National University of San Marcos

Faculty of Mathematical Sciences

School of Scientific Computing



Curricular Plan 2026

– Reporte Final–

Última modificación: Febrero de 2026

This document is based on the international proposal of ACM/IEEE-CS *Computing Curricula*, which also served as the basis for the Spanish version created by the Peruvian Computer Society (SPC).

ISBN:

All rights reserved. Ernesto Cuadros-Vargas, National University of San Marcos (UNMSM).

Task Force

Dr. Maria Zegarra-Garay
Dean of the Faculty of Mathematical Sciences
email: *mzegarrag@unmsm.edu.pe*

Mag. Jorge Icaro Condado Jáuregui
Acting Director of Scientific Computing
email: *jcondadoj@unmsm.edu.pe*

Dr. Frida Mereyda López Córdova
Ph.D. in Systems and Information Engineering, UNMSM (2025)
Professor, Academic Department of Mathematics
email: *flopezc@unmsm.edu.pe*

MSc Victoria Ysabel Rojas Rojas
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *vrojasr@unmsm.edu.pe*

MSc Rubén Arbañil
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *rarbanilr@unmsm.edu.pe*

BSc Máximo Gerardo Gonzales Chávez
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *mgonzalesc1@unmsm.edu.pe*

MSc Oscar Benito Pacheco
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *obenitop@unmsm.edu.pe*

Dr. Rosa Medina Aguilar
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *rmedinaa1@unmsm.edu.pe*

MSc Luz Teresa Quispe Vega
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *lquispev@unmsm.edu.pe*

MSc Miriam Gisell Pescorán Florencio
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *mpescoranf@unmsm.edu.pe*

BSc Joseph Wilmer Simpe Laura
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *jsimpel@unmsm.edu.pe*

MSc Francisco Felipe Quiroz Chueca
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *fquirozc@unmsm.edu.pe*

MSc Rómulo Fernando Lomparte Alvarado
Professor at the Faculty of Mathematical Sciences, UNMSM
email: *rlompartea@unmsm.edu.pe*

Dr. Omar Stevenson Guzmán Rea
Ph.D. in Mathematics. Federal University of Pernambuco (2020)
Professor, Academic Department of Mathematics
email: *oguzmanr@unmsm.edu.pe*

Ernesto Cuadros-Vargas <*ecuadros@spc.org.pe*>
PhD in Computer Science, University of Sao Paulo, Brazil (2004)
Distinguished Speaker for the Association of Computing Machinery (ACM) (2025-2027)
Member of the Board of Governors of the IEEE Computer Society (2020-2023)
Member of the Steering Committee ACM/ IEEE-CS Computing Curricula 2020 (CS2020)
Member of the Steering Committee ACM/ IEEE-CS Computing Curricula/Computer Science (CS2013)
Executive Secretary of the Latin American Center for Computing Studies (CLEI) 2008-2016
email: *ecuadros@spc.org.pe*

Student Representatives

Maria Fernanda Rupay Caballero
Student of Scientific Computing, UNMSM, 2024
email: *maria.rupayc@unmsm.edu.pe*

Sebastián Alfredo Borja Ramos
Student of Scientific Computing, UNMSM, 2024
email: *sebastian.borja@unmsm.edu.pe*

Angel Aarón Reyes Cáceres
Student of Scientific Computing, UNMSM, 2024
email: *angel.reyesc@unmsm.edu.pe*

Carlos Jesús López Bravo
Student of Scientific Computing, UNMSM, 2024
email: *carlos.lopez22@unmsm.edu.pe*

Julio André Carrillo Montero
Student of Scientific Computing, UNMSM, 2023
email: *julio.carrillom@unmsm.edu.pe*

Adrian Enrique Borstelmann Segura
Student of Scientific Computing, UNMSM, 2025
email: *adrian.borstelmanns@unmsm.edu.pe*

Jan Lucas Mancinelli Vite
Student of Scientific Computing, UNMSM, 2023
email: *jan.mancinelli@unmsm.edu.pe*

Adriano Emer Valdivia
Student of Scientific Computing, UNMSM, 2025
email: *adriano.valdivia@unmsm.edu.pe*

Cristian Josue Bernal Nuñez
Student of Scientific Computing, UNMSM, 2023
email: *cristian.bernal@unmsm.edu.pe*

Jhanilde Pozo Hernandez
Student of Scientific Computing, UNMSM, 2023
email: *jhanilde.pozo@unmsm.edu.pe*

Franco David Zevallos Silva
Student of Scientific Computing, UNMSM, 2025
email: *franco.zevalloss@unmsm.edu.pe*

Angello Andre Manyari Medina
Student of Scientific Computing, UNMSM, 2023
email: *angello.manyari@unmsm.edu.pe*

Resumen ejecutivo

Este documento representa el informe de la nueva malla curricular 2026 del Escuela Profesional de Computación Científica (Matemática Computacional) de la Universidad Nacional Mayor de San Marcos (*https://www.unmsm.edu.pe*) en la ciudad de Lima-Perú.

Este documento presenta el diseño curricular de la carrera de Scientific Computing bajo los estándares internacionales del *Computing Curricula* CC2020, marco de referencia desarrollado conjuntamente por la *Association for Computing Machinery* (ACM) y la *IEEE Computer Society* (IEEE-CS). El CC2020, disponible en http://www.acm.org/education, sintetiza tres décadas de evolución disciplinar (desde CS2001 hasta CC2020) y responde a las demandas actuales de la cuarta revolución industrial.

La propuesta adapta este referente global a nuestro contexto regional, incorporando:

- Requerimientos del mercado laboral local en tecnología
- Fortalezas institucionales en investigación aplicada
- Tendencias emergentes en educación computacional

La computación hoy en día presenta los siguientes perfiles de formación profesional para pregrado:

- Ingeniería de Computación (*Computer Engineering* – CE),
- Ciencia de la Computación (*Computer Science* – CS),
- Sistemas de Información (*Information Systems* – IS),
- Ingeniería de Software (*Software Engineering* – SE),
- Tecnología de la Información (*Information Technology* – IT),
- Ciberseguridad (*Cybersecurity* – CY) y
- Ciencia de Datos (*Data Science* – DS).

El modelo pedagógico se sustenta en tres ejes transformadores:

- **Excelencia técnica**: Dominio de fundamentos computacionales y especialización progresiva
- **Innovación disruptiva**: Metodologías ágiles, pensamiento computacional y creación de startups
- **Responsabilidad social**: Ética profesional, sostenibilidad tecnológica e impacto comunitario

Esta malla producirá profesionales con:

- Capacidad para resolver problemas complejos en entornos globales
- Competencias para liderar la transformación digital
- Sensibilidad hacia los desafíos éticos de la inteligencia artificial, privacidad y gobernanza tecnológica

Esta propuesta incorpora mecanismos de actualización continua, asegurando relevancia frente a la acelerada obsolescencia tecnológica y alineamiento con los estándares de acreditación internacional.

Contents

Agradecimientos **xxvii**

Acronyms **xxix**

1 Introducción **1**
1.1 Basic definitions . . . 1
1.2 Occupational Field and Job Market . . . 1
1.3 Importancia de la carrera en la sociedad . . . 2
1.4 Institutional Info . . . 3
1.4.1 Mission . . . 3
1.4.2 Vision . . . 3
1.5 Accreditation information . . . 3
1.5.1 Career Outcomes . . . 3
1.5.2 Educational objectives . . . 3
1.5.3 Learning Outcomes . . . 4
1.6 Profiles . . . 4
1.6.1 Applicant profile . . . 4
1.6.2 Graduate profile . . . 5
1.6.3 ProfessionalProfile . . . 5
1.6.4 Faculty profile . . . 6
1.7 Degrees and Titles . . . 6
1.8 Resources to teach . . . 6
1.9 Document Organization . . . 6

2 Body of knowledge of Computational Mathematics **9**
2.1 Numerical and Scientific Analysis (NSA) . . . 10
2.1.1 NSA/Computational Arithmetic and Error Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 10
2.1.2 NSA/Numerical Linear Algebra (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 10
2.1.3 NSA/Roots Finding and Basic Numerical Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 11
2.1.4 NSA/Interpolation and Function Approximation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 11
2.1.5 NSA/Numerical Differentiation and Integration (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 11
2.1.6 NSA/Numerical Methods for Ordinary Differential Equations (ODEs) (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 12
2.2 Mathematical Modeling and Simulation (MMS) . . . 12
2.2.1 MMS/Mathematical Model Formulation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 12
2.2.2 MMS/Discrete Systems Simulation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 13
2.2.3 MMS/Continuous Systems Simulation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 13
2.2.4 MMS/Sensitivity Analysis and Robustness (Core Tier1: 1 hr, Core Tier2: 1 hr) 13
2.2.5 MMS/Model Validation and Verification (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 13
2.3 Probability and Statistical Computing (PSC) . . . 14

2.3.1 PSC/Advanced Probability and Stochastic Processes (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 14
2.3.2 PSC/Computational Statistical Inference (Core Tier1: 1 hr, Core Tier2: 1 hr) . 14
2.3.3 PSC/Design of Computational Experiments (Core Tier1: 1 hr, Core Tier2: 1 hr) 15
2.3.4 PSC/Resampling and Validation Methods (Core Tier1: 1 hr, Core Tier2: 1 hr) 15
2.4 Data Structures and Algorithms (DSA) . . . 15
2.4.1 DSA/Algorithmic Complexity and Asymptotic Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 16
2.4.2 DSA/Fundamental and Advanced Data Structures (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 16
2.4.3 DSA/Searching and Sorting Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) . 16
2.4.4 DSA/Graph Algorithms and Combinatorial Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 17
2.4.5 DSA/Parallel and Distributed Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) 17
2.5 High-Performance Computing (HPC) . . . 17
2.5.1 HPC/Parallel Computer Architectures (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 18
2.5.2 HPC/Shared Memory Parallel Programming (Core Tier1: 1 hr, Core Tier2: 1 hr) 18
2.5.3 HPC/Distributed Memory Parallel Programming (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 18
2.5.4 HPC/Accelerator Programming (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 18
2.5.5 HPC/Parallel Numerical Methods (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 19
2.6 Optimization and Machine Learning (OML) . . . 19
2.6.1 OML/Convex and Non-Convex Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 19
2.6.2 OML/Constrained Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 20
2.6.3 OML/Optimization for Machine Learning (Core Tier1: 1 hr, Core Tier2: 1 hr) 20
2.6.4 OML/Large-Scale and Distributed Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 20
2.6.5 OML/Metaheuristics and Bio-Inspired Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 21
2.7 Data Science and Machine Learning Mathematics (DSM) . . . 21
2.7.1 DSM/Foundations of Statistical Learning (Core Tier1: 1 hr, Core Tier2: 1 hr) 21
2.7.2 DSM/Supervised Learning Methods (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 21
2.7.3 DSM/Unsupervised Learning Methods (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 22
2.7.4 DSM/Model Evaluation and Selection (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 22
2.8 Artificial Intelligence Mathematics (AIM) . . . 22
2.8.1 AIM/Deep Learning Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 23
2.8.2 AIM/Advanced Neural Architectures (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 23
2.8.3 AIM/Reinforcement Learning (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 23
2.8.4 AIM/Natural Language Processing with Transformers (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 24
2.8.5 AIM/Generative Models and GANs (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 24
2.9 Symbolic and Discrete Computation (SDC) . . . 24
2.9.1 SDC/Group Theory and Algebraic Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 24
2.9.2 SDC/Computational Number Theory and Cryptography (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 25
2.9.3 SDC/Symbolic Computation Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 25
2.9.4 SDC/Computational Logic and Formal Verification (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 25
2.10 Signal, Image, and Visualization (SIV) . . . 26
2.10.1 SIV/Fourier Analysis and Wavelets (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 26
2.10.2 SIV/Digital Image Processing and Computer Vision (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 26
2.10.3 SIV/Scientific Visualization and Visual Computing (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 27

2.10.4 SIV/Computer Graphics and Computational Geometry (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 27
2.11 Quantum Computing and Algorithms (QCA) . . . 27
2.11.1 QCA/Quantum Mechanics Foundations for Computation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 27
2.11.2 QCA/Qubits and Quantum Circuits (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 28
2.11.3 QCA/Quantum Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 28
2.11.4 QCA/Post-Quantum Cryptography and Quantum Simulation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 28
2.12 Cybersecurity and Privacy (CSP) . . . 29
2.12.1 CSP/Symmetric Cryptography (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 29
2.12.2 CSP/Asymmetric Cryptography (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 29
2.12.3 CSP/Privacy-Preserving Computation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 30
2.12.4 CSP/Information Theory and Data Compression (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 30
2.13 Computational Mathematics for Specific Domains (CDM) . . . 30
2.13.1 CDM/Bioinformatics and Computational Biology (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 30
2.13.2 CDM/Computational Finance and Risk Science (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 31
2.13.3 CDM/Computational Physics and Quantum Chemistry (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 31
2.13.4 CDM/Computational Materials Science (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 31
2.13.5 CDM/Climate Modeling and Complex Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 32
2.14 Data Governance and Management (DGM) . . . 32
2.14.1 DGM/Ethics in AI and Responsible Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 32
2.14.2 DGM/Data Project Management and MLOps (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 33
2.14.3 DGM/Data Governance and Regulatory Compliance (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 33

3 Body of knowledge of Computer Science 35
3.1 Artificial Intelligence (AI) . . . 37
3.1.1 AI/Introduction to AI: Concepts and Intelligent Behavior (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 38
3.1.2 AI/Problem Characteristics and Agent Nature (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 39
3.1.3 AI/State and Uninformed Search (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 39
3.1.4 AI/Heuristic Search and CSP (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 39
3.1.5 AI/Adversarial Search and Stochastic Optimization (Core Tier1: 1 hr, Core Tier2: 3 hrs) . . . 40
3.1.6 AI/Knowledge Representation and Probabilistic Reasoning Foundations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 40
3.1.7 AI/Independence, Markov Models and Decision Making (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 41
3.1.8 AI/Machine Learning Fundamentals (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 41
3.1.9 AI/Machine Learning Data, Representations and Evaluation (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 41
3.1.10 AI/Neural Networks and Ethics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 42
3.1.11 AI/Machine Learning Formulation and Evaluation (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 42
3.1.12 AI/Deep Learning, RL and Ethics (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 43
3.1.13 AI/Neural Network Training and Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 43

3.1.14 AI/Regularization Techniques for Deep Networks (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 43
3.1.15 AI/Convolutional and Residual Network Architectures (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 43
3.1.16 AI/Transformer Architectures and Generative Models (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 44
3.1.17 AI/AI Applications and Societal Impact (Core Tier1: 2 hrs, Core Tier2: 2 hrs) 44
3.1.18 AI/Additional AI Applications and Societal Impact (Core Tier1: 2 hrs, Core Tier2: 2 hrs) . . . 44
3.1.19 AI/Logical Representation and Reasoning . . . 45
3.1.20 AI/Probabilistic Representation and Reasoning . . . 45
3.1.21 AI/Planning . . . 46
3.1.22 AI/Agents and Cognitive Systems . . . 46
3.1.23 AI/Agentic Loop Engineering . . . 47
3.1.24 AI/Natural Language Processing . . . 47
3.1.25 AI/Robotics . . . 48
3.1.26 AI/Perception and Computer Vision . . . 49
3.2 Algorithmic Foundations (AL) . . . 50
3.2.1 AL/Foundational Data Structures (Core Tier1: 5 hrs, Core Tier2: 3 hrs) . . . 50
3.2.2 AL/Foundational Algorithms (Core Tier1: 4 hrs, Core Tier2: 2 hrs) . . . 51
3.2.3 AL/Advanced Algorithms (Core Tier1: 3 hrs, Core Tier2: 2 hrs) . . . 51
3.2.4 AL/Algorithmic Strategies (Core Tier1: 6 hrs) . . . 52
3.2.5 AL/Complexity Analysis Framework (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 52
3.2.6 AL/Asymptotic Notation and Complexity Classes (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 53
3.2.7 AL/Complexity Analysis: Recursion, Amortization, and Tight Bounds (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 53
3.2.8 AL/Computational Complexity Theory (Core Tier1: 2 hrs, Core Tier2: 1 hr) . 53
3.2.9 AL/Formal Languages and Automata (Core Tier1: 4 hrs, Core Tier2: 10 hrs) . 54
3.2.10 AL/Computability and Decidability (Core Tier1: 6 hrs, Core Tier2: 14 hrs) . . 54
3.2.11 AL/Society, Ethics, and the Profession . . . 55
3.3 Architecture and Organization (AR) . . . 56
3.3.1 AR/Digital Logic and Digital Systems (Core Tier2: 3 hrs) . . . 56
3.3.2 AR/Machine-Level Data Representation (Core Tier1: 1 hr) . . . 57
3.3.3 AR/Assembly Level Machine Organization (Core Tier1: 1 hr, Core Tier2: 2 hrs) 57
3.3.4 AR/Memory Hierarchy (Core Tier1: 6 hrs) . . . 58
3.3.5 AR/Interfacing and Communication (Core Tier1: 1 hr) . . . 58
3.3.6 AR/Functional Organization (Core Tier2: 2 hrs) . . . 58
3.3.7 AR/Performance and Energy Efficiency (Core Tier2: 3 hrs) . . . 59
3.3.8 AR/Heterogeneous Architectures (Core Tier2: 2 hrs) . . . 59
3.3.9 AR/Secure Processor Architectures (Core Tier2: 2 hrs) . . . 60
3.3.10 AR/Quantum Architectures (Core Tier2: 2 hrs) . . . 60
3.3.11 AR/Sustainability Issues . . . 61
3.4 Data Management (DM) . . . 61
3.4.1 DM/The Role of Data and the Data Life Cycle (Core Tier1: 2 hrs) . . . 63
3.4.2 DM/Core Database System Concepts (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 63
3.4.3 DM/Data Modeling (Core Tier1: 2 hrs, Core Tier2: 3 hrs) . . . 64
3.4.4 DM/Relational Databases (Core Tier1: 1 hr, Core Tier2: 3 hrs) . . . 64
3.4.5 DM/Query Construction (Core Tier1: 2 hrs, Core Tier2: 4 hrs) . . . 64
3.4.6 DM/Query Processing (Core Tier2: 4 hrs) . . . 65
3.4.7 DM/DBMS Internals (Core Tier2: 4 hrs) . . . 66
3.4.8 DM/NoSQL Systems (Core Tier2: 2 hrs) . . . 66
3.4.9 DM/Data Security and Privacy (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 66
3.4.10 DM/Data Analytics (Core Tier2: 3 hrs) . . . 67
3.4.11 DM/Distributed Databases/Cloud Computing . . . 67
3.4.12 DM/Semi-structured and Unstructured Databases . . . 67

3.4.13 DM/Society, Ethics, and the Profession . . . 68
3.5 Foundations of Programming Languages (FPL) . . . 68
3.5.1 FPL/Object-Oriented Programming Fundamentals (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 69
3.5.2 FPL/Object-Oriented Programming: Encapsulation, Subtyping, and Reflection (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 70
3.5.3 FPL/Functional Programming (Core Tier1: 4 hrs, Core Tier2: 3 hrs) . . . 70
3.5.4 FPL/Logic Programming (Core Tier2: 2 hrs) . . . 71
3.5.5 FPL/Shell Scripting (Core Tier1: 2 hrs) . . . 72
3.5.6 FPL/Event-Driven and Reactive Programming (Core Tier1: 2 hrs, Core Tier2: 2 hrs) . . . 72
3.5.7 FPL/Parallel and Distributed Computing (Core Tier1: 2 hrs, Core Tier2: 2 hrs) 73
3.5.8 FPL/Aspect-Oriented Programming . . . 73
3.5.9 FPL/Type Systems Fundamentals (Core Tier1: 2 hrs, Core Tier2: 2 hrs) . . . 74
3.5.10 FPL/Type Systems: Polymorphism and Complementary Typing (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 74
3.5.11 FPL/Advanced Type Theory (Core Tier1: 2 hrs, Core Tier2: 2 hrs) . . . 75
3.5.12 FPL/Systems Execution and Memory Model (Core Tier1: 2 hrs) . . . 75
3.5.13 FPL/Memory Management (Core Tier1: 1 hr) . . . 75
3.5.14 FPL/Language Translation and Execution (Core Tier1: 2 hrs, Core Tier2: 3 hrs) 76
3.5.15 FPL/Program Abstraction and Representation (Core Tier2: 3 hrs) . . . 76
3.5.16 FPL/Syntax Analysis . . . 77
3.5.17 FPL/Compiler Semantic Analysis . . . 78
3.5.18 FPL/Program Analysis and Analyzers . . . 78
3.5.19 FPL/Code Generation . . . 78
3.5.20 FPL/Run-time Behavior and Systems . . . 79
3.5.21 FPL/Advanced Programming Constructs . . . 80
3.5.22 FPL/Language Pragmatics . . . 80
3.5.23 FPL/Formal Semantics . . . 81
3.5.24 FPL/Formal Development Methodologies . . . 81
3.5.25 FPL/Design Principles of Programming Languages . . . 81
3.5.26 FPL/Society, Ethics, and the Profession . . . 82
3.6 Graphics and Interactive Techniques (GIT) . . . 82
3.6.1 GIT/Fundamental Concepts of Graphics and Interactive Techniques (Core Tier1: 4 hrs, Core Tier2: 3 hrs) . . . 83
3.6.2 GIT/Visualization (Core Tier2: 6 hrs) . . . 83
3.6.3 GIT/Applied Rendering and Techniques (Core Tier2: 15 hrs) . . . 83
3.6.4 GIT/Geometric Modeling (Core Tier2: 6 hrs) . . . 84
3.6.5 GIT/Shading and Advanced Rendering (Core Tier2: 6 hrs) . . . 84
3.6.6 GIT/Computer Animation (Core Tier2: 6 hrs) . . . 84
3.6.7 GIT/Simulation (Core Tier2: 6 hrs) . . . 84
3.6.8 GIT/Immersion (Core Tier2: 6 hrs) . . . 85
3.6.9 GIT/Interaction (Core Tier2: 4 hrs) . . . 85
3.6.10 GIT/Image Processing (Core Tier2: 6 hrs) . . . 85
3.6.11 GIT/Tangible/Physical Computing (Core Tier2: 6 hrs) . . . 86
3.6.12 GIT/Society, Ethics, and the Profession . . . 86
3.7 Human-Computer Interaction (HCI) . . . 86
3.7.1 HCI/Understanding the User: Individual goals and interactions with others (Core Tier1: 2 hrs, Core Tier2: 5 hrs) . . . 87
3.7.2 HCI/Accountability and Responsibility in Design (Core Tier1: 2 hrs, Core Tier2: 2 hrs) . . . 87
3.7.3 HCI/Accessibility and Inclusive Design (Core Tier1: 2 hrs, Core Tier2: 2 hrs) . 88
3.7.4 HCI/Evaluating the Design (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 88
3.7.5 HCI/System Design (Core Tier1: 1 hr, Core Tier2: 5 hrs) . . . 88
3.7.6 HCI/Society, Ethics, and the Profession . . . 89
3.8 Mathematical and Statistical Foundations (MSF) . . . 89

3.8.1 MSF/Discrete Mathematics (Core Tier1: 15 hrs, Core Tier2: 6 hrs) 89
3.8.2 MSF/Advanced Discrete Mathematics (Core Tier1: 15 hrs, Core Tier2: 6 hrs) 90
3.8.3 MSF/Probability (Core Tier1: 11 hrs, Core Tier2: 29 hrs) 90
3.8.4 MSF/Statistics (Core Tier1: 10 hrs, Core Tier2: 30 hrs) 91
3.8.5 MSF/Linear Algebra (Core Tier1: 5 hrs, Core Tier2: 35 hrs) 91
3.8.6 MSF/Calculus (Core Tier2: 40 hrs) 92
3.9 Networking and Communication (NC) 92
3.9.1 NC/Fundamentals of Networks and Communications (Core Tier1: 3 hrs) . . . 93
3.9.2 NC/Networked Applications (Core Tier1: 4 hrs) 93
3.9.3 NC/Reliability Support (Core Tier2: 6 hrs) 94
3.9.4 NC/Routing and Forwarding (Core Tier2: 4 hrs) 94
3.9.5 NC/Single Hop Communication (Core Tier2: 3 hrs) 94
3.9.6 NC/Network Security (Core Tier2: 3 hrs) 95
3.9.7 NC/Mobility Support (Core Tier2: 4 hrs) 95
3.9.8 NC/Emerging Topics (Core Tier2: 4 hrs) 95
3.10 Operating Systems (OS) 96
3.10.1 OS/Role and Purpose of Operating Systems (Core Tier1: 2 hrs) 96
3.10.2 OS/Principles of Operating System (Core Tier1: 2 hrs) 97
3.10.3 OS/Concurrency (Core Tier1: 2 hrs, Core Tier2: 1 hr) 97
3.10.4 OS/Protection and Safety (Core Tier1: 2 hrs, Core Tier2: 1 hr) 98
3.10.5 OS/Scheduling (Core Tier2: 2 hrs) 99
3.10.6 OS/Process Model (Core Tier2: 2 hrs) 99
3.10.7 OS/Memory Management (Core Tier2: 1 hr) 100
3.10.8 OS/Device management (Core Tier2: 1 hr) 100
3.10.9 OS/File Systems API and Implementation (Core Tier2: 2 hrs) 101
3.10.10 OS/Advanced File systems (Core Tier2: 1 hr) 101
3.10.11 OS/Virtualization (Core Tier2: 1 hr) 102
3.10.12 OS/Real-time and Embedded Systems (Core Tier2: 1 hr) 102
3.10.13 OS/Fault tolerance (Core Tier2: 1 hr) 103
3.10.14 OS/Society, Ethics, and the Profession 103
3.11 Parallel and Distributed Computing (PDC) 104
3.11.1 PDC/Programs: Declarative Parallelism (Core Tier1: 1 hr, Core Tier2: 1 hr) . 105
3.11.2 PDC/Programs: Starting Activities (Core Tier1: 1 hr, Core Tier2: 1 hr) 105
3.11.3 PDC/Programs: Execution Properties (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 105
3.11.4 PDC/Programs: Distribution (Core Tier1: 1 hr, Core Tier2: 1 hr) 106
3.11.5 PDC/Programs: Implementation Mappings (Core Tier1: 1 hr, Core Tier2: 1 hr) 106
3.11.6 PDC/GPU Programming 106
3.11.7 PDC/Communication (Core Tier1: 2 hrs, Core Tier2: 4 hrs) 107
3.11.8 PDC/Communication: Properties and Extensions (Core Tier1: 1 hr, Core Tier2: 1 hr) 107
3.11.9 PDC/Memory and Consistency (Core Tier1: 1 hr, Core Tier2: 1 hr) 108
3.11.10 PDC/Coordination (Core Tier1: 1 hr, Core Tier2: 3 hrs) 108
3.11.11 PDC/Coordination: Synchronization and Atomicity (Core Tier1: 1 hr, Core Tier2: 2 hrs) 109
3.11.12 PDC/Coordination: Advanced Properties (Core Tier1: 1 hr, Core Tier2: 2 hrs) 109
3.11.13 PDC/Evaluation (Core Tier1: 1 hr, Core Tier2: 3 hrs) 109
3.11.14 PDC/Evaluation: Formal Methods (Core Tier1: 1 hr, Core Tier2: 1 hr) 110
3.11.15 PDC/Evaluation: Testing and Measurement (Core Tier1: 1 hr, Core Tier2: 1 hr) 110
3.11.16 PDC/Algorithms (Core Tier1: 1 hr, Core Tier2: 3 hrs) 111
3.11.17 PDC/Algorithms: Application Domains Survey (Core Tier1: 1 hr, Core Tier2: 3 hrs) 111
3.11.18 PDC/Algorithms: Algorithmic Domains (Core Tier1: 1 hr, Core Tier2: 3 hrs) . 112
3.12 Software Development Fundamentals (SDF) 112
3.12.1 SDF/Basic Programming Concepts and Constructs (Core Tier1: 10 hrs) 113
3.12.2 SDF/Programming Practices: Testing, Debugging, and Documentation (Core Tier1: 10 hrs) 113

3.12.3 SDF/Fundamental Data Structures (Core Tier1: 12 hrs) . . . 114
3.12.4 SDF/Algorithms (Core Tier1: 6 hrs) . . . 114
3.12.5 SDF/Software Development Practices (Core Tier1: 5 hrs) . . . 114
3.12.6 SDF/Society, Ethics, and the Profession . . . 115
3.13 Software Engineering (SE) . . . 115
3.13.1 SE/Team Collaboration and Dynamics (Core Tier1: 2 hrs, Core Tier2: 2 hrs) . 116
3.13.2 SE/Diversity, Stakeholders and Distributed Teams (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 117
3.13.3 SE/Version Control and CI/CD (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 117
3.13.4 SE/Development Environment and IDE (Core Tier1: 1 hr, Core Tier2: 2 hrs) . 118
3.13.5 SE/Requirements Engineering (Core Tier2: 2 hrs) . . . 118
3.13.6 SE/Requirements Evolution and Estimation (Core Tier2: 1 hr) . . . 119
3.13.7 SE/Principles and Software Architecture (Core Tier1: 1 hr, Core Tier2: 2 hrs) 119
3.13.8 SE/API Design and Modeling (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 120
3.13.9 SE/Software Design Quality and Evaluation (Core Tier1: 1 hr, Core Tier2: 2 hrs) 120
3.13.10 SE/Coding Practices (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 121
3.13.11 SE/Large-Scale Construction and Process (Core Tier1: 1 hr, Core Tier2: 2 hrs) 122
3.13.12 SE/Verification and validation Concepts (Core Tier1: 1 hr, Core Tier2: 1 hr) . 122
3.13.13 SE/Test Planning and Types (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 122
3.13.14 SE/Advanced Testing Practices (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 123
3.13.15 SE/Testing Tools and Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 123
3.13.16 SE/Performance Testing and Benchmarking (Core Tier1: 1 hr, Core Tier2: 1 hr) 124
3.13.17 SE/API Compatibility and Versioning (Core Tier2: 2 hrs) . . . 124
3.13.18 SE/Refactoring Techniques (Core Tier2: 1 hr) . . . 125
3.13.19 SE/Reliability Concepts (Core Tier2: 2 hrs) . . . 125
3.13.20 SE/Reliability Engineering (Core Tier2: 1 hr) . . . 125
3.13.21 SE/Formal Methods . . . 126
3.14 Security (SEC) . . . 126
3.14.1 SEC/Security Mindset and Principles (Core Tier1: 1 hr, Core Tier2: 3 hrs) . . 127
3.14.2 SEC/Access Control and Applications (Core Tier1: 1 hr, Core Tier2: 3 hrs) . . 128
3.14.3 SEC/AI and Hardware Security (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 128
3.14.4 SEC/Security Ethics, Law and Society (Core Tier1: 1 hr, Core Tier2: 2 hrs) . 128
3.14.5 SEC/Privacy and Security by Design (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 129
3.14.6 SEC/Injection and Input Validation (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 129
3.14.7 SEC/Memory Safety and Types (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 130
3.14.8 SEC/Information Flow and Non-Interference (Core Tier1: 1 hr, Core Tier2: 1 hr) 130
3.14.9 SEC/Malware Analysis and Advanced Security (Core Tier1: 1 hr, Core Tier2: 3 hrs) . . . 130
3.14.10 SEC/Cryptography Fundamentals (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 131
3.14.11 SEC/Cryptographic Algorithms (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 131
3.14.12 SEC/Protocols, Applications and Advanced Cryptography (Core Tier1: 1 hr, Core Tier2: 2 hrs) . . . 132
3.14.13 SEC/Threat Analysis and Security Engineering (Core Tier1: 1 hr, Core Tier2: 4 hrs) . . . 132
3.14.14 SEC/Security Design and Controls Engineering (Core Tier1: 1 hr, Core Tier2: 3 hrs) . . . 133
3.14.15 SEC/Trusted Computing and Privacy Engineering (Core Tier1: 1 hr, Core Tier2: 3 hrs) . . . 133
3.14.16 SEC/Digital Forensics Techniques (Core Tier2: 4 hrs) . . . 134
3.14.17 SEC/Legal Aspects and Incident Handling (Core Tier2: 3 hrs) . . . 134
3.14.18 SEC/Security Governance, Management and Policy (Core Tier2: 2 hrs) . . . 135
3.14.19 SEC/Risk and Data Management (Core Tier2: 1 hr) . . . 135
3.15 Society, Ethics, and the Profession (SEP) . . . 135
3.15.1 SEP/Social Impact and AI (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 136
3.15.2 SEP/Digital Equity and Accessibility (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . 137
3.15.3 SEP/Ethical Foundations and Frameworks (Core Tier1: 2 hrs, Core Tier2: 1 hr) 137

3.15.4 SEP/Ethical Analysis Methods and Tools (Core Tier1: 1 hr, Core Tier2: 1 hr) 138
3.15.5 SEP/Professional Values, Nature and Continuing Development (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 138
3.15.6 SEP/Ethics Codes and Legal Accountability (Core Tier1: 1 hr, Core Tier2: 1 hr) 139
3.15.7 SEP/Professional Responsibility and Public Policy (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 139
3.15.8 SEP/Ethical Dissent and Cultural Context (Core Tier1: 1 hr, Core Tier2: 1 hr) 140
3.15.9 SEP/Workplace Ethics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 140
3.15.10 SEP/Intellectual Property Foundations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 140
3.15.11 SEP/Licensing, Plagiarism and Authorship (Core Tier1: 1 hr, Core Tier2: 1 hr) 141
3.15.12 SEP/Digital IP Rights and Copyright (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 141
3.15.13 SEP/Open Source, Piracy and DRM (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 141
3.15.14 SEP/Data, Surveillance and Privacy Technology (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 142
3.15.15 SEP/Privacy Philosophical Foundations and Legislation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 142
3.15.16 SEP/Civil Liberties and Expression (Core Tier1: 1 hr, Core Tier2: 1 hr) 142
3.15.17 SEP/Technical and Professional Communication (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 143
3.15.18 SEP/Team Communication and Culture (Core Tier1: 1 hr, Core Tier2: 1 hr) . 143
3.15.19 SEP/Environmental Footprint of Computing Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 144
3.15.20 SEP/Systemic Effects and Social Context (Core Tier1: 1 hr, Core Tier2: 1 hr) 144
3.15.21 SEP/Sustainable Design and Pervasive Computing (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 144
3.15.22 SEP/Foundations and Early History of Computing (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 145
3.15.23 SEP/Modern History: PC, Internet and AI (Core Tier1: 1 hr, Core Tier2: 1 hr) 145
3.15.24 SEP/Economic Models and Strategies (Core Tier2: 1 hr) . . . 146
3.15.25 SEP/Automation, AI and Economic Ethics (Core Tier2: 1 hr) . . . 146
3.15.26 SEP/Computer Crimes and Attacks (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 146
3.15.27 SEP/Security Laws, Policies and Responsibility (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 147
3.15.28 SEP/Identity and Diversity in Computing (Core Tier1: 1 hr, Core Tier2: 1 hr) 147
3.15.29 SEP/DEIA Historical and Cross-Cultural Context (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 148
3.15.30 SEP/Inclusion and Computing Practices (Core Tier1: 1 hr, Core Tier2: 1 hr) . 148
3.15.31 SEP/Accessible Design (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 148
3.16 System Fundamentals (SF) . . . 149
3.16.1 SF/Hardware Components and Logic (Core Tier1: 2 hrs) . . . 149
3.16.2 SF/Layers, Abstractions and Scalability (Core Tier1: 2 hrs) . . . 150
3.16.3 SF/Digital Systems and Logic (Core Tier1: 2 hrs) . . . 150
3.16.4 SF/State Machines (Core Tier1: 2 hrs) . . . 150
3.16.5 SF/Sequential and Parallel Processing (Core Tier1: 2 hrs) . . . 151
3.16.6 SF/Resource Management (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 151
3.16.7 SF/Latency, Cache and Memory Hierarchy (Core Tier1: 2 hrs, Core Tier2: 2 hrs) 152
3.16.8 SF/Virtualization and Isolation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 152
3.16.9 SF/Performance Metrics and Benchmarks (Core Tier1: 1 hr, Core Tier2: 1 hr) 153
3.16.10 SF/Analytical Performance Models (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 153
3.16.11 SF/Events, Tools and Experimentation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 153
3.16.12 SF/System Reliability (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 154
3.16.13 SF/System Security (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 154
3.16.14 SF/System Design (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 155
3.16.15 SF/Society, Ethics, and the Profession . . . 155
3.17 Specialized Platform Development (SPD) . . . 155

3.17.1 SPD/Common Aspects: Platforms, APIs and Constraints (Core Tier1: 3 hrs, Core Tier2: 2 hrs) . . . 156
3.17.2 SPD/Common Aspects: Platform Languages and Web Programming Patterns (Core Tier1: 2 hrs, Core Tier2: 1 hr) . . . 157
3.17.3 SPD/Frontend, Frameworks and Web Standards (Core Tier2: 3 hrs) . . . 157
3.17.4 SPD/Cloud Architecture and Data Management (Core Tier2: 3 hrs) . . . 157
3.17.5 SPD/Mobile Platforms: Development Fundamentals (Core Tier2: 2 hrs) . . . 158
3.17.6 SPD/Mobile Platforms: Architecture and Constraints (Core Tier2: 2 hrs) . . . 158
3.17.7 SPD/Mobile Platforms: Capabilities and Asynchronous Computing (Core Tier2: 1 hr) . . . 158
3.17.8 SPD/Robot Platforms (Core Tier2: 4 hrs) . . . 158
3.17.9 SPD/Embedded Characteristics and Architectures (Core Tier2: 1 hr) . . . 159
3.17.10 SPD/Systems, Programming and Sensors (Core Tier2: 2 hrs) . . . 159
3.17.11 SPD/Real-Time, Memory and Verification (Core Tier2: 3 hrs) . . . 160
3.17.12 SPD/Platform History, Types and Characteristics (Core Tier2: 1 hr) . . . 160
3.17.13 SPD/Sensors, Controllers and eSports Ecosystems (Core Tier2: 1 hr) . . . 161
3.17.14 SPD/Architectures and Rendering (Core Tier2: 1 hr) . . . 161
3.17.15 SPD/Algorithms, Media and Virtual Worlds (Core Tier2: 1 hr) . . . 161
3.17.16 SPD/Languages, APIs and Development Tools (Core Tier2: 1 hr) . . . 161
3.17.17 SPD/Game Engines (Core Tier2: 1 hr) . . . 162
3.17.18 SPD/Game Design Foundations (Core Tier2: 1 hr) . . . 162
3.17.19 SPD/Genres, Audiences and Social Reach (Core Tier2: 1 hr) . . . 162
3.17.20 SPD/Interactive Data and Analytics Platforms (Core Tier2: 2 hrs) . . . 163
3.17.21 SPD/AI Programming and Quantum Platforms (Core Tier2: 1 hr) . . . 163
3.17.22 SPD/SPD-SEP/Mobile . . . 164
3.17.23 SPD/SPD-SEP/Web . . . 164
3.17.24 SPD/Intellectual Property in Games . . . 164
3.17.25 SPD/Game Interface and Accessibility . . . 165
3.17.26 SPD/Health, Safety and Sustainability . . . 165
3.17.27 SPD/Representation, Inclusion and Equality . . . 165
3.17.28 SPD/SPD-SEP/Robotics . . . 166
3.17.29 SPD/SPD-SEP/Interactive . . . 166

4 Body of knowledge of Mathematics 167
4.1 Calculus and Analysis (CAN) . . . 168
4.1.1 CAN/Limits and Continuity (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 168
4.1.2 CAN/Differential Calculus (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 169
4.1.3 CAN/Antiderivatives and Integration Techniques (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 169
4.1.4 CAN/Improper Integrals (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 169
4.1.5 CAN/Sequences, Series, and Power Series (Core Tier1: 1 hr, Core Tier2: 1 hr) 169
4.1.6 CAN/Applications of the Definite Integral (Core Tier1: 1 hr, Core Tier2: 1 hr) 170
4.1.7 CAN/Vectors and the Geometry of Space (Core Tier1: 1 hr, Core Tier2: 1 hr) 170
4.1.8 CAN/Quadric Surfaces (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 171
4.1.9 CAN/Parametric Equations and Polar Coordinates (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 171
4.1.10 CAN/Multivariable Differential Calculus (Core Tier1: 1 hr, Core Tier2: 1 hr) . 171
4.1.11 CAN/Multiple Integrals (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 172
4.1.12 CAN/Line Integrals and Vector Fields (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 172
4.1.13 CAN/Surface Integrals and Fundamental Theorems (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 172
4.1.14 CAN/Vector Functions of a Real Variable (Core Tier1: 1 hr, Core Tier2: 1 hr) 173
4.1.15 CAN/Multivariable Transformations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 173
4.1.16 CAN/Real Analysis and Lebesgue Integration (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 173

4.1.17 CAN/Complex Analysis and Conformal Mapping (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 174
4.1.18 CAN/Functional Analysis and Banach/Hilbert Spaces . . . 174
4.1.19 CAN/Harmonic Analysis and Fourier Theory . . . 174
4.1.20 CAN/Calculus of Variations . . . 175
4.1.21 CAN/Measure Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 175
4.2 Algebra and Number Theory (ANT) . . . 175
4.2.1 ANT/Matrices and Systems of Linear Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 176
4.2.2 ANT/Vector Spaces and Linear Transformations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 176
4.2.3 ANT/Eigenvalues, Inner Product Spaces, and Quadratic Forms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 177
4.2.4 ANT/Group Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 177
4.2.5 ANT/Rings, Fields, and Galois Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 177
4.2.6 ANT/Module Theory and Commutative Algebra . . . 178
4.2.7 ANT/Analytic Number Theory . . . 178
4.2.8 ANT/Algebraic Number Theory . . . 178
4.2.9 ANT/Representation Theory . . . 179
4.2.10 ANT/Cryptography and Coding Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 179
4.3 Geometry and Topology (GTO) . . . 179
4.3.1 GTO/Euclidean and Projective Geometry (Core Tier1: 1 hr, Core Tier2: 1 hr) 180
4.3.2 GTO/General Topology (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 180
4.3.3 GTO/Differential Geometry of Curves and Surfaces (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 181
4.3.4 GTO/Smooth Manifolds and Differential Forms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 181
4.3.5 GTO/Riemannian Geometry . . . 181
4.3.6 GTO/Algebraic Topology . . . 182
4.3.7 GTO/Symplectic Geometry . . . 182
4.3.8 GTO/Geometric Group Theory . . . 182
4.4 Discrete Mathematics and Combinatorics (DMC) . . . 183
4.4.1 DMC/Enumerative Combinatorics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 183
4.4.2 DMC/Graph Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 183
4.4.3 DMC/Extremal and Probabilistic Combinatorics . . . 184
4.4.4 DMC/Discrete Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 184
4.4.5 DMC/Design Theory and Combinatorial Codes . . . 184
4.4.6 DMC/Mathematical Logic and Set Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) 185
4.5 Probability and Statistics (PST) . . . 185
4.5.1 PST/Probability Axioms and Distributions (Core Tier1: 1 hr, Core Tier2: 1 hr) 185
4.5.2 PST/Limit Theorems (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 186
4.5.3 PST/Stochastic Processes (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 186
4.5.4 PST/Estimation Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 186
4.5.5 PST/Hypothesis Testing (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 187
4.5.6 PST/Bayesian Inference (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 187
4.5.7 PST/High-Dimensional Statistics . . . 187
4.5.8 PST/Time Series Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 188
4.5.9 PST/Nonparametric Methods (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 188
4.6 Numerical and Scientific Analysis (NSA) . . . 188
4.6.1 NSA/Error Analysis and Floating-Point Arithmetic (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 189
4.6.2 NSA/Root-Finding for Nonlinear Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) 189
4.6.3 NSA/Numerical Linear Algebra (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 190
4.6.4 NSA/Approximation Theory and Interpolation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 190
4.6.5 NSA/Numerical Integration and Quadrature (Core Tier1: 1 hr, Core Tier2: 1 hr) 190

4.6.6 NSA/Numerical Methods for Differential Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 191
4.6.7 NSA/Finite Element Method (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 191
4.6.8 NSA/Optimization Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 191
4.6.9 NSA/Parallel and High-Performance Computing (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 192
4.7 Mathematical Modeling and Simulation (MMS) . . . 192
4.7.1 MMS/First-Order Ordinary Differential Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 193
4.7.2 MMS/Higher-Order Differential Equations and Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 193
4.7.3 MMS/Partial Differential Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 193
4.7.4 MMS/Dynamical Systems and Chaos (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 194
4.7.5 MMS/Asymptotic Analysis and Perturbation Methods (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 194
4.7.6 MMS/Mathematical Fluid Dynamics . . . 194
4.7.7 MMS/Mathematical Control Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 195
4.7.8 MMS/Stochastic Differential Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) . 195
4.7.9 MMS/Agent-Based and Complex Systems Modeling . . . 195
4.8 Mathematical Foundations of Data Science (FDS) . . . 196
4.8.1 FDS/Statistical Learning Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 196
4.8.2 FDS/Optimization for Machine Learning (Core Tier1: 1 hr, Core Tier2: 1 hr) . 196
4.8.3 FDS/Topological Data Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 197
4.8.4 FDS/Dimensionality Reduction (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 197
4.8.5 FDS/Kernel Methods and Reproducing Kernel Hilbert Spaces (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 198
4.8.6 FDS/Information Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 198
4.8.7 FDS/Mathematical Foundations of Deep Learning (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 198
4.9 Computational Geometry and Topology (CGT) . . . 199
4.9.1 CGT/Voronoi Diagrams and Delaunay Triangulations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 199
4.9.2 CGT/Convex Hulls and Polytopes (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 199
4.9.3 CGT/Digital Topology and Image Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr) 200
4.9.4 CGT/Persistent Homology and Applied Topology (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 200
4.9.5 CGT/Mesh Generation and Geometric Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 201
4.10 Quantum Information and Computing (QIC) . . . 201
4.10.1 QIC/Quantum Mechanics Foundations for Computing (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 201
4.10.2 QIC/Quantum Circuits and Gates (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 202
4.10.3 QIC/Quantum Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 202
4.10.4 QIC/Entanglement and Bell Inequalities (Core Tier1: 1 hr, Core Tier2: 1 hr) . 202
4.10.5 QIC/Quantum Error Correction (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 203
4.10.6 QIC/Tensor Networks and Quantum Simulation . . . 203
4.11 Mathematical Biology and Life Sciences (MBL) . . . 204
4.11.1 MBL/Population Dynamics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 204
4.11.2 MBL/Mathematical Epidemiology (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 204
4.11.3 MBL/Systems Biology and Network Dynamics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 205
4.11.4 MBL/Bioinformatics Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 205
4.11.5 MBL/Mathematical Neuroscience (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 205
4.12 Financial and Actuarial Mathematics (FAM) . . . 206
4.12.1 FAM/Interest Theory and Time Value of Money (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 206

4.12.2 FAM/Arbitrage-Free Pricing and Market Models (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 207
4.12.3 FAM/Black-Scholes and Option Pricing (Core Tier1: 1 hr, Core Tier2: 1 hr) . 207
4.12.4 FAM/Risk Measures and Portfolio Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) 207
4.12.5 FAM/Actuarial Models and Life Tables (Core Tier1: 1 hr, Core Tier2: 1 hr) . 208
4.12.6 FAM/Computational Finance and Monte Carlo Methods (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 208

5 Body of knowledge of Physics 209
5.1 Classical Mechanics (CME) . . . 210
5.1.1 CME/Kinematics and Dynamics of Particles (Core Tier1: 1 hr, Core Tier2: 1 hr) 211
5.1.2 CME/Work, Energy, and Conservative Forces (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 211
5.1.3 CME/Systems of Particles and Linear Momentum (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 212
5.1.4 CME/Rotational Motion and Angular Momentum (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 213
5.1.5 CME/Rigid Body Dynamics in 3D (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 213
5.1.6 CME/Non-Inertial Reference Frames and Fictitious Forces (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 214
5.1.7 CME/Lagrangian and Hamiltonian Mechanics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 214
5.1.8 CME/Central Force Motion and Orbital Mechanics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 215
5.1.9 CME/Theory of Small Oscillations and Coupled Oscillators (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 215
5.1.10 CME/Continuum Mechanics Foundations . . . 216
5.1.11 CME/Elasticity (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 216
5.1.12 CME/Fluids: Hydrostatics, Hydrodynamics, and Viscosity (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 217
5.1.13 CME/Vibrations and Waves (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 217
5.1.14 CME/Transport Phenomena (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 218
5.2 Electromagnetism (ELM) . . . 218
5.2.1 ELM/Electrostatics: Electric Field and Gauss's Law (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 219
5.2.2 ELM/Electric Potential and Poisson/Laplace Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 219
5.2.3 ELM/Electrostatic Energy and Dielectrics (Core Tier1: 1 hr, Core Tier2: 1 hr) 220
5.2.4 ELM/Magnetostatics: Biot-Savart and Ampere's Laws (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 220
5.2.5 ELM/Magnetic Properties of Matter (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 221
5.2.6 ELM/Electrodynamics: Faraday's Law and Induction (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 221
5.2.7 ELM/Maxwell's Equations and Gauge Transformations (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 222
5.2.8 ELM/Electromagnetic Waves and Propagation in Media (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 222
5.2.9 ELM/Radiation Theory and Dipole Radiation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 223
5.2.10 ELM/Special Relativity in Electromagnetism (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 223
5.2.11 ELM/Electric Current, Circuits, and Kirchhoff's Laws (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 223
5.3 Thermodynamics and Statistical Mechanics (TSM) . . . 224
5.3.1 TSM/Laws of Thermodynamics and State Functions (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 224

5.3.2 TSM/Kinetic Theory of Gases (Core Tier1: 1 hr, Core Tier2: 1 hr) 224
5.3.3 TSM/Heat Engines, Entropy, and the Second Law (Core Tier1: 1 hr, Core Tier2: 1 hr) 225
5.3.4 TSM/Thermodynamic Potentials and Maxwell Relations (Core Tier1: 1 hr, Core Tier2: 1 hr) 225
5.3.5 TSM/Phase Transitions and Critical Phenomena (Core Tier1: 1 hr, Core Tier2: 1 hr) 226
5.3.6 TSM/Microcanonical, Canonical, and Grand Canonical Ensembles (Core Tier1: 1 hr, Core Tier2: 1 hr) 226
5.3.7 TSM/Quantum Statistics: Fermi-Dirac and Bose-Einstein (Core Tier1: 1 hr, Core Tier2: 1 hr) 226
5.3.8 TSM/Blackbody Radiation and Phonons (Core Tier1: 1 hr, Core Tier2: 1 hr) . 227
5.3.9 TSM/Non-Equilibrium Statistical Mechanics (Core Tier1: 1 hr, Core Tier2: 1 hr) 227
5.3.10 TSM/Applications to Biological and Complex Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) 227
5.4 Quantum Mechanics (QME) 228
5.4.1 QME/Origins of Quantum Theory and Wave-Particle Duality (Core Tier1: 1 hr, Core Tier2: 1 hr) 228
5.4.2 QME/The Schrödinger Equation and Wave Functions (Core Tier1: 1 hr, Core Tier2: 1 hr) 228
5.4.3 QME/Formalism: Hilbert Space and Dirac Notation (Core Tier1: 1 hr, Core Tier2: 1 hr) 229
5.4.4 QME/One-Dimensional Potentials (Wells, Barriers, Tunneling) (Core Tier1: 1 hr, Core Tier2: 1 hr) 229
5.4.5 QME/The Quantum Harmonic Oscillator (Core Tier1: 1 hr, Core Tier2: 1 hr) 229
5.4.6 QME/Angular Momentum and Spin Physics (Core Tier1: 1 hr, Core Tier2: 1 hr) 230
5.4.7 QME/The Hydrogen Atom and Central Potentials (Core Tier1: 1 hr, Core Tier2: 1 hr) 230
5.4.8 QME/Time-Independent and Time-Dependent Perturbation Theory (Core Tier1: 1 hr, Core Tier2: 1 hr) 230
5.4.9 QME/Identical Particles and Exchange Symmetry (Core Tier1: 1 hr, Core Tier2: 1 hr) 231
5.4.10 QME/Scattering Theory and Cross Sections (Core Tier1: 1 hr, Core Tier2: 1 hr) 231
5.5 Relativity and Cosmology (RCO) 232
5.5.1 RCO/Special Relativity: Lorentz Transformations and 4-Vectors (Core Tier1: 1 hr, Core Tier2: 1 hr) 232
5.5.2 RCO/Relativistic Dynamics and Energy-Momentum (Core Tier1: 1 hr, Core Tier2: 1 hr) 232
5.5.3 RCO/General Relativity Foundations: Equivalence Principle (Core Tier1: 1 hr, Core Tier2: 1 hr) 233
5.5.4 RCO/Spacetime Curvature and Einstein Field Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) 233
5.5.5 RCO/Schwarzschild Geometry and Black Holes (Core Tier1: 1 hr, Core Tier2: 1 hr) 233
5.5.6 RCO/Gravitational Waves (Core Tier1: 1 hr, Core Tier2: 1 hr) 233
5.5.7 RCO/Big Bang Theory and Friedmann Equations (Core Tier1: 1 hr, Core Tier2: 1 hr) 234
5.5.8 RCO/Cosmic Microwave Background (CMB) (Core Tier1: 1 hr, Core Tier2: 1 hr) 234
5.5.9 RCO/Dark Matter and Dark Energy (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 234
5.5.10 RCO/Early Universe Inflation and Nucleosynthesis (Core Tier1: 1 hr, Core Tier2: 1 hr) 235
5.6 Condensed Matter Physics (SSP) 235
5.6.1 SSP/Crystal Structures and Reciprocal Lattices (Core Tier1: 1 hr, Core Tier2: 1 hr) 235

5.6.2 SSP/X-Ray Diffraction and Lattice Dynamics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 236
5.6.3 SSP/Free Electron Model and Band Theory of Solids (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 236
5.6.4 SSP/Semiconductors, Insulators, and Metals (Core Tier1: 1 hr, Core Tier2: 1 hr) 236
5.6.5 SSP/Magnetic Properties: Para-, Dia-, and Ferromagnetism (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 237
5.6.6 SSP/Superconductivity: BCS Theory and High-Tc (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 237
5.6.7 SSP/Soft Matter Physics and Polymers (Core Tier1: 1 hr, Core Tier2: 1 hr) . 237
5.6.8 SSP/Nanostructures and Low-Dimensional Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 238
5.6.9 SSP/Spintronics and Topological Insulators (Core Tier1: 1 hr, Core Tier2: 1 hr) 238
5.6.10 SSP/Strongly Correlated Electron Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) 238
5.7 Nuclear and Particle Physics (NPP) . . . 239
5.7.1 NPP/Nuclear Structure, Binding Energy, and Shell Model (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 239
5.7.2 NPP/Radioactivity and Nuclear Decays (Alpha, Beta, Gamma) (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 239
5.7.3 NPP/Nuclear Reactions, Fission, and Fusion (Core Tier1: 1 hr, Core Tier2: 1 hr) 240
5.7.4 NPP/Particle Accelerators and Detectors (Core Tier1: 1 hr, Core Tier2: 1 hr) 240
5.7.5 NPP/The Standard Model of Particle Physics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 240
5.7.6 NPP/Quarks, Leptons, Gauge Bosons (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 241
5.7.7 NPP/Symmetries and Conservation Laws (CPT) (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 241
5.7.8 NPP/Fundamental Interactions and Feynman Diagrams (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 241
5.7.9 NPP/Higgs Mechanism and Beyond the Standard Model (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 242
5.7.10 NPP/Neutrino Physics and Oscillation (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 242
5.8 Quantum Information and Computing (QIC) . . . 242
5.8.1 QIC/Qubits, Superposition, and Entanglement (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 243
5.8.2 QIC/Quantum Logic Gates and Circuits (Core Tier1: 1 hr, Core Tier2: 1 hr) . 243
5.8.3 QIC/Quantum Algorithms (Shor, Grover) (Core Tier1: 1 hr, Core Tier2: 1 hr) 243
5.8.4 QIC/Quantum Teleportation and Cryptography (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 244
5.8.5 QIC/Decoherence and Quantum Error Correction (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 244
5.8.6 QIC/Physical Realizations of Quantum Computers (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 244
5.8.7 QIC/Quantum Sensing and Metrology (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 245
5.8.8 QIC/Quantum Networks and Communication (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 245
5.9 Optics and Photonics (OPH) . . . 245
5.9.1 OPH/Geometrical Optics and Optical Instruments (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 246
5.9.2 OPH/Wave Optics: Interference and Diffraction (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 246
5.9.3 OPH/Polarization and Crystal Optics (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 247
5.9.4 OPH/Lasers: Principles and Applications (Core Tier1: 1 hr, Core Tier2: 1 hr) 247
5.9.5 OPH/Nonlinear Optics and Frequency Conversion (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 247
5.9.6 OPH/Fiber Optics and Photonic Crystals (Core Tier1: 1 hr, Core Tier2: 1 hr) 248
5.9.7 OPH/Ultrafast Optics and Spectroscopy (Core Tier1: 1 hr, Core Tier2: 1 hr) . 248

5.9.8 OPH/Nano-optics and Plasmonics (Core Tier1: 1 hr, Core Tier2: 1 hr) 248
5.10 Computational and Data-Driven Physics (CDP) 249
5.10.1 CDP/Numerical Methods for Differential Equations in Physics (Core Tier1: 1 hr, Core Tier2: 1 hr) 249
5.10.2 CDP/Monte Carlo Methods and Simulations (Core Tier1: 1 hr, Core Tier2: 1 hr) 249
5.10.3 CDP/High-Performance Computing (HPC) for Large-Scale Simulations (Core Tier1: 1 hr, Core Tier2: 1 hr) 250
5.10.4 CDP/Machine Learning for Physical Discovery (Core Tier1: 1 hr, Core Tier2: 1 hr) 250
5.10.5 CDP/Data Analysis and Uncertainty Quantification (Core Tier1: 1 hr, Core Tier2: 1 hr) 250
5.10.6 CDP/Modeling Complex Adaptive Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) 251
5.10.7 CDP/Neural Networks for Quantum Systems (Core Tier1: 1 hr, Core Tier2: 1 hr) 251
5.10.8 CDP/Physics-Informed Neural Networks (PINNs) (Core Tier1: 1 hr, Core Tier2: 1 hr) 251
5.11 Biophysics and Medical Physics (BMP) 252
5.11.1 BMP/Physics of Biological Macromolecules (Core Tier1: 1 hr, Core Tier2: 1 hr) 252
5.11.2 BMP/Biomechanics and Cellular Forces (Core Tier1: 1 hr, Core Tier2: 1 hr) . 252
5.11.3 BMP/Neural Physics and Information Processing in the Brain (Core Tier1: 1 hr, Core Tier2: 1 hr) 253
5.11.4 BMP/Imaging Physics (MRI, CT, PET, Ultrasound) (Core Tier1: 1 hr, Core Tier2: 1 hr) 253
5.11.5 BMP/Radiation Therapy and Dosimetry (Core Tier1: 1 hr, Core Tier2: 1 hr) . 253
5.11.6 BMP/Statistical Mechanics of Biological Networks (Core Tier1: 1 hr, Core Tier2: 1 hr) 254
5.11.7 BMP/Microfluidics and Lab-on-a-Chip (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 254
5.12 Astrophysics and Space Science (ASS) 254
5.12.1 ASS/Stellar Structure and Evolution (Core Tier1: 1 hr, Core Tier2: 1 hr) . . . 255
5.12.2 ASS/Planetary Science and Exoplanets (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 255
5.12.3 ASS/Galactic Dynamics and Structure (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 255
5.12.4 ASS/High-Energy Astrophysics (X-rays, Gamma-rays) (Core Tier1: 1 hr, Core Tier2: 1 hr) 256
5.12.5 ASS/Observational Techniques and Telescopes (Core Tier1: 1 hr, Core Tier2: 1 hr) 256
5.12.6 ASS/Solar Physics and Space Weather (Core Tier1: 1 hr, Core Tier2: 1 hr) . . 256
5.12.7 ASS/Multi-messenger Astronomy (Gravitational Waves + Neutrinos) (Core Tier1: 1 hr, Core Tier2: 1 hr) 257

6 Curriculum **259**
6.1 Course coding 259
6.2 Estructura Curricular 260
6.3 Topics distributed by course 265
6.3.1 Computational Mathematics topics/course 265
6.3.2 Computer Science topics/course 271
6.3.3 Mathematics topics/course 279
6.3.4 Physics topics/course 281
6.4 Expected outcomes distributed by course 283
6.5 Courses by Outcome 287
6.5.1 Outcome: ABET-1) Identify, formulate, and solve applied scientific problems. . 287
6.5.2 Outcome: ABET-2) Formulate or design systems, processes, or programs. . . . 287
6.5.3 Outcome: ABET-3) Experimentation, data analysis, and scientific judgment. . 288
6.5.4 Outcome: ABET-4) Effective communication with diverse audiences. 289
6.5.5 Outcome: ABET-5) Ethical responsibilities and impact of scientific solutions. . 289
6.5.6 Outcome: ABET-6) Teamwork, planning, and risk management. 289
6.6 Distribution of courses in the program 289

6.7 Compatibility of the program with international standards . . . 292

7 Detailed content by course **295**
First Semester . . . 295
CS111 Introduction to Programming (Mandatory) . . . 295
MA100 Basic Mathematics (Mandatory) . . . 300
FG101 Language (Mandatory) . . . 303
FG105 Gender and Society (Mandatory) . . . 306
FG109 Environment and Sustainable Development (Mandatory) . . . 309
FG112 Methods of managing university personnel (Mandatory) . . . 312
FG140 Fundamentals of Scientific Research (Mandatory) . . . 315
Second Semester . . . 318
CS100 Introduction to Computer Science (Mandatory) . . . 318
CS112 Objects-oriented programming I (Mandatory) . . . 328
AI161 Applied AI (Mandatory) . . . 333
MA111 Calculus I (Mandatory) . . . 338
FI101 General Physics (Mandatory) . . . 340
Third Semester . . . 343
CS113 Objects-oriented programming II (Mandatory) . . . 343
CS1D1 Discrete Structures I (Mandatory) . . . 349
CS2B1 Platform Based Development (Mandatory) . . . 352
MA112 Calculus II (Mandatory) . . . 356
MA121 Linear Algebra I (Mandatory) . . . 358
ID101 Technical and professional English I (Mandatory) . . . 361
Fourth Semester . . . 365
CS1D2 Discrete Structures II (Mandatory) . . . 365
CS210 Algorithms and Data Structures (Mandatory) . . . 367
CS271 Data Management (Mandatory) . . . 372
MA122 Linear Algebra II (Mandatory) . . . 377
MA210 Calculus III (Mandatory) . . . 380
ID102 Technical and professional English II (Mandatory) . . . 383
Fifth Semester . . . 387
CS251 Computer graphics (Mandatory) . . . 387
MA211 Calculus IV (Mandatory) . . . 390
MA221 Numerical Analysis I (Mandatory) . . . 393
MA222 Real Analysis I (Mandatory) . . . 396
MA251 Introduction to Statistics (Mandatory) . . . 399
MA281 Numbers theory (Mandatory) . . . 401
ID201 Technical and professional English III (Mandatory) . . . 405
Sixth Semester . . . 409
MA212 Ordinary Differential Equations (Mandatory) . . . 409
MA325 Numerical Analysis II (Mandatory) . . . 412
MA351 Statistics and Probabilities (Mandatory) . . . 415
MA371 Functional analysis I (Mandatory) . . . 420
ID202 Technical and professional English IV (Mandatory) . . . 423
Seventh Semester . . . 427
CS261 Artificial Intelligence (Mandatory) . . . 427
CS3P1 Parallel and Distributed Computing (Mandatory) . . . 434
MA213 Partial Differential Equations (Mandatory) . . . 443
MA326 Numerical Analysis III (Mandatory) . . . 446
MA361 Mathematical Modeling (Mandatory) . . . 449
ID203 Technical and professional English V (Mandatory) . . . 452
Eighth Semester . . . 455
AI263 Introduction to Machine Learning (Mandatory) . . . 455
MA401 Capstone Project I (Mandatory) . . . 458
CS2S1 Operating systems (Elective) . . . 461

CS3I1 Computer Security (Elective) . . . 466
QC251 Quantum Mechanics (Elective) . . . 472
CS253 Exploratory data analysis (Elective) . . . 476
CS370 Big Data and Analytics (Elective) . . . 480
STA252 Inferential Statistics (Elective) . . . 484
MA327 Numerical Analysis IV (Elective) . . . 487
MA352 Stochastic processes (Elective) . . . 490
MA372 Functional analysis II (Elective) . . . 493
Ninth Semester . . . 496
AI264 Deep Learning (Mandatory) . . . 496
MA402 Capstone Project II (Mandatory) . . . 499
CS231 Networking and Communication (Elective) . . . 503
MA381 Algebraic Structures (Elective) . . . 508
AI262 Natural Language Processing (Elective) . . . 512
CS273 Time series analysis (Elective) . . . 515
MA341 Optimization I (Elective) . . . 519
MA355 Statistical Learning Theory (Elective) . . . 522
Tenth Semester . . . 525
CS400 Pre-professional internships (Mandatory) . . . 525
AI268 Computational Vision (Mandatory) . . . 527
AI367 Topics in Artificial Intelligence (Mandatory) . . . 533
MA403 Capstone Project III (Mandatory) . . . 539
QC381 Introduction to Quantum Computing (Elective) . . . 543
CS254 Advanced Data Visualization (Elective) . . . 547

8 Professors & Courses **551**
Taught courses by professor . . . 551
Professor assigned per course . . . 551

9 Equivalences with Other Curriculum Plans **555**
Equivalence from 2026 to 2018 . . . 555
Equivalence from 2018 to 2026 . . . 559

10 Laboratories **563**

List of Figures

6.1 Course coding scheme. . . . 259
6.2 Credits by area by semester . . . 291
6.3 Distribution of courses by area considering credits. . . . 291
6.4 Distribution of credits by course levels. . . . 292

7.1 Connection Map. CS111 Introduction to Programming . . . 295
7.2 Connection Map. MA100 Basic Mathematics . . . 300
7.3 Connection Map. FG101 Language . . . 303
7.4 Connection Map. FG105 Gender and Society . . . 306
7.5 Connection Map. FG109 Environment and Sustainable Development . . . 309
7.6 Connection Map. FG112 Methods of managing university personnel . . . 312
7.7 Connection Map. FG140 Fundamentals of Scientific Research . . . 315
7.8 Connection Map. CS100 Introduction to Computer Science . . . 318
7.9 Connection Map. CS112 Objects-oriented programming I . . . 328
7.10 Connection Map. AI161 Applied AI . . . 333
7.11 Connection Map. MA111 Calculus I . . . 338
7.12 Connection Map. FI101 General Physics . . . 340
7.13 Connection Map. CS113 Objects-oriented programming II . . . 343
7.14 Connection Map. CS1D1 Discrete Structures I . . . 349
7.15 Connection Map. CS2B1 Platform Based Development . . . 352
7.16 Connection Map. MA112 Calculus II . . . 356
7.17 Connection Map. MA121 Linear Algebra I . . . 358
7.18 Connection Map. ID101 Technical and professional English I . . . 361
7.19 Connection Map. CS1D2 Discrete Structures II . . . 365
7.20 Connection Map. CS210 Algorithms and Data Structures . . . 367
7.21 Connection Map. CS271 Data Management . . . 372
7.22 Connection Map. MA122 Linear Algebra II . . . 377
7.23 Connection Map. MA210 Calculus III . . . 380
7.24 Connection Map. ID102 Technical and professional English II . . . 383
7.25 Connection Map. CS251 Computer graphics . . . 387
7.26 Connection Map. MA211 Calculus IV . . . 390
7.27 Connection Map. MA221 Numerical Analysis I . . . 393
7.28 Connection Map. MA222 Real Analysis I . . . 396
7.29 Connection Map. MA251 Introduction to Statistics . . . 399
7.30 Connection Map. MA281 Numbers theory . . . 401
7.31 Connection Map. ID201 Technical and professional English III . . . 405
7.32 Connection Map. MA212 Ordinary Differential Equations . . . 409
7.33 Connection Map. MA325 Numerical Analysis II . . . 412
7.34 Connection Map. MA351 Statistics and Probabilities . . . 415
7.35 Connection Map. MA371 Functional analysis I . . . 420
7.36 Connection Map. ID202 Technical and professional English IV . . . 423
7.37 Connection Map. CS261 Artificial Intelligence . . . 427
7.38 Connection Map. CS3P1 Parallel and Distributed Computing . . . 434
7.39 Connection Map. MA213 Partial Differential Equations . . . 443

7.40 Connection Map. MA326 Numerical Analysis III 446
7.41 Connection Map. MA361 Mathematical Modeling 449
7.42 Connection Map. ID203 Technical and professional English V 452
7.43 Connection Map. AI263 Introduction to Machine Learning 455
7.44 Connection Map. MA401 Capstone Project I 458
7.45 Connection Map. CS2S1 Operating systems 461
7.46 Connection Map. CS3I1 Computer Security 466
7.47 Connection Map. QC251 Quantum Mechanics 472
7.48 Connection Map. CS253 Exploratory data analysis 476
7.49 Connection Map. CS370 Big Data and Analytics 480
7.50 Connection Map. STA252 Inferential Statistics 484
7.51 Connection Map. MA327 Numerical Analysis IV 487
7.52 Connection Map. MA352 Stochastic processes 490
7.53 Connection Map. MA372 Functional analysis II 493
7.54 Connection Map. AI264 Deep Learning 496
7.55 Connection Map. MA402 Capstone Project II 499
7.56 Connection Map. CS231 Networking and Communication 503
7.57 Connection Map. MA381 Algebraic Structures 508
7.58 Connection Map. AI262 Natural Language Processing 512
7.59 Connection Map. CS273 Time series analysis 515
7.60 Connection Map. MA341 Optimization I 519
7.61 Connection Map. MA355 Statistical Learning Theory 522
7.62 Connection Map. CS400 Pre-professional internships 525
7.63 Connection Map. AI268 Computational Vision 527
7.64 Connection Map. AI367 Topics in Artificial Intelligence 533
7.65 Connection Map. MA403 Capstone Project III 539
7.66 Connection Map. QC381 Introduction to Quantum Computing 543
7.67 Connection Map. CS254 Advanced Data Visualization 547

List of Tables

2.1 List of KUs in the Numerical and Scientific Analysis area. 10
2.2 List of KUs in the Mathematical Modeling and Simulation area. 12
2.3 List of KUs in the Probability and Statistical Computing area. 14
2.4 List of KUs in the Data Structures and Algorithms area. 16
2.5 List of KUs in the High-Performance Computing area. 17
2.6 List of KUs in the Optimization and Machine Learning area. 19
2.7 List of KUs in the Data Science and Machine Learning Mathematics area. 21
2.8 List of KUs in the Artificial Intelligence Mathematics area. 23
2.9 List of KUs in the Symbolic and Discrete Computation area. 24
2.10 List of KUs in the Signal, Image, and Visualization area. 26
2.11 List of KUs in the Quantum Computing and Algorithms area. 27
2.12 List of KUs in the Cybersecurity and Privacy area. 29
2.13 List of KUs in the Computational Mathematics for Specific Domains area. 30
2.14 List of KUs in the Data Governance and Management area. 32

3.1 List of KUs in the Artificial Intelligence area. 38
3.2 List of KUs in the Algorithmic Foundations area. 50
3.3 List of KUs in the Architecture and Organization area. 56
3.4 List of KUs in the Data Management area. 63
3.5 List of KUs in the Foundations of Programming Languages area. 69
3.6 List of KUs in the Graphics and Interactive Techniques area. 82
3.7 List of KUs in the Human-Computer Interaction area. 87
3.8 List of KUs in the Mathematical and Statistical Foundations area. 89
3.9 List of KUs in the Networking and Communication area. 93
3.10 List of KUs in the Operating Systems area. 96
3.11 List of KUs in the Parallel and Distributed Computing area. 104
3.12 List of KUs in the Software Development Fundamentals area. 113
3.13 List of KUs in the Software Engineering area. 116
3.14 List of KUs in the Security area. 127
3.15 List of KUs in the Society, Ethics, and the Profession area. 136
3.16 List of KUs in the System Fundamentals area. 149
3.17 List of KUs in the Specialized Platform Development area. 156

4.1 List of KUs in the Calculus and Analysis area. 168
4.2 List of KUs in the Algebra and Number Theory area. 176
4.3 List of KUs in the Geometry and Topology area. 180
4.4 List of KUs in the Discrete Mathematics and Combinatorics area. 183
4.5 List of KUs in the Probability and Statistics area. 185
4.6 List of KUs in the Numerical and Scientific Analysis area. 189
4.7 List of KUs in the Mathematical Modeling and Simulation area. 192
4.8 List of KUs in the Mathematical Foundations of Data Science area. 196
4.9 List of KUs in the Computational Geometry and Topology area. 199
4.10 List of KUs in the Quantum Information and Computing area. 201
4.11 List of KUs in the Mathematical Biology and Life Sciences area. 204

4.12 List of KUs in the Financial and Actuarial Mathematics area. 206

5.1 List of KUs in the Classical Mechanics area. 210
5.2 List of KUs in the Electromagnetism area. 219
5.3 List of KUs in the Thermodynamics and Statistical Mechanics area. 224
5.4 List of KUs in the Quantum Mechanics area. 228
5.5 List of KUs in the Relativity and Cosmology area. 232
5.6 List of KUs in the Condensed Matter Physics area. 235
5.7 List of KUs in the Nuclear and Particle Physics area. 239
5.8 List of KUs in the Quantum Information and Computing area. 243
5.9 List of KUs in the Optics and Photonics area. 246
5.10 List of KUs in the Computational and Data-Driven Physics area. 249
5.11 List of KUs in the Biophysics and Medical Physics area. 252
5.12 List of KUs in the Astrophysics and Space Science area. 255

6.1 Computational Mathematics topics by course from 1st to 2nd Semester 265
6.2 Computational Mathematics topics by course from 3rd to 4th Semester 266
6.3 Computational Mathematics topics by course from 5th to 6th Semester 267
6.4 Computational Mathematics topics by course 7th Semester 267
6.5 Computational Mathematics topics by course 8th Semester 268
6.6 Computational Mathematics topics by course from 9th to 10th Semester 269
6.7 Computer Science topics by course from 1st to 2nd Semester (Part 1 of 2) 271
6.8 Computer Science topics by course from 1st to 2nd Semester (Part 2 of 2) 272
6.9 Computer Science topics by course from 3rd to 4th Semester 273
6.10 Computer Science topics by course from 5th to 6th Semester 274
6.11 Computer Science topics by course 7th Semester (Part 1 of 2) 275
6.12 Computer Science topics by course 7th Semester (Part 2 of 2) 276
6.13 Computer Science topics by course 8th Semester 277
6.14 Computer Science topics by course from 9th to 10th Semester (Part 1 of 2) 278
6.15 Computer Science topics by course from 9th to 10th Semester (Part 2 of 2) 279
6.16 Mathematics topics by course from 1st to 2nd Semester 279
6.17 Mathematics topics by course from 3rd to 4th Semester 279
6.18 Mathematics topics by course from 5th to 6th Semester 280
6.19 Mathematics topics by course 7th Semester 280
6.20 Mathematics topics by course 8th Semester 280
6.21 Mathematics topics by course from 9th to 10th Semester 281
6.22 Physics topics by course from 1st to 2nd Semester 281
6.23 Physics topics by course from 3rd to 4th Semester 281
6.24 Physics topics by course from 5th to 6th Semester 282
6.25 Physics topics by course 7th Semester 282
6.26 Physics topics by course 8th Semester 282
6.27 Physics topics by course from 9th to 10th Semester 283
6.28 Outcomes by course 1st to 2nd Semester 283
6.29 Outcomes by course 3rd to 4th Semester 284
6.30 Outcomes by course 5th to 6th Semester 284
6.31 Outcomes by course 7th Semester 285
6.32 Outcomes by course 8th Semester 285
6.33 Outcomes by course 9th to 10th Semester 286
6.34 Distribution of credits by area by semester 290

Agradecimientos

Además de los autores directos de este documento, también deseamos dejar manifiesto de nuestro agradecimiento a otros colegas de diversas universidades del país y del mundo que gentilmente han aportado parte de su tiempo a darnos sus sugerencias. Entre ellos debemos mencionar a:

- Todo de equipo de ACM/ IEEE-CS Computing Curricula (CC2020, CS2023 y CS2013).
- Los docentes que participaron en las Jornadas Curriculares (JC).
- A todo el personal administrativo por la organización las JC.
- A los estudiantes que participaron en las JC.
- A los colegas de otras universidades que nos han dado sus sugerencias.
- Sociedad Peruana de Computación por apoyarnos y facilitarnos su documento de referencia.

Todo este equipo de trabajo asumió como premisa que el centro de nuestro esfuerzo, es la formación académica y humana de los estudiantes para formar agentes de cambio positivo y disruptivo en la sociedad.

A todos ellos deseamos agradecerles por su aporte que ha permitido generar este documento, único en su género en nuestro país, que servirá para sentar las bases de una carrera más sólida en esta fantástica área que nos ha tocado estudiar y de la cual nos sentimos orgullosos de formar parte.

Acronyms

.

Chapter 1

Introducción

Computing has undergone impressive development in recent decades, becoming the engine of scientific, technological, industrial, social, economic, and cultural development, significantly transforming our daily actions.

The emergence of the computer has marked a new era in human history that was impossible to imagine several decades ago. The vast number of applications that have been developed in recent years are transforming the development of all fields of knowledge, commerce in the globalized world we live in, the way we communicate, teaching-learning processes, and even the way we entertain ourselves.

To give an idea of the relevance and importance that this discipline has reached in our country, it is enough to mention that currently approximately more than 100 Computing degree programs are offered nationally. This without considering non-university higher technical programs that are available.

All these existing programs have computing as the core of their study but do so under more than 50 different names such as: Systems Engineering, Computer Engineering, Computer and Systems Engineering, among others. Despite all of them targeting the same job market, it is at least surprising that it is not possible to find even two that share the same curriculum, even in the first year.

Commonly, during the 1970s, Computing developed within Science Faculties in most American, British, and other countries' universities. During the 1980s, computing groups in universities struggled to achieve academic legitimacy within their local context. Frequently, they transformed into Mathematics and Computing departments, until finally splitting into two separate departments of Mathematics and Computing in the 1990s. It is in this decade that a growing number of institutions recognized the pervasive influence of Computing, creating independent units as departments, schools, or institutes dedicated to this area of study, a change that has demonstrated both insight and foresight.

In Perú, an increasing number of higher education institutions have tried to follow the development of foreign universities (though not always in a very serious or successful manner), recognizing Computing as a field of study in itself, as well as its strategic importance in education, and creating departments, schools, or institutes dedicated to its study. The Facultad de Ciencias Matemáticas cannot be an exception to this change, in which there is already a relative delay compared to many educational institutions within and outside of Peru.

1.1 Basic definitions

1.2 Occupational Field and Job Market

Nuestro egresado podrá prestar sus servicios profesionales en empresas e instituciones públicas y privadas que requieran sus capacidades en función del desarrollo que oferta, entre ellas:

- Empresas dedicadas a la producción de software con calidad internacional.
- Empresas, instituciones y organizaciones que requieran software de calidad para mejorar sus actividades y/o servicios ofertados.

Nuestro egresado puede desempeñarse en el mercado laboral sin ningún problema ya que, en general, la exigencia del mercado y campo ocupacional está mucho más orientada al uso de herramientas.

Sin embargo, es poco común que los propios profesionales de esta carrera se pregunten: ¿qué tipo de formación debería tener si yo quisiera crear esas herramientas además de saber usarlas?. Ambos perfiles (usuario y creador) son bastante diferentes pues no sería posible usar algo que todavía no fue creado. En otras palabras, los creadores de tecnología son los que dan origen a nuevos puestos de trabajo y abren la posibilidad de que otros puedan usar esa tecnología.

Debido a la formación basada en la investigación, nuestro profesional debe siempre ser un innovador donde trabaje. Esta misma formación permite que el egresado piense también en crear su propia empresa de desarrollo de software. Considerando que países en vias de desarrollo tienen un costo de vida mucho menor que Norte América ó Europa, una posibilidad que se muestra interesante es la exportación de software pero eso requiere que la calidad del producto sea al mismo nivel de lo ofrecido a nivel internacional.

Este perfil profesional también posibilita que los egresados se queden en el país; producir software en nuestro país y venderlo fuera es más rentable que salir al extranjero y comercializarlo allá.

El campo ocupacional de un egresado es amplio y está en continua expansión y cambio. Prácticamente toda empresa u organización hace uso de servicios de computación de algún tipo, y la buena formación básica de nuestros egresados hace que puedan responder a los requerimientos de las mismas exitosamente. Este egresado, no sólo podrá dar soluciones a los problemas existentes sino que deberá proponer innovaciones tecnológicas que impulsen la empresa hacia un progreso constante.

A medida que la informatización básica de las empresas del país avanza, la necesidad de personas capacitadas para resolver los problemas de mayor complejidad aumenta y el plan de estudios que hemos desarrollado tiene como objetivo satisfacer esta demanda considerandola a mediano y largo plazo. El campo para las tareas de investigación y desarrollo de problemas complejos en computación es también muy amplio y está creciendo día a día a nivel mundial.

Debido a la capacidad innovadora de nuestro egresado, existe una mayor la probabilidad de registrar patentes con un alto nivel inventivo lo cual es especialmente importante en nuestros países.

1.3 Importancia de la carrera en la sociedad

Uno de los caminos que se espera que siga un profesional del área de Computación Científica es que el se dedique a producir software o que se integre a las empresas productoras de software. En el ámbito de la computación, es común observar que los países cuentas con Asociaciones de Productores de Software cuyas políticas están orientadas a la exportación. Siendo así, no tendría sentido preparar a nuestros alumnos sólo para el mercado local o nacional. Nuestros egresados deben estar preparados para desenvolverse en el mundo globalizado que nos ha tocado vivir.

Nuestros futuros profesionales deben estar orientados a crear nuevas empresas de base tecnológica que puedan incrementar las exportaciones de software peruano. Este nuevo perfil está orientado a generar industria innovadora. Si nosotros somos capaces de exportar software competitivo también estaremos en condiciones de atraer nuevas inversiones. Las nuevas inversiones generarían más puestos de empleo bien remunerados y con un costo bajo en relación a otros tipos de industria. Bajo esta perspectiva, podemos afirmar que esta carrera será un motor que impulsará al desarrollo del país de forma decisiva con una inversión muy baja en relación a otros campos.

Es necesario recordar que la mayor innovación de productos comerciales de versiones recientes utiliza tecnología que se conocía en el mundo académico hace varios años, décadas o más. Un ejemplo claro son las bases de datos que soportan datos y consultas espaciales desde hace muy pocos años. Sin embargo, utilizan estructuras de datos que ya existían hace algunas décadas. Es lógico pensar que la gente del área académica no se dedique a estudiar en profundidad la última versión de un determinado software cuando esa tecnología ya la conocían hace mucho tiempo. Por esa misma razón es raro en el mundo observar que una universidad tenga convenios con una transnacional de software para dictar solamente esa tecnología pues, nuestra función es generar esa tecnología **y no sólo saber usarla**.

Tampoco debemos olvidar que los alumnos que ingresan hoy saldrán al mercado dentro de 5 años aproximadamente y, en un mundo que cambia tan rápido, no podemos ni debemos enseñarles tomando en cuenta solamente el mercado actual. Nuestros profesionales deben estar preparados para resolver los problemas que habrá dentro de 10 o 15 años y eso sólo es posible a través de la investigación.

1.4 Institutional Info

1.4.1 Mission

Based on the aforementioned, our mission is:

To be the comprehensive, high-level national reference in Mathematics, Statistics, Scientific Computing, and Operations Research, at both the Graduate and Undergraduate levels, contributing to the preservation of Peru's cultural diversity and its scientific and technological progress. Its members are distinguished by the application of the scientific method for the development of knowledge and the solution of the country's problems, while respecting and protecting its environment.

1.4.2 Vision

- Queremos ser una carrera profesional acreditada con estándares internacionales que cuente con el reconocimiento en función de la calidad y competitividad de sus docentes y egresados.
- Queremos ser una carrera que trascienda por la relevancia y pertinencia de sus proyectos de investigación básica y aplicada.
- Buscamos ser una carrera que promueva el desarrollo de la industria del software relacionada a Computación Científica a nivel internacional, incorporando a sus egresados a la industria ya establecida o generando nuevas empresas en este mismo rubro.
- Queremos ser una carrera que comparta y difunda el conocimiento con todos los sectores de la población y contribuya a la solución de los problemas estratégicos de nuestra sociedad.

1.5 Accreditation information

1.5.1 Career Outcomes

Upon completion of this program, graduates will have achieved the following outcomes:

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline.

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose.

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions.

ABET-4) An ability to communicate effectively with a range of audiences.

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts.

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty.

The competences and/or outcomes proposed in this curriculum correspond to the competences and/or outcomes proposed by ABET.

1.5.2 Educational objectives

Después de cinco años de egresado de la carrera profesional de Computación Científica, nuestros profesionales deben ser capaces de:

1. Cumplir y superar las expectativas de trabajo definidas por el entorno laboral.
2. Desempeñarse como miembro, o líder, de un equipo de trabajo tanto especializado como multidisciplinario.
3. Proponer soluciones al contexto laboral, donde se desenvuelve, basadas en la implementación o mejora del estado del arte en Scientific Computing.

4. Comunicar de forma efectiva propuestas tecnológicas a personas de distintos niveles de conocimiento y de diferentes ámbitos sociales.
5. Actualizarse y adaptarse a nuevos conocimientos en Scientific Computing y a diferentes ámbitos laborales, de forma autónoma o mediante estudios complementarios.
6. Demostrar un claro entendimiento de las consecuencias que surgen a partir de creaciones tecnológicas relacionadas a la Scientific Computing en aspectos tales como lo social, ético, humano, moral, legal, ambiental, económico, entre otros.

1.5.3 Learning Outcomes

Each KU within a KA lists both a set of topics and the learning outcomes that students must achieve regarding the specified topics. Learning outcomes are not of equal size and do not have a uniform allocation of curricular hours; topics with the same number of hours may have very different numbers of associated learning outcomes.

Each learning outcome has an associated level of mastery. In defining the different levels, we draw from other curricular approaches, especially Bloom's taxonomy, which has been well explored within Computación Científica. In this document, Bloom's levels were not applied directly, partly because several of them are driven by pedagogical context, which would introduce too much plurality in a document of this type; and partly because we intend the mastery levels to be indicative and not impose theoretical restrictions on the users of this document.

We use three expected mastery levels, which are:

Level 1 Familiarity : The student **understands** basic concepts. Answers: **"What is this?"**

Level 2 Usage : The student **applies** concepts practically. Answers: **"How to use this?"**

Level 3 Assessment : The student **evaluates and justifies** approaches. Answers: **"Why this method?"**

For example, to assess mastery levels, let us consider the notion of the derivative in calculus. At the "familiarity" level, a student is expected to know the definition of a derivative, understand its meaning as a rate of change, and know why it is a fundamental tool in basic sciences.

In order to demonstrate mastery at the "Usage" level, the student must be able to compute derivatives of elementary functions, apply differentiation rules, and use them to solve simple optimization problems or rates of change.

At the "Assessment" level, the student would be required to understand the multiple interpretations of the derivative (geometric, physical, economic, etc.), analyze its limitations, and be able to select the most appropriate approach to model and solve complex phenomena in different scientific contexts.

1.6 Profiles

1.6.1 Applicant profile

The applicant to enter the Escuela Profesional de Computación Científica at Universidad Nacional Mayor de San Marcos must have:
Knowledge of:

1. Fundamental concepts of algebra, geometry, trigonometry, and elementary mathematical analysis.
2. Basic physical and chemical phenomena in their everyday environment.
3. Elementary notions of probability and descriptive statistics.

Skills to:

1. Identify cause-effect relationships in natural and abstract systems, and predict their behaviors.
2. Distinguish between empirical observations and theoretical models, recognizing their limits and applications.
3. Perceive and describe quantitative and qualitative patterns in data and phenomena.
4. Formulate hypotheses, design verification procedures, and interpret results rigorously.
5. Communicate arguments and conclusions clearly, both orally and in writing, using appropriate technical language.

6. Maintain concentration on analytical tasks that require sustained effort.
7. Synthesize complex information and extract essential concepts.
8. Acquire and apply systematic and self-regulated study methodologies.

Attitudes of:

1. Genuine interest and curiosity in the exact and natural sciences, and their application in engineering.
2. Willingness for collaborative work in laboratories, workshops, and academic projects.
3. Initiative to explore problems beyond the curriculum and perseverance in the face of intellectual challenges.

1.6.2 Graduate profile

The graduate of the Scientific Computing Professional Program is able to:

1. Apply theoretical foundations of pure mathematics (linear algebra, mathematical analysis, number theory) to model complex problems in science and engineering.
2. Design and implement efficient numerical and symbolic algorithms for solving differential equations, optimization systems, and computational algebra problems.
3. Develop mathematical software using scientific programming languages (Python, Julia, MATLAB, C++) and specialized libraries for high-performance computing.
4. Model and simulate phenomena using Monte Carlo methods, stability analysis, graph theory, and combinatorial optimization in real-world applications.
5. Apply machine learning, deep learning, and big data analysis techniques to extract patterns, build predictors, and solve classification and clustering problems.
6. Collaborate on interdisciplinary projects in bioinformatics, cryptography, quantitative finance, computational physics, and artificial intelligence, providing mathematical rigor.
7. Analyze the computational complexity of algorithms, implement parallelization techniques, and optimize resource usage in clusters and GPU architectures.
8. Communicate mathematical and computational results through scientific publications, interactive data visualization, and rigorous technical documentation for diverse audiences.
9. Adapt mathematical tools to solve industrial problems in sectors such as telecommunications, logistics, cybersecurity, data mining, and video game development.
10. Continue graduate studies in applied mathematics, data science, or scientific computing, contributing to knowledge generation in emerging areas.

1.6.3 ProfessionalProfile

The professional in Computación Científica has the following characteristics:

- Apply fundamental theories of pure and applied mathematics to model and solve complex problems in science and engineering.
- Design and implement efficient numerical and symbolic algorithms using scientific languages (Python, Julia, MATLAB, C++).
- Develop computational models to simulate phenomena, solve differential equations, and optimize systems through advanced mathematical methods.
- Apply machine learning, deep learning, and massive data analysis techniques to extract patterns and build robust predictors.
- Work with high-performance computing architectures (clusters, GPU) to solve large-scale problems in reasonable time.
- Collaborate on interdisciplinary projects in bioinformatics, cryptography, quantitative finance, computational physics, and artificial intelligence.
- Communicate mathematical and computational results with rigor through publications, interactive visualizations, and specialized technical documentation.
- Analyze algorithmic computational complexity, implement parallelization techniques, and optimize computational resource usage.
- Adapt mathematical tools to solve challenges in sectors such as telecommunications, mining, logistics, cybersecurity, and technological development.

- Continue graduate studies in applied mathematics, data science, or scientific computing, generating original impactful knowledge.

1.6.4 Faculty profile

The professor of the Computación Científica program is a person who:

- Promotes holistic student formation through teaching, research, and extension, transmitting solid knowledge with clarity and simplicity.
- Fosters research and continuous learning so that students adapt to area advancements and seek the common good of individuals, society, and organizations.
- Pursues a PhD degree with full-time dedication, maintaining advanced academic training oriented towards international excellence.
- Publishes research in international events and high-impact indexed journals, contributing to knowledge advancement and staying updated.
- Establishes cooperation networks with international-level institutions, enriching the educational experience.
- Actively participates in student formation and integration into the academic community , promoting continuous rigor and academic excellence.

1.7 Degrees and Titles

Cumplida la aprobación de los créditos académicos propuestos en la malla curricular y luego de haber cumplido con los requisitos estipulados en el reglamento de grados y títulos de la carrera, el egresado recibirá:

Grado Académico: Bachiller en Computación Científica y

Titulo Profesional: Licenciado en Computación Científica

1.8 Resources to teach

A innovative professional must always stay up to date with the latest advances in their field. The latest developments in this area are not necessarily presented in books. We must use publications from indexed journals with worldwide circulation.

For this reason, we rely on institutional subscriptions to ACM and IEEE-CS. It is recommended that faculty use this material to discuss trends in all areas during class.

Computer laboratories must also be renewed according to the rapid pace inherent to this field, and this implies constant renovations to ensure that students remain up-to-date.

The physical resources regarding laboratories for each course are detailed in Chapter 10.

1.9 Document Organization

The remainder of this document is organized as follows: Chapter 1 defines and explains the field of action of the Computación Científica, in addition to providing a very brief explanation of the different programs related to this discipline.

In Chapter 2, we present the Knowledge Areas of Computational Mathematics, indicating the topics and learning objectives of each subject,

In Chapter 3, we present the Knowledge Areas of Computer Science, indicating the topics and learning objectives of each subject,

In Chapter 4, we present the Knowledge Areas of Mathematics, indicating the topics and learning objectives of each subject,

In Chapter 5, we present the Knowledge Areas of Physics, indicating the topics and learning objectives of each subject,

Chapter 6 contains the distribution by semesters, by areas, by levels, graphical view of the curriculum, comparison with various international proposals, distribution of topics per course, as well as the distribution of skills per subject.

Chapter 7 contains detailed information for each of the courses, including the skills to which it contributes, bibliography per unit, as well as the minimum number of hours per unit.

In Chapter 8 we can observe a list of courses by professor and of professor by course.

In Chapter 9, the equivalence tables with other curricula are presented.

In Chapter 10, a suggestion of the laboratories required for teaching classes is presented, which could vary depending on the number of students.

Chapter 2

Body of knowledge of Computational Mathematics

The 14 knowledge areas in Computational Mathematics are:

2.1 Numerical and Scientific Analysis (NSA) (Pg.. 10)
- 2.1.1 Computational Arithmetic and Error Analysis (Pg.. 10)
- 2.1.2 Numerical Linear Algebra (Pg.. 10)
- 2.1.3 Roots Finding and Basic Numerical Optimization (Pg.. 11)
- 2.1.4 Interpolation and Function Approximation (Pg.. 11)
- 2.1.5 Numerical Differentiation and Integration (Pg.. 11)
- 2.1.6 Numerical Methods for Ordinary Differential Equations (ODEs) (Pg.. 12)

2.2 Mathematical Modeling and Simulation (MMS) (Pg.. 12)
- 2.2.1 Mathematical Model Formulation (Pg.. 12)
- 2.2.2 Discrete Systems Simulation (Pg.. 13)
- 2.2.3 Continuous Systems Simulation (Pg.. 13)
- 2.2.4 Sensitivity Analysis and Robustness (Pg.. 13)
- 2.2.5 Model Validation and Verification (Pg.. 13)

2.3 Probability and Statistical Computing (PSC) (Pg.. 14)
- 2.3.1 Advanced Probability and Stochastic Processes (Pg.. 14)
- 2.3.2 Computational Statistical Inference (Pg.. 14)
- 2.3.3 Design of Computational Experiments (Pg.. 15)
- 2.3.4 Resampling and Validation Methods (Pg.. 15)

2.4 Data Structures and Algorithms (DSA) (Pg.. 15)
- 2.4.1 Algorithmic Complexity and Asymptotic Analysis (Pg.. 16)
- 2.4.2 Fundamental and Advanced Data Structures (Pg.. 16)
- 2.4.3 Searching and Sorting Algorithms (Pg.. 16)
- 2.4.4 Graph Algorithms and Combinatorial Optimization (Pg.. 17)
- 2.4.5 Parallel and Distributed Algorithms (Pg.. 17)

2.5 High-Performance Computing (HPC) (Pg.. 17)
- 2.5.1 Parallel Computer Architectures (Pg.. 18)
- 2.5.2 Shared Memory Parallel Programming (Pg.. 18)
- 2.5.3 Distributed Memory Parallel Programming (Pg.. 18)
- 2.5.4 Accelerator Programming (Pg.. 18)
- 2.5.5 Parallel Numerical Methods (Pg.. 19)

2.6 Optimization and Machine Learning (OML) (Pg.. 19)
- 2.6.1 Convex and Non-Convex Optimization (Pg.. 19)
- 2.6.2 Constrained Optimization (Pg.. 20)
- 2.6.3 Optimization for Machine Learning (Pg.. 20)
- 2.6.4 Large-Scale and Distributed Optimization (Pg.. 20)
- 2.6.5 Metaheuristics and Bio-Inspired Optimization (Pg.. 21)

2.7 Data Science and Machine Learning Mathematics (DSM) (Pg.. 21)
- 2.7.1 Foundations of Statistical Learning (Pg.. 21)
- 2.7.2 Supervised Learning Methods (Pg.. 21)
- 2.7.3 Unsupervised Learning Methods (Pg.. 22)
- 2.7.4 Model Evaluation and Selection (Pg.. 22)

2.8 Artificial Intelligence Mathematics (AIM) (Pg.. 22)
- 2.8.1 Deep Learning Theory (Pg.. 23)
- 2.8.2 Advanced Neural Architectures (Pg.. 23)
- 2.8.3 Reinforcement Learning (Pg.. 23)
- 2.8.4 Natural Language Processing with Transformers (Pg.. 24)
- 2.8.5 Generative Models and GANs (Pg.. 24)

2.9 Symbolic and Discrete Computation (SDC) (Pg.. 24)
- 2.9.1 Group Theory and Algebraic Algorithms (Pg.. 24)
- 2.9.2 Computational Number Theory and Cryptography (Pg.. 25)
- 2.9.3 Symbolic Computation Systems (Pg.. 25)
- 2.9.4 Computational Logic and Formal Verification (Pg.. 25)

2.10 Signal, Image, and Visualization (SIV) (Pg.. 26)
- 2.10.1 Fourier Analysis and Wavelets (Pg.. 26)
- 2.10.2 Digital Image Processing and Computer Vision (Pg.. 26)
- 2.10.3 Scientific Visualization and Visual Computing (Pg.. 27)
- 2.10.4 Computer Graphics and Computational Geometry (Pg.. 27)

2.11 Quantum Computing and Algorithms (QCA) (Pg.. 27)
- 2.11.1 Quantum Mechanics Foundations for Computation (Pg.. 27)
- 2.11.2 Qubits and Quantum Circuits (Pg.. 28)
- 2.11.3 Quantum Algorithms (Pg.. 28)
- 2.11.4 Post-Quantum Cryptography and Quantum Simulation (Pg.. 28)

2.12 Cybersecurity and Privacy (CSP) (Pg.. 29)
- 2.12.1 Symmetric Cryptography (Pg.. 29)
- 2.12.2 Asymmetric Cryptography (Pg.. 29)
- 2.12.3 Privacy-Preserving Computation (Pg.. 30)
- 2.12.4 Information Theory and Data Compression (Pg.. 30)

2.13 Computational Mathematics for Specific Domains (CDM) (Pg.. 30)
- 2.13.1 Bioinformatics and Computational Biology (Pg.. 30)
- 2.13.2 Computational Finance and Risk Science (Pg.. 31)
- 2.13.3 Computational Physics and Quantum Chemistry (Pg.. 31)
- 2.13.4 Computational Materials Science (Pg.. 31)
- 2.13.5 Climate Modeling and Complex Systems (Pg.. 32)

2.14 Data Governance and Management (DGM) (Pg.. 32)
- 2.14.1 Ethics in AI and Responsible Algorithms (Pg.. 32)
- 2.14.2 Data Project Management and MLOps (Pg.. 33)
- 2.14.3 Data Governance and Regulatory Compliance (Pg.. 33)

2.1 Numerical and Scientific Analysis (NSA)

This area covers the fundamental numerical methods and their analysis that form the foundation of computational mathematics. It focuses on the development, implementation, and analysis of algorithms for solving mathematical problems numerically, with a strong emphasis on understanding and controlling errors that arise from finite-precision arithmetic and discrete approximations. This knowledge is essential for all subsequent areas in computational mathematics, as it provides the tools to ensure that computed solutions are accurate, stable, and reliable.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.1.1 Computational Arithmetic and Error Analysis (Pg.. 10)	1	1
⭐ 2.1.2 Numerical Linear Algebra (Pg.. 10)	1	1
⭐ 2.1.3 Roots Finding and Basic Numerical Optimization (Pg.. 11)	1	1
⭐ 2.1.4 Interpolation and Function Approximation (Pg.. 11)	1	1
⭐ 2.1.5 Numerical Differentiation and Integration (Pg.. 11)	1	1
⭐ 2.1.6 Numerical Methods for Ordinary Differential Equations (ODEs) (Pg.. 12)	1	1

Table 2.1: List of KUs in the Numerical and Scientific Analysis area.

2.1.1 NSA/Computational Arithmetic and Error Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Floating-point arithmetic and the IEEE standard
- Sources of error: truncation, rounding, and discretization
- Error propagation and condition number
- Stability of numerical algorithms
- Absolute, relative, and forward/backward error

Learning Outcomes:
Core:

1. Define the different types of errors in numerical computation (rounding, truncation, discretization) [Familiarity]
2. Explain the principles of floating-point representation and arithmetic according to the IEEE standard [Familiarity]
3. Calculate the absolute and relative error of an approximation [Usage]
4. Analyze the propagation of errors in a sequence of arithmetic operations [Usage]
5. Distinguish between the condition of a problem and the stability of an algorithm [Assessment]
6. Compute the condition number of a simple problem (e.g., matrix-vector multiplication) [Usage]

2.1.2 NSA/Numerical Linear Algebra (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- LU decomposition and its applications
- QR decomposition and least squares problems
- Singular Value Decomposition (SVD) and its applications
- Condition number and numerical stability of linear systems
- Iterative methods for linear systems (Jacobi, Gauss-Seidel, Conjugate Gradient)

Learning Outcomes:
Core:

1. Implement LU and QR decomposition algorithms for solving linear systems [Usage]
2. Explain the geometric and algebraic meaning of the Singular Value Decomposition [Familiarity]
3. Use SVD for applications such as matrix approximation and principal component analysis [Usage]
4. Analyze the stability of a linear system using its condition number [Assessment]

5. Select and implement an appropriate iterative method for solving large, sparse linear systems [Assessment]

2.1.3 NSA/Roots Finding and Basic Numerical Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Bisection method and fixed-point iteration
- Newton's method and secant method for root finding
- Convergence analysis and order of convergence
- Gradient descent for univariate optimization
- Rate of convergence and stopping criteria

Learning Outcomes:
Core:
1. Implement root-finding algorithms (bisection, Newton, secant) and analyze their convergence [Usage]
2. Classify the order of convergence of iterative methods [Assessment]
3. Apply gradient descent to solve simple univariate optimization problems [Usage]
4. Design appropriate stopping criteria for iterative methods [Usage]

2.1.4 NSA/Interpolation and Function Approximation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Polynomial interpolation (Lagrange, Newton forms)
- Spline interpolation (cubic splines)
- Least squares approximation
- Runge's phenomenon and Chebyshev nodes
- Error analysis for interpolation

Learning Outcomes:
Core:
1. Construct interpolating polynomials using different methods [Usage]
2. Implement cubic spline interpolation [Usage]
3. Apply least squares method for data fitting [Usage]
4. Explain Runge's phenomenon and strategies to mitigate it [Familiarity]
5. Analyze the error of an interpolation scheme [Assessment]

2.1.5 NSA/Numerical Differentiation and Integration (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Finite difference approximations (forward, backward, central)
- Richardson extrapolation
- Newton-Cotes formulas (Trapezoidal rule, Simpson's rule)
- Gaussian quadrature
- Error analysis for numerical integration

Learning Outcomes:
Core:
1. Compute derivatives of functions using finite difference schemes [Usage]
2. Apply Richardson extrapolation to improve the accuracy of numerical approximations [Usage]
3. Implement numerical integration rules (Trapezoidal, Simpson, Gaussian) [Usage]
4. Compare the accuracy and efficiency of different numerical integration methods [Assessment]
5. Analyze the error for numerical differentiation and integration formulas [Assessment]

2.1.6 NSA/Numerical Methods for Ordinary Differential Equations (ODEs) (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Euler's method and its variants (forward, backward)
- Runge-Kutta methods
- Multistep methods (Adams-Bashforth, Adams-Moulton)
- Stability and convergence analysis for ODE solvers
- Systems of ODEs and higher-order ODEs

Learning Outcomes:
Core:

1. Implement basic ODE solvers (Euler, Runge-Kutta) [Usage]
2. Analyze the stability and convergence properties of ODE numerical methods [Assessment]
3. Select an appropriate numerical method for a given ODE based on stability and accuracy requirements [Assessment]
4. Solve systems of ODEs and convert higher-order ODEs to first-order systems [Usage]

2.2 Mathematical Modeling and Simulation (MMS)

This area focuses on the process of creating, analyzing, and validating mathematical models to represent real-world systems. It covers both continuous and discrete modeling paradigms, as well as deterministic and stochastic approaches. Students learn to formulate models, implement simulations, and critically evaluate the results through sensitivity analysis and uncertainty quantification. This knowledge is essential for applying computational mathematics to complex problems in science, engineering, and other domains.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.2.1 Mathematical Model Formulation (Pg.. 12)	1	1
⭐ 2.2.2 Discrete Systems Simulation (Pg.. 13)	1	1
⭐ 2.2.3 Continuous Systems Simulation (Pg.. 13)	1	1
⭐ 2.2.4 Sensitivity Analysis and Robustness (Pg.. 13)	1	1
⭐ 2.2.5 Model Validation and Verification (Pg.. 13)	1	1

Table 2.2: List of KUs in the Mathematical Modeling and Simulation area.

2.2.1 MMS/Mathematical Model Formulation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Modeling process and cycle
- Continuous vs. discrete models
- Deterministic vs. stochastic models
- Assumptions identification and simplification techniques
- Dimensional analysis and scaling

Learning Outcomes:
Core:

1. Describe the complete modeling cycle from problem definition to solution interpretation [Familiarity]
2. Classify models as continuous/discrete and deterministic/stochastic [Assessment]
3. Identify and justify appropriate assumptions for model simplification [Assessment]
4. Apply dimensional analysis to reduce complexity of physical models [Usage]

2.2.2 MMS/Discrete Systems Simulation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Queueing theory and models
- Cellular automata and agent-based models
- Monte Carlo simulation methods
- Markov chains and processes
- Discrete-event simulation

Learning Outcomes:
Core:

1. Formulate and analyze simple queueing models [Usage]
2. Implement cellular automata and agent-based simulations [Usage]
3. Apply Monte Carlo methods for probabilistic simulation [Usage]
4. Analyze Markov chains for state transitions and steady-state behavior [Assessment]
5. Design discrete-event simulations for system analysis [Usage]

2.2.3 MMS/Continuous Systems Simulation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Ordinary Differential Equations (ODEs) models
- Partial Differential Equations (PDEs) models
- Finite difference methods for PDEs
- Solver selection and implementation
- Stiff systems and adaptive methods

Learning Outcomes:
Core:

1. Formulate ODE and PDE models for physical systems [Usage]
2. Implement finite difference methods for simple PDEs [Usage]
3. Select appropriate numerical solvers for different types of ODEs/PDEs [Assessment]
4. Handle stiff systems using appropriate numerical methods [Usage]

2.2.4 MMS/Sensitivity Analysis and Robustness (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Local sensitivity analysis (partial derivatives)
- Global sensitivity analysis (Sobol indices)
- Morris method and screening techniques
- Robustness analysis and worst-case scenarios
- Uncertainty propagation

Learning Outcomes:
Core:

1. Perform local sensitivity analysis using derivative-based methods [Usage]
2. Apply global sensitivity analysis techniques (e.g., Sobol indices) [Usage]
3. Use Morris method for factor screening [Usage]
4. Assess model robustness under parameter variations [Assessment]
5. Quantify uncertainty propagation through models [Assessment]

2.2.5 MMS/Model Validation and Verification (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Verification: numerical code correctness
- Validation: comparison with experimental data

- Uncertainty Quantification (UQ) in models
- Model calibration and parameter estimation
- Goodness-of-fit tests and metrics

Learning Outcomes:
Core:

1. Distinguish between verification and validation in modeling [Familiarity]
2. Apply verification techniques to ensure code correctness [Usage]
3. Validate models using experimental data and statistical tests [Usage]
4. Calibrate model parameters using optimization techniques [Usage]
5. Apply uncertainty quantification methods to assess model reliability [Usage]

2.3 Probability and Statistical Computing (PSC)

This area bridges the theoretical foundations of probability and statistics with their computational implementation. It covers advanced probabilistic modeling, computational inference methods, and the design of experiments in silico. Students learn to apply stochastic processes, Bayesian methods, and resampling techniques to analyze complex systems and make data-driven decisions under uncertainty. This knowledge is crucial for modern data analysis, machine learning, and scientific computing.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.3.1 Advanced Probability and Stochastic Processes (Pg.. 14)	1	1
⭐ 2.3.2 Computational Statistical Inference (Pg.. 14)	1	1
⭐ 2.3.3 Design of Computational Experiments (Pg.. 15)	1	1
⭐ 2.3.4 Resampling and Validation Methods (Pg.. 15)	1	1

Table 2.3: List of KUs in the Probability and Statistical Computing area.

2.3.1 PSC/Advanced Probability and Stochastic Processes (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Measure-theoretic probability foundations
- Random variables and distributions
- Stochastic processes characterization
- Markov chains and Markov processes
- Brownian motion and diffusion processes

Learning Outcomes:
Core:

1. Apply measure-theoretic concepts to probability spaces [Familiarity]
2. Characterize different types of stochastic processes [Assessment]
3. Analyze Markov chains for steady-state and transient behavior [Usage]
4. Model systems using continuous-time stochastic processes [Usage]

2.3.2 PSC/Computational Statistical Inference (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Maximum Likelihood Estimation (MLE)
- Bayesian inference methods
- Markov Chain Monte Carlo (MCMC) methods
- Expectation-Maximization (EM) algorithm
- Bootstrap and jackknife resampling

Learning Outcomes:
Core:

1. Implement Maximum Likelihood Estimation for various distributions [Usage]
2. Apply Bayesian methods for parameter estimation [Usage]
3. Implement MCMC algorithms for posterior sampling [Usage]
4. Use EM algorithm for models with latent variables [Usage]
5. Apply bootstrap methods for confidence intervals [Usage]

2.3.3 PSC/Design of Computational Experiments (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Factorial designs and fractional factorials
- Response surface methodology
- Space-filling designs (Latin hypercube)
- Optimal design of experiments
- Computer experiments and surrogate modeling

Learning Outcomes:
Core:

1. Design factorial experiments for factor screening [Usage]
2. Apply response surface methodology for optimization [Usage]
3. Implement space-filling designs for computer experiments [Usage]
4. Select optimal designs for specific objectives [Assessment]
5. Construct surrogate models for expensive simulations [Usage]

2.3.4 PSC/Resampling and Validation Methods (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Bootstrap estimation and confidence intervals
- Cross-validation techniques
- Permutation tests and randomization
- Model selection and information criteria
- Ensemble methods and bagging

Learning Outcomes:
Core:

1. Implement bootstrap for standard error estimation [Usage]
2. Apply cross-validation for model assessment [Usage]
3. Use permutation tests for hypothesis testing [Usage]
4. Compare models using information criteria (AIC, BIC) [Assessment]
5. Implement bagging for variance reduction [Usage]

2.4 Data Structures and Algorithms (DSA)

This area provides the fundamental toolkit for designing and analyzing efficient computational solutions. It covers the theoretical foundations of algorithm complexity, the practical implementation of fundamental and advanced data structures, and the design principles behind efficient algorithms for searching, sorting, graph processing, and combinatorial optimization. This knowledge forms the computational core that enables students to develop efficient solutions across all other areas of computational mathematics.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.4.1 Algorithmic Complexity and Asymptotic Analysis (Pg.. 16)	1	1
⭐ 2.4.2 Fundamental and Advanced Data Structures (Pg.. 16)	1	1
⭐ 2.4.3 Searching and Sorting Algorithms (Pg.. 16)	1	1
⭐ 2.4.4 Graph Algorithms and Combinatorial Optimization (Pg.. 17)	1	1
⭐ 2.4.5 Parallel and Distributed Algorithms (Pg.. 17)	1	1

Table 2.4: List of KUs in the Data Structures and Algorithms area.

2.4.1 DSA/Algorithmic Complexity and Asymptotic Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Complexity measures and asymptotic notation
- Best, average, and worst-case analysis
- Recurrence relations and solving techniques
- Empirical analysis of algorithms
- Space complexity and trade-offs

Learning Outcomes:
Core:
1. Analyze algorithms using asymptotic notation (Big O, Omega, Theta) [Usage]
2. Differentiate between best, average, and worst-case complexities [Assessment]
3. Solve recurrence relations for algorithm analysis [Usage]
4. Compare algorithms through empirical measurements [Assessment]

2.4.2 DSA/Fundamental and Advanced Data Structures (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Arrays, linked lists, and dynamic arrays
- Stacks, queues, and deques
- Hash tables and collision resolution
- Trees and binary search trees
- Heaps and priority queues

Learning Outcomes:
Core:
1. Implement basic data structures from scratch [Usage]
2. Analyze time/space complexity of data structure operations [Assessment]
3. Select appropriate data structures for given problems [Assessment]
4. Handle hash collisions using different strategies [Usage]

2.4.3 DSA/Searching and Sorting Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Linear and binary search
- Comparison-based sorting (QuickSort, MergeSort, HeapSort)
- Non-comparison sorting (RadixSort, CountingSort)
- Selection algorithms and order statistics
- External sorting and large-scale processing

Learning Outcomes:
Core:
1. Implement and compare searching algorithms [Usage]

2. Analyze time/space complexity of sorting algorithms [Assessment]
3. Select appropriate sorting algorithm for data characteristics [Assessment]
4. Implement order statistics algorithms [Usage]

2.4.4 DSA/Graph Algorithms and Combinatorial Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Graph representation and traversal (BFS, DFS)
- Shortest path algorithms (Dijkstra, Bellman-Ford)
- Minimum spanning trees (Prim, Kruskal)
- Network flow and matching
- NP-completeness and approximation algorithms

Learning Outcomes:
Core:

1. Implement graph traversal algorithms [Usage]
2. Solve shortest path problems using appropriate algorithms [Usage]
3. Construct minimum spanning trees [Usage]
4. Analyze complexity of graph algorithms [Assessment]
5. Design approximation algorithms for NP-hard problems [Usage]

2.4.5 DSA/Parallel and Distributed Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Parallel computation models (PRAM)
- Basic parallel algorithms (reduction, scan)
- Distributed algorithms and consensus
- Synchronization and load balancing
- MapReduce and distributed data processing

Learning Outcomes:
Core:

1. Analyze parallel algorithms using work-span model [Assessment]
2. Implement parallel reduction and scan operations [Usage]
3. Design distributed algorithms for consensus [Usage]
4. Apply MapReduce paradigm to data processing [Usage]

2.5 High-Performance Computing (HPC)

This area focuses on the principles and practices of developing computational solutions that leverage parallel and distributed computing resources to solve large-scale scientific and engineering problems. It covers computer architectures, parallel programming models, optimization techniques for modern hardware, and the implementation of numerical methods on high-performance systems. This knowledge enables students to tackle computationally intensive problems that are infeasible on single-processor systems.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.5.1 Parallel Computer Architectures (Pg.. 18)	1	1
⭐ 2.5.2 Shared Memory Parallel Programming (Pg.. 18)	1	1
⭐ 2.5.3 Distributed Memory Parallel Programming (Pg.. 18)	1	1
⭐ 2.5.4 Accelerator Programming (Pg.. 18)	1	1
⭐ 2.5.5 Parallel Numerical Methods (Pg.. 19)	1	1

Table 2.5: List of KUs in the High-Performance Computing area.

2.5.1 HPC/Parallel Computer Architectures (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Flynn's taxonomy and parallel architectures
- Memory hierarchies and cache coherence
- Multi-core processors and shared memory
- Distributed memory systems and clusters
- Accelerators (GPUs, FPGAs) and heterogeneous computing

Learning Outcomes:
Core:

1. Classify parallel architectures using Flynn's taxonomy [Familiarity]
2. Analyze memory hierarchy effects on performance [Assessment]
3. Compare multi-core and distributed memory systems [Assessment]
4. Select appropriate accelerators for specific computational tasks [Assessment]

2.5.2 HPC/Shared Memory Parallel Programming (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- OpenMP programming model
- Thread creation and management
- Synchronization primitives (locks, barriers)
- Data sharing and scoping
- Performance optimization and load balancing

Learning Outcomes:
Core:

1. Implement parallel algorithms using OpenMP [Usage]
2. Manage threads and task parallelism [Usage]
3. Apply synchronization mechanisms correctly [Usage]
4. Optimize shared memory performance [Usage]

2.5.3 HPC/Distributed Memory Parallel Programming (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- MPI programming model
- Point-to-point communication
- Collective communication operations
- Process topologies and groups
- Scalability analysis and optimization

Learning Outcomes:
Core:

1. Implement distributed algorithms using MPI [Usage]
2. Handle point-to-point and collective communications [Usage]
3. Design efficient process topologies [Usage]
4. Analyze scalability of distributed applications [Assessment]

2.5.4 HPC/Accelerator Programming (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- GPU architecture and programming models
- CUDA programming model
- OpenACC directives
- Kernel design and optimization
- Memory management on accelerators

Learning Outcomes:
Core:

1. Program GPUs using CUDA [Usage]
2. Apply OpenACC directives for accelerator programming [Usage]
3. Design efficient GPU kernels [Usage]
4. Manage GPU memory hierarchy effectively [Usage]

2.5.5 HPC/Parallel Numerical Methods (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Parallel linear algebra algorithms
- Parallel PDE solvers
- Domain decomposition methods
- Load balancing for numerical computations
- Performance tuning and profiling

Learning Outcomes:
Core:

1. Implement parallel numerical algorithms [Usage]
2. Apply domain decomposition for PDEs [Usage]
3. Balance computational load across processors [Usage]
4. Tune performance of parallel numerical codes [Usage]

2.6 Optimization and Machine Learning (OML)

This area covers the mathematical foundations and computational methods for optimization, with special emphasis on techniques relevant to machine learning. It includes convex and non-convex optimization, constrained optimization, large-scale optimization methods, and metaheuristics. The area bridges traditional optimization theory with modern machine learning applications, providing the tools needed to train complex models and solve large-scale optimization problems efficiently.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.6.1 Convex and Non-Convex Optimization (Pg.. 19)	1	1
⭐ 2.6.2 Constrained Optimization (Pg.. 20)	1	1
⭐ 2.6.3 Optimization for Machine Learning (Pg.. 20)	1	1
⭐ 2.6.4 Large-Scale and Distributed Optimization (Pg.. 20)	1	1
⭐ 2.6.5 Metaheuristics and Bio-Inspired Optimization (Pg.. 21)	1	1

Table 2.6: List of KUs in the Optimization and Machine Learning area.

2.6.1 OML/Convex and Non-Convex Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Convex sets and functions
- Optimality conditions
- Gradient methods for convex optimization
- Non-convex optimization challenges
- Convergence analysis

Learning Outcomes:
Core:
1. Characterize convex optimization problems [Assessment]
2. Apply optimality conditions to verify solutions [Usage]
3. Implement gradient-based methods for convex problems [Usage]
4. Analyze challenges in non-convex optimization [Assessment]

2.6.2 OML/Constrained Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Linear programming (simplex method)
- Integer programming
- Lagrange multipliers and KKT conditions
- Penalty and barrier methods
- Duality theory

Learning Outcomes:
Core:
1. Solve linear programming problems [Usage]
2. Formulate and solve integer programming problems [Usage]
3. Apply KKT conditions to constrained problems [Usage]
4. Use penalty methods for constrained optimization [Usage]

2.6.3 OML/Optimization for Machine Learning (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Stochastic gradient descent
- Momentum and adaptive methods (Adam, RMSProp)
- Regularization and sparsity
- Distributed optimization for ML
- Second-order methods for ML

Learning Outcomes:
Core:
1. Implement stochastic gradient descent variants [Usage]
2. Apply adaptive optimization methods [Usage]
3. Use regularization for model generalization [Usage]
4. Design distributed optimization strategies [Usage]

2.6.4 OML/Large-Scale and Distributed Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- First-order methods for large-scale problems
- Coordinate descent and block methods
- Consensus and distributed optimization
- Federated learning optimization
- Communication-efficient distributed optimization

Learning Outcomes:
Core:
1. Apply first-order methods to large-scale problems [Usage]
2. Implement coordinate descent algorithms [Usage]
3. Design consensus-based distributed algorithms [Usage]
4. Optimize federated learning systems [Usage]

2.6.5 OML/Metaheuristics and Bio-Inspired Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Genetic algorithms and evolutionary strategies
- Swarm intelligence (PSO, ant colony)
- Simulated annealing
- Tabu search
- Hybrid metaheuristics

Learning Outcomes:
Core:

1. Implement genetic algorithms [Usage]
2. Apply swarm intelligence methods [Usage]
3. Use simulated annealing for global optimization [Usage]
4. Design hybrid metaheuristic approaches [Usage]

2.7 Data Science and Machine Learning Mathematics (DSM)

This area provides the mathematical foundations underlying modern data science and machine learning. It covers statistical learning theory, supervised and unsupervised learning methods, model evaluation techniques, and the mathematical principles that govern the behavior of learning algorithms. The focus is on understanding why machine learning methods work, their theoretical guarantees, and their limitations, rather than just their application.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.7.1 Foundations of Statistical Learning (Pg.. 21)	1	1
⭐ 2.7.2 Supervised Learning Methods (Pg.. 21)	1	1
⭐ 2.7.3 Unsupervised Learning Methods (Pg.. 22)	1	1
⭐ 2.7.4 Model Evaluation and Selection (Pg.. 22)	1	1

Table 2.7: List of KUs in the Data Science and Machine Learning Mathematics area.

2.7.1 DSM/Foundations of Statistical Learning (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Bias-variance tradeoff
- Vapnik-Chervonenkis (VC) theory
- Empirical risk minimization
- Generalization bounds
- Learning theory fundamentals

Learning Outcomes:
Core:

1. Analyze the bias-variance tradeoff in model selection [Assessment]
2. Apply VC theory to analyze model complexity [Usage]
3. Implement empirical risk minimization [Usage]
4. Interpret generalization bounds [Familiarity]

2.7.2 DSM/Supervised Learning Methods (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Linear models and extensions

- Support Vector Machines (SVM)
- Decision trees and ensemble methods
- Kernel methods
- Probabilistic supervised learning

Learning Outcomes:
Core:

1. Implement and analyze linear models [Usage]
2. Apply SVM for classification and regression [Usage]
3. Construct ensemble methods (bagging, boosting) [Usage]
4. Use kernel methods for nonlinear problems [Usage]

2.7.3 DSM/Unsupervised Learning Methods (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Clustering algorithms (k-means, hierarchical)
- Dimensionality reduction (PCA, t-SNE)
- Density estimation
- Manifold learning
- Autoencoders and deep unsupervised learning

Learning Outcomes:
Core:

1. Apply clustering algorithms to data exploration [Usage]
2. Implement PCA and other dimensionality reduction methods [Usage]
3. Use density estimation for unsupervised learning [Usage]
4. Analyze manifold learning techniques [Assessment]

2.7.4 DSM/Model Evaluation and Selection (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Cross-validation strategies
- Performance metrics for classification and regression
- Model comparison and statistical tests
- Hyperparameter optimization
- Learning curves and diagnostic tools

Learning Outcomes:
Core:

1. Implement cross-validation for model assessment [Usage]
2. Select appropriate performance metrics [Assessment]
3. Compare models using statistical tests [Usage]
4. Optimize hyperparameters using systematic methods [Usage]

2.8 Artificial Intelligence Mathematics (AIM)

This area covers the advanced mathematical foundations of modern artificial intelligence, with emphasis on deep learning theory, advanced neural architectures, reinforcement learning, and generative models. It provides the theoretical understanding needed to develop novel AI architectures, analyze their properties, and understand their limitations. The focus is on the mathematical principles that enable machines to learn complex representations and make intelligent decisions.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.8.1 Deep Learning Theory (Pg.. 23)	1	1
⭐ 2.8.2 Advanced Neural Architectures (Pg.. 23)	1	1
⭐ 2.8.3 Reinforcement Learning (Pg.. 23)	1	1
⭐ 2.8.4 Natural Language Processing with Transformers (Pg.. 24)	1	1
⭐ 2.8.5 Generative Models and GANs (Pg.. 24)	1	1

Table 2.8: List of KUs in the Artificial Intelligence Mathematics area.

2.8.1 AIM/Deep Learning Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Universal approximation theorems
- Optimization landscape of deep networks
- Generalization in deep learning
- Representation learning theory
- Training dynamics and gradient flow

Learning Outcomes:
Core:
1. Explain universal approximation theorems and their implications [Familiarity]
2. Analyze the optimization landscape of deep networks [Assessment]
3. Study generalization phenomena in deep learning [Assessment]
4. Analyze training dynamics of neural networks [Assessment]

2.8.2 AIM/Advanced Neural Architectures (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Convolutional Neural Networks (CNNs)
- Recurrent Neural Networks (RNNs) and LSTMs
- Transformer architectures
- Attention mechanisms
- Graph Neural Networks (GNNs)

Learning Outcomes:
Core:
1. Design CNN architectures for visual tasks [Usage]
2. Implement RNNs for sequence modeling [Usage]
3. Apply transformer architectures to various domains [Usage]
4. Use attention mechanisms in neural networks [Usage]

2.8.3 AIM/Reinforcement Learning (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Markov Decision Processes (MDPs)
- Value iteration and policy iteration
- Q-learning and temporal difference learning
- Policy gradient methods
- Deep reinforcement learning

Learning Outcomes:
Core:
1. Formulate problems as Markov Decision Processes [Usage]
2. Implement value and policy iteration [Usage]
3. Apply Q-learning for model-free RL [Usage]
4. Use policy gradient methods [Usage]

2.8.4 AIM/Natural Language Processing with Transformers (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Transformer architecture and self-attention
- BERT and pre-trained language models
- GPT models and autoregressive generation
- Multimodal transformers
- Efficient transformer variants

Learning Outcomes:
Core:
1. Implement transformer models from scratch [Usage]
2. Fine-tune pre-trained language models [Usage]
3. Apply GPT models for text generation [Usage]
4. Use multimodal transformers for cross-modal tasks [Usage]

2.8.5 AIM/Generative Models and GANs (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Generative Adversarial Networks (GANs)
- Variational Autoencoders (VAEs)
- Normalizing flows
- Diffusion models
- Evaluation of generative models

Learning Outcomes:
Core:
1. Implement GANs for image generation [Usage]
2. Apply VAEs for latent representation learning [Usage]
3. Use diffusion models for generation [Usage]
4. Evaluate generative model quality [Assessment]

2.9 Symbolic and Discrete Computation (SDC)

This area bridges classical mathematical disciplines with computational methods, focusing on symbolic computation, algebraic algorithms, number theory applications, and computational logic. It covers the manipulation of mathematical expressions in exact form, the computational aspects of discrete mathematics, and the application of these techniques to cryptography and formal verification. This knowledge is essential for developing computer algebra systems, cryptographic protocols, and formally verified software.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.9.1 Group Theory and Algebraic Algorithms (Pg.. 24)	1	1
⭐ 2.9.2 Computational Number Theory and Cryptography (Pg.. 25)	1	1
⭐ 2.9.3 Symbolic Computation Systems (Pg.. 25)	1	1
⭐ 2.9.4 Computational Logic and Formal Verification (Pg.. 25)	1	1

Table 2.9: List of KUs in the Symbolic and Discrete Computation area.

2.9.1 SDC/Group Theory and Algebraic Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Group theory fundamentals and computational representation

- Algebraic algorithms for group operations
- Permutation groups and computational symmetry
- Applications in cryptography and coding theory
- Complexity of algebraic algorithms

Learning Outcomes:

Core:

1. Implement fundamental group operations computationally [Usage]
2. Analyze algebraic algorithms for group theory [Assessment]
3. Apply permutation groups to symmetry problems [Usage]
4. Use group theory in cryptographic applications [Usage]

2.9.2 SDC/Computational Number Theory and Cryptography (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Modular arithmetic and algorithms
- Primality testing and factorization
- Elliptic curve cryptography
- Discrete logarithm problem
- Post-quantum cryptography foundations

Learning Outcomes:

Core:

1. Implement modular arithmetic algorithms [Usage]
2. Apply primality testing methods [Usage]
3. Use elliptic curves in cryptographic protocols [Usage]
4. Analyze discrete logarithm based systems [Assessment]

2.9.3 SDC/Symbolic Computation Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Computer algebra systems (Mathematica, SymPy)
- Symbolic integration and differentiation
- Equation solving and simplification
- Theorem proving and automated reasoning
- Applications in scientific computing

Learning Outcomes:

Core:

1. Use computer algebra systems for mathematical computation [Usage]
2. Implement symbolic computation algorithms [Usage]
3. Apply theorem proving techniques [Usage]
4. Connect symbolic and numerical computation [Assessment]

2.9.4 SDC/Computational Logic and Formal Verification (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Propositional and predicate logic
- SAT solvers and satisfiability
- Model checking and temporal logic
- Program verification and Hoare logic
- Interactive theorem provers

Learning Outcomes:

Core:

1. Apply computational logic to problem solving [Usage]
2. Use SAT solvers for constraint satisfaction [Usage]
3. Implement model checking algorithms [Usage]
4. Verify programs using formal methods [Usage]

2.10 Signal, Image, and Visualization (SIV)

This area covers the mathematical and computational foundations for processing, analyzing, and visualizing signals, images, and data. It includes Fourier and wavelet analysis, digital image processing, computer vision algorithms, scientific visualization techniques, and computer graphics. The focus is on developing the mathematical understanding and computational skills needed to extract information from sensory data and create effective visual representations of complex information.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.10.1 Fourier Analysis and Wavelets (Pg.. 26)	1	1
⭐ 2.10.2 Digital Image Processing and Computer Vision (Pg.. 26)	1	1
⭐ 2.10.3 Scientific Visualization and Visual Computing (Pg.. 27)	1	1
⭐ 2.10.4 Computer Graphics and Computational Geometry (Pg.. 27)	1	1

Table 2.10: List of KUs in the Signal, Image, and Visualization area.

2.10.1 SIV/Fourier Analysis and Wavelets (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Fourier series and transforms
- Discrete Fourier Transform (DFT) and FFT
- Wavelet transforms and multiresolution analysis
- Filter design and signal processing
- Applications in signal and image analysis

Learning Outcomes:
Core:

1. Implement FFT algorithms for signal processing [Usage]
2. Apply wavelet transforms for multiscale analysis [Usage]
3. Design digital filters for signal processing [Usage]
4. Analyze signals in frequency domain [Assessment]

2.10.2 SIV/Digital Image Processing and Computer Vision (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Image representation and color spaces
- Image filtering and enhancement
- Image segmentation and feature detection
- Morphological image processing
- Computer vision algorithms

Learning Outcomes:
Core:

1. Implement image filtering algorithms [Usage]
2. Apply image segmentation techniques [Usage]
3. Use morphological operations for image analysis [Usage]
4. Develop computer vision applications [Usage]

2.10.3 SIV/Scientific Visualization and Visual Computing (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Visualization principles and perception
- Flow visualization and vector fields
- Volume rendering and isosurfaces
- Information visualization
- Visual analytics and interactive exploration

Learning Outcomes:
Core:

1. Design effective scientific visualizations [Usage]
2. Implement flow visualization techniques [Usage]
3. Apply volume rendering to scientific data [Usage]
4. Create interactive visual analytics tools [Usage]

2.10.4 SIV/Computer Graphics and Computational Geometry (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Rendering pipeline and transformations
- Geometric algorithms and data structures
- Curves and surfaces representation
- Collision detection and computational geometry
- Real-time graphics and GPU programming

Learning Outcomes:
Core:

1. Implement basic rendering algorithms [Usage]
2. Apply computational geometry algorithms [Usage]
3. Design curves and surfaces using mathematical representations [Usage]
4. Develop GPU-accelerated graphics applications [Usage]

2.11 Quantum Computing and Algorithms (QCA)

This area introduces the fundamental principles of quantum computing and quantum algorithms, bridging the gap between quantum mechanics and computational mathematics. It covers quantum bits, quantum gates, quantum circuits, and the development of quantum algorithms that offer potential speedups over classical approaches. The focus is on understanding the mathematical foundations of quantum computation and implementing quantum algorithms using classical simulation tools.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.11.1 Quantum Mechanics Foundations for Computation (Pg.. 27)	1	1
⭐ 2.11.2 Qubits and Quantum Circuits (Pg.. 28)	1	1
⭐ 2.11.3 Quantum Algorithms (Pg.. 28)	1	1
⭐ 2.11.4 Post-Quantum Cryptography and Quantum Simulation (Pg.. 28)	1	1

Table 2.11: List of KUs in the Quantum Computing and Algorithms area.

2.11.1 QCA/Quantum Mechanics Foundations for Computation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Quantum states and state vectors
- Superposition and entanglement
- Quantum measurement and observables
- Hilbert spaces and Dirac notation
- Unitary evolution and quantum dynamics

Learning Outcomes:

Core:

1. Describe quantum states using Dirac notation [Familiarity]
2. Explain superposition and entanglement phenomena [Familiarity]
3. Apply quantum measurement principles [Usage]
4. Analyze unitary transformations in quantum systems [Assessment]

2.11.2 QCA/Qubits and Quantum Circuits (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Qubit representation and Bloch sphere
- Quantum gates and circuits
- Universal quantum gate sets
- Quantum noise and error models
- Quantum circuit simulation

Learning Outcomes:

Core:

1. Represent qubits using mathematical formalism [Usage]
2. Design quantum circuits using standard gates [Usage]
3. Implement universal quantum gate sets [Usage]
4. Simulate quantum circuits on classical computers [Usage]

2.11.3 QCA/Quantum Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Deutsch-Jozsa algorithm
- Grover's search algorithm
- Shor's factoring algorithm
- Quantum Fourier Transform
- Quantum walk algorithms

Learning Outcomes:

Core:

1. Implement Deutsch-Jozsa algorithm [Usage]
2. Apply Grover's algorithm for search problems [Usage]
3. Analyze Shor's factoring algorithm [Assessment]
4. Use Quantum Fourier Transform in algorithms [Usage]

2.11.4 QCA/Post-Quantum Cryptography and Quantum Simulation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Quantum threat to classical cryptography
- Lattice-based cryptography
- Code-based cryptography
- Quantum simulation of physical systems
- Quantum error correction

Learning Outcomes:

Core:

1. Analyze quantum threats to cryptographic systems [Assessment]
2. Implement lattice-based cryptographic schemes [Usage]
3. Simulate quantum systems using classical computers [Usage]
4. Apply quantum error correction codes [Usage]

2.12 Cybersecurity and Privacy (CSP)

This area covers the mathematical foundations of modern cybersecurity and privacy-preserving technologies. It includes cryptographic protocols, security analysis, privacy-enhancing technologies, and information theory applications. The focus is on developing the mathematical understanding needed to design, analyze, and implement secure systems and privacy-preserving algorithms in the context of computational mathematics.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.12.1 Symmetric Cryptography (Pg.. 29)	1	1
⭐ 2.12.2 Asymmetric Cryptography (Pg.. 29)	1	1
⭐ 2.12.3 Privacy-Preserving Computation (Pg.. 30)	1	1
⭐ 2.12.4 Information Theory and Data Compression (Pg.. 30)	1	1

Table 2.12: List of KUs in the Cybersecurity and Privacy area.

2.12.1 CSP/Symmetric Cryptography (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Block ciphers and modes of operation
- Stream ciphers
- Cryptographic hash functions
- Message authentication codes
- Cryptanalysis techniques

Learning Outcomes:
Core:

1. Implement block cipher algorithms [Usage]
2. Apply modes of operation for encryption [Usage]
3. Use cryptographic hash functions [Usage]
4. Analyze cryptanalysis attacks [Assessment]

2.12.2 CSP/Asymmetric Cryptography (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Public-key cryptography principles
- RSA cryptosystem
- Elliptic curve cryptography
- Diffie-Hellman key exchange
- Digital signatures

Learning Outcomes:
Core:

1. Implement RSA encryption and decryption [Usage]
2. Apply elliptic curve cryptography [Usage]
3. Use Diffie-Hellman for secure key exchange [Usage]
4. Implement digital signature schemes [Usage]

2.12.3 CSP/Privacy-Preserving Computation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Differential privacy
- Homomorphic encryption
- Secure multi-party computation
- Federated learning privacy
- Privacy metrics and analysis

Learning Outcomes:
Core:

1. Apply differential privacy mechanisms [Usage]
2. Use homomorphic encryption for private computation [Usage]
3. Implement secure multi-party computation protocols [Usage]
4. Analyze privacy guarantees of algorithms [Assessment]

2.12.4 CSP/Information Theory and Data Compression (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Entropy and information measures
- Source coding theorem
- Channel capacity and noisy channels
- Compression algorithms
- Cryptographic security and information theory

Learning Outcomes:
Core:

1. Calculate entropy and mutual information [Usage]
2. Apply source coding to data compression [Usage]
3. Analyze channel capacity for communication [Assessment]
4. Implement compression algorithms [Usage]

2.13 Computational Mathematics for Specific Domains (CDM)

This area applies computational mathematics methods to specific application domains, demonstrating how mathematical modeling, simulation, and data analysis techniques are used to solve real-world problems in various scientific and engineering fields. It covers computational biology, quantitative finance, computational physics, materials science, and climate modeling, providing students with domain-specific knowledge and interdisciplinary problem-solving skills.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.13.1 Bioinformatics and Computational Biology (Pg.. 30)	1	1
⭐ 2.13.2 Computational Finance and Risk Science (Pg.. 31)	1	1
⭐ 2.13.3 Computational Physics and Quantum Chemistry (Pg.. 31)	1	1
⭐ 2.13.4 Computational Materials Science (Pg.. 31)	1	1
⭐ 2.13.5 Climate Modeling and Complex Systems (Pg.. 32)	1	1

Table 2.13: List of KUs in the Computational Mathematics for Specific Domains area.

2.13.1 CDM/Bioinformatics and Computational Biology (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Sequence alignment and analysis
- Genomic data analysis
- Protein structure prediction
- Systems biology and network analysis
- Phylogenetic tree construction

Learning Outcomes:

Core:

1. Implement sequence alignment algorithms [Usage]
2. Analyze genomic datasets using statistical methods [Usage]
3. Model protein structures using computational methods [Usage]
4. Apply systems biology approaches [Usage]

2.13.2 CDM/Computational Finance and Risk Science (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Option pricing models
- Portfolio optimization
- Risk management and value at risk
- Stochastic calculus in finance
- Algorithmic trading strategies

Learning Outcomes:

Core:

1. Price options using numerical methods [Usage]
2. Optimize investment portfolios [Usage]
3. Calculate risk measures and value at risk [Usage]
4. Apply stochastic calculus to financial modeling [Usage]

2.13.3 CDM/Computational Physics and Quantum Chemistry (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Molecular dynamics simulation
- Quantum chemistry calculations
- Monte Carlo methods in physics
- Continuum mechanics simulation
- Electromagnetic field computation

Learning Outcomes:

Core:

1. Simulate molecular dynamics [Usage]
2. Implement quantum chemistry algorithms [Usage]
3. Apply Monte Carlo methods to physical systems [Usage]
4. Model continuum mechanics problems [Usage]

2.13.4 CDM/Computational Materials Science (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Crystal structure prediction
- Phase field modeling
- Density Functional Theory (DFT) calculations
- Multiscale materials modeling
- Materials informatics and machine learning

Learning Outcomes:
Core:
1. Predict crystal structures using computational methods [Usage]
2. Apply phase field models for material evolution [Usage]
3. Implement DFT calculations for electronic structure [Usage]
4. Use multiscale modeling approaches [Usage]

2.13.5 CDM/Climate Modeling and Complex Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Climate system modeling
- Fluid dynamics and atmospheric models
- Complex systems and network theory
- Ecosystem modeling
- Uncertainty quantification in climate models

Learning Outcomes:
Core:
1. Model climate systems using differential equations [Usage]
2. Simulate fluid dynamics for atmospheric modeling [Usage]
3. Analyze complex systems using network theory [Assessment]
4. Quantify uncertainty in environmental models [Usage]

2.14 Data Governance and Management (DGM)

This area covers the ethical, managerial, and organizational aspects of data science and computational mathematics. It addresses the responsible development and deployment of AI systems, project management methodologies for data-intensive projects, data governance frameworks, and regulatory compliance. The focus is on developing the professional skills needed to lead computational mathematics projects responsibly and effectively in real-world contexts.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 2.14.1 Ethics in AI and Responsible Algorithms (Pg.. 32)	1	1
⭐ 2.14.2 Data Project Management and MLOps (Pg.. 33)	1	1
⭐ 2.14.3 Data Governance and Regulatory Compliance (Pg.. 33)	1	1

Table 2.14: List of KUs in the Data Governance and Management area.

2.14.1 DGM/Ethics in AI and Responsible Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Algorithmic fairness and bias mitigation
- Transparency and explainable AI (XAI)
- Accountability and responsibility in AI systems
- Ethical frameworks for AI development
- FAT/ML principles (Fairness, Accountability, Transparency)

Learning Outcomes:
Core:
1. Detect and mitigate algorithmic bias [Usage]
2. Implement explainable AI techniques [Usage]
3. Apply ethical frameworks to AI projects [Assessment]
4. Evaluate systems using FAT/ML principles [Assessment]

2.14.2 DGM/Data Project Management and MLOps (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- MLOps and model lifecycle management
- Data project management methodologies
- Version control for models and data
- Model deployment and monitoring
- Infrastructure for data science

Learning Outcomes:
Core:

1. Implement MLOps practices [Usage]
2. Manage data science projects effectively [Usage]
3. Deploy and monitor machine learning models [Usage]
4. Setup data science infrastructure [Usage]

2.14.3 DGM/Data Governance and Regulatory Compliance (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Data governance frameworks
- GDPR and data protection regulations
- Data quality management
- Data ethics and stewardship
- Compliance monitoring and auditing

Learning Outcomes:
Core:

1. Implement data governance frameworks [Usage]
2. Apply GDPR and privacy regulations [Usage]
3. Manage data quality throughout lifecycle [Usage]
4. Ensure regulatory compliance in data projects [Usage]

Chapter 3

Body of knowledge of Computer Science

The 17 knowledge areas in Computer Science are:

3.1 Artificial Intelligence (AI) (Pg.. 37)

- 3.1.1 Introduction to AI: Concepts and Intelligent Behavior (Pg.. 38)
- 3.1.2 Problem Characteristics and Agent Nature (Pg.. 39)
- 3.1.3 State and Uninformed Search (Pg.. 39)
- 3.1.4 Heuristic Search and CSP (Pg.. 39)
- 3.1.5 Adversarial Search and Stochastic Optimization (Pg.. 40)
- 3.1.6 Knowledge Representation and Probabilistic Reasoning Foundations (Pg.. 40)
- 3.1.7 Independence, Markov Models and Decision Making (Pg.. 41)
- 3.1.8 Machine Learning Fundamentals (Pg.. 41)
- 3.1.9 Machine Learning Data, Representations and Evaluation (Pg.. 41)
- 3.1.10 Neural Networks and Ethics (Pg.. 42)
- 3.1.11 Machine Learning Formulation and Evaluation (Pg.. 42)
- 3.1.12 Deep Learning, RL and Ethics (Pg.. 43)
- 3.1.13 Neural Network Training and Optimization (Pg.. 43)
- 3.1.14 Regularization Techniques for Deep Networks (Pg.. 43)
- 3.1.15 Convolutional and Residual Network Architectures (Pg.. 43)
- 3.1.16 Transformer Architectures and Generative Models (Pg.. 44)
- 3.1.17 AI Applications and Societal Impact (Pg.. 44)
- 3.1.18 Additional AI Applications and Societal Impact (Pg.. 44)
- 3.1.19 Logical Representation and Reasoning (Pg.. 45)
- 3.1.20 Probabilistic Representation and Reasoning (Pg.. 45)
- 3.1.21 Planning (Pg.. 46)
- 3.1.22 Agents and Cognitive Systems (Pg.. 46)
- 3.1.23 Agentic Loop Engineering (Pg.. 47)
- 3.1.24 Natural Language Processing (Pg.. 47)
- 3.1.25 Robotics (Pg.. 48)
- 3.1.26 Perception and Computer Vision (Pg.. 49)

3.2 Algorithmic Foundations (AL) (Pg.. 50)

- 3.2.1 Foundational Data Structures (Pg.. 50)
- 3.2.2 Foundational Algorithms (Pg.. 51)
- 3.2.3 Advanced Algorithms (Pg.. 51)
- 3.2.4 Algorithmic Strategies (Pg.. 52)
- 3.2.5 Complexity Analysis Framework (Pg.. 52)
- 3.2.6 Asymptotic Notation and Complexity Classes (Pg.. 53)
- 3.2.7 Complexity Analysis: Recursion, Amortization, and Tight Bounds (Pg.. 53)
- 3.2.8 Computational Complexity Theory (Pg.. 53)
- 3.2.9 Formal Languages and Automata (Pg.. 54)
- 3.2.10 Computability and Decidability (Pg.. 54)
- 3.2.11 Society, Ethics, and the Profession (Pg.. 55)

3.3 Architecture and Organization (AR) (Pg.. 56)

- 3.3.1 Digital Logic and Digital Systems (Pg.. 56)
- 3.3.2 Machine-Level Data Representation (Pg.. 57)
- 3.3.3 Assembly Level Machine Organization (Pg.. 57)
- 3.3.4 Memory Hierarchy (Pg.. 58)
- 3.3.5 Interfacing and Communication (Pg.. 58)
- 3.3.6 Functional Organization (Pg.. 58)
- 3.3.7 Performance and Energy Efficiency (Pg.. 59)
- 3.3.8 Heterogeneous Architectures (Pg.. 59)
- 3.3.9 Secure Processor Architectures (Pg.. 60)
- 3.3.10 Quantum Architectures (Pg.. 60)
- 3.3.11 Sustainability Issues (Pg.. 61)

3.4 Data Management (DM) (Pg.. 61)

- 3.4.1 The Role of Data and the Data Life Cycle (Pg.. 63)
- 3.4.2 Core Database System Concepts (Pg.. 63)
- 3.4.3 Data Modeling (Pg.. 64)
- 3.4.4 Relational Databases (Pg.. 64)
- 3.4.5 Query Construction (Pg.. 64)
- 3.4.6 Query Processing (Pg.. 65)
- 3.4.7 DBMS Internals (Pg.. 66)
- 3.4.8 NoSQL Systems (Pg.. 66)
- 3.4.9 Data Security and Privacy (Pg.. 66)
- 3.4.10 Data Analytics (Pg.. 67)
- 3.4.11 Distributed Databases/Cloud Computing (Pg.. 67)
- 3.4.12 Semi-structured and Unstructured Databases (Pg.. 67)
- 3.4.13 Society, Ethics, and the Profession (Pg.. 68)

3.5 Foundations of Programming Languages (FPL) (Pg.. 68)

- 3.5.1 Object-Oriented Programming Fundamentals (Pg.. 69)
- 3.5.2 Object-Oriented Programming: Encapsulation, Subtyping, and Reflection (Pg.. 70)
- 3.5.3 Functional Programming (Pg.. 70)
- 3.5.4 Logic Programming (Pg.. 71)
- 3.5.5 Shell Scripting (Pg.. 72)
- 3.5.6 Event-Driven and Reactive Programming (Pg.. 72)
- 3.5.7 Parallel and Distributed Computing (Pg.. 73)
- 3.5.8 Aspect-Oriented Programming (Pg.. 73)
- 3.5.9 Type Systems Fundamentals (Pg.. 74)
- 3.5.10 Type Systems: Polymorphism and Complementary Typing (Pg.. 74)
- 3.5.11 Advanced Type Theory (Pg.. 75)
- 3.5.12 Systems Execution and Memory Model (Pg.. 75)
- 3.5.13 Memory Management (Pg.. 75)
- 3.5.14 Language Translation and Execution (Pg.. 76)
- 3.5.15 Program Abstraction and Representation (Pg.. 76)
- 3.5.16 Syntax Analysis (Pg.. 77)
- 3.5.17 Compiler Semantic Analysis (Pg.. 78)
- 3.5.18 Program Analysis and Analyzers (Pg.. 78)
- 3.5.19 Code Generation (Pg.. 78)
- 3.5.20 Run-time Behavior and Systems (Pg.. 79)
- 3.5.21 Advanced Programming Constructs (Pg.. 80)
- 3.5.22 Language Pragmatics (Pg.. 80)
- 3.5.23 Formal Semantics (Pg.. 81)
- 3.5.24 Formal Development Methodologies (Pg.. 81)
- 3.5.25 Design Principles of Programming Languages (Pg.. 81)
- 3.5.26 Society, Ethics, and the Profession (Pg.. 82)

3.6 Graphics and Interactive Techniques (GIT) (Pg.. 82)
- 3.6.1 Fundamental Concepts of Graphics and Interactive Techniques (Pg.. 83)
- 3.6.2 Visualization (Pg.. 83)
- 3.6.3 Applied Rendering and Techniques (Pg.. 83)
- 3.6.4 Geometric Modeling (Pg.. 84)
- 3.6.5 Shading and Advanced Rendering (Pg.. 84)
- 3.6.6 Computer Animation (Pg.. 84)
- 3.6.7 Simulation (Pg.. 84)
- 3.6.8 Immersion (Pg.. 85)
- 3.6.9 Interaction (Pg.. 85)
- 3.6.10 Image Processing (Pg.. 85)
- 3.6.11 Tangible/Physical Computing (Pg.. 86)
- 3.6.12 Society, Ethics, and the Profession (Pg.. 86)

3.7 Human-Computer Interaction (HCI) (Pg.. 86)
- 3.7.1 Understanding the User: Individual goals and interactions with others (Pg.. 87)
- 3.7.2 Accountability and Responsibility in Design (Pg.. 87)
- 3.7.3 Accessibility and Inclusive Design (Pg.. 88)
- 3.7.4 Evaluating the Design (Pg.. 88)
- 3.7.5 System Design (Pg.. 88)
- 3.7.6 Society, Ethics, and the Profession (Pg.. 89)

3.8 Mathematical and Statistical Foundations (MSF) (Pg.. 89)
- 3.8.1 Discrete Mathematics (Pg.. 89)
- 3.8.2 Advanced Discrete Mathematics (Pg.. 90)
- 3.8.3 Probability (Pg.. 90)
- 3.8.4 Statistics (Pg.. 91)
- 3.8.5 Linear Algebra (Pg.. 91)
- 3.8.6 Calculus (Pg.. 92)

3.9 Networking and Communication (NC) (Pg.. 92)
- 3.9.1 Fundamentals of Networks and Communications (Pg.. 93)
- 3.9.2 Networked Applications (Pg.. 93)
- 3.9.3 Reliability Support (Pg.. 94)
- 3.9.4 Routing and Forwarding (Pg.. 94)
- 3.9.5 Single Hop Communication (Pg.. 94)
- 3.9.6 Network Security (Pg.. 95)
- 3.9.7 Mobility Support (Pg.. 95)
- 3.9.8 Emerging Topics (Pg.. 95)

3.10 Operating Systems (OS) (Pg.. 96)
- 3.10.1 Role and Purpose of Operating Systems (Pg.. 96)
- 3.10.2 Principles of Operating System (Pg.. 97)
- 3.10.3 Concurrency (Pg.. 97)
- 3.10.4 Protection and Safety (Pg.. 98)
- 3.10.5 Scheduling (Pg.. 99)
- 3.10.6 Process Model (Pg.. 99)
- 3.10.7 Memory Management (Pg.. 100)
- 3.10.8 Device management (Pg.. 100)
- 3.10.9 File Systems API and Implementation (Pg.. 101)
- 3.10.10 Advanced File systems (Pg.. 101)
- 3.10.11 Virtualization (Pg.. 102)
- 3.10.12 Real-time and Embedded Systems (Pg.. 102)
- 3.10.13 Fault tolerance (Pg.. 103)
- 3.10.14 Society, Ethics, and the Profession (Pg.. 103)

3.11 Parallel and Distributed Computing (PDC) (Pg.. 104)
- 3.11.1 Programs: Declarative Parallelism (Pg.. 105)
- 3.11.2 Programs: Starting Activities (Pg.. 105)
- 3.11.3 Programs: Execution Properties (Pg.. 105)
- 3.11.4 Programs: Distribution (Pg.. 106)
- 3.11.5 Programs: Implementation Mappings (Pg.. 106)
- 3.11.6 GPU Programming (Pg.. 106)
- 3.11.7 Communication (Pg.. 107)
- 3.11.8 Communication: Properties and Extensions (Pg.. 107)
- 3.11.9 Memory and Consistency (Pg.. 108)
- 3.11.10 Coordination (Pg.. 108)
- 3.11.11 Coordination: Synchronization and Atomicity (Pg.. 109)
- 3.11.12 Coordination: Advanced Properties (Pg.. 109)
- 3.11.13 Evaluation (Pg.. 109)
- 3.11.14 Evaluation: Formal Methods (Pg.. 110)
- 3.11.15 Evaluation: Testing and Measurement (Pg.. 110)
- 3.11.16 Algorithms (Pg.. 111)
- 3.11.17 Algorithms: Application Domains Survey (Pg.. 111)
- 3.11.18 Algorithms: Algorithmic Domains (Pg.. 112)

3.12 Software Development Fundamentals (SDF) (Pg.. 112)
- 3.12.1 Basic Programming Concepts and Constructs (Pg.. 113)
- 3.12.2 Programming Practices: Testing, Debugging, and Documentation (Pg.. 113)
- 3.12.3 Fundamental Data Structures (Pg.. 114)
- 3.12.4 Algorithms (Pg.. 114)
- 3.12.5 Software Development Practices (Pg.. 114)
- 3.12.6 Society, Ethics, and the Profession (Pg.. 115)

3.13 Software Engineering (SE) (Pg.. 115)
- 3.13.1 Team Collaboration and Dynamics (Pg.. 116)
- 3.13.2 Diversity, Stakeholders and Distributed Teams (Pg.. 117)
- 3.13.3 Version Control and CI/CD (Pg.. 117)
- 3.13.4 Development Environment and IDE (Pg.. 118)
- 3.13.5 Requirements Engineering (Pg.. 118)
- 3.13.6 Requirements Evolution and Estimation (Pg.. 119)
- 3.13.7 Principles and Software Architecture (Pg.. 119)
- 3.13.8 API Design and Modeling (Pg.. 120)
- 3.13.9 Software Design Quality and Evaluation (Pg.. 120)
- 3.13.10 Coding Practices (Pg.. 121)
- 3.13.11 Large-Scale Construction and Process (Pg.. 122)
- 3.13.12 Verification and validation Concepts (Pg.. 122)
- 3.13.13 Test Planning and Types (Pg.. 122)
- 3.13.14 Advanced Testing Practices (Pg.. 123)
- 3.13.15 Testing Tools and Analysis (Pg.. 123)
- 3.13.16 Performance Testing and Benchmarking (Pg.. 124)
- 3.13.17 API Compatibility and Versioning (Pg.. 124)
- 3.13.18 Refactoring Techniques (Pg.. 125)
- 3.13.19 Reliability Concepts (Pg.. 125)
- 3.13.20 Reliability Engineering (Pg.. 125)
- 3.13.21 Formal Methods (Pg.. 126)

3.14 Security (SEC) (Pg.. 126)
- 3.14.1 Security Mindset and Principles (Pg.. 127)
- 3.14.2 Access Control and Applications (Pg.. 128)
- 3.14.3 AI and Hardware Security (Pg.. 128)
- 3.14.4 Security Ethics, Law and Society (Pg.. 128)
- 3.14.5 Privacy and Security by Design (Pg.. 129)
- 3.14.6 Injection and Input Validation (Pg.. 129)
- 3.14.7 Memory Safety and Types (Pg.. 130)
- 3.14.8 Information Flow and Non-Interference (Pg.. 130)
- 3.14.9 Malware Analysis and Advanced Security (Pg.. 130)
- 3.14.10 Cryptography Fundamentals (Pg.. 131)
- 3.14.11 Cryptographic Algorithms (Pg.. 131)
- 3.14.12 Protocols, Applications and Advanced Cryptography (Pg.. 132)
- 3.14.13 Threat Analysis and Security Engineering (Pg.. 132)
- 3.14.14 Security Design and Controls Engineering (Pg.. 133)
- 3.14.15 Trusted Computing and Privacy Engineering (Pg.. 133)
- 3.14.16 Digital Forensics Techniques (Pg.. 134)
- 3.14.17 Legal Aspects and Incident Handling (Pg.. 134)
- 3.14.18 Security Governance, Management and Policy (Pg.. 135)
- 3.14.19 Risk and Data Management (Pg.. 135)

3.15 Society, Ethics, and the Profession (SEP) (Pg.. 135)
- 3.15.1 Social Impact and AI (Pg.. 136)
- 3.15.2 Digital Equity and Accessibility (Pg.. 137)
- 3.15.3 Ethical Foundations and Frameworks (Pg.. 137)
- 3.15.4 Ethical Analysis Methods and Tools (Pg.. 138)
- 3.15.5 Professional Values, Nature and Continuing Development (Pg.. 138)
- 3.15.6 Ethics Codes and Legal Accountability (Pg.. 139)
- 3.15.7 Professional Responsibility and Public Policy (Pg.. 139)
- 3.15.8 Ethical Dissent and Cultural Context (Pg.. 140)
- 3.15.9 Workplace Ethics (Pg.. 140)
- 3.15.10 Intellectual Property Foundations (Pg.. 140)
- 3.15.11 Licensing, Plagiarism and Authorship (Pg.. 141)
- 3.15.12 Digital IP Rights and Copyright (Pg.. 141)
- 3.15.13 Open Source, Piracy and DRM (Pg.. 141)
- 3.15.14 Data, Surveillance and Privacy Technology (Pg.. 142)
- 3.15.15 Privacy Philosophical Foundations and Legislation (Pg.. 142)
- 3.15.16 Civil Liberties and Expression (Pg.. 142)
- 3.15.17 Technical and Professional Communication (Pg.. 143)
- 3.15.18 Team Communication and Culture (Pg.. 143)

- 3.15.19 Environmental Footprint of Computing Systems (Pg.. 144)
- 3.15.20 Systemic Effects and Social Context (Pg.. 144)
- 3.15.21 Sustainable Design and Pervasive Computing (Pg.. 144)
- 3.15.22 Foundations and Early History of Computing (Pg.. 145)
- 3.15.23 Modern History: PC, Internet and AI (Pg.. 145)
- 3.15.24 Economic Models and Strategies (Pg.. 146)
- 3.15.25 Automation, AI and Economic Ethics (Pg.. 146)
- 3.15.26 Computer Crimes and Attacks (Pg.. 146)
- 3.15.27 Security Laws, Policies and Responsibility (Pg.. 147)
- 3.15.28 Identity and Diversity in Computing (Pg.. 147)
- 3.15.29 DEIA Historical and Cross-Cultural Context (Pg.. 148)
- 3.15.30 Inclusion and Computing Practices (Pg.. 148)
- 3.15.31 Accessible Design (Pg.. 148)

3.16 System Fundamentals (SF) (Pg.. 149)

- 3.16.1 Hardware Components and Logic (Pg.. 149)
- 3.16.2 Layers, Abstractions and Scalability (Pg.. 150)
- 3.16.3 Digital Systems and Logic (Pg.. 150)
- 3.16.4 State Machines (Pg.. 150)
- 3.16.5 Sequential and Parallel Processing (Pg.. 151)
- 3.16.6 Resource Management (Pg.. 151)
- 3.16.7 Latency, Cache and Memory Hierarchy (Pg.. 152)
- 3.16.8 Virtualization and Isolation (Pg.. 152)
- 3.16.9 Performance Metrics and Benchmarks (Pg.. 153)
- 3.16.10 Analytical Performance Models (Pg.. 153)
- 3.16.11 Events, Tools and Experimentation (Pg.. 153)
- 3.16.12 System Reliability (Pg.. 154)
- 3.16.13 System Security (Pg.. 154)
- 3.16.14 System Design (Pg.. 155)
- 3.16.15 Society, Ethics, and the Profession (Pg.. 155)

3.17 Specialized Platform Development (SPD) (Pg.. 155)

- 3.17.1 Common Aspects: Platforms, APIs and Constraints (Pg.. 156)
- 3.17.2 Common Aspects: Platform Languages and Web Programming Patterns (Pg.. 157)
- 3.17.3 Frontend, Frameworks and Web Standards (Pg.. 157)
- 3.17.4 Cloud Architecture and Data Management (Pg.. 157)
- 3.17.5 Mobile Platforms: Development Fundamentals (Pg.. 158)
- 3.17.6 Mobile Platforms: Architecture and Constraints (Pg.. 158)
- 3.17.7 Mobile Platforms: Capabilities and Asynchronous Computing (Pg.. 158)
- 3.17.8 Robot Platforms (Pg.. 158)
- 3.17.9 Embedded Characteristics and Architectures (Pg.. 159)
- 3.17.10 Systems, Programming and Sensors (Pg.. 159)
- 3.17.11 Real-Time, Memory and Verification (Pg.. 160)
- 3.17.12 Platform History, Types and Characteristics (Pg.. 160)
- 3.17.13 Sensors, Controllers and eSports Ecosystems (Pg.. 161)
- 3.17.14 Architectures and Rendering (Pg.. 161)
- 3.17.15 Algorithms, Media and Virtual Worlds (Pg.. 161)
- 3.17.16 Languages, APIs and Development Tools (Pg.. 161)
- 3.17.17 Game Engines (Pg.. 162)
- 3.17.18 Game Design Foundations (Pg.. 162)
- 3.17.19 Genres, Audiences and Social Reach (Pg.. 162)
- 3.17.20 Interactive Data and Analytics Platforms (Pg.. 163)
- 3.17.21 AI Programming and Quantum Platforms (Pg.. 163)
- 3.17.22 SPD-SEP/Mobile (Pg.. 164)
- 3.17.23 SPD-SEP/Web (Pg.. 164)
- 3.17.24 Intellectual Property in Games (Pg.. 164)
- 3.17.25 Game Interface and Accessibility (Pg.. 165)
- 3.17.26 Health, Safety and Sustainability (Pg.. 165)
- 3.17.27 Representation, Inclusion and Equality (Pg.. 165)
- 3.17.28 SPD-SEP/Robotics (Pg.. 166)
- 3.17.29 SPD-SEP/Interactive (Pg.. 166)

3.1 Artificial Intelligence (AI)

Artificial intelligence (AI) studies problems that are difficult or impractical to solve with traditional algorithmic approaches. These problems are often reminiscent of those considered to require human intelligence, and the resulting AI solution strategies typically generalize over classes of problems. AI techniques are now pervasive in computing, supporting everyday applications such as email, social media, photography, financial markets, and intelligent virtual assistants (e.g., Siri, Alexa). These techniques are also used in the design and analysis of autonomous agents that perceive their environment and interact rationally with it, such as self-driving vehicles and other robots.

Traditionally, AI has included a mix of symbolic and subsymbolic approaches. The solutions it provides rely on a broad set of general and specialized knowledge representation schemes, problem solving mechanisms, and optimization techniques. These approaches deal with perception (e.g., speech recognition, natural language understanding, computer vision), problem solving (e.g., search, planning, optimization), generation (e.g., narrative, conversation, images, models, recommendations), acting (e.g., robotics, task-automation, control), and the architectures needed to support them (e.g., single agents, multi-agent systems). Machine learning may be used within each of these aspects and can even be employed end-to-end across all of them. The study of Artificial Intelligence prepares students to determine when an AI approach is appropriate for a given problem, identify appropriate representations and reasoning mechanisms, implement them, and evaluate them with respect to both performance and their broader societal impact.

Over the past decade, the term ”artificial intelligence” has become commonplace within businesses, news articles, and everyday conversation, driven largely by a series of high-impact machine learning applications. These advances were made possible by the widespread availability of large datasets, increased computational power, and algorithmic improvements. In particular, there has been a shift from engineered representations to representations learned automatically through optimization over large datasets. The resulting advances have put such terms as ”neural networks” and ”deep learning” into everyday vernacular. Businesses now advertise AI-based solutions as value-additions to their

services, so that ”artificial intelligence” is now both a technical term and a marketing buzzword. Other disciplines, such as biology, art, architecture, and finance, increasingly use AI techniques to solve problems within their disciplines.

For the first time in our history, the broader population has access to sophisticated AI-driven tools, including tools to generate essays or poems from a prompt, artwork from a description, and fake photographs or videos depicting real people. AI technology is now in widespread use in stock trading, curating our news and social media feeds, automated evaluation of job applicants, detection of medical conditions, and influencing prison sentencing through recidivism prediction. Consequently, AI technology can have significant societal impacts and ethical considerations that must be understood and considered when developing and applying it.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.1.1 Introduction to AI: Concepts and Intelligent Behavior (Pg.. 38)	2	1
⭐ 3.1.2 Problem Characteristics and Agent Nature (Pg.. 39)	1	1
⭐ 3.1.3 State and Uninformed Search (Pg.. 39)	1	2
⭐ 3.1.4 Heuristic Search and CSP (Pg.. 39)	1	2
⭐ 3.1.5 Adversarial Search and Stochastic Optimization (Pg.. 40)	1	3
⭐ 3.1.6 Knowledge Representation and Probabilistic Reasoning Foundations (Pg.. 40)	1	1
⭐ 3.1.7 Independence, Markov Models and Decision Making (Pg.. 41)	1	1
⭐ 3.1.8 Machine Learning Fundamentals (Pg.. 41)	1	2
⭐ 3.1.9 Machine Learning Data, Representations and Evaluation (Pg.. 41)	1	2
⭐ 3.1.10 Neural Networks and Ethics (Pg.. 42)	1	1
⭐ 3.1.11 Machine Learning Formulation and Evaluation (Pg.. 42)	1	2
⭐ 3.1.12 Deep Learning, RL and Ethics (Pg.. 43)	1	2
⭐ 3.1.13 Neural Network Training and Optimization (Pg.. 43)	1	1
⭐ 3.1.14 Regularization Techniques for Deep Networks (Pg.. 43)	1	1
⭐ 3.1.15 Convolutional and Residual Network Architectures (Pg.. 43)	1	1
⭐ 3.1.16 Transformer Architectures and Generative Models (Pg.. 44)	1	1
⭐ 3.1.17 AI Applications and Societal Impact (Pg.. 44)	2	2
⭐ 3.1.18 Additional AI Applications and Societal Impact (Pg.. 44)	2	2
3.1.19 Logical Representation and Reasoning (Pg.. 45)	Elective	
3.1.20 Probabilistic Representation and Reasoning (Pg.. 45)	Elective	
3.1.21 Planning (Pg.. 46)	Elective	
3.1.22 Agents and Cognitive Systems (Pg.. 46)	Elective	
3.1.23 Agentic Loop Engineering (Pg.. 47)	Elective	
3.1.24 Natural Language Processing (Pg.. 47)	Elective	
3.1.25 Robotics (Pg.. 48)	Elective	
3.1.26 Perception and Computer Vision (Pg.. 49)	Elective	

Table 3.1: List of KUs in the Artificial Intelligence area.

3.1.1 AI/Introduction to AI: Concepts and Intelligent Behavior (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:

Core

- Overview of AI problems, Examples of successful recent AI applications
- Definitions of agents with examples (e.g., reactive, deliberative)
- What is intelligent behavior?: 1. The Turing test and its flaws 2. Multimodal input and output 3. Simulation of intelligent behavior 4. Rational versus non-rational reasoning
- Overview of AI Applications, growth, and impact (economic, societal, ethics)

Non Core

- Philosophical issues
- History of AI

Learning Outcomes:
Core:
1. Describe the Turing test and the "Chinese Room" thought experiment [Explain]
2. Differentiate between optimal reasoning/behavior and human-like reasoning/behavior [Evaluate]
3. Differentiate the terms: AI, machine learning, and deep learning [Explain]

3.1.2 AI/Problem Characteristics and Agent Nature (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Problem characteristics: 1. Fully versus partially observable 2. Single versus multi-agent 3. Deterministic versus stochastic 4. Static versus dynamic 5. Discrete versus continuous
- Nature of agents: 1. Autonomous, semi-autonomous, mixed-initiative autonomy 2. Reflexive, goal-based, and utility-based 3. Decision making under uncertainty and with incomplete information 4. The importance of perception and environmental interactions 5. Learning-based agents 6. Embodied agents: (a) sensors, dynamics, effectors
- Practice identifying problem characteristics in example environments
- Additional depth on nature of agents with examples
- Additional depth on AI Applications, Growth, and Impact (economic, societal, ethics, security)

Learning Outcomes:
Core:
1. Enumerate the characteristics of a specific problem [Apply]

3.1.3 AI/State and Uninformed Search (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core
- State space representation of a problem: 1. Specifying states, goals, and operators 2. Factoring states into representations (hypothesis spaces) 3. Problem solving by graph search: (a) e.g., Graphs as a space, and tree traversals as exploration of that space (b) Dynamic construction of the graph (not given upfront)
- Uninformed graph search for problem solving: 1. Breadth-first search 2. Depth-first search: (a) With iterative deepening 3. Uniform cost search
- Space and time complexities of graph search algorithms
- Bidirectional search

Non Core
- Understanding the search space: 1. Constructing search trees 2. Dynamic search spaces 3. Combinatorial explosion of search space 4. Search space topology (e.g., ridges, saddle points, local minima)

Learning Outcomes:
Core:
1. Design the state space representation for a puzzle (e.g., N-queens or 3-jug problem) [Apply]
2. Select and implement an appropriate uninformed search algorithm for a problem (e.g., tic-tac-toe), and characterize its time and space complexities [Apply]

3.1.4 AI/Heuristic Search and CSP (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core
- Heuristic graph search for problem solving: 1. Heuristic construction and admissibility 2. Hill-climbing 3. Local minima and the search landscape: (a) Local vs global solutions 4. Greedy best-first search 5. A* search
- Beam search
- Implementation of A* search

- Constraint satisfaction

Non Core

- Variations on A* (IDA*, SMA*, RBFS)

Learning Outcomes:

Core:

1. Select and implement an appropriate informed search algorithm for a problem after designing a helpful heuristic function (e.g., a robot navigating a 2D gridworld) [Apply]
2. Evaluate whether a heuristic for a given problem is admissible/can guarantee an optimal solution [Evaluate]
3. Design and implement A*/beam search to solve a problem, and compare it against other search algorithms in terms of the solution cost, number of nodes expanded, etc [Apply]
4. Compare and contrast various heuristic searches vis-a-vis applicability to a given problem [Evaluate]
5. Model a logic or Sudoku puzzle as a constraint satisfaction problem, solve it with backtrack search, and determine how much arc consistency can reduce the search space [Apply]

3.1.5 AI/Adversarial Search and Stochastic Optimization (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:

Core

- Two-player adversarial games: 1. Minimax search 2. Alpha-beta pruning: (a) Ply cutoff

Non Core

- Local search
- Tabu search
- Two-player adversarial games: 1. The horizon effect 2. Opening playbooks/endgame solutions 3. What it means to ”solve” a game (e.g., checkers)
- Implementation of minimax search, beam search
- Expectimax search (MDP-solving) and chance nodes
- Stochastic search: 1. Simulated annealing 2. Genetic algorithms 3. Monte-Carlo tree search

Learning Outcomes:

Core:

1. Apply minimax search in a two-player adversarial game (e.g., connect four), using heuristic evaluation at a particular depth to compute the scores to back up [Apply]
2. Apply minimax search with alpha-beta pruning to prune search space in a two-player adversarial game (e.g., connect four) [Apply]
3. Compare and contrast genetic algorithms with classic search techniques, explaining when it is most appropriate to use a genetic algorithm to learn a model versus other forms of optimization (e.g., gradient descent) [Evaluate]

NonCore:

4. Design and implement a genetic algorithm solution to a problem [Apply]
5. Design and implement a simulated annealing schedule to avoid local minima in a problem [Apply]

3.1.6 AI/Knowledge Representation and Probabilistic Reasoning Foundations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Types of representations: 1. Symbolic, logical: (a) Creating a representation from a natural language problem statement 2. Learned subsymbolic representations 3. Graphical models (e.g., naive Bayes, Bayesian network)
- Review of probabilistic reasoning, Bayes theorem
- Bayesian reasoning: 1. Bayesian inference
- Random variables and probability distributions: 1. Axioms of probability 2. Probabilistic inference 3. Bayes' Rule (derivation) 4. Bayesian inference (more complex examples)

Learning Outcomes:
Core:

1. Given a natural language problem statement, encode it as a symbolic or logical representation [Apply]
2. Compute a probabilistic inference in a real-world problem using Bayes' theorem to determine the probability of a hypothesis given evidence [Apply]
3. Apply Bayes' rule to determine the probability of a hypothesis given evidence [Apply]
4. Compute the probability of outcomes and test whether outcomes are independent [Apply]

3.1.7 AI/Independence, Markov Models and Decision Making (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Independence
- Conditional Independence
- Markov chains and Markov models
- Utility and decision making

Learning Outcomes:
Core:

1. Explain how we can make decisions under uncertainty, using concepts such as Bayes theorem and utility [Explain]

3.1.8 AI/Machine Learning Fundamentals (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Definition and examples of a broad variety of machine learning tasks: 1. Supervised learning: (a) Classification (b) Regression 2. Reinforcement learning 3. Unsupervised learning: (a) Clustering
- Fundamental ideas: 1. No free lunch theorem: no one learner can solve all problems; representational design decisions have consequences. 2. Sources of error and undecidability in machine learning
- A simple statistical-based supervised learning such as linear regression or decision trees: 1. Focus on how they work without going into mathematical or optimization details; enough to understand and use existing implementations correctly
- The overfitting problem/controlling solution complexity (regularization, pruning - intuition only): 1. The bias (underfitting) - variance (overfitting) tradeoff

Learning Outcomes:
Core:

1. Describe the differences among the three main styles of learning (supervised, reinforcement, and unsupervised) and determine which is appropriate to a particular problem domain [Explain]
2. Differentiate the terms of AI, machine learning, and deep learning [Evaluate]
3. Frame an application as a classification problem, including the available input features and output to be predicted (e.g., identifying alphabetic characters from pixel grid input) [Apply]
4. Identify overfitting in the context of a problem and learning curves and describe solutions to overfitting [Evaluate]
5. Explain how machine learning works as an optimization/search process [Explain]

3.1.9 AI/Machine Learning Data, Representations and Evaluation (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Working with Data: 1. Data preprocessing: (a) Importance and pitfalls of preprocessing choices 2. Handling missing values (imputing, flag-as-missing): (a) Implications of imputing vs flag-as-missing 3. Encoding categorical variables, encoding real-valued data 4. Normalization/standardization 5. Emphasis on real data, not textbook examples
- Representations: 1. Hypothesis spaces and complexity 2. Simple basis feature expansion, such as squaring univariate features 3. Learned feature representations
- Machine learning evaluation: 1. Separation of train, validation, and test sets 2. Performance metrics for classifiers 3. Estimation of test performance on held-out data 4. Tuning the parameters of a machine learning model with a validation set 5. Importance of understanding what a model is doing, where its pitfalls/shortcomings are, and the implications of its decisions

Learning Outcomes:

Core:

1. Apply two or more simple statistical learning algorithms to a classification task and measure the classifiers' accuracy [Apply]
2. Implement a statistical learning algorithm and the corresponding optimization process to train the classifier and obtain a prediction on new data [Apply]
3. Explain proper ML evaluation procedures, including the differences between training and testing performance, and what can go wrong with the evaluation process leading to inaccurate reporting of ML performance [Explain]
4. Compare two machine learning algorithms on a dataset, implementing the data preprocessing and evaluation methodology (e.g., metrics and handling of train/test splits) from scratch [Apply]

3.1.10 AI/Neural Networks and Ethics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Basic neural networks: 1. Fundamentals of understanding how neural networks work and their training process, without details of the calculations 2. Basic introduction to generative neural networks (e.g., large language models)
- Ethics for Machine Learning: 1. Focus on real data, real scenarios, and case studies 2. Dataset/algorithmic/evaluation bias and unintended consequences

Learning Outcomes:

Core:

1. Describe the neural network training process and resulting learned representations [Explain]

3.1.11 AI/Machine Learning Formulation and Evaluation (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:

Core

- Formulation of simple machine learning as an optimization problem, such as least squares linear regression or logistic regression: 1. Objective function 2. Gradient descent 3. Regularization to avoid overfitting (mathematical formulation)
- Ensembles of models: 1. Simple weighted majority combination
- Performance evaluation: 1. Other metrics for classification (e.g., error, precision, recall) 2. Performance metrics for regressors 3. Confusion matrix 4. Cross-validation: (a) Parameter tuning (grid/random search, via cross-validation)

Learning Outcomes:

Core:

1. Compare and contrast several learning techniques (e.g., decision trees, logistic regression, naive Bayes, neural networks, and belief networks), providing examples of when each strategy is superior [Compare]
2. Evaluate the performance of a simple learning system on a real-world dataset [Evaluate]

NonCore:

3. Characterize the state of the art in learning theory, including its achievements and shortcomings [Explain]

4. Explain the problem of overfitting, along with techniques for detecting and managing the problem [Explain]
5. Explain the triple tradeoff among the size of a hypothesis space, the size of the training set, and performance accuracy [Explain]

3.1.12 AI/Deep Learning, RL and Ethics (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Deep learning: 1. Deep feed-forward networks (intuition only, no mathematics) 2. Convolutional neural networks (intuition only, no mathematics) 3. Visualization of learned feature representations from deep nets 4. Other architectures (generative NN, recurrent NN, transformers, etc.)
- Overview of reinforcement learning methods
- Two or more applications of machine learning algorithms: 1. E.g., medicine and health, economics, vision, natural language, robotics, game play
- Ethics for Machine Learning: 1. Continued focus on real data, real scenarios, and case studies 2. Privacy 3. Fairness 4. Intellectual property 5. Explainability

Learning Outcomes:
Core:

1. Visualize the training progress of a neural network through learning curves in a well-established toolkit (e.g., TensorBoard) and visualize the learned features of the network [Apply]

NonCore:

2. Given a real-world application of machine learning, describe ethical issues regarding the choices of data, preprocessing steps, algorithm selection, and visualization/presentation of results [Apply]

3.1.13 AI/Neural Network Training and Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Modern backpropagation: computational graphs and automatic differentiation
- Advanced optimizers: momentum, RMSProp, Adam, and learning-rate schedules

Learning Outcomes:
Core:

1. Implement a deep neural network from scratch, including its backpropagation pass [Apply]
2. Select and tune an optimizer and its learning-rate schedule for training stability [Apply]

3.1.14 AI/Regularization Techniques for Deep Networks (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Dropout regularization
- Batch normalization

Learning Outcomes:
Core:

1. Diagnose overfitting in a training run using regularization diagnostics [Analyze]
2. Apply dropout and batch normalization to improve a model's generalization [Apply]

3.1.15 AI/Convolutional and Residual Network Architectures (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Deep convolutional neural network architectures

- Residual connections and ResNets

Learning Outcomes:

Core:

1. Design a deep CNN architecture for a given vision task [Apply]
2. Adapt a residual architecture to a specific vision domain [Apply]

3.1.16 AI/Transformer Architectures and Generative Models (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Transformers and the attention mechanism
- Generative models: variational autoencoders (VAEs) and generative adversarial networks (GANs)

Learning Outcomes:

Core:

1. Train a transformer model on a real dataset [Apply]
2. Analyze and contrast VAE and GAN generative-model architectures [Analyze]

3.1.17 AI/AI Applications and Societal Impact (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:

Core

- At least one application of AI to a specific problem and field, such as medicine, health, sustainability, social media, economics, education, robotics, etc. (choose at least one for the CS Core): 1. Formulating and evaluating a specific application as an AI problem: (a) How to deal with underspecified or ill-posed problems 2. Data availability/scarcity and cleanliness: (a) Basic data cleaning and preprocessing (b) Data set bias 3. Algorithmic bias 4. Evaluation bias 5. Assessment of societal implications of the application
- Deployed deep generative models: 1. High-level overview of deep image generative models (e.g., as of 2023, DALL-E, Midjourney, Stable Diffusion, etc.), their uses, and their shortcomings/pitfalls. 2. High-level overview of large language models (e.g., as of 2023, ChatGPT, Bard, etc.), their uses, and their shortcomings/pitfalls.
- Overview of societal impact of AI: 1. Ethics 2. Fairness 3. Trust/explainability 4. Privacy and usage of training data 5. Human autonomy and oversight/regulations/legal requirements 6. Sustainability

Learning Outcomes:

Core:

1. Given a real-world application domain and problem, formulate an AI solution to it, identifying proper data/input, preprocessing, representations, AI techniques, and evaluation metrics/methodology [Apply]

3.1.18 AI/Additional AI Applications and Societal Impact (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:

Core

- One or more additional applications of AI to a broad set of problems and diverse fields, such as medicine, health, sustainability, social media, economics, education, robotics, etc. (choose a different area from that chosen for the CS Core): 1. Formulating and evaluating a specific application as an AI problem: (a) How to deal with underspecified or ill-posed problems 2. Data availability/scarcity and cleanliness: (a) Basic data cleaning and preprocessing (b) Data set bias 3. Algorithmic bias 4. Evaluation bias 5. Assessment of societal implications of the application
- Additional depth on deployed deep generative models: 1. Introduction to how deep image generative models work, (e.g., as of 2023, DALL-E, Midjourney, Stable Diffusion) including discussion

of attention 2. Introduction to how large language models work, (e.g., as of 2023, ChatGPT, Bard) including discussion of attention 3. Idea of foundational models, how to use them, and the benefits/issues with training them from big data
- Analysis and discussion of the societal impact of AI: 1. Ethics 2. Fairness 3. Trust/explainability 4. Privacy and usage of training data 5. Human autonomy and oversight/regulations/legal requirements 6. Sustainability

Learning Outcomes:
Core:

1. Analyze the societal impact of one or more specific real-world AI applications, identifying issues regarding ethics, fairness, bias, trust, and explainability [Evaluate]
2. Describe some of the failure modes of current deep generative models for language or images, and how this could affect their use in an application [Describe]

3.1.19 AI/Logical Representation and Reasoning

Topics:
Non Core

- Review of propositional and predicate logic
- Resolution and theorem proving (propositional logic only): 1. Forward chaining, backward chaining
- Knowledge representation issues: 1. Description logics 2. Ontology engineering
- Semantic web
- Non-monotonic reasoning (e.g., non-classical logics, default reasoning)
- Argumentation
- Reasoning about action and change (e.g., situation and event calculus)
- Temporal and spatial reasoning
- Logic programming: 1. Prolog, Answer Set Programming
- Rule-based Expert Systems
- Semantic networks
- Model-based and Case-based reasoning

Learning Outcomes:
NonCore:

1. Translate a natural language (e.g., English) sentence into a predicate logic statement [Apply]
2. Convert a logic statement into causal form [Apply]
3. Apply resolution to a set of logic statements to answer a query [Apply]
4. Compare and contrast the most common models used for structured knowledge representation, highlighting their strengths and weaknesses [Evaluate]
5. Identify the components of non-monotonic reasoning and its usefulness as a representational mechanism for belief systems [Evaluate]
6. Compare and contrast the basic techniques for representing uncertainty [Evaluate]
7. Compare and contrast the basic techniques for qualitative representation [Evaluate]
8. Apply situation and event calculus to problems of action and change [Apply]
9. Explain the distinction between temporal and spatial reasoning, and how they interrelate [Explain]
10. Explain the difference between rule-based, case-based, and model-based reasoning techniques [Explain]
11. Define the concept of a planning system and how it differs from classical search techniques [Define]
12. Describe the differences between planning as search, operator-based planning, and propositional planning, providing examples of domains where each is most applicable [Describe]
13. Explain the distinction between monotonic and non-monotonic inference [Explain]

3.1.20 AI/Probabilistic Representation and Reasoning

Topics:
Non Core

- Conditional Independence review
- Knowledge representations: 1. Bayesian Networks: (a) Exact inference and its complexity (b) Markov blankets and d-separation (c) Randomized sampling (Monte Carlo) methods (e.g., Gibbs sampling) 2. Markov Networks 3. Relational probability models 4. Hidden Markov Models

- Decision Theory: 1. Preferences and utility functions 2. Maximizing expected utility 3. Game theory

Learning Outcomes:
NonCore:

1. Compute the probability of a hypothesis given the evidence in a Bayesian network [Apply]
2. Explain how conditional independence assertions allow for greater efficiency of probabilistic systems [Explain]
3. Identify examples of knowledge representations for reasoning under uncertainty [List]
4. State the complexity of exact inference. Identify methods for approximate inference [State]
5. Design and implement at least one knowledge representation for reasoning under uncertainty [Apply]
6. Describe the complexities of temporal probabilistic reasoning [Describe]
7. Design and implement an HMM as one example of a temporal probabilistic system [Apply]
8. Describe the relationship between preferences and utility functions [Describe]
9. Explain how utility functions and probabilistic reasoning can be combined to make rational decisions [Explain]

3.1.21 AI/Planning

Topics:
Non Core

- Review of propositional and first-order logic
- Planning operators and state representations
- Total order planning
- Partial-order planning
- Plan graphs and GraphPlan
- Hierarchical planning
- Planning languages and representations: 1. PDDL
- Multi-agent planning
- MDP-based planning
- Interconnecting planning, execution, and dynamic replanning: 1. Conditional planning 2. Continuous planning 3. Probabilistic planning

Learning Outcomes:
NonCore:

1. Construct the state representation, goal, and operators for a given planning problem [Apply]
2. Encode a planning problem in PDDL and use a planner to solve it [Apply]
3. Given a set of operators, initial state, and goal state, draw the partial-order planning graph and include ordering constraints to resolve all conflicts [Apply]
4. Construct the complete planning graph for GraphPlan to solve a given problem [Apply]

3.1.22 AI/Agents and Cognitive Systems

Topics:
Non Core

- Agent architectures (e.g., reactive, layered, cognitive)
- Agent theory (including mathematical formalisms)
- Rationality, Game Theory: 1. Decision-theoretic agents 2. Markov decision processes (MDP) 3. Bandit algorithms
- Software agents, personal assistants, and information access: 1. Collaborative agents 2. Information-gathering agents 3. Believable agents (synthetic characters, modeling emotions in agents)
- Learning agents

- Cognitive systems: 1. Cognitive architectures (e.g., ACT-R, SOAR, ICARUS, FORR) 2. Capabilities (e.g., perception, decision making, prediction, knowledge maintenance) 3. Knowledge representation, organization, utilization, acquisition, and refinement 4. Applications and evaluation of cognitive systems
- Multi-agent systems: 1. Collaborating agents 2. Agent teams 3. Competitive agents (e.g., auctions, voting) 4. Swarm systems and biologically inspired models 5. Multi-agent learning
- Human-agent interaction: 1. Communication methodologies (verbal and non-verbal) 2. Practical issues 3. Applications: (a) Trading agents, supply chain management (b) Ethical issues of AI interactions with humans (c) Regulation and legal requirements of AI systems for interacting with humans

Learning Outcomes:
NonCore:

1. Characterize and contrast the standard agent architectures [Evaluate]
2. Describe the applications of agent theory to domains such as software agents, personal assistants, and believable agents, and discuss associated ethical implications [Describe]
3. Describe the primary paradigms used by learning agents [Describe]
4. Demonstrate using appropriate examples how multi-agent systems support agent interaction [Demonstrate]
5. Construct an intelligent agent using a well-established cognitive architecture (ACT-R, SOAR) for solving a specific problem [Apply]

3.1.23 AI/Agentic Loop Engineering

The discipline of designing the iterative observe-act-check cycle an AI agent runs through, as distinct from engineering a single prompt. Includes loop patterns with multiple specialized agents (builder/critic) that refine a result against a predefined quality bar until it is approved.

Topics:
Non Core

- Loop engineering as a discipline: designing the iterative observe-act-check cycle an agent runs through, as distinct from writing a single prompt
- The Gauntlet Loop pattern (Matt Shumer, 2026): a lead agent breaks a goal and a quality bar into the smallest independently judgeable pieces, then fans out builder and critic sub-agents with fresh context
- Builder/critic separation: an agent that created a result is usually worse at objectively judging its own quality than at explaining it, hence the need for an independent critic
- Blind A/B comparison of the result against a predefined quality bar, and loop stopping criteria (critic approval or an iteration budget)
- Relationship to other agentic loop patterns: ReAct (reason-act), self-reflection, and multi-agent debate

Learning Outcomes:
NonCore:

1. Explain how loop engineering differs from engineering a single prompt [Explain]
2. Design a Gauntlet-Loop-style loop (builder agent + independent critic agent + blind comparison against a quality bar) to iteratively refine an AI-generated result [Design]
3. Justify why separating the builder and critic roles improves evaluation quality compared to having the same agent judge its own result [Justify]
4. Define appropriate stopping criteria for an iterative agentic loop given a quality bar and an iteration budget [Define]

3.1.24 AI/Natural Language Processing

Topics:
Non Core

- Deterministic and stochastic grammars
- Parsing algorithms: 1. CFGs and chart parsers (e.g., CYK) 2. Probabilistic CFGs and weighted CYK

- Representing meaning/Semantics: 1. Logic-based knowledge representations 2. Semantic roles 3. Temporal representations 4. Beliefs, desires, and intentions
- Corpus-based methods
- N-grams and HMMs
- Smoothing and backoff
- Examples of use: POS tagging and morphology
- Information retrieval: 1. Vector space model: (a) TF & IDF 2. Precision and recall
- Information extraction
- Language translation
- Text classification, categorization: 1. Bag of words model
- Deep learning for NLP: 1. RNNs 2. Transformers 3. Multi-modal embeddings (e.g., images + text) 4. Generative language models

Learning Outcomes:
NonCore:

1. Define and contrast deterministic and stochastic grammars, providing examples to show the adequacy of each [Define]
2. Simulate, apply, or implement classic and stochastic algorithms for parsing natural language [Apply]
3. Identify the challenges of representing meaning [Analyze]
4. List the advantages of using standard corpora. Identify examples of current corpora for a variety of NLP tasks [List]
5. Identify techniques for information retrieval, language translation, and text classification [Analyze]
6. Implement a TF/IDF transform, use it to extract features from a corpus, and train an off-the-shelf machine learning algorithm using those features to do text classification [Apply]

3.1.25 AI/Robotics

Topics:
Non Core

- Overview: problems and progress: 1. State-of-the-art robot systems, including their sensors and an overview of their sensor processing 2. Robot control architectures, e.g., deliberative vs reactive control and Braitenberg vehicles 3. World modeling and world models 4. Inherent uncertainty in sensing and in control
- Sensors and effectors: 1. Sensors: e.g., LIDAR, sonar, vision, depth, stereoscopic, event cameras, microphones, haptics 2. Effectors: e.g., wheels, arms, grippers
- Coordinate frames, translation, and rotation (2D and 3D)
- Configuration space and environmental maps
- Interpreting uncertain sensor data
- Localization and mapping
- Navigation and control
- Forward and inverse kinematics
- Motion path planning and trajectory optimization
- Manipulation and grasping
- Joint control and dynamics
- Vision-based control
- Multiple-robot coordination and collaboration
- Human-robot interaction: 1. Shared workspaces 2. Human-robot teaming and physical HRI 3. Social assistive robots 4. Motion/task/goal prediction 5. Collaboration and communication (explicit vs implicit, verbal or symbolic vs non-verbal or visual) 6. Trust
- Applications and Societal, Economic, and Ethical Issues: 1. Societal, economic, right-to-work implications 2. Ethical and privacy implications of robotic applications 3. Liability in autonomous robotics 4. Autonomous weapons and ethics 5. Human oversight and control

Learning Outcomes:
NonCore:

1. List capabilities and limitations of today's state-of-the-art robot systems, including their sensors

and the crucial sensor processing that informs those systems [List]
2. Integrate sensors, actuators, and software into a robot designed to undertake a specific task [Apply]
3. Program a robot to accomplish simple tasks using deliberative, reactive, and/or hybrid control architectures [Apply]
4. Implement fundamental motion planning algorithms within a robot configuration space [Apply]
5. Characterize the uncertainties associated with common robot sensors and actuators; articulate strategies for mitigating these uncertainties [Characterize]
6. List the differences among robots' representations of their external environment, including their strengths and shortcomings [List]
7. Compare and contrast at least three strategies for robot navigation within known and/or unknown environments, including their strengths and shortcomings [Compare]
8. Describe at least one approach for coordinating the actions and sensing of several robots to accomplish a single task [Describe]
9. Compare and contrast a multi-robot coordination and a human-robot collaboration approach and attribute their differences to differences between the problem settings [Compare]
10. Analyze the societal, economic, and ethical issues of a real-world robotics application [Analyze]

3.1.26 AI/Perception and Computer Vision

Topics:
Non Core

- Computer vision: 1. Image acquisition, representation, processing, and properties 2. Shape representation, object recognition, and segmentation 3. Motion analysis 4. Generative models
- Audio and speech recognition
- Touch and proprioception
- Other modalities (e.g., olfaction)
- Modularity in recognition
- Approaches to pattern recognition: 1. Classification algorithms and measures of classification quality 2. Statistical techniques 3. Deep learning techniques

Learning Outcomes:
NonCore:

1. Summarize the importance of image and object recognition in AI and indicate several significant applications of this technology [Summarize]
2. List at least three image-segmentation approaches, such as thresholding, edge-based and region-based algorithms, along with their defining characteristics, strengths, and weaknesses [List]
3. Implement 2d object recognition based on contour-based and/or region-based shape representations [Apply]
4. Distinguish the goals of sound-recognition, speech-recognition, and speaker-recognition and identify how the raw audio signal will be handled differently in each of these cases [Distinguish]
5. Provide at least two examples of a transformation of a data source from one sensory domain to another, e.g., tactile data interpreted as single-band 2d images [Generate]
6. Implement a feature-extraction algorithm on real data, e.g., an edge or corner detector for images or vectors of Fourier coefficients describing a short slice of audio signal [Apply]
7. Implement an algorithm combining features into higher-level percepts, e.g., a contour or polygon from visual primitives or phoneme hypotheses from an audio signal [Apply]
8. Implement a classification algorithm that segments input percepts into output categories and quantitatively evaluates the resulting classification [Apply]
9. Evaluate the performance of the underlying feature-extraction, relative to at least one alternative possible approach (whether implemented or not) in its contribution to the classification task (8), above [Evaluate]
10. Describe at least three classification approaches, their prerequisites for applicability, their strengths, and their shortcomings [Describe]
11. Implement and evaluate a deep learning solution to problems in computer vision, such as object or scene recognition [Apply]

3.2 Algorithmic Foundations (AL)

Algorithms and data structures are fundamental to computer science, since every theoretical computation and applied program consists of algorithms that operate on data elements possessing some underlying structure. Selecting appropriate computational solutions to real-world problems benefits from understanding the theoretical and practical capabilities and limitations of available algorithms and paradigms, including their impact on the environment and society. Moreover, this understanding provides insight into the intrinsic nature of computation, computational problems, and computational problem-solving as well as possible solution techniques independent of programming language, programming paradigm, computer hardware, or other implementation aspects.

This knowledge area focuses on the nature of computation including the concepts and skills required to design and analyze algorithms for solving real-world computational problems. It complements the implementation of algorithms and data structures found in the Software Development Foundations (SDF) knowledge area. As algorithms and data structures are essential in all advanced areas of computer science, this area provides the algorithmic foundations that every computer science graduate is expected to know. Exposure to the breadth of these foundational AL topics is designed to provide students with the basis for studying these topics in more depth, for studying additional computation and algorithm topics, and for learning advanced algorithms across a variety of CS knowledge areas and CS+X disciplines.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.2.1 Foundational Data Structures (Pg.. 50)	5	3
⭐ 3.2.2 Foundational Algorithms (Pg.. 51)	4	2
⭐ 3.2.3 Advanced Algorithms (Pg.. 51)	3	2
⭐ 3.2.4 Algorithmic Strategies (Pg.. 52)	6	
⭐ 3.2.5 Complexity Analysis Framework (Pg.. 52)	2	1
⭐ 3.2.6 Asymptotic Notation and Complexity Classes (Pg.. 53)	1	1
⭐ 3.2.7 Complexity Analysis: Recursion, Amortization, and Tight Bounds (Pg.. 53)	2	1
⭐ 3.2.8 Computational Complexity Theory (Pg.. 53)	2	1
⭐ 3.2.9 Formal Languages and Automata (Pg.. 54)	4	10
⭐ 3.2.10 Computability and Decidability (Pg.. 54)	6	14
3.2.11 Society, Ethics, and the Profession (Pg.. 55)	Elective	

Table 3.2: List of KUs in the Algorithmic Foundations area.

3.2.1 AL/Foundational Data Structures (Core Tier1: 5 hrs, Core Tier2: 3 hrs)

Topics:

Core

- Abstract Data Type (ADT) and operations on an ADT: 1. Dictionary operations (insert, delete, find)
- Arrays: 1. Numeric vs non-numeric, character strings 2. Single (vector) vs multidimensional (matrix)
- Records/Structs/Tuples and Objects
- Linked lists (for historical reasons): 1. Single vs Double and Linear vs Circular
- Stacks
- Queues and deques: 1. Heap-based priority queue
- Hash tables/maps: 1. Collision resolution and complexity (e.g., probing, chaining, rehash)
- Trees: 1. Binary, n-ary, and search trees 2. Balanced (e.g., AVL, Red-Black, Heap)
- Sets

Learning Outcomes:

Core:

1. For each ADT/Data-Structure in this unit: (a) Explain its definition, properties, representation(s), and associated ADT operations. (b) Explain step-by-step how the ADT operations associated with the data structure transform it. [Explain]

2. Explain how collision avoidance and collision resolution is handled in hash tables [Explain]
3. Explain the heap property and the use of heaps as an implementation of a priority queue [Explain]

3.2.2 AL/Foundational Algorithms (Core Tier1: 4 hrs, Core Tier2: 2 hrs)

Topics:
Core

- Graphs (e.g., [un]directed, [a]cyclic, [un]connected, and [un]weighted): 1. Graph representation: adjacency list vs matrix
- Search algorithms: 1. $O(n)$ complexity (e.g., linear/sequential array/list search) 2. $O(\log_2 n)$ complexity (e.g., binary search) 3. $O(\log_b n)$ complexity (e.g., uninformed depth/breadth-first tree search)
- Sorting algorithms (e.g., stable, unstable): 1. $O(n^2)$ complexity (e.g., insertion, selection) 2. $O(n \log n)$ complexity (e.g., quicksort, merge, timsort)
- Graph algorithms: 1. Shortest path (e.g., Dijkstra's, Floyd's) 2. Minimal spanning tree (e.g., Prim's, Kruskal's)
- Sorting algorithms: 1. $O(n \log n)$ complexity heapsort 2. Pseudo $O(n)$ complexity (e.g., bucket, counting, radix)
- Graph algorithms: 1. Transitive closure (e.g., Warshall's) 2. Topological sort
- Matching: 1. Efficient string matching (e.g., Boyer-Moore, Knuth-Morris-Pratt) 2. Longest common subsequence matching 3. Regular expression matching

Learning Outcomes:
Core:

1. For each algorithm in this unit explain step-by-step how the algorithm operates [Explain]
2. For each algorithmic approach (e.g., sorting) in this unit apply a prototypical example of the approach (e.g., merge sort) [Apply]
3. Given requirements for a problem, develop multiple solutions using various data structures and algorithms. Subsequently, evaluate the suitability, strengths, and weaknesses selecting an approach that best satisfies the requirements [Create]
4. Explain factors beyond computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data [Explain]
5. For each of the algorithms and algorithmic approaches in the KA Core topics: (a) Explain a prototypical example of the algorithm, and (b) Explain step-by-step how the algorithm operates. [Explain]

3.2.3 AL/Advanced Algorithms (Core Tier1: 3 hrs, Core Tier2: 2 hrs)

Topics:
Non Core

- Cryptography algorithms (e.g., SHA-256)
- Parallel algorithms
- Consensus algorithms (e.g., Blockchain): 1. Proof of work vs proof of stake
- Quantum computing algorithms: 1. Oracle-based (e.g., Deutsch-Jozsa, Bernstein-Vazirani, Simon) 2. Superpolynomial speed-up via QFT (e.g., Shor's) 3. Polynomial speed-up via amplitude amplification (e.g., Grover's)
- Fast-Fourier Transform (FFT) algorithm
- Differential evolution algorithm

Learning Outcomes:
NonCore:

1. An appreciation of quantum computation and its application to certain problems [Explain]

3.2.4 AL/Algorithmic Strategies (Core Tier1: 6 hrs)

Topics:
Core

- Paradigms: 1. Brute-Force (e.g., linear search, selection sort, traveling salesperson, knapsack) 2. Decrease-and-Conquer: (a) By a Constant (e.g., insertion sort, topological sort) (b) By a Constant Factor (e.g., binary search) (c) By a Variable Size (e.g., Euclid's) 3. Divide-and-Conquer (e.g., binary search, quicksort, mergesort, Strassen's) 4. Greedy (e.g., Dijkstra's, Kruskal's, Knapsack) 5. Transform-and-Conquer: (a) Instance simplification (e.g., find duplicates via list presort) (b) Representation change (e.g., heapsort) (c) Problem reduction (e.g., least-common-multiple, linear programming) (d) Dynamic programming (e.g., Floyd's, Marshall, Bellman-Ford) 6. Space vs time tradeoffs (e.g., hashing)
- Handling exponential growth (e.g., heuristic A*, branch-and-bound, backtracking)
- Iteration vs recursion (e.g., factorial, tree search)
- Paradigms: 1. Approximation algorithms 2. Iterative improvement (e.g., Ford-Fulkerson, simplex) 3. Randomized/Stochastic algorithms (e.g., max-cut, balls and bins)

Non Core

- Quantum computing

Learning Outcomes:
Core:

1. For each of the paradigms in this unit: (a) Explain its definitional characteristics (b) Explain an example that demonstrates the paradigm including how this example satisfies the paradigm's characteristics. [Explain]
2. For each of the algorithms in the Foundational Data Structures, Foundational Algorithms, Advanced Algorithms unit, explain the paradigm used by the algorithm and how it exemplifies this paradigm [Explain]
3. Given an algorithm, explain the paradigm used by the algorithm and how it exemplifies this paradigm [Explain]
4. Give a real-world problem, evaluate appropriate algorithmic paradigms and algorithms from these paradigms that address the problem including evaluating the tradeoffs among the paradigms and algorithms selected [Evaluate]
5. Give examples of iterative and recursive algorithms that solve the same problem, explain the benefits and disadvantages of each approach [Explain]
6. Evaluate whether a greedy approach leads to an optimal solution [Evaluate]
7. Explain various approaches for addressing computational problems whose algorithmic solutions are exponential [Explain]

3.2.5 AL/Complexity Analysis Framework (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:
Core

- Complexity Analysis Framework: 1. Best, average, and worst-case performance of an algorithm 2. Empirical and relative (Order of Growth) measurements 3. Input size and primitive operations 4. Time and space efficiency
- Empirical measurements of performance
- Time and space tradeoffs in algorithms

Learning Outcomes:
Core:

1. Prepare a presentation that explains to first year students the basic concepts of algorithmic complexity including best, average, and worst-case algorithm behavior, Big-O, Omega, and Theta notations, complexity classes, time and space tradeoffs, empirical measurement, and impact on practical problems [Explain]
2. For each algorithm in the Foundational Data Structures, Foundational Algorithms, Advanced Algorithms unit, explain its runtime complexity class and why it belongs to this class [Explain]
3. Informally evaluate the foundational complexity class of simple algorithms [Evaluate]
4. Develop empirical studies to determine and validate hypotheses about the runtime complexity of various algorithms by running algorithms on input of various sizes and comparing actual

performance to the theoretical analysis [Create]
5. Explain examples that illustrate time-space tradeoffs of algorithms [Explain]
6. Explain how tree balance affects the efficiency of binary search tree operations [Explain]

3.2.6 AL/Asymptotic Notation and Complexity Classes (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Asymptotic complexity analysis (average and worst-case bounds): 1. Big-O, Big-Omega, and Big-Theta formal notations 2. Foundational Complexity Classes and Representative Examples/Problems: (a) $O(1)$ Constant (e.g., array access) (b) $O(\log_2 n)$ Logarithmic (e.g., binary search) (c) $O(n)$ Linear (e.g., linear search) (d) $O(n \log_2 n)$ Log Linear (e.g., mergesort) (e) $O(n^2)$ Quadratic (e.g., selection sort) (f) $O(n^c)$ Polynomial (e.g., $O(n^3)$ Gaussian elimination) (g) $O(2^n)$ Exponential (e.g., Knapsack, Satisfiability (SAT), Traveling Sales-Person (TSP), all subsets) (h) $O(n!)$ Factorial (e.g., Hamiltonian circuit, all permutations)

Learning Outcomes:
Core:

1. Using examples, explain each of the foundational complexity classes in this unit [Explain]
2. For each foundational complexity class in this unit, explain an algorithm that demonstrates the associated runtime complexity [Explain]
3. Explain to a non-technical audience the significance of tractable versus intractable algorithms using an intuitive explanation of Big-O complexity [Explain]
4. Apply Big-O notation to give upper case bounds on time/space complexity of algorithms [Apply]

3.2.7 AL/Complexity Analysis: Recursion, Amortization, and Tight Bounds (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:
Core

- Little-o, Little-Omega, and Little Theta notations
- Formal recursive analysis
- Amortized analysis

Learning Outcomes:
Core:

1. Given a problem to program for which there may be several algorithmic approaches, evaluate them and determine which are feasible, and select one that is optimal in implementation and run-time behavior [Evaluate]
2. Use recurrence relations to evaluate the time complexity of recursively defined algorithms [Apply]
3. Apply elementary recurrence relations using a form of the Master Theorem [Apply]

3.2.8 AL/Computational Complexity Theory (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:
Core

- Tractability and intractability: 1. P, NP, and NP-Complete Complexity Classes 2. NP-Complete Problems (e.g., SAT, Knapsack, TSP) 3. Reductions
- Turing Machine-based models of complexity: 1. Time complexity: (a) P, NP, NP-C, and EXP classes (b) Cook-Levin theorem 2. Space Complexity: (a) NSpace and PSpace (b) Savitch's theorem

Learning Outcomes:
Core:

1. Explain the significance of NP-Completeness [Explain]
2. Explain how NP-Hard is a lower bound and NP is an upper bound for NP-Completeness [Explain]

3. Explain examples of NP-complete problems [Explain]
4. Explain the Cook-Levin Theorem and the NP-Completeness of SAT [Explain]
5. Explain the classes P and NP [Explain]
6. Prove that a problem is NP-Complete by reducing a classic known NP-C problem to it (e.g., 3SAT and Clique) [Create]
7. Explain the P-space class and its relation to the EXP class [Explain]

3.2.9 AL/Formal Languages and Automata (Core Tier1: 4 hrs, Core Tier2: 10 hrs)

Topics:
Core

- Formal automata: 1. Finite State 2. Pushdown 3. Linear Bounded 4. Turing Machine
- Formal languages, grammars and Chomsky Hierarchy: 1. Regular (Type-3): (a) Regular Expressions 2. Context-Free (Type-2) 3. Context-Sensitive (Type-1) 4. Recursively Enumerable (Type-0)
- Relations among formal automata, languages, and grammars
- Deterministic and nondeterministic automata
- Pumping Lemma proofs: 1. Proof of Finite State/Regular-Language limitation 2. Pushdown Automata/Context-Free-Language limitation
- Equivalent models of algorithmic computation: 1. Turing Machines and Variations (e.g., multi-tape, non-deterministic) 2. Lambda Calculus 3. Mu-Recursive Functions

Learning Outcomes:
Core:

1. For each formal automaton in this unit: (a) Explain its definition comparing its characteristics with this unit's other automata (b) Using an example, explain step-by-step how the automaton operates on input including whether it accepts the associated input (c) Explain an example of inputs that can and cannot be accepted by the automaton. [Explain]
2. Given a problem, develop an appropriate automaton that addresses the problem [Create]
3. Develop a regular expression for a given regular language expressed in natural language [Create]
4. Explain the difference between regular expressions (Type-3 acceptors) and the regular expressions (Type-2 acceptors) used in programming languages [Explain]
5. For each formal model in this unit: (a) Explain its definition comparing its characteristics with the others in this unit (b) Explain example inputs that are and cannot be accepted by the language/grammar. [Explain]
6. For each formal automaton in this unit explain (compare/contrast) its deterministic and nondeterministic capabilities [Explain]
7. Apply pumping lemmas, or alternative means, to prove the limitations of Finite State and Pushdown automata [Apply]
8. Convert among equivalently powerful notations for a language, including among DFAs, NFAs, and regular expressions, and between PDAs and CFGs [Apply]

3.2.10 AL/Computability and Decidability (Core Tier1: 6 hrs, Core Tier2: 14 hrs)

Topics:
Core

- Decidability, (un)computability, and halting
- The Church-Turing thesis
- Algorithmic correctness: 1. Invariants (e.g., in iteration, recursion, tree search)
- Decidability: 1. Arithmetization and diagonalization
- Reducibility and reductions
- Time complexity based on Turing Machine
- Space complexity (e.g., Pspace, Savitch's Theorem)

Non Core

- Quantum computation: 1. Postulates of quantum mechanics: (a) State space (b) State evolution (c) State composition (d) State measurement 2. Column vector representations of qubits

3. Matrix representations of quantum operations 4. Simple quantum gates (e.g., XNOT, CNOT)

Learning Outcomes:
Core:

1. Explain a universal Turing Machine and its operation [Explain]
2. Present to an audience of co-workers and managers the impossibility of providing them a program that checks all other programs, including some seemingly simple ones, for infinite loops including an explanation of the Halting problem, why it has no algorithmic solution, and its significance for real-world algorithmic computation [Explain]
3. Explain examples of classic uncomputable problems [Explain]
4. Explain the Church-Turing Thesis and its significance for algorithmic computation [Explain]
5. Explain how (loop) invariants can be used to prove the correctness of an algorithm [Explain]
6. Apply arithmetization and diagonalization to prove the Halting Problem for Turing Machines is Undecidability [Apply]
7. Given a known undecidable language, apply a mapping reduction or computational history to prove that another language is undecidable [Apply]
8. Explain Rice's theorem and its significance [Explain]
9. Explain an example proof of a problem that is uncomputable by reducing a classic known uncomputable problem to it [Explain]
10. Explain the Primitive and General Recursive functions (zero, successor, selection, primitive recursion, composition, and Mu), their significance, and Turing Machine implementations [Explain]
11. Explain how computation is performed in Lambda Calculus (e.g., Alpha conversion and Beta reduction) [Explain]

NonCore:

12. For a quantum system give examples that explain the following postulates: (a) State Space - system state represented as a unit vector in Hilbert space (b) State Evolution - the use of unitary operators to evolve system state (c) State Composition - the use of tensor product to compose systems states (d) State Measurement - the probabilistic output of measuring a system state. [Explain]
13. Explain the operation of a quantum XNOT or CNOT gate on a quantum bit represented as a matrix and column vector, respectively [Explain]

3.2.11 AL/Society, Ethics, and the Profession

Topics:
Core

- Social, ethical, and secure algorithms
- Algorithmic fairness
- Anonymity (e.g., Differential Privacy)
- Accountability/Transparency
- Responsible algorithms
- Economic and other impacts of inefficient algorithms
- Sustainability
- Context aware computing

Learning Outcomes:
Core:

1. Develop algorithmic solutions to real-world societal problems, such as routing an ambulance to a hospital [Create]
2. Explain the impact that an algorithm may have on the environment and society when used to solve a real-world problem while considering its sustainability and that it can affect different societal groups in different ways [Explain]
3. Prepare a presentation that justifies the selection of appropriate data structures and/or algorithms to solve a given real-world problem [Explain]
4. Explain an example that articulates how differential privacy protects knowledge of an individual's data [Explain]

5. Explain the environmental impacts of design choices that relate to algorithm design [Explain]
6. Explain the tradeoffs involved in proof-of-work and proof-of-stake algorithms [Explain]

3.3 Architecture and Organization (AR)

Computing professionals spend considerable time writing efficient code to solve a particular problem in an application domain. As the shift from sequential to parallel processing occurs, a deeper understanding of the underlying computer architectures is necessary. Architecture can no longer be viewed as a black box where principles from one architecture can be applied to another. Instead, programmers should look inside the black box and use specific components to enhance system performance and energy efficiency.

The Architecture and Organization (AR) knowledge area aims to develop a deeper understanding of the hardware environments upon which almost all computing is based, and the relevant interfaces provided to higher software layers. The target hardware comprises low-end embedded system processors up to high-end enterprise multiprocessors.

The topics in this knowledge area will benefit students by enabling them to appreciate the fundamental architectural principles of modern computer systems, including the challenge of harnessing parallelism to sustain performance and energy improvements into the future. This KA will help computer science students depart from the black box approach and become more aware of the underlying computer system and the efficiencies specific architectures can achieve.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.3.1 Digital Logic and Digital Systems (Pg.. 56)		3
⭐ 3.3.2 Machine-Level Data Representation (Pg.. 57)	1	
⭐ 3.3.3 Assembly Level Machine Organization (Pg.. 57)	1	2
⭐ 3.3.4 Memory Hierarchy (Pg.. 58)	6	
⭐ 3.3.5 Interfacing and Communication (Pg.. 58)	1	
⭐ 3.3.6 Functional Organization (Pg.. 58)		2
⭐ 3.3.7 Performance and Energy Efficiency (Pg.. 59)		3
⭐ 3.3.8 Heterogeneous Architectures (Pg.. 59)		2
⭐ 3.3.9 Secure Processor Architectures (Pg.. 60)		2
⭐ 3.3.10 Quantum Architectures (Pg.. 60)		2
3.3.11 Sustainability Issues (Pg.. 61)	Elective	

Table 3.3: List of KUs in the Architecture and Organization area.

3.3.1 AR/Digital Logic and Digital Systems (Core Tier2: 3 hrs)

Topics:
Core

- Combinational vs sequential logic/field programmable gate arrays (FPGAs): 1. Fundamental combinational 2. Sequential logic building block
- Computer-aided design tools that process hardware and architectural representations
- High-level synthesis: 1. Register transfer notation 2. Hardware description language (e.g., Verilog/VHDL/Chisel)
- System-on-chip (SoC) design flow
- Physical constraints: 1. Gate delays 2. Fan-in and fan-out 3. Energy/power 4. Speed of light

Learning Outcomes:
Core:

1. Discuss the progression of computer technology components from vacuum tubes to VLSI, from mainframe computer architectures to the organization of warehouse-scale computers [Debate]
2. Describe parallelism and data dependencies between and within components in a modern heterogeneous computer architecture [Describe]
3. Explain the relationship between parallelism and power consumption [Explain]

4. Construct the design of basic building blocks for a computer: arithmetic-logic unit (gate-level), registers (gate-level), central processing unit (register transfer-level), and memory (register transfer-level) [Create]
5. Evaluate simple building blocks (e.g., arithmetic-logic unit, registers, movement between registers) of a simple computer design [Evaluate]
6. Analyze the timing behavior of a pipelined processor, identifying data dependency issues [Analyze]

3.3.2 AR/Machine-Level Data Representation (Core Tier1: 1 hr)

Topics:
Core

- Overview and history of computer architecture
- Bits, bytes, and words
- Unsigned, signed and two's complement representations
- Numeric data representation and number bases: 1. Fixed-point 2. Floating-point
- Representation of non-numeric data
- Representation of records, arrays and UTF data types

Learning Outcomes:
Core:

1. Discuss why everything in computers are data, including instructions [Debate]
2. Explain how fixed-length number representations can affect accuracy and precision [Explain]
3. Describe how negative integers are stored in sign-magnitude and two's-complement representations [Describe]
4. Discuss how different formats can represent numerical data [Debate]
5. Explain the bit-level representation of non-numeric data, such as characters, strings, records, and arrays [Explain]
6. Translate numerical data from one format to another [Translate]
7. Describe how a single adder (without overflow detection) can handle both signed (two's complement) and unsigned (binary) input without "knowing" which format a given input is using [Describe]

3.3.3 AR/Assembly Level Machine Organization (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- von Neumann machine architecture
- Control unit: instruction fetch, decode, and execution
- Introduction to SIMD vs MIMD and the Flynn taxonomy
- Shared memory multiprocessors/multicore organization
- Instruction set architecture (ISA) (e.g., x86, ARM and RISC-V): 1. Fixed vs variable-width instruction sets 2. Instruction formats 3. Data manipulation, control, I/O 4. Addressing modes 5. Machine language programming 6. Assembly language programming
- Subroutine call and return mechanisms
- I/O and interrupts
- Heap, static, stack, and code segments

Learning Outcomes:
Core:

1. Discuss how the classical von Neumann functional units are implemented in embedded systems, particularly on-chip and off-chip memory [Debate]
2. Describe how instructions are executed in a classical von Neumann machine, with extensions for threads, multiprocessor synchronization, and SIMD execution [Describe]
3. Assess an example diagram with instruction-level parallelism and hazards to describe how they are managed in typical processor pipelines [Assess]

4. Discuss how instructions are represented at the machine level and in the context of a symbolic assembler [Debate]
5. Map an example of high-level language patterns into assembly/machine language notations [Map]
6. Contrast different instruction formats considering aspects such as addresses per instruction and variable-length vs fixed-length formats [Contrast]
7. Analyze a subroutine diagram to comment on how subroutine calls are handled at the assembly level [Analyze]
8. Describe basic concepts of interrupts and I/O operations [Describe]
9. Write a simple assembly language program for string/array processing and manipulation [Write]

3.3.4 AR/Memory Hierarchy (Core Tier1: 6 hrs)

Topics:
Core

- Memory hierarchy: the importance of temporal and spatial locality
- Main memory organization and operations
- Persistent memory (e.g., SSD, standard disks)
- Latency, cycle time, bandwidth, and interleaving
- Cache memories: 1. Address mapping 2. Block size 3. Replacement and store policy 4. Prefetching
- Multiprocessor cache coherence
- Virtual memory (hardware support)
- Fault handling and reliability
- Reliability: 1. Error coding 2. Data compression 3. Data integrity
- Processing In-Memory (PIM)

Learning Outcomes:
Core:

1. Using a memory system diagram, identify the main types of memory technology (e.g., SRAM, DRAM) and their relative cost and performance [Analyze]
2. Measure the effect of memory latency on running time [Evaluate]
3. Enumerate the functions of a system with virtual memory management [Enumerate]
4. Compute average memory access time under various cache and memory configurations and mixes of instruction and data references [Compute]

3.3.5 AR/Interfacing and Communication (Core Tier1: 1 hr)

Topics:
Core

- I/O fundamentals: 1. Handshaking and buffering 2. Programmed I/O 3. Interrupt-driven I/O
- Interrupt structures: vectored and prioritized, interrupt acknowledgment
- I/O devices (e.g., mouse, keyboard, display, camera, sensors, actuators)
- External storage, physical organization, and drives
- Buses fundamentals: 1. Bus protocols 2. Arbitration 3. Direct-memory access (DMA)

Learning Outcomes:
Core:

1. Analyze an interrupt control diagram to comment on how interrupts are used to implement I/O control and data transfers [Analyze]
2. Enumerate various types of buses in a computer system [Enumerate]
3. List the advantages of magnetic disks and contrast them with those of solid-state disks [List]

3.3.6 AR/Functional Organization (Core Tier2: 2 hrs)

Topics:
Core

- Implementation of simple datapaths, including instruction pipelining, hazard detection, and resolution (e.g., stalls, forwarding)
- Control unit: 1. Hardwired implementation 2. Microprogrammed realization
- Instruction pipelining
- Introduction to instruction-level parallelism (ILP)

Learning Outcomes:
Core:

1. Compare alternative implementation of datapaths in modern computer architectures [Compare]
2. Produce a set of control signals for adding two integers using hardwired and microprogrammed implementations [Produce]
3. Discuss instruction-level parallelism using pipelining and significant hazards that may occur [Debate]
4. Design a complete processor, including datapath and control [Design]
5. Compute the average cycles per instruction for a given processor and memory system implementation [Compute]

3.3.7 AR/Performance and Energy Efficiency (Core Tier2: 3 hrs)

Topics:
Core

- Performance-energy evaluation (introduction): performance, power consumption, memory, and communication costs
- Branch prediction, speculative execution, out-of-order execution, Tomasulo's algorithm
- Enhancements for vector processors and GPUs
- Hardware support for multithreading: 1. Race conditions 2. Lock implementations 3. Point-to-point synchronization 4. Barrier implementation
- Scalability
- Alternative architectures including VLIW/EPIC, accelerators, and other special purpose processors
- Dynamic voltage and frequency scaling (DVFS)
- Dark Silicon

Learning Outcomes:
Core:

1. Discuss performance and energy efficiency evaluation metrics [Debate]
2. Analyze a speculative execution diagram and write about the decisions that can be made [Analyze]
3. Create a GPU performance-watt benchmarking diagram [Create]
4. Write a multithreaded program that adds (in parallel) elements of two integer vectors [Write]
5. Recommend a set of design choices for alternative computer architectures [Recommend]
6. Enumerate key concepts associated with dynamic voltage and frequency scaling [Enumerate]
7. Measure energy savings improvement for an 8-bit integer quantization compared to a 32-bit quantization [Evaluate]

3.3.8 AR/Heterogeneous Architectures (Core Tier2: 2 hrs)

Topics:
Core

- SIMD and MIMD architectures (e.g., General-Purpose GPUs, TPUs, and NPUs)
- Heterogeneous memory systems: 1. Shared memory versus distributed memory 2. Volatile vs non-volatile memory 3. Coherence protocols
- Domain-Specific Architectures (DSAs): 1. Machine Learning Accelerator 2. In-networking computing 3. Embedded systems for emerging applications 4. Neuromorphic computing 5. Edge computing devices
- Packaging and integration solutions such as 3DIC and chiplets
- Machine learning in architecture design: 1. AI algorithms for workload analysis 2. Optimization of architecture configurations for performance and power efficiency

Learning Outcomes:
Core:

1. Analyze a system diagram with alternative parallel architectures, e.g., SIMD and MIMD, and identify the key differences [Analyze]
2. Discuss what memory-management issues are found in multiprocessors that are not present in uniprocessors and how these issues might be resolved [Debate]
3. Indicate the differences between memory backplane, processor memory interconnect, and remote memory via networks, their implications for access latency, and their impact on program performance [Differentiate]
4. Discuss how you would determine when to use a domain-specific accelerator instead of a general-purpose CPU [Debate]
5. Enumerate key differences in architectural design principles between a vector and scalar-based processing unit [Enumerate]
6. List the advantages and disadvantages of a PIM architecture [List]

3.3.9 AR/Secure Processor Architectures (Core Tier2: 2 hrs)

Topics:
Core

- Principles of Secure Hardware: 1. Security Risk Analysis, Asset Protection, and Threat Model 2. Cryptographic Acceleration with Hardware 3. Support for virtualization (e.g., OS isolation)
- Roots of trust in hardware, Physically Unclonable Functions (PUF)
- Hardware Random Number Generators
- Memory protection extensions: 1. Runtime pointer bounds checking (e.g., buffer overflow) 2. Protection at the microarchitectural level 3. Protection at the ISA level
- Trusted Execution Environment (TEE): 1. Trusted Computer Base Protections 2. Protecting virtual machines 3. Protecting containers 4. Trusted software modules (Enclaves)
- Homomorphic encryption for privacy-preserving data processing

Learning Outcomes:
Core:

1. Discuss principles of secure hardware, exploring a framework for risk analysis and asset protection [Debate]
2. Summarize how Physically Unclonable Functions (PUF) can be a unique device identifier in security applications [Summarize]
3. Distinguish a random number generator with dedicated hardware support from generators without hardware dedicated to generating entropy [Distinguish]
4. List the advantages and disadvantages of memory protection at the ISA level [List]
5. Describe key design issues of a trusted execution environment (TEE) to support virtual machines [Describe]

3.3.10 AR/Quantum Architectures (Core Tier2: 2 hrs)

Topics:
Core

- Principles: 1. The wave-particle duality principle 2. The uncertainty principle in the double-slit experiment 3. What is a Qubit? Superposition, interference, and measurement. Photons as qubits 4. Systems of two qubits, Entanglement, Bell states, The No-Signaling theorem
- Axioms of QM: superposition principle, measurement axiom, unitary evolution
- Single qubit gates for the circuit model of quantum computation: X, Z, H
- Two qubit gates and tensor products, working with matrices
- The No-Cloning Theorem. The Quantum Teleportation protocol
- Algorithms: 1. Simple quantum algorithms: Bernstein-Vazirani, Simon's algorithm 2. Implementing Deutsch-Josza with Mach-Zehnder Interferometers 3. Quantum factoring (Shor's Algorithm) 4. Quantum search (Grover's Algorithm)
- Implementation aspects: 1. The physical implementation of qubits 2. Classical control of a Quantum Processing Unit (QPU) 3. Error mitigation and control, NISQ and beyond 4. Measurement

approaches
- Emerging Applications: 1. Post-quantum encryption 2. The Quantum Internet 3. Adiabatic quantum computation (AQC) and quantum annealing

Learning Outcomes:
Core:

1. Discuss how a quantum object produced as a particle propagates like a wave and is detected as a particle with a probability distribution corresponding to the wave [Debate]
2. Discuss the quantum-level nature that is inherently probabilistic [Debate]
3. Express your view on entanglement that can be used to create non-classical correlations, but there is no way to use quantum entanglement to send messages faster than the speed of light [Evaluate]
4. Describe quantum parallelism and the role of constructive vs destructive interference in quantum algorithms given the probabilistic nature of measurement(s) [Describe]
5. Analyze a code snippet providing the role of quantum Fourier transform (QFT) in Shor's algorithm [Analyze]
6. Write a program to implement Shor's algorithm in a simulator, highlighting the classical components and aspects of Shor's algorithm [Write]
7. Enumerate the specifics of each qubit modality (e.g., trapped ion, superconducting, silicon spin, photonic, quantum dot, neutral atom, topological, color center, electron-on-helium) [Enumerate]
8. Contrast AQC with the gate model of quantum computation and the problems each is better suited to solve [Contrast]

3.3.11 AR/Sustainability Issues

Topics:
Non Core

- Environmental impacts of implementation decisions: 1. Sustainability goals, resource consumption, and economic viability 2. Carbon footprint, hardware electronic waste 3. The energy footprint of data centers at various workloads (e.g., AI model training and use) 4. Guidelines for sustainable design standards

Learning Outcomes:
NonCore:

1. Assess the environmental impacts of a given project's deployment (e.g., the energy consumption of CPUs and GPUs, contribution to e-waste, and effect of hardware virtualization in data centers) [Assess]

3.4 Data Management (DM)

Since the mid-1970s, the study of Data Management (DM) has meant an almost exclusive study of relational database systems. Depending on institutional context, students have studied, in varying proportions, the following.

- Data modeling and database design: for example, E-R Data model, relational model, normalization theory
- Query construction: e.g., relational algebra, SQL
- Query processing: e.g., indices (B+tree, hash), algorithms (e.g., external sorting, select, project, join), query optimization (transformations, index selection)
- DBMS internals: e.g., concurrency/locking, transaction management, buffer management

Today's graduates are expected to possess DBMS user (rather than implementor) skills. These primarily include data modeling and query construction; ability to take an unorganized collection of data, organize it using a DBMS, and access/update the collection via queries.

Additionally, students need to study the following.

- The role data plays in an organization. This includes the Data Life Cycle: Creation-Processing-Review/Reporting-Retention/Retrieval-Destruction.

- The social/legal aspects of data collection: e.g., scale, data privacy, database privacy (compliance) by design, de-identification, ownership, reliability, database security, and intended and unintended applications.
- Emerging and advanced technologies that are augmenting/replacing traditional relational systems, particularly those used to support (big) data analytics, including NoSQL (e.g., JSON, XML, key-value store databases), cloud databases, MapReduce, and dataframes.
- The existing and emerging roles for those involved with data management, which include the following:
  - Product feature engineers: those who use both SQL and NoSQL operational databases.
  - Analytical engineers/data engineers: those who write analytical SQL, Python, and Scala code to build data assets for business groups.
  - Business analysts: those who build/manage data most frequently with Excel spreadsheets.
  - Data infrastructure engineers: those who implement a data management system in a variety of data applications (e.g., OLTP).
  - "Everyone" who produces or consumes data must understand the associated social, ethical, and professional issues.

One role that transcends all the above categories is that of data custodian. Previously, data were seen as a resource to be managed (Information Systems Management) just like other enterprise resources. Today, data are seen in a larger context. Data about customers can now be seen as belonging to (or in some national contexts, as owned by) those customers. There is now an accepted understanding that the safe and ethical storage, and use, of institutional data is part of being a responsible data custodian.

Furthermore, we acknowledge the tension between a curricular focus on professional preparation versus the study of a knowledge area as a scientific endeavor. This is particularly true with Data Management. For example, proving (or at least knowing) the completeness of Armstrong's Axioms is fundamental in functional dependency theory. However, most computer science graduates will never utilize this concept during their professional careers. The same can be said for many other topics in the Data Management canon. Conversely, if our graduates can only normalize data into Boyce-Codd normal form (using an automated tool) and write SQL queries, without understanding the role that indices play in efficient query execution, we have done them and society a disservice.

To this end, the number of CS Core hours is relatively small relative to the KA Core hours. This approach is designed to allow institutions with differing contexts to customize their curricula appropriately. An institution that focuses on OLTP implementation, for example, would prioritize efficient storage and data access, while an institution that focuses on product features would prioritize programmatic access to extant databases.

However, an institution manages this tension, we wish to give voice to one of the ironies of computer science curricula. Students typically spend much of their educational life reading (and writing) data from a file or interactively, while outside of the academy the predominant data comes from databases accessed programmatically. Perhaps in the not-too-distant future students will learn programmatic database access early on and then continue this practice as they progress through their curriculum.

Finally, we understand that while the Data Management KA may be orthogonal to the SEC (Security) and SEP (Society, Ethics, and the Profession) KAs, it is also ground zero for these (and other) knowledge areas. When designing persistent data stores, the question of what should be stored must be examined from both legal and ethical perspectives. Are there privacy concerns? And just as importantly, how well protected is the data?

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.4.1 The Role of Data and the Data Life Cycle (Pg.. 63)	2	
⭐ 3.4.2 Core Database System Concepts (Pg.. 63)	2	1
⭐ 3.4.3 Data Modeling (Pg.. 64)	2	3
⭐ 3.4.4 Relational Databases (Pg.. 64)	1	3
⭐ 3.4.5 Query Construction (Pg.. 64)	2	4
⭐ 3.4.6 Query Processing (Pg.. 65)		4
⭐ 3.4.7 DBMS Internals (Pg.. 66)		4
⭐ 3.4.8 NoSQL Systems (Pg.. 66)		2
⭐ 3.4.9 Data Security and Privacy (Pg.. 66)	1	2
⭐ 3.4.10 Data Analytics (Pg.. 67)		3
3.4.11 Distributed Databases/Cloud Computing (Pg.. 67)	Elective	
3.4.12 Semi-structured and Unstructured Databases (Pg.. 67)	Elective	
3.4.13 Society, Ethics, and the Profession (Pg.. 68)	Elective	

Table 3.4: List of KUs in the Data Management area.

3.4.1 DM/The Role of Data and the Data Life Cycle (Core Tier1: 2 hrs)

Topics:
Core

- The Data Life Cycle: Creation-Processing-Review/Reporting-Retention/Retrieval-Destruction

Learning Outcomes:
Core:

1. Identify the five stages of the Data Life Cycle [Analyze]

3.4.2 DM/Core Database System Concepts (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:
Core

- Purpose and advantages of database systems
- Components of database systems
- Design of core DBMS functions (e.g., query mechanisms, transaction management, buffer management, access methods)
- Database architecture, data independence, and data abstraction
- Transaction management
- Normalization
- Approaches for managing large volumes of data (e.g., NoSQL database systems, use of MapReduce)
- How to support CRUD-only applications
- Distributed databases/cloud-based systems
- Structured, semi-structured, and unstructured data
- Use of a declarative query language
- Systems supporting structured and/or stream content

Learning Outcomes:
Core:

1. Identify at least four advantages that using a database system provides [Analyze]
2. Enumerate the components of a (relational) database system [Enumerate]
3. Follow a query as it is processed by the components of a (relational) database system [Analyze]
4. Defend the value of data independence [Defend]
5. Compose a simple select-project-join query in SQL [Compose]
6. Enumerate the four properties of a correct transaction manager [Enumerate]
7. Describe the advantages for eliminating duplicate repeated data [Describe]
8. Outline how MapReduce uses parallelism to process data efficiently [Outline]
9. Evaluate the differences between structured and semi/unstructured databases [Evaluate]

3.4.3 DM/Data Modeling (Core Tier1: 2 hrs, Core Tier2: 3 hrs)

Topics:
Core

- Data modeling
- Relational data model
- Conceptual models (e.g., entity-relationship, UML diagrams)
- Semi-structured data models (expressed using DTD, XML, or JSON Schema, for example)

Non Core

- Spreadsheet models
- Object-oriented models: 1. GraphQL
- New features in SQL
- Specialized Data Modeling topics: 1. Time series data (aggregation, join) 2. Graph data (link traversal) 3. Techniques for avoiding inefficient raw data access (e.g., "avg daily price"): materialized views and special data structures (e.g., Hyperloglog, bitmap) 4. Geo-Spatial data (e.g., GIS databases)

Learning Outcomes:
Core:

1. Describe the components of the relational data model [Describe]
2. Model 1:1, 1:n, and n:m relationships using the relational data model [Use]
3. Describe the components of the E-R (or some other non-relational) data model [Describe]
4. Model a given environment using a conceptual data model [Analyze]
5. Model a given environment using the document-based or key-value store-based data model [Apply]

3.4.4 DM/Relational Databases (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- Entity and referential integrity: Candidate key, superkeys
- Relational database design
- Mapping conceptual schema to a relational schema
- Physical database design: file and storage structures
- Introduction to Functional dependency theory
- Normalization Theory: 1. Decomposition of a schema; lossless-join, and dependency-preservation properties of a decomposition 2. Normal forms (BCNF) 3. Denormalization (for efficiency)

Non Core

- Functional dependency theory: 1. Closure of a set of attributes 2. Canonical Cover
- Normalization theory: 1. Multi-valued dependency (4NF) 2. Join dependency (PJNF, 5NF) 3. Representation theory

Learning Outcomes:
Core:

1. Describe the defining characteristics behind the relational data model [Describe]
2. Comment on the difference between a foreign key and a superkey [Explain]
3. Enumerate the different types of integrity constraints [Enumerate]
4. Compose a relational schema from a conceptual schema which contains 1:1, 1:n, and n:m relationships [Compose]
5. Map appropriate file structure to relations and indices [Map]
6. Describe how functional dependency theory generalizes the notion of key [Describe]
7. Defend a given decomposition as lossless and or dependency preserving [Defend]
8. Detect which normal form a given decomposition yields [Detect]
9. Comment on reasons for denormalizing a relation [Interpret]

3.4.5 DM/Query Construction (Core Tier1: 2 hrs, Core Tier2: 4 hrs)

Topics:
Core

- SQL Query Formation: 1. Interactive SQL execution 2. Programmatic execution of an SQL query
- Relational Algebra
- SQL: 1. Data definition including integrity and other constraint specifications 2. Update sublanguage

Non Core

- Relational Calculus
- QBE and 4th-generation environments
- Different ways to invoke non-procedural queries in conventional languages
- Introduction to other major query languages (e.g., XPATH, SPARQL)
- Stored procedures

Learning Outcomes:
Core:

1. Compose SQL queries that incorporate select, project, join, union, intersection, set difference, and set division [Compose]
2. Determine when a nested SQL query is correlated or not [Determine]
3. Iterate over data retrieved programmatically from a database via an SQL query [Create]
4. Define, in SQL, a relation schema, including all integrity constraints and delete/update triggers [Define]
5. Compose an SQL query to update a tuple in a relation [Compose]

3.4.6 DM/Query Processing (Core Tier2: 4 hrs)

Topics:
Core

- Page structures
- Index structures: 1. B+ trees 2. Hash indices: static and dynamic 3. Index creation in SQL
- File structures: 1. Heap files 2. Hash files
- Algorithms for query operators: 1. External Sorting 2. Selection 3. Projection; with and without duplicate elimination 4. Natural Joins: Nested loop, Sort-merge, Hash join 5. Analysis of algorithm efficiency
- Query transformations
- Query optimization: 1. Access paths 2. Query plan construction 3. Selectivity estimation 4. Index-only plans
- Parallel Query Processing (e.g., parallel scan, parallel join, parallel aggregation)
- Database tuning/performance: 1. Index selection 2. Impact of indices on query performance 3. Denormalization

Learning Outcomes:
Core:

1. Describe the purpose and organization of both B+ tree and hash index structures [Describe]
2. Compose an SQL command to create an index (any kind) [Compose]
3. Specify the steps for the various query operator algorithms: external sorting, projection with duplicate elimination, sort-merge join, hash-join, block nested-loop join [Analyze]
4. Derive the run-time (in I/O requests) for each of the above algorithms [Analyze]
5. Transform a query in relational algebra to its equivalent appropriate for a left-deep, pipelined execution [Redesign]
6. Compute selectivity estimates for a given selection and/or join operation [Compute]
7. Describe how to modify an index structure to facilitate an index-only operation for a given relation [Describe]
8. For a given scenario decide on which indices to support for the efficient execution of a set of queries [Assess]
9. Describe how DBMSs leverage parallelism to speed up query processing by dividing the work across multiple processors or nodes [Describe]

3.4.7 DM/DBMS Internals (Core Tier2: 4 hrs)

Topics:
Core
- DB Buffer Management
- Transaction Management: 1. Isolation Levels 2. ACID 3. Serializability 4. Distributed Transactions
- Concurrency Control: 1. 2-Phase Locking 2. Deadlocks handling strategies 3. Quorum-based consistency models
- Recovery Manager: 1. Relation with Buffer Manager

Non Core
- Concurrency Control: 1. Optimistic concurrency control 2. Timestamp concurrency control
- Recovery Manager: 1. Write-Ahead logging 2. ARIES recovery system (Analysis, REDO, UNDO)

Learning Outcomes:
Core:
1. Describe how a DBMS manages its Buffer Pool [Describe]
2. Describe the four properties for a correct transaction manager [Describe]
3. Outline the principle of serializability [Outline]

3.4.8 DM/NoSQL Systems (Core Tier2: 2 hrs)

Topics:
Core
- Why NoSQL? (e.g., Impedance mismatch between Application [CRUD] and RDBMS)
- Key-Value and Document data model

Non Core
- Storage systems (e.g., Key-Value systems, Data Lakes)
- Distribution Models (Sharding and Replication)
- Graph Databases
- Consistency Models (Update and Read, Quorum consistency, CAP theorem)
- Processing model (e.g., Map-Reduce, multi-stage map-reduce, incremental map-reduce)
- Case Studies: Cloud storage system (e.g., S3); Graph databases; "When not to use NoSQL"

Learning Outcomes:
Core:
1. Develop a use case for the use of NoSQL over RDBMS [Create]
2. Describe the defining characteristics behind Key-Value and Document-based data models [Describe]

3.4.9 DM/Data Security and Privacy (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core
- Differences between data security and data privacy
- Protecting data and database systems from attacks, including injection attacks such as SQL injection
- Personally identifying information (PII) and its protection
- Ethical considerations in ensuring the security and privacy of data
- Need for, and different approaches to securing data at rest, in transit, and during processing
- Database auditing and its role in digital forensics
- Data inferencing and preventing attacks
- Laws and regulations governing data security and data privacy

Non Core
- Typical risk factors and prevention measures for ensuring data integrity
- Ransomware and prevention of data loss and destruction

Learning Outcomes:
Core:

1. Describe the differences in the goals for data security and data privacy [Describe]
2. Identify and mitigate risks associated with different approaches to protecting data [Analyze]
3. Describe legal and ethical considerations of end-to-end data security and privacy [Describe]
4. Develop a database auditing system given risk considerations [Create]
5. Apply several data exploration approaches to understanding unfamiliar datasets [Apply]

3.4.10 DM/Data Analytics (Core Tier2: 3 hrs)

Topics:
Core

- Exploratory data techniques (motivation, representation, descriptive statistics, visualizations)
- Data science lifecycle: business understanding, data understanding, data preparation, modeling, evaluation, deployment, and user acceptance
- Data mining and machine learning algorithms: e.g., classification, clustering, association, regression
- Data acquisition and governance
- Data security and privacy considerations
- Data fairness and bias
- Data visualization techniques and their use in data analytics
- Entity Resolution

Learning Outcomes:
Core:

1. Describe several data exploration approaches, including visualization, to understanding unfamiliar datasets [Describe]
2. Apply several data exploration approaches to understanding unfamiliar datasets [Apply]
3. Describe basic machine learning/data mining algorithms and when they are appropriate for use [Describe]
4. Apply several machine learning/data mining algorithms [Apply]
5. Describe legal and ethical considerations in acquiring, using, and modifying datasets [Describe]
6. Describe issues of fairness and bias in data collection and usage [Describe]

3.4.11 DM/Distributed Databases/Cloud Computing

Topics:
Non Core

- Distributed DBMS: 1. Distributed data storage 2. Distributed query processing 3. Distributed transaction model 4. Homogeneous and heterogeneous solutions 5. Client-server distributed databases
- Parallel DBMS: 1. Parallel DBMS architectures: shared memory, shared disk, shared nothing 2. Speedup and scale-up, e.g., use of the MapReduce processing model 3. Data replication and weak consistency models

Learning Outcomes:
NonCore:

1. Describe the key components of a distributed DBMS, including distributed data storage, query processing, and transaction management [Describe]
2. Analyze the tradeoffs between parallel DBMS architectures: shared memory, shared disk, and shared nothing [Analyze]
3. Describe data replication strategies and weak consistency models in distributed database systems [Describe]

3.4.12 DM/Semi-structured and Unstructured Databases

Topics:
Non Core

- Vectorized unstructured data (text, video, audio, etc.) and vector storage: 1. TF-IDF Vectorizer with ngram 2. Word2Vec 3. Array database or array data type handling

- Semi-structured databases (e.g., JSON): 1. Storage: (a) Encoding and compression of nested data types 2. Indexing: (a) Btree, skip index, Bloom filter (b) Inverted index and bitmap compression (c) Space filling curve indexing for semi-structured geo-data 3. Query processing for OLTP and OLAP use cases: (a) Insert, Select, update/delete tradeoffs (b) Case studies on Postgres/JSON, MongoDB, and Snowflake/JSON

Learning Outcomes:
NonCore:

1. Describe vectorized representations of unstructured data (e.g., TF-IDF, Word2Vec) and their storage models [Describe]
2. Apply indexing techniques (e.g., B-tree, inverted index, Bloom filter) to semi-structured data stored in JSON format [Apply]
3. Analyze query processing tradeoffs for semi-structured databases in OLTP and OLAP use cases [Analyze]

3.4.13 DM/Society, Ethics, and the Profession

Topics:
Core

- Issues related to scale
- Data privacy overall: 1. Privacy compliance by design
- Data anonymity
- Data ownership/custodianship
- Intended and unintended applications of stored data
- Reliability of data
- Provenance, data lineage, and metadata management
- Data security

Learning Outcomes:
Core:

1. Enumerate three social and three legal issues related to large data collections [Enumerate]
2. Describe the value of data privacy [Describe]
3. Identify the competing stakeholders with respect to data ownership [Analyze]
4. Enumerate three negative unintended consequences from a given (well known) data-centric application (e.g., Facebook, LastPass, Ashley Madison) [Enumerate]
5. Describe the meaning of data provenance and lineage [Describe]
6. Identify how a database might contribute to data security as well as how it may introduce insecurities [Analyze]

3.5 Foundations of Programming Languages (FPL)

The foundations of programming languages are rooted in discrete mathematics, logic, and formal languages, and provide a basis for the understanding of complex modern programming languages. Although programming languages vary according to the language paradigm and the problem domain and evolve in response to both societal needs and technological advancement, they share an underlying abstract model of computation and program development. This remains true even as processor hardware and their interface with programming tools become increasingly intertwined and progressively more complex. An understanding of the common abstractions and programming paradigms enables faster learning of programming languages.

The Foundations of Programming Languages knowledge area is concerned with articulating the underlying concepts and principles of programming languages, the formal specification of a programming language and the behavior of a program, explaining how programming languages are implemented, comparing the strengths and weaknesses of various programming paradigms, and describing how programming languages interface with entities such as operating systems and hardware. The concepts covered here are applicable to several languages and an understanding of these principles assists a learner to move readily from one language to another, as well as select a programming paradigm and language that best suits the problem at hand.

Programming languages are the medium through which programmers precisely describe concepts, formulate algorithms, and reason about solutions. Over the course of a career, a computer scientist will learn and work with many different languages, separately or together. Software developers must understand different programming models, programming features and constructs, and underlying concepts to make informed design choices among languages that support multiple complementary approaches. It would be useful to know how programming language features are defined, composed, and implemented to improve execution efficiency and long-term maintenance of developed software. Also useful is a basic knowledge of language translation, program analysis, run-time behavior, memory management and interplay of concurrent processes communicating with each other through message-passing, shared memory, and synchronization. Finally, some developers and researchers will need to design new languages, an exercise which requires greater familiarity with basic principles.

Knowledge Area (KA)	**Core Tier1**	**Core Tier2**
⭐ 3.5.1 Object-Oriented Programming Fundamentals (Pg.. 69)	2	1
⭐ 3.5.2 Object-Oriented Programming: Encapsulation, Subtyping, and Reflection (Pg.. 70)	2	1
⭐ 3.5.3 Functional Programming (Pg.. 70)	4	3
⭐ 3.5.4 Logic Programming (Pg.. 71)		2
⭐ 3.5.5 Shell Scripting (Pg.. 72)	2	
⭐ 3.5.6 Event-Driven and Reactive Programming (Pg.. 72)	2	2
⭐ 3.5.7 Parallel and Distributed Computing (Pg.. 73)	2	2
3.5.8 Aspect-Oriented Programming (Pg.. 73)	Elective	
⭐ 3.5.9 Type Systems Fundamentals (Pg.. 74)	2	2
⭐ 3.5.10 Type Systems: Polymorphism and Complementary Typing (Pg.. 74)	1	1
⭐ 3.5.11 Advanced Type Theory (Pg.. 75)	2	2
⭐ 3.5.12 Systems Execution and Memory Model (Pg.. 75)	2	
⭐ 3.5.13 Memory Management (Pg.. 75)	1	
⭐ 3.5.14 Language Translation and Execution (Pg.. 76)	2	3
⭐ 3.5.15 Program Abstraction and Representation (Pg.. 76)		3
3.5.16 Syntax Analysis (Pg.. 77)	Elective	
3.5.17 Compiler Semantic Analysis (Pg.. 78)	Elective	
3.5.18 Program Analysis and Analyzers (Pg.. 78)	Elective	
3.5.19 Code Generation (Pg.. 78)	Elective	
3.5.20 Run-time Behavior and Systems (Pg.. 79)	Elective	
3.5.21 Advanced Programming Constructs (Pg.. 80)	Elective	
3.5.22 Language Pragmatics (Pg.. 80)	Elective	
3.5.23 Formal Semantics (Pg.. 81)	Elective	
3.5.24 Formal Development Methodologies (Pg.. 81)	Elective	
3.5.25 Design Principles of Programming Languages (Pg.. 81)	Elective	
3.5.26 Society, Ethics, and the Profession (Pg.. 82)	Elective	

Table 3.5: List of KUs in the Foundations of Programming Languages area.

3.5.1 FPL/Object-Oriented Programming Fundamentals (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:

Core

- Imperative programming as a subset of object-oriented programming.
- Object-oriented design: 1. Decomposition into objects carrying state and having behavior. 2. Class-hierarchy design for modeling.
- Definition of classes: fields, methods, and constructors.
- Subclasses, inheritance (including multiple inheritance), and method overriding.
- Dynamic dispatch: definition of method-call.
- Exception handling.

Learning Outcomes:
Core:

1. Enumerate the differences between imperative and object-oriented programming paradigms [Enumerate]
2. Compose a class through design, implementation, and testing to meet behavioral requirements [Compose]
3. Build a simple class hierarchy utilizing subclassing that allows code to be reused for distinct subclasses [Create]
4. Predict and validate control flow in a program using dynamic dispatch [Evaluate]
5. Compare and contrast how computational solutions to a problem differ in procedural, functional, and object-oriented approaches [Compare]
6. Compare and contrast mechanisms to define and protect data elements within procedural, functional, and object-oriented approaches [Compare]

3.5.2 FPL/Object-Oriented Programming: Encapsulation, Subtyping, and Reflection (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:
Core

- Object-oriented idioms for encapsulation: 1. Privacy, data hiding, and visibility of class members. 2. Interfaces revealing only method signatures. 3. Abstract base classes, traits and mixins.
- Dynamic vs static properties.
- Composition vs inheritance.
- Subtyping: 1. Subtype polymorphism; implicit upcasts in typed languages. 2. Notion of behavioral replacement: subtypes acting like supertype. 3. Relationship between subtyping and inheritance.
- Collection classes, iterators, and other common library components.
- Metaprogramming and reflection.

Learning Outcomes:
Core:

1. Compare and contrast the benefits and costs/impact of using inheritance (subclasses) and composition (specifically, how to base composition on higher order functions) [Compare]
2. Explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping (the idea of a subtype being usable in a context that expects the supertype) [Explain]
3. Use object-oriented encapsulation mechanisms such as interfaces and private members [Use]
4. Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language [Define]
5. Use collection classes and iterators effectively to solve a problem [Use]
6. Compare and contrast how computational solutions to a problem differ in procedural, functional, and object-oriented approaches [Compare]
7. Compare and contrast mechanisms to define and protect data elements within procedural, functional, and object-oriented approaches [Compare]

3.5.3 FPL/Functional Programming (Core Tier1: 4 hrs, Core Tier2: 3 hrs)

Topics:
Core

- Lambda expressions and evaluation: 1. Variable binding and scope rules. 2. Parameter-passing. 3. Nested lambda expressions and reduction order.
- Effect-free programming: 1. Function calls have no side effects, facilitating compositional reasoning. 2. Immutable variables and data copying vs reduction. 3. Use of recursion vs loops vs pipelining (map/reduce).
- Processing structured data (e.g., trees) via functions with cases for each data variant: 1. Functions defined over compound data in terms of functions applied to the constituent pieces. 2. Persistent data structures.

- Using higher-order functions (taking, returning, and storing functions).
- Metaprogramming and reflection.
- Function closures (functions using variables in the enclosing lexical environment): 1. Basic meaning and definition - creating closures at run-time by capturing the environment. 2. Canonical idioms: call-backs, arguments to iterators, reusable code via function arguments. 3. Using a closure to encapsulate data in its environment. 4. Delayed versus eager evaluation.

Non Core

- Graph reduction machine and call-by-need.
- Implementing delayed evaluation.
- Integration with logic programming paradigm using concepts such as equational logic, narrowing, residuation and semantic unification.
- Integration with other programming paradigms such as imperative and object-oriented.

Learning Outcomes:

Core:

1. Develop basic algorithms that avoid assigning to mutable states or considering reference equality [Create]
2. Develop useful functions that take and return other functions [Create]
3. Explain a simple example of lambda expression being implemented using a virtual machine, such as a SECD machine, showing storage and reclaim of the environment [Explain]
4. Correctly interpret variables and lexical scope in a program using function closures [Interpret]
5. Use functional encapsulation mechanisms such as closures and modular interfaces [Use]
6. Compare and contrast stateful vs stateless execution [Compare]
7. Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language [Define]

NonCore:

8. Illustrate graph reduction using a λ-expression using a shared subexpression [Apply]
9. Illustrate the execution of a simple nested λ-expression using an abstract machine, such as an ABC machine [Apply]
10. Illustrate narrowing, residuation, and semantic unification using simple illustrative examples [Apply]
11. Illustrate the concurrency constructs using simple programming examples of known concepts such as a buffer being read and written concurrently or sequentially [Apply]

3.5.4 FPL/Logic Programming (Core Tier2: 2 hrs)

Topics:

Core

- Universal vs existential quantifiers.
- First order predicate logic vs higher order logic.
- Expressing complex relations using logical connectives and simpler relations.
- Definitions of Horn clause, facts, goals and subgoals.
- Unification and unification algorithm; unification vs assertion vs expression evaluation.
- Mixing relations with functions.
- Cuts, backtracking, and non-determinism.
- Closed-world vs open-world assumptions.

Non Core

- Memory overhead of variable copying in handling iterative programs.
- Programming constructs to store partial computation and pruning search trees.
- Mixing functional programming and logic programming using concepts such as equational logic, narrowing, residuation, and semantic unification.
- Higher-order, constraint, and inductive logic programming.
- Integration with other programming paradigms such as object-oriented programming.
- Advance programming constructs such as difference-lists, creating user defined data structures, set of, etc.

Learning Outcomes:
Core:

1. Use a logic language to implement a conventional algorithm [Use]
2. Use a logic language to implement an algorithm employing implicit search using clauses, relations, and cuts [Use]
3. Use a simple illustrative example to show correspondence between First Order Predicate Logic (FOPL) and logic programs using Horn clauses [Use]
4. Use examples to illustrate the unification algorithm and its role of parameter-passing in query reduction [Use]
5. Use simple logic programs interleaving relations, functions, and recursive programming such as factorial and Fibonacci numbers and simple complex relationships between entities and illustrate execution and parameter-passing using unification and backtracking [Use]

NonCore:

6. Illustrate computation of simple programs such as Fibonacci and show overhead of recomputation, and then show how to improve execution overhead [Apply]

3.5.5 FPL/Shell Scripting (Core Tier1: 2 hrs)

Topics:
Core

- Error/exception handling
- Piping
- System commands: 1. Interface with operating systems
- Environment variables
- File abstraction and operators
- Data structures, such as arrays and lists.
- Regular expressions
- Programs and processes
- Workflow

Learning Outcomes:
Core:

1. Create and execute automated scripts to manage various system tasks [Create]
2. Solve various text processing problems through scripting [Solve]

3.5.6 FPL/Event-Driven and Reactive Programming (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:
Core

- Procedural programming vs reactive programming: advantages of reactive programming in capturing events.
- Components of reactive programming: event-source, event signals, listeners and dispatchers, event objects, adapters, event-handlers.
- Stateless and state-transition models of event-based programming.
- Canonical uses such as GUIs, mobile devices, robots, servers.
- Using a reactive framework: 1. Defining event handlers/listeners 2. Parameterization of event senders and event arguments 3. Externally generated events and program-generated events
- Separation of model, view, and controller
- Event-driven and reactive programs as state-transition systems

Learning Outcomes:
Core:

1. Implement event handlers for use in reactive systems, such as GUIs [Implement]
2. Examine why an event-driven programming style is natural in domains where programs react to external events [Examine]
3. Define and use a reactive framework [Define]
4. Describe an interactive system in terms of a model, a view, and a controller [Describe]

3.5.7 FPL/Parallel and Distributed Computing (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:
Core

- Safety and liveness: 1. Race conditions 2. Dependencies/preconditions 3. Fault models 4. Termination
- Programming models: 1. Actor models 2. Procedural and reactive models 3. Synchronous/asynchronous programming models 4. Data parallelism
- Properties: 1. Order-based properties: (a) Commutativity (b) Independence 2. Consistency-based properties: (a) Atomicity (b) Consensus
- Execution control: 1. Async await 2. Promises 3. Threads
- Communication and coordination: 1. Mutexes 2. Message-passing 3. Shared memory 4. Cobegin-coend 5. Monitors 6. Channels 7. Threads 8. Guards
- Futures
- Language support for data parallelism such as forall, loop unrolling, map/reduce
- Effect of memory-consistency models on language semantics and correct code generation
- Representational State Transfer Application Programming Interfaces (REST APIs)
- Technologies and approaches: cloud computing, high performance computing, quantum computing, ubiquitous computing
- Overheads of message-passing
- Granularity of program for efficient exploitation of concurrency
- Concurrency and other programming paradigms (e.g., functional)

Learning Outcomes:
Core:

1. Explain why programming languages do not guarantee sequential consistency in the presence of data races and what programmers must do as a result [Explain]
2. Implement correct concurrent programs using multiple programming models, such as shared memory, actors, futures, synchronization constructs, and data-parallelism primitives [Implement]
3. Use a message-passing model to analyze a communication protocol [Use]
4. Use synchronization constructions such as monitor/synchronized methods in a simple program [Use]
5. Modeling data dependency using simple programming constructs involving variables, read and write [Implement]
6. Modeling control dependency using simple constructs such as selection and iteration [Implement]
7. Explain how REST API's integrate applications and automate processes [Explain]
8. Explain benefits, constraints and challenges related to distributed and parallel computing [Explain]

3.5.8 FPL/Aspect-Oriented Programming

Topics:
Non Core

- Aspects
- Join points
- Advice: 1. Before 2. After (as finally, returning or throwing) 3. Around
- Point cuts: 1. Designators
- Weaving - static and dynamic
- Alternatives including annotations and IDEs

Learning Outcomes:
NonCore:

1. Describe the core concepts of aspect-oriented programming, including aspects, join points, advice, and point cuts [Describe]
2. Apply static and dynamic weaving strategies to modularize cross-cutting concerns in a program [Apply]

3. Analyze alternatives to aspect-oriented programming, such as annotations and IDE-based approaches, for handling cross-cutting concerns [Analyze]

3.5.9 FPL/Type Systems Fundamentals (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:
Core

- A type as a set of values together with a set of operations: 1. Primitive types (e.g., numbers, Booleans) 2. Compound types built from other types (e.g., records/structs, unions, arrays, lists, functions, references using set operations)
- Association of types to variables, arguments, results, and fields
- Type safety as an aspect of program correctness
- Type safety and errors caused by using values inconsistently given their intended types
- Goals and limitations of static and dynamic typing: detecting and eliminating errors as early as possible.
- Type equivalence: structural vs name equivalence

Learning Outcomes:
Core:

1. Describe, for both a primitive and a compound type, the values that have that type [Describe]
2. Describe, for a language with a static type system, the operations that are forbidden statically, such as passing the wrong type of value to a function or method [Describe]
3. Describe examples of program errors detected by a type system [Describe]
4. Identify program properties, for multiple programming languages, that are checked statically and program properties that are checked dynamically [Analyze]
5. Describe an example program that does not type-check in a particular language and yet would have no error if run [Describe]
6. Use types and type-error messages to write and debug programs [Use]
7. Explain how typing rules define the set of operations that are legal for a type [Explain]
8. List the type rules governing the use of a particular compound type [List]
9. Explain why undecidability requires type systems to conservatively approximate program behavior [Explain]
10. Explain multiple benefits and limitations of static typing in writing, maintaining, and debugging software [Explain]

3.5.10 FPL/Type Systems: Polymorphism and Complementary Typing (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Generic types (parametric polymorphism): 1. Definition and advantages of polymorphism: parametric, subtyping, overloading, and coercion 2. Comparison of monomorphic and polymorphic types 3. Comparison with ad-hoc polymorphism (overloading) and subtype polymorphism 4. Generic parameters and typing 5. Use of generic libraries such as collections 6. Comparison with ad hoc polymorphism (overloading) and subtype polymorphism 7. Prescriptive vs descriptive polymorphism 8. Implementation models of polymorphic types 9. Subtyping
- Complementary benefits of static and dynamic typing: 1. Errors early vs errors late/avoided 2. Enforce invariants during code development and code maintenance vs postpone typing decisions while prototyping and conveniently allow flexible coding patterns such as heterogeneous collections. 3. Typing rules for function, product, and sum types 4. Avoiding misuse of code vs allowing more code reuse 5. Detect incomplete programs vs allow incomplete programs to run 6. Relationship to static analysis 7. Decidability

Learning Outcomes:
Core:

1. Define and use program pieces (such as functions, classes, methods) that use generic types, including for collections [Define]
2. Discuss the differences among generics, subtyping, and overloading [Debate]

3.5.11 FPL/Advanced Type Theory (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:
Non Core

- Compositional type constructors, such as product types (for aggregates), sum types (for unions), function types, quantified types, and recursive types
- Type checking
- Subtyping: 1. Subtype polymorphism; implicit upcasts in typed languages 2. Notion of behavioral replacement: subtypes acting like supertype 3. Relationship between subtyping and inheritance
- Type safety as preservation plus progress
- Type inference
- Static overloading
- Propositions as types (implication as a function, conjunction as a product, disjunction as a sum)
- Dependent types (universal quantification as dependent function, existential quantification as dependent product).

Learning Outcomes:
NonCore:

1. Define a type system precisely and compositionally [Define]
2. For various foundational type constructors, identify the values they describe and the invariants they enforce [Analyze]
3. Precisely describe the invariants preserved by a sound type system [Describe]
4. Prove type safety for a simple language in terms of preservation and progress theorems [Prove]
5. Implement a unification-based type-inference algorithm for a simple language [Implement]
6. Explain how static overloading and associated resolution algorithms influence the dynamic behavior of programs [Explain]

3.5.12 FPL/Systems Execution and Memory Model (Core Tier1: 2 hrs)

Topics:
Core

- Data structures for translation, execution, translation, and code mobility such as stack, heap, aliasing (sharing using pointers), indexed sequence and string
- Direct, indirect, and indexed access to memory location
- Run-time representation of data abstractions such as variables, arrays, vectors, records, pointer-based data elements such as linked-lists and trees, and objects
- Abstract low-level machine with simple instruction, stack, and heap to explain translation and execution
- Run-time layout of memory: activation record (with various pointers), static data, call-stack, heap: 1. Translating selection and iterative constructs to control-flow diagrams 2. Translating control-flow diagrams to low level abstract code 3. Implementing loops, recursion, and tail calls 4. Translating function/procedure calls and return from calls, including different parameter-passing mechanisms using an abstract machine
- Green computing.

Learning Outcomes:
Core:

1. Explain how a core language construct, such as data abstractions and control abstractions, is executed [Explain]

3.5.13 FPL/Memory Management (Core Tier1: 1 hr)

Topics:
Core

- Low level allocation and accessing of high-level data structures such as basic data types, n-dimensional array, vector, record, and objects

- Return from procedure as automatic deallocation mechanism for local data elements in the stack

- Manual memory management: allocating, de-allocating, and reusing heap memory
- Automated memory management: garbage collection as an automated technique using the notion of reachability

Learning Outcomes:
Core:

1. Explain how programming language implementations typically organize memory into global data, text, heap, and stack sections and how features such as recursion and memory management map to this memory model [Explain]
2. Explain why memory leaks and dangling pointer problems occur, and what can be done by a programmer to avoid/fix them [Explain]

3.5.14 FPL/Language Translation and Execution (Core Tier1: 2 hrs, Core Tier2: 3 hrs)

Topics:
Core

- Execution models for JIT (Just-In-Time), compiler, interpreter
- Use of intermediate code, e.g., bytecode
- Limitations and benefits of JIT, compiler, and interpreter
- Cross compilers/transpilers
- BNF and extended BNF representation of context-free grammar
- Parse tree using a simple sentence such as arithmetic expression or if-then-else statement
- Execution as native code or within a virtual machine
- Language translation pipeline: syntax analysis, parsing, optional type-checking, translation/code generation and optimization, linking, loading, execution
- Run-time representation of core language constructs such as objects (method tables) and functions that can be passed as parameters to and returned from functions (closures)
- Secure compiler development

Learning Outcomes:
Core:

1. Explain and understand the differences between compiled, JIT, and interpreted language implementations, including the benefits and limitations of each [Explain]
2. Differentiate syntax and parsing from semantics and evaluation [Differentiate]
3. Use BNF and extended BNF to specify the syntax of simple constructs such as if-then-else, type declaration and iterative constructs for known languages such as C++ or Python [Use]
4. Illustrate the parse tree using a simple sentence/arithmetic expression [Apply]
5. Illustrate translation of syntax diagrams to BNF/extended BNF for simple constructs such as if-then-else, type declaration, iterative constructs, etc [Apply]
6. Illustrate ambiguity in parsing using nested if-then-else/arithmetic expression and show resolution using precedence order [Apply]
7. Discuss the benefits and limitations of garbage collection, including the notion of reachability [Debate]

3.5.15 FPL/Program Abstraction and Representation (Core Tier2: 3 hrs)

Topics:
Core

- BNF and regular expressions
- Programs that take (other) programs as input such as interpreters, compilers, type-checkers, documentation generators
- Components of a language: 1. Definitions of alphabets, delimiters, sentences, syntax, and semantics 2. Syntax vs semantics
- Program as a set of non-ambiguous meaningful sentences

- Basic programming abstractions: constants, variables, declarations (including nested declarations), command, expression, assignment, selection, definite and indefinite iteration, iterators, function, procedure, modules, exception handling
- Mutable vs immutable variables: advantages and disadvantages of reusing existing memory location vs advantages of copying and keeping old values; storing partial computation vs recomputation
- Types of variables: static, local, nonlocal, global; need and issues with nonlocal and global variables.
- Scope rules: static vs dynamic; visibility of variables; side-effects.
- Side-effects induced by nonlocal variables, global variables and aliased variables.

Non Core

- L-values and R-values: mapping mutable variable-name to L-values; mapping immutable variable-names to R-values
- Environment vs store and their properties
- Data and control abstraction
- Mechanisms for information exchange between program units such as procedures, functions, and modules: nonlocal variables, global variables, parameter-passing, import-export between modules
- Data structures to represent code for execution, translation, or transmission.
- Low level instruction representation such as virtual machine instructions, assembly language, and binary representation
- Lambda calculus, variable binding, and variable renaming.
- Types of semantics: operational, axiomatic, denotational, behavioral; define and use abstract syntax trees; contrast with concrete syntax.

Learning Outcomes:

Core:

1. Illustrate the scope of variables and visibility using simple programs [Apply]
2. Illustrate different types of parameter-passing using simple pseudo programming language [Apply]
3. Explain side-effect using global and nonlocal variables and how to fix such programs [Explain]
4. Explain how programs that process other programs treat the other programs as their input data [Explain]
5. Describe a grammar and an abstract syntax tree for a small language [Describe]
6. Describe the benefits of having program representations other than strings of source code [Describe]
7. Implement a program to process some representation of code for some purpose, such as an interpreter, an expression optimizer, or a documentation generator [Implement]

3.5.16 FPL/Syntax Analysis

Topics:

Non Core

- Regular grammars vs context-free grammars
- Scanning and parsing based on language specifications
- Lexical analysis using regular expressions
- Tokens and their use
- Parsing strategies including top-down (e.g., recursive descent, or LL) and bottom-up (e.g., LR or GLR) techniques: 1. Lookahead tables and their application to parsing
- Language theory: 1. Chomsky hierarchy 2. Left-most/right-most derivation and ambiguity 3. Grammar transformation
- Parser error recovery mechanisms
- Generating scanners and parsers from declarative specifications

Learning Outcomes:

NonCore:

1. Use formal grammars to specify the syntax of languages [Use]
2. Illustrate the role of lookahead tables in parsing [Apply]

3. Use declarative tools to generate parsers and scanners [Use]
4. Recognize key issues in syntax definitions: ambiguity, associativity, precedence [Recognize]

3.5.17 FPL/Compiler Semantic Analysis

Topics:
Non Core

- Abstract syntax trees; contrast with concrete syntax
- Defining, traversing, and modifying high-level program representations
- Scope and binding resolution
- Static semantics: 1. Type checking. 2. Define before use 3. Annotation and extended static checking frameworks.
- L-values/R-values
- Call semantics
- Types of parameter-passing with simple illustrations and comparison: call by value, call by reference, call by value-result, call by name, call by need and their variations
- Declarative specifications such as attribute grammars and their applications in handling limited context-base grammar

Learning Outcomes:
NonCore:

1. Draw the abstract syntax tree for a small language [Diagram]
2. Implement context-sensitive, source-level static analyses such as type-checkers or resolving identifiers to identify their binding occurrences [Implement]
3. Describe semantic analyses using an attribute grammar [Describe]

3.5.18 FPL/Program Analysis and Analyzers

Topics:
Non Core

- Relevant program representations, such as basic blocks, control-flow graphs, def-use chains, and static single assignment
- Undecidability and consequences for program analysis
- Flow-insensitive analysis, such as type-checking and scalable pointer and alias analysis
- Flow-sensitive analysis, such as forward and backward dataflow analyses
- Path-sensitive analysis, such as software model checking and software verification
- Tools and frameworks for implementing analyzers
- Role of static analysis in program optimization and data dependency analysis during exploitation of concurrency
- Role of program analysis in (partial) verification and bug-finding
- Parallelization: 1. Analysis for auto-parallelization 2. Analysis for detecting concurrency bugs

Learning Outcomes:
NonCore:

1. Explain the difference between dataflow graph and control flow graph [Explain]
2. Explain why non-trivial sound program analyses must be approximate [Explain]
3. Argue why an analysis is correct (sound and terminating) [Argue]
4. Explain why potential aliasing limits sound program analysis and how alias analysis can help [Explain]
5. Use the results of a program analysis for program optimization and/or partial program correctness [Use]

3.5.19 FPL/Code Generation

Topics:
Non Core

- Instruction sets
- Control flow

- Memory management
- Procedure calls and method dispatching
- Separate compilation; linking
- Instruction selection
- Instruction scheduling (e.g., pipelining)
- Register allocation
- Code optimization as a form of program analysis
- Program generation through generative AI

Learning Outcomes:
NonCore:

1. Identify all essential steps for automatically converting source code into assembly or other low-level languages [Analyze]
2. Explain the low-level code necessary for calling functions/methods in modern languages [Explain]
3. Discuss why separate compilation requires uniform calling conventions [Debate]
4. Discuss why separate compilation limits optimization because of unknown effects of calls [Debate]
5. Discuss opportunities for optimization introduced by naive translation and approaches for achieving, optimization, such as instruction selection, instruction scheduling, register allocation, and peephole optimization [Debate]

3.5.20 FPL/Run-time Behavior and Systems

Topics:
Non Core

- Process models using stacks and heaps to allocate and deallocate activation records and recovering environments using frame pointers and return addresses during a procedure call including parameter-passing examples
- Schematics of code lookup using hash tables for methods in implementations of object-oriented programs
- Data layout for objects and activation records
- Object allocation in heap
- Implementing virtual entities and virtual methods; virtual method tables and their application
- Run-time behavior of object-oriented programs
- Compare and contrast allocation of memory during information exchange using parameter-passing and non-local variables (using chain of static links).
- Dynamic memory management approaches and techniques: malloc/free, garbage collection (mark-sweep, copying, reference counting), regions (also known as arenas or zones)
- Just-in-time compilation and dynamic recompilation
- Interface to operating system (e.g., for program initialization)
- Interoperability between programming languages including parameter-passing mechanisms and data representation: 1. Big endian, little endian 2. Data layout of composite data types such as arrays
- Other common features of virtual machines, such as class loading, threads, and security checking
- Sandboxing

Learning Outcomes:
NonCore:

1. Discuss benefits and limitations of automatic memory management [Debate]
2. Explain the use of metadata in run-time representations of objects and activation records, such as class pointers, array lengths, return addresses, and frame pointers [Explain]
3. Compare and contrast static allocation vs stack-based allocation vs heap-based allocation of data elements [Compare]
4. Explain why some data elements cannot be automatically deallocated at the end of a procedure/method call (need for garbage collection) [Explain]

5. Discuss advantages, disadvantages, and difficulties of just-in-time and dynamic recompilation [Debate]
6. Discuss the use of sandboxing in mobile code [Debate]
7. Identify the services provided by modern language run-time systems [Analyze]

3.5.21 FPL/Advanced Programming Constructs

Topics:
Non Core

- Encapsulation mechanisms
- Delayed evaluation and infinite streams
- Compare and contrast delayed evaluation vs eager evaluation
- Unification vs assertion vs expression evaluation
- Control abstractions: exception handling, continuations, monads.
- Object-oriented abstractions: multiple inheritance, mixins, traits, multimethods
- Metaprogramming: macros, generative programming, model-based development
- String manipulation via pattern-matching (regular expressions)
- Dynamic code evaluation ("eval")
- Language support for checking assertions, invariants, and pre/post-conditions
- Domain specific languages, such as database languages, data science languages, embedded computing languages, synchronous languages, hardware interface languages
- Massive parallel high performance computing models and languages

Learning Outcomes:
NonCore:

1. Use various advanced programming constructs and idioms correctly [Use]
2. Discuss how various advanced programming constructs aim to improve program structure, software quality, and programmer productivity [Debate]
3. Discuss how various advanced programming constructs interact with the definition and implementation of other language features [Debate]

3.5.22 FPL/Language Pragmatics

Topics:
Non Core

- Effect of technology needs and software requirements on programming language development and evolution
- Problem domains and programming paradigm
- Criteria for good programming language design: 1. Principles of language design such as orthogonality 2. Defining control and iteration constructs 3. Modularization of large software
- Evaluation order, precedence, and associativity
- Eager vs delayed evaluation
- Defining control and iteration constructs
- External calls and system libraries

Learning Outcomes:
NonCore:

1. Discuss the role of concepts such as orthogonality and well-chosen defaults in language design [Debate]
2. Objectively evaluate and justify language-design decisions [Evaluate]
3. Implement an example program whose result can differ under different rules for evaluation order, precedence, or associativity [Implement]
4. Illustrate uses of delayed evaluation, such as user-defined control abstractions [Apply]
5. Discuss the need for allowing calls to external calls and system libraries and the consequences for language implementation [Debate]

3.5.23 FPL/Formal Semantics

Topics:
Non Core

- Syntax vs semantics
- Approaches to semantics: axiomatic, operational, denotational, type-based
- Axiomatic semantics of abstract constructs such as assignment, selection, iteration using pre-condition, post-conditions, and loop invariant
- Operational semantics analysis of abstract constructs and sequence of such as assignment, expression evaluation, selection, iteration using environment and store: 1. Symbolic execution 2. Constraint checkers
- Denotational semantics: 1. Lambda Calculus.
- Proofs by induction over language semantics
- Formal definitions and proofs for type systems: 1. Propositions as types (implication as a function, conjunction as a product, disjunction as a sum) 2. Dependent types (universal quantification as dependent function, existential quantification as dependent product) 3. Parametricity

Learning Outcomes:
NonCore:

1. Construct formal semantics for a small language [Create]
2. Write a lambda-calculus program and show its evaluation to a normal form [Write]
3. Discuss the different approaches of operational, denotational, and axiomatic semantics [Debate]
4. Use induction to prove properties of all programs in a language [Use]
5. Use induction to prove properties of all programs in a language that is well-typed according to a formally defined type system [Use]
6. Use parametricity to establish the behavior of code given only its type [Use]

3.5.24 FPL/Formal Development Methodologies

Topics:
Non Core

- Formal specification languages and methodologies
- Theorem provers, proof assistants, and logics
- Constraint checkers
- Dependent types (universal quantification as dependent function, existential quantification as dependent product)
- Specification and proof discharge for fully verified software systems using pre/post conditions, refinement types, etc.
- Formal modeling and manual refinement/implementation of software systems.
- Use of symbolic testing and fuzzing in software development.
- Model checking.
- Understanding of situations where formal methods can be effectively applied and how to structure development to maximize their value.

Learning Outcomes:
NonCore:

1. Use formal modeling techniques to develop and validate architectures [Use]
2. Use proof assisted programming languages to develop fully specified and verified software artifacts [Use]
3. Use verifier and specification support in programming languages to formally validate system properties [Use]
4. Integrate symbolic validation tooling into a programming workflow [Integrate]
5. Discuss when and how formal methods can be effectively used in the development process [Debate]

3.5.25 FPL/Design Principles of Programming Languages

Topics:
Non Core

- Language design principles: 1. Simplicity 2. Security 3. Fast translation 4. Efficient object code 5. Orthogonality 6. Readability 7. Completeness 8. Implementation strategies
- Designing a language to fit a specific domain or problem
- Interoperability between programming languages
- Language portability
- Formal description of a programming language
- Green computing principles

Learning Outcomes:
NonCore:

1. Understand what constitutes good language design and apply that knowledge to evaluate a real programming language [Evaluate]

3.5.26 FPL/Society, Ethics, and the Profession

Topics:
Non Core

- Impact of English-centric programming languages
- Enhancing accessibility and inclusivity for people with disabilities - Supporting assistive technologies
- Human factors related to programming languages and usability: 1. Impact of syntax on accessibility 2. Supporting cultural differences (e.g., currency, decimals, dates) 3. Neurodiversity
- Etymology of terms such as ”class,” ”master,” and ”slave” in programming languages
- Increasing accessibility by supporting multiple languages within applications (UTF)

Learning Outcomes:
NonCore:

1. Consciously design programming languages to be inclusive and non-offensive [Design]

3.6 Graphics and Interactive Techniques (GIT)

Graphics and Interactive Techniques (GIT) encompass the principles, algorithms, and technologies involved in rendering, modeling, animating, and interacting with visual and interactive content. This knowledge area integrates concepts from computer graphics, human-computer interaction, visualization, and image processing to create immersive and effective digital experiences. With the proliferation of interactive applications in entertainment, education, scientific visualization, and user interfaces, GIT plays a crucial role in modern computing. Students studying GIT learn to design, implement, and evaluate systems that generate visual content, facilitate user interaction, and leverage emerging technologies such as virtual reality, augmented reality, and tangible computing. The area emphasizes both theoretical foundations and practical skills, preparing graduates to contribute to diverse fields including game development, simulation, data visualization, and interactive media.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.6.1 Fundamental Concepts of Graphics and Interactive Techniques (Pg.. 83)	4	3
⭐ 3.6.2 Visualization (Pg.. 83)		6
⭐ 3.6.3 Applied Rendering and Techniques (Pg.. 83)		15
⭐ 3.6.4 Geometric Modeling (Pg.. 84)		6
⭐ 3.6.5 Shading and Advanced Rendering (Pg.. 84)		6
⭐ 3.6.6 Computer Animation (Pg.. 84)		6
⭐ 3.6.7 Simulation (Pg.. 84)		6
⭐ 3.6.8 Immersion (Pg.. 85)		6
⭐ 3.6.9 Interaction (Pg.. 85)		4
⭐ 3.6.10 Image Processing (Pg.. 85)		6
⭐ 3.6.11 Tangible/Physical Computing (Pg.. 86)		6
3.6.12 Society, Ethics, and the Profession (Pg.. 86)	Elective	

Table 3.6: List of KUs in the Graphics and Interactive Techniques area.

3.6.1 GIT/Fundamental Concepts of Graphics and Interactive Techniques (Core Tier1: 4 hrs, Core Tier2: 3 hrs)

Topics:
Core
- Overview of computer graphics pipeline and interactive systems
- Basic rendering concepts: rasterization, ray tracing, shading models
- Color theory and color spaces
- Coordinate systems and transformations
- Basic interaction techniques: input devices, event handling
- Advanced rendering concepts: global illumination, physically based rendering
- Graphics hardware and acceleration techniques
- Human perception and visual cognition

Learning Outcomes:
Core:
1. Explain the basic stages of the computer graphics pipeline [Explain]
2. Describe basic rendering techniques such as rasterization and ray tracing [Describe]
3. Apply color theory to design visually effective graphics [Apply]
4. Implement coordinate transformations for 2D and 3D graphics [Implement]
5. Design basic interactive applications using common input devices and event handling [Design]
6. Compare advanced rendering techniques such as global illumination and physically based rendering [Compare]
7. Analyze the role of graphics hardware in accelerating rendering tasks [Analyze]
8. Explain how human perception influences graphics and interaction design [Explain]

3.6.2 GIT/Visualization (Core Tier2: 6 hrs)

Topics:
Core
- Principles of data visualization: perception, cognition, and effective encoding
- Types of visualizations: scientific, information, geographical
- Visualization techniques for multivariate, temporal, and hierarchical data
- Interactive visualization and exploration
- Evaluation of visualization effectiveness

Learning Outcomes:
Core:
1. Apply principles of perception and cognition to design effective visualizations [Apply]
2. Select appropriate visualization techniques for different data types and tasks [Evaluate]
3. Implement interactive visualizations for data exploration [Implement]
4. Evaluate the effectiveness of a visualization using empirical methods [Evaluate]

3.6.3 GIT/Applied Rendering and Techniques (Core Tier2: 15 hrs)

Topics:
Core
- Rasterization algorithms: scan conversion, antialiasing
- Ray tracing: acceleration structures, sampling, shading
- Texture mapping and filtering
- Shading models: Lambertian, Phong, BRDF
- Advanced techniques: shadow mapping, environment mapping, deferred rendering

Learning Outcomes:
Core:
1. Implement rasterization algorithms for primitive rendering [Implement]
2. Implement basic ray tracing with acceleration structures [Implement]

3. Apply texture mapping and filtering techniques to enhance visual realism [Apply]
4. Compare different shading models and their applications [Compare]
5. Implement advanced rendering techniques such as shadow mapping or deferred rendering [Implement]

3.6.4 GIT/Geometric Modeling (Core Tier2: 6 hrs)

Topics:
Core

- Geometric representations: polygons, curves, surfaces, implicit forms
- Modeling techniques: splines, subdivision surfaces, constructive solid geometry
- Procedural modeling and generation
- Geometric algorithms: intersection, distance, decomposition

Learning Outcomes:
Core:

1. Describe different geometric representations and their trade-offs [Describe]
2. Implement modeling techniques such as spline curves or subdivision surfaces [Implement]
3. Create procedural models using algorithms or rules [Create]
4. Apply geometric algorithms to solve problems like intersection testing or mesh processing [Apply]

3.6.5 GIT/Shading and Advanced Rendering (Core Tier2: 6 hrs)

Topics:
Core

- Advanced shading: physically based shading, subsurface scattering
- Global illumination: radiosity, photon mapping, path tracing
- Real-time rendering techniques: level of detail, culling, occlusion
- GPU programming and shaders

Learning Outcomes:
Core:

1. Implement advanced shading models for realistic materials [Implement]
2. Compare global illumination algorithms and their trade-offs [Compare]
3. Optimize real-time rendering using techniques like level of detail and culling [Create]
4. Develop GPU shaders for custom rendering effects [Create]

3.6.6 GIT/Computer Animation (Core Tier2: 6 hrs)

Topics:
Core

- Animation principles: timing, spacing, anticipation, follow-through
- Keyframe animation and interpolation
- Procedural animation: particle systems, flocking, physics-based
- Character animation: skeletal animation, skinning, inverse kinematics
- Motion capture and retargeting

Learning Outcomes:
Core:

1. Apply animation principles to create believable motion [Apply]
2. Implement keyframe animation with interpolation techniques [Implement]
3. Create procedural animations such as particle systems or physics-based simulations [Create]
4. Implement character animation using skeletal systems and skinning [Implement]
5. Process motion capture data for animation retargeting [Implement]

3.6.7 GIT/Simulation (Core Tier2: 6 hrs)

Topics:
Core

- Physics-based simulation: rigid bodies, fluids, cloth
- Numerical methods for simulation: integration, collision detection
- Real-time simulation constraints and optimizations
- Applications in games, visual effects, and scientific visualization

Learning Outcomes:

Core:

1. Implement physics-based simulations for rigid bodies or fluids [Implement]
2. Apply numerical integration methods for dynamic simulations [Apply]
3. Optimize simulations for real-time performance [Create]
4. Develop simulations for specific applications such as games or scientific visualization [Create]

3.6.8 GIT/Immersion (Core Tier2: 6 hrs)

Topics:

Core

- Virtual reality (VR) and augmented reality (AR) concepts
- Immersive display technologies: head-mounted displays, CAVEs
- Tracking and input for immersive environments
- Presence and comfort in immersive systems

Learning Outcomes:

Core:

1. Explain the core concepts and differences between VR and AR [Explain]
2. Evaluate immersive display technologies and their applications [Evaluate]
3. Implement tracking and input solutions for immersive environments [Implement]
4. Design immersive experiences that maximize presence and minimize discomfort [Design]

3.6.9 GIT/Interaction (Core Tier2: 4 hrs)

Topics:

Core

- Interaction paradigms: direct manipulation, gesture-based, tangible
- User interface design for graphics applications
- Multimodal interaction: combining visual, auditory, haptic feedback
- Evaluation of interaction techniques: usability, user experience

Learning Outcomes:

Core:

1. Compare different interaction paradigms and their suitability for various tasks [Compare]
2. Design user interfaces for graphics-intensive applications [Design]
3. Implement multimodal interaction systems incorporating multiple feedback channels [Implement]
4. Conduct usability evaluations of interaction techniques [Evaluate]

3.6.10 GIT/Image Processing (Core Tier2: 6 hrs)

Topics:

Core

- Image representation and formats
- Filtering and enhancement: convolution, frequency domain
- Image segmentation and feature detection
- Morphological operations
- Applications in graphics, vision, and multimedia

Learning Outcomes:

Core:

1. Manipulate images using different representations and formats [Process]
2. Apply filtering techniques for image enhancement and analysis [Apply]
3. Implement image segmentation algorithms for feature extraction [Implement]

4. Use morphological operations for image processing tasks [Use]
5. Develop applications that integrate image processing with graphics or vision systems [Create]

3.6.11 GIT/Tangible/Physical Computing (Core Tier2: 6 hrs)

Topics:
Core

- Tangible interfaces and physical computing
- Sensors and actuators for interactive systems
- Embedded systems for interactive art and design
- Interactive installations and environments

Learning Outcomes:
Core:

1. Design tangible interfaces that bridge physical and digital interactions [Design]
2. Integrate sensors and actuators to create responsive systems [Integrate]
3. Develop embedded systems for interactive art or design projects [Create]
4. Create interactive installations that engage users in physical spaces [Create]

3.6.12 GIT/Society, Ethics, and the Profession

Topics:
Core

- Ethical considerations in graphics and interaction: representation, accessibility, privacy
- Social impact of immersive technologies and visualization
- Professional practice in graphics and interactive industries

Learning Outcomes:
Core:

1. Analyze ethical issues related to representation, accessibility, and privacy in graphics and interaction [Analyze]
2. Discuss the social implications of immersive technologies and data visualization [Debate]
3. Describe professional practices and standards in graphics and interactive industries [Describe]

3.7 Human-Computer Interaction (HCI)

Human-Computer Interaction (HCI) is a multidisciplinary field that focuses on the design, evaluation, and implementation of interactive computing systems for human use, and the study of major phenomena surrounding them. HCI integrates knowledge and methods from computer science, psychology, design, sociology, and other fields to create technologies that are useful, usable, and satisfying for people. The core goal of HCI is to improve interactions between users and computers by making systems more usable, accessible, and efficient while considering human capabilities and limitations.

In today's world where computing technology permeates every aspect of daily life-from smartphones and websites to medical devices and smart home systems-HCI principles are essential for creating systems that people can effectively and safely use. HCI emphasizes user-centered design processes, iterative prototyping, and empirical evaluation to ensure that systems meet real user needs. The field addresses diverse topics including usability, user experience (UX), accessibility, interaction design, user research methods, and the social implications of technology.

HCI prepares students to design, implement, and evaluate interactive systems that are not only functional but also consider human factors, ethical implications, and societal impact. Graduates with HCI expertise work in roles such as UX designers, interaction designers, usability engineers, and product managers across industries including software development, healthcare, education, entertainment, and public policy.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.7.1 Understanding the User: Individual goals and interactions with others (Pg.. 87)	2	5
⭐ 3.7.2 Accountability and Responsibility in Design (Pg.. 87)	2	2
⭐ 3.7.3 Accessibility and Inclusive Design (Pg.. 88)	2	2
⭐ 3.7.4 Evaluating the Design (Pg.. 88)	1	2
⭐ 3.7.5 System Design (Pg.. 88)	1	5
3.7.6 Society, Ethics, and the Profession (Pg.. 89)	Elective	

Table 3.7: List of KUs in the Human-Computer Interaction area.

3.7.1 HCI/Understanding the User: Individual goals and interactions with others (Core Tier1: 2 hrs, Core Tier2: 5 hrs)

Topics:
Core

- Human capabilities and limitations: perception, cognition, motor skills
- User characteristics: demographics, expertise, cultural background
- User goals, tasks, and contexts of use
- Social and organizational aspects of interaction
- Cognitive models and theories relevant to HCI
- Emotional and affective aspects of interaction
- User research methods: interviews, surveys, observations

Learning Outcomes:
Core:

1. Describe human capabilities and limitations that affect interaction design [Describe]
2. Analyze how user characteristics influence interaction with computing systems [Analyze]
3. Identify user goals and tasks in specific contexts of use [Analyze]
4. Explain social and organizational factors that impact technology adoption and use [Explain]
5. Apply cognitive models to predict and explain user behavior [Apply]
6. Consider emotional and affective factors in interaction design [Analyze]
7. Conduct user research using appropriate methods such as interviews or observations [Evaluate]

3.7.2 HCI/Accountability and Responsibility in Design (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:
Core

- Ethical considerations in HCI: fairness, transparency, accountability
- Design for values and value-sensitive design
- Responsible innovation and design ethics
- Algorithmic accountability and bias in interactive systems
- Professional responsibility in HCI practice
- Legal and regulatory considerations for interactive systems

Learning Outcomes:
Core:

1. Identify ethical issues in HCI design and implementation [Analyze]
2. Apply value-sensitive design approaches to technology development [Apply]
3. Discuss responsible innovation practices in HCI [Debate]
4. Analyze algorithmic accountability and bias in interactive systems [Analyze]
5. Describe professional responsibilities in HCI practice [Describe]
6. Explain legal and regulatory considerations relevant to interactive systems [Explain]

3.7.3 HCI/Accessibility and Inclusive Design (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:
Core

- Accessibility principles and guidelines
- Disabilities and assistive technologies
- Inclusive design and universal design principles
- Evaluation methods for accessibility
- Designing for diverse abilities and contexts
- Accessibility standards and compliance

Learning Outcomes:
Core:

1. Apply accessibility principles to interactive system design [Apply]
2. Describe how different disabilities affect interaction with technology [Describe]
3. Design inclusive systems using universal design principles [Design]
4. Conduct accessibility evaluations of interactive systems [Evaluate]
5. Create designs that accommodate diverse abilities and usage contexts [Create]
6. Ensure compliance with accessibility standards and regulations [Evaluate]

3.7.4 HCI/Evaluating the Design (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Usability principles and heuristics
- Evaluation methods: heuristic evaluation, cognitive walkthrough
- User testing: planning, conducting, analyzing
- Advanced evaluation methods: controlled experiments, field studies
- UX metrics and measurement
- Analytical evaluation methods: modeling and simulation

Learning Outcomes:
Core:

1. Apply usability principles and heuristics to evaluate designs [Apply]
2. Conduct heuristic evaluations and cognitive walkthroughs [Evaluate]
3. Plan, conduct, and analyze user testing sessions [Plan]
4. Design and conduct controlled experiments for HCI research [Design]
5. Measure user experience using appropriate metrics [Evaluate]
6. Use analytical methods to model and predict user performance [Use]

3.7.5 HCI/System Design (Core Tier1: 1 hr, Core Tier2: 5 hrs)

Topics:
Core

- Design process: user-centered design, iterative design
- Interaction design: concepts, principles, patterns
- Prototyping techniques: low-fidelity, high-fidelity, interactive
- Information architecture and navigation design
- Advanced interaction techniques: gesture, voice, multimodal
- Design for specific domains: mobile, web, ubiquitous computing
- Collaborative and social computing systems

Learning Outcomes:
Core:

1. Apply user-centered design processes to develop interactive systems [Apply]
2. Design interactions using established principles and patterns [Design]
3. Create prototypes at different fidelity levels for design exploration [Create]
4. Structure information and navigation for effective user interaction [Structure]

5. Implement advanced interaction techniques such as gesture or voice interfaces [Implement]
6. Design interactive systems for specific domains and platforms [Design]
7. Design collaborative systems that support group interaction [Design]

3.7.6 HCI/Society, Ethics, and the Profession

Topics:
Core
- Social impact of interactive technologies
- Ethical issues in HCI research and practice
- Professional practice in HCI
- Cultural and cross-cultural considerations in HCI
- Sustainability and HCI
- Future trends and challenges in HCI

Learning Outcomes:
Core:
1. Analyze the social impact of interactive technologies on individuals and society [Analyze]
2. Address ethical issues in HCI research and design projects [Evaluate]
3. Describe professional practices and career paths in HCI [Describe]
4. Design systems that account for cultural differences and contexts [Design]
5. Incorporate sustainability considerations into HCI design [Integrate]
6. Discuss emerging trends and future challenges in HCI [Debate]

3.8 Mathematical and Statistical Foundations (MSF)

The mathematical and statistical foundations of computer science provide the theoretical underpinnings for the discipline. They include discrete mathematics, probability, statistics, linear algebra, and calculus. These areas are essential for understanding algorithms, data structures, machine learning, graphics, cryptography, and many other topics in computer science.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.8.1 Discrete Mathematics (Pg.. 89)	15	6
⭐ 3.8.2 Advanced Discrete Mathematics (Pg.. 90)	15	6
⭐ 3.8.3 Probability (Pg.. 90)	11	29
⭐ 3.8.4 Statistics (Pg.. 91)	10	30
⭐ 3.8.5 Linear Algebra (Pg.. 91)	5	35
⭐ 3.8.6 Calculus (Pg.. 92)		40

Table 3.8: List of KUs in the Mathematical and Statistical Foundations area.

3.8.1 MSF/Discrete Mathematics (Core Tier1: 15 hrs, Core Tier2: 6 hrs)

Topics:
Core
- Sets: operations, cardinality, power sets, Cartesian products
- Relations: properties, equivalence relations, partial orders
- Functions: injective, surjective, bijective, composition, inverse
- Logic: propositional and predicate logic, quantifiers, proofs
- Graphs: definitions, paths, cycles, trees, connectivity
- Counting: permutations, combinations, binomial theorem
- Recurrence relations: solving simple recurrences

Learning Outcomes:
Core:
1. Define basic set operations and relations [Explain]
2. Apply set operations to solve problems [Apply]

3. Define properties of relations [Explain]
4. Determine if a relation is an equivalence relation [Apply]
5. Define types of functions [Explain]
6. Compose functions and find inverses [Apply]
7. Explain propositional and predicate logic [Explain]
8. Construct simple logical proofs [Apply]
9. Define basic graph concepts [Explain]
10. Solve basic graph problems [Apply]
11. Explain permutations and combinations [Explain]
12. Apply counting techniques to solve problems [Apply]
13. Define recurrence relations [Explain]
14. Solve simple recurrence relations [Apply]

3.8.2 MSF/Advanced Discrete Mathematics (Core Tier1: 15 hrs, Core Tier2: 6 hrs)

Topics:
Core

- Advanced set theory: infinite sets, Cantor's diagonal argument
- Advanced relations: closures, lattices
- Advanced functions: recursive functions, computability
- Advanced logic: proof systems, completeness, decidability
- Advanced graph theory: planarity, coloring, networks
- Advanced counting: generating functions, inclusion-exclusion
- Advanced recurrence: generating functions, asymptotic behavior

Learning Outcomes:
Core:

1. Explain concepts of infinite sets and cardinality [Explain]
2. Apply closure operations to relations [Apply]
3. Analyze advanced graph properties [Evaluate]
4. Develop generating functions for counting problems [Create]

3.8.3 MSF/Probability (Core Tier1: 11 hrs, Core Tier2: 29 hrs)

Topics:
Core

- Basic probability: sample spaces, events, axioms
- Conditional probability, independence, Bayes' theorem
- Random variables: discrete and continuous, distributions
- Expectation, variance, standard deviation
- Common distributions: uniform, binomial, normal
- Joint distributions, covariance, correlation
- Limit theorems: law of large numbers, central limit theorem
- Stochastic processes: Markov chains
- Information theory: entropy, mutual information

Learning Outcomes:
Core:

1. Define basic probability concepts [Explain]
2. Calculate probabilities of events [Apply]
3. Define conditional probability and independence [Explain]
4. Apply Bayes' theorem to update probabilities [Apply]
5. Define random variables and distributions [Explain]
6. Compute expectation and variance [Apply]
7. Describe common probability distributions [Explain]
8. Apply common distributions to model real-world phenomena [Apply]
9. Explain joint distributions and dependence [Explain]

10. Compute covariance and correlation [Apply]
11. Analyze the implications of limit theorems [Evaluate]
12. Model simple systems using Markov chains [Create]

3.8.4 MSF/Statistics (Core Tier1: 10 hrs, Core Tier2: 30 hrs)

Topics:
Core

- Descriptive statistics: measures of center and spread
- Data visualization: histograms, box plots, scatter plots
- Sampling distributions: sampling variability, central limit theorem
- Estimation: point estimates, confidence intervals
- Hypothesis testing: null and alternative hypotheses, p-values
- Regression: simple and multiple linear regression
- Analysis of variance (ANOVA)
- Nonparametric methods
- Experimental design: randomization, control, blinding

Learning Outcomes:
Core:

1. Define descriptive statistics measures [Explain]
2. Compute descriptive statistics for a dataset [Apply]
3. Explain common data visualization techniques [Explain]
4. Create appropriate visualizations for given data [Apply]
5. Define sampling distributions and variability [Explain]
6. Apply the central limit theorem to approximate distributions [Apply]
7. Define point estimates and confidence intervals [Explain]
8. Compute confidence intervals for means and proportions [Apply]
9. Define hypothesis testing concepts [Explain]
10. Conduct a simple hypothesis test and interpret p-values [Apply]
11. Explain simple and multiple linear regression [Explain]
12. Fit a regression model and interpret coefficients [Apply]
13. Analyze variance using ANOVA [Evaluate]
14. Design a simple experiment with appropriate controls [Create]

3.8.5 MSF/Linear Algebra (Core Tier1: 5 hrs, Core Tier2: 35 hrs)

Topics:
Core

- Vectors: operations, dot product, cross product
- Matrices: operations, inverses, determinants
- Systems of linear equations: Gaussian elimination, matrix form
- Vector spaces: subspaces, basis, dimension
- Linear transformations: matrix representation, kernel, image
- Eigenvalues and eigenvectors: diagonalization
- Orthogonality: orthonormal bases, Gram-Schmidt
- Singular value decomposition (SVD)
- Applications: least squares, computer graphics, machine learning

Learning Outcomes:
Core:

1. Define vector operations and properties [Explain]
2. Compute vector operations and products [Apply]
3. Define matrix operations and properties [Explain]
4. Compute matrix operations, inverses, and determinants [Apply]
5. Explain systems of linear equations and solution methods [Explain]
6. Solve systems of linear equations using elimination [Apply]
7. Define vector spaces, subspaces, basis, and dimension [Explain]

8. Determine if a set of vectors forms a basis [Apply]
9. Define linear transformations and their representations [Explain]
10. Find the kernel and image of a linear transformation [Apply]
11. Explain eigenvalues and eigenvectors [Explain]
12. Diagonalize a matrix when possible [Apply]
13. Analyze orthogonality and orthonormal bases [Evaluate]
14. Apply singular value decomposition to a matrix [Create]

3.8.6 MSF/Calculus (Core Tier2: 40 hrs)

Topics:
Core

- Limits and continuity
- Derivatives: definition, rules, applications
- Integrals: definite and indefinite, fundamental theorem
- Series: convergence, Taylor series
- Multivariable calculus: partial derivatives, gradients
- Optimization: local and global extrema, Lagrange multipliers
- Differential equations: basic solution techniques
- Numerical methods: integration, differentiation

Learning Outcomes:
Core:

1. Define limits and continuity [Explain]
2. Compute limits of functions [Apply]
3. Define derivatives and differentiation rules [Explain]
4. Compute derivatives of functions [Apply]
5. Define integrals and the fundamental theorem of calculus [Explain]
6. Compute definite and indefinite integrals [Apply]
7. Define series and convergence [Explain]
8. Compute Taylor series expansions [Apply]
9. Explain partial derivatives and gradients [Explain]
10. Compute partial derivatives and gradients [Apply]
11. Analyze optimization problems using calculus [Evaluate]
12. Solve simple differential equations [Create]

3.9 Networking and Communication (NC)

Networking and communication play a central role in interconnected computer systems that are transforming the daily lives of billions of people. The public internet provides connectivity for networked applications that serve ever-increasing numbers of individuals and organizations around the world. Complementing the public sector, major proprietary networks leverage their global footprints to support cost-effective distributed computing, storage, and content delivery. Advances in satellite networks expand connectivity to rural areas. Device-to-device communication underlies the emerging Internet of Things.

This knowledge area deals with key concepts in networking and communication, as well as their representative instantiations in the internet and other computer networks. Besides the basic principles of switching and layering, the area at its core provides knowledge on naming, addressing, reliability, error control, flow control, congestion control, domain hierarchy, routing, forwarding, modulation, encoding, framing, and access control. The area also covers knowledge units in network security and mobility, such as security threats, countermeasures, device-to-device communication, and multi-hop wireless networking. In addition to the fundamental principles, the area includes their specific realization of the Internet as well as hands-on skills in the implementation of networking and communication concepts. Finally, the area comprises emerging topics such as network virtualization and quantum networking.

As the main learning outcome, learners develop a thorough understanding of the role and operation of networking and communication in networked computer systems. They learn how network structure

and communication protocols affect the behavior of distributed applications. The area can be used to educate not only key principles but also their specific instantiations in the internet and equip the student with hands-on implementation skills. While computer-system, networking, and communication technologies are advancing at a fast pace, the gained fundamental knowledge enables the student to readily apply the concepts in new technological settings.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.9.1 Fundamentals of Networks and Communications (Pg.. 93)	3	
⭐ 3.9.2 Networked Applications (Pg.. 93)	4	
⭐ 3.9.3 Reliability Support (Pg.. 94)		6
⭐ 3.9.4 Routing and Forwarding (Pg.. 94)		4
⭐ 3.9.5 Single Hop Communication (Pg.. 94)		3
⭐ 3.9.6 Network Security (Pg.. 95)		3
⭐ 3.9.7 Mobility Support (Pg.. 95)		4
⭐ 3.9.8 Emerging Topics (Pg.. 95)		4

Table 3.9: List of KUs in the Networking and Communication area.

3.9.1 NC/Fundamentals of Networks and Communications (Core Tier1: 3 hrs)

Topics:
Core

- Importance of networking in contemporary computing, and associated challenges. See also: Digital Equity and Accessibility, Social Impact and AI, Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation
- Organization of the internet (e.g., users, Internet Service Providers, autonomous systems, content providers, content delivery networks)
- Switching techniques (e.g., circuit and packet)
- Layers and their roles (application, transport, network, datalink, and physical)
- Layering principles (e.g., encapsulation and hourglass model). See also: Digital Systems and Logic, Sequential and Parallel Processing, State Machines
- Network elements (e.g., routers, switches, hubs, access points, and hosts)
- Basic queueing concepts (e.g., relationship with latency, congestion, service levels, etc.)

Learning Outcomes:
Core:

1. Articulate the organization of the internet [Articulate]
2. List and define the appropriate network terminology [List]
3. Describe the layered structure of a typical networked architecture [Describe]
4. Identify the different types of complexity in a network (edges, core, etc.) [Analyze]

3.9.2 NC/Networked Applications (Core Tier1: 4 hrs)

Topics:
Core

- Naming and address schemes (e.g., DNS, and Uniform Resource Identifiers)
- Distributed application paradigms (e.g., client/server, peer-to-peer, cloud, edge, and fog). See also: Communication, Coordination
- Diversity of networked application demands (e.g., latency, bandwidth, and loss tolerance). See also: Communication, Sustainable Design and Pervasive Computing, Environmental Footprint of Computing Systems, Systemic Effects and Social Context, Digital Equity and Accessibility, Social Impact and AI
- Coverage of application-layer protocols (e.g., HTTP)
- Interactions with TCP, UDP, and Socket APIs. See also: Programs: Distribution, Programs: Execution Properties, GPU Programming, Programs: Implementation Mappings, Programs: Declarative Parallelism, Programs: Starting Activities

Learning Outcomes:
Core:
1. Define the principles of naming, addressing, resource location [Define]
2. Analyze the needs of specific networked application demands [Analyze]
3. Describe the details of one application layer protocol [Describe]
4. Implement a simple client-server socket-based application [Implement]

3.9.3 NC/Reliability Support (Core Tier2: 6 hrs)

Topics:
Core
- Unreliable delivery (e.g., UDP)
- Principles of reliability (e.g., delivery without loss, duplication, or out of order). See also: System Reliability
- Error control (e.g., retransmission, error correction)
- Flow control (e.g., stop and wait, window based)
- Congestion control (e.g., implicit and explicit congestion notification)
- TCP and performance issues (e.g., Tahoe, Reno, Vegas, Cubic)

Learning Outcomes:
Core:
1. Describe the operation of reliable delivery protocols [Describe]
2. List the factors that affect the performance of reliable delivery protocols [List]
3. Describe some TCP reliability design issues [Describe]
4. Design and implement a simple reliable protocol [Design]

3.9.4 NC/Routing and Forwarding (Core Tier2: 4 hrs)

Topics:
Core
- Routing paradigms and hierarchy (e.g., intra/inter domain, centralized and decentralized, source routing, virtual circuits, QoS)
- Forwarding methods (e.g., forwarding tables and matching algorithms)
- IP and Scalability issues (e.g., NAT, CIDR, BGP, different versions of IP)

Learning Outcomes:
Core:
1. Describe various routing paradigms and hierarchies [Describe]
2. Describe how packets are forwarded in an IP network [Describe]
3. Describe how the Internet tackles scalability challenges [Describe]

3.9.5 NC/Single Hop Communication (Core Tier2: 3 hrs)

Topics:
Core
- Introduction to modulation, bandwidth, and communication media
- Encoding and Framing
- Medium Access Control (MAC) (e.g., random access and scheduled access)
- Ethernet and WiFi
- Switching (e.g., spanning trees, VLANS).
- Local Area Network Topologies (e.g., data center, campus networks).

Learning Outcomes:
Core:
1. Describe some basic aspects of modulation, bandwidth, and communication media [Describe]
2. Describe in detail a MAC protocol [Describe]
3. Demonstrate understanding of encoding and framing solution tradeoffs [Demonstrate]
4. Describe details of the implementation of Ethernet [Describe]
5. Describe how switching works [Describe]

6. Describe one kind of a LAN topology [Describe]

3.9.6 NC/Network Security (Core Tier2: 3 hrs)

Topics:
Core

- General intro about security (Threats, vulnerabilities, and countermeasures). See also: Computer Crimes and Attacks, Security Laws, Policies and Responsibility, AI and Hardware Security, Access Control and Applications, Security Mindset and Principles, Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering
- Network specific threats and attack types (e.g., denial of service, spoofing, sniffing and traffic redirection, attacker-in-the-middle, message integrity attacks, routing attacks, ransomware, and traffic analysis). See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles, Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering
- Countermeasures: 1. Cryptography (e.g. SSL, TLS, symmetric/asymmetric) 2. Architectures for secure networks (e.g., secure channels, secure routing protocols, secure DNS, VPNs, DMZ, Zero Trust Network Access, hyper network security, anonymous communication protocols, isolation) 3. Network monitoring, intrusion detection, firewalls, spoofing and DoS protection, honeypots, tracebacks, BGP Sec, RPKI

Learning Outcomes:
Core:

1. Describe some of the threat models of network security [Describe]
2. Describe specific network-based countermeasures [Describe]
3. Analyze various aspects of network security from a case study [Analyze]

3.9.7 NC/Mobility Support (Core Tier2: 4 hrs)

Topics:
Core

- Principles of cellular communication (e.g., 4G, 5G)
- Principles of Wireless LANs (mainly 802.11)
- Device to device communication (e.g., IoT communication)
- Multi-hop wireless networks (e.g., ad hoc networks, opportunistic, delay tolerant)

Learning Outcomes:
Core:

1. Describe some aspects of cellular communication such as registration [Describe]
2. Describe how 802.11 supports mobile users [Describe]
3. Describe practical uses of device-to-device communication, as well as multihop [Describe]
4. Describe one type of mobile network such as ad hoc [Describe]

3.9.8 NC/Emerging Topics (Core Tier2: 4 hrs)

Topics:
Core

- Middleboxes (e.g., advances in usage of AI, intent-based networking, filtering, deep packet inspection, load balancing, NAT, CDN)
- Network Virtualization (e.g., SDN, Data Center Networks)
- Quantum Networking (e.g., Intro to the domain, teleportation, security, Quantum Internet)
- Satellite, mmWave, Visible Light

Learning Outcomes:
Core:

1. Describe the value of advances in middleboxes in networks [Describe]
2. Describe the importance of Software Defined Networks [Describe]
3. Describe some of the added value achieved by using Quantum Networking [Describe]

3.10 Operating Systems (OS)

The operating system is a collection of services needed to safely interface the hardware with applications. Core topics focus on the mechanisms and policies needed to virtualize computation, memory, and Input/Output (I/O). Overarching themes that are reused at many levels in computer systems are well illustrated in operating systems (e.g., polling vs interrupts, caching, flexibility vs costs, scheduling approaches to processes, page replacement, etc.). The Operating Systems knowledge area contains the key underlying concepts for other knowledge areas - trust boundaries, concurrency, persistence, and safe extensibility.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.10.1 Role and Purpose of Operating Systems (Pg.. 96)	2	
⭐ 3.10.2 Principles of Operating System (Pg.. 97)	2	
⭐ 3.10.3 Concurrency (Pg.. 97)	2	1
⭐ 3.10.4 Protection and Safety (Pg.. 98)	2	1
⭐ 3.10.5 Scheduling (Pg.. 99)		2
⭐ 3.10.6 Process Model (Pg.. 99)		2
⭐ 3.10.7 Memory Management (Pg.. 100)		1
⭐ 3.10.8 Device management (Pg.. 100)		1
⭐ 3.10.9 File Systems API and Implementation (Pg.. 101)		2
⭐ 3.10.10 Advanced File systems (Pg.. 101)		1
⭐ 3.10.11 Virtualization (Pg.. 102)		1
⭐ 3.10.12 Real-time and Embedded Systems (Pg.. 102)		1
⭐ 3.10.13 Fault tolerance (Pg.. 103)		1
3.10.14 Society, Ethics, and the Profession (Pg.. 103)	Elective	

Table 3.10: List of KUs in the Operating Systems area.

3.10.1 OS/Role and Purpose of Operating Systems (Core Tier1: 2 hrs)

Topics:
Core

- Operating systems mediate between general purpose hardware and application-specific software.
- Universal operating system functions (e.g., process, user and device interfaces, persistence of data)
- Extended and/or specialized operating system functions (e.g., embedded systems, server types such as file, web, multimedia, boot loaders and boot security)
- Design issues (e.g., efficiency, robustness, flexibility, portability, security, compatibility, power, safety, tradeoffs between error checking and performance, flexibility and performance, and security and performance). See also: Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering
- Influences of security, networking, multimedia, parallel and distributed computing
- Overarching concern of security/protection: Neglecting to consider security at every layer creates an opportunity to inappropriately access resources. 1. Unauthorized access to files on an unencrypted drive can be achieved by moving the media to another computer. 2. Operating systems enforced security can be defeated by infiltrating the boot layer before the operating system is loaded. 3. Process isolation can be subverted by inadequate authorization checking at API boundaries. 4. Vulnerabilities in system firmware can provide attack vectors that bypass the operating system entirely. 5. Improper isolation of virtual machine memory, computing, and hardware can expose the host system to attacks from guest systems. 6. The operating system may need to mitigate exploitation of hardware and firmware vulnerabilities, leading to potential performance reductions (e.g., Spectre and Meltdown mitigations).
- Exposure of operating systems functions in shells and systems programming. See also: Shell Scripting

Learning Outcomes:
Core:

1. Understand the objectives and functions of modern operating systems [Explain]
2. Evaluate the design issues in different usage scenarios (e.g., real time OS, mobile, server) [Evaluate]
3. Understand the functions of a contemporary operating system with respect to convenience, efficiency, and the ability to evolve [Explain]
4. Understand how evolution and stability are desirable and mutually antagonistic in operating systems function [Explain]

3.10.2 OS/Principles of Operating System (Core Tier1: 2 hrs)

Topics:
Core

- Operating system software design and approaches (e.g., monolithic, layered, modular, microkernel, unikernel)
- Abstractions, processes, and resources
- Concept of system calls and links to application program interfaces (e.g., Win32, Java, Posix). See also: Assembly Level Machine Organization
- The evolution of the link between hardware architecture and the operating system functions
- Protection of resources means protecting some machine instructions/functions. See also: Assembly Level Machine Organization 1. Applications cannot arbitrarily access memory locations or file storage device addresses. 2. Protection of coprocessors and network devices
- Leveraging interrupts from hardware level: service routines and implementations. See also: Assembly Level Machine Organization 1. Timer interrupts for implementing time slices 2. I/O interrupts for putting blocking threads to sleep without polling
- Concept of user/system state and protection, transition to kernel mode using system calls. See also: Assembly Level Machine Organization
- Mechanism for invoking system calls, the corresponding mode and context switch and return from interrupt. See also: Assembly Level Machine Organization
- Performance costs of context switches and associated cache flushes when performing process switches in Spectre-mitigated environments.

Learning Outcomes:
Core:

1. Understand how the application of software design approaches to operating systems design/implementation (e.g., layered, modular, etc.) affects the robustness and maintainability of an operating system [Explain]
2. Categorize system calls by purpose [Categorize]
3. Understand dynamics of invoking a system call (e.g., passing parameters, mode change) [Explain]
4. Evaluate whether a function can be implemented in the application layer or can only be accomplished by system calls [Evaluate]
5. Apply OS techniques for isolation, protection, and throughput across OS functions (e.g., starvation similarities in process scheduling, disk request scheduling, semaphores, etc.) and beyond [Apply]
6. Understand how the separation into kernel and user mode affects safety and performance [Explain]
7. Understand the advantages and disadvantages of using interrupt processing in enabling multiprogramming [Explain]
8. Analyze potential vectors of attack via the operating systems and the security features designed to guard against them [Analyze]

3.10.3 OS/Concurrency (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:
Core

- Thread abstraction relative to concurrency
- Race conditions, critical regions (role of interrupts, if needed). See also: Programs: Distribution, Programs: Execution Properties, GPU Programming, Programs: Implementation Mappings, Programs: Declarative Parallelism, Programs: Starting Activities
- Deadlocks and starvation. See also: Coordination
- Multiprocessor issues (spin-locks, reentrancy).
- Multiprocess concurrency vs multithreading
- Thread creation, states, structures. See also: Digital Systems and Logic, Sequential and Parallel Processing, State Machines
- Thread APIs
- Deadlocks and starvation (necessary conditions/mitigations). See also: Coordination
- Implementing thread safe code (semaphores, mutex locks, condition variables). See also: Performance and Energy Efficiency, Events, Tools and Experimentation, Performance Metrics and Benchmarks, Analytical Performance Models, Evaluation
- Race conditions in shared memory. See also: Coordination

Non Core

- Managing atomic access to OS objects (e.g., big kernel lock vs many small locks vs lockless data structures like lists)

Learning Outcomes:

Core:

1. Understand the advantages and disadvantages of concurrency as inseparable functions within the operating system framework [Explain]
2. Understand how architecture level implementation results in concurrency problems including race conditions [Explain]
3. Understand concurrency issues in multiprocessor systems [Explain]
4. Understand the range of mechanisms that can be employed at the operating system level to realize concurrent systems and describe the benefits of each [Explain]
5. Understand techniques for achieving synchronization in an operating system (e.g., describe how a semaphore can be implemented using OS primitives) including intra-concurrency control and use of hardware atomics [Explain]
6. Accurately analyze code to identify race conditions and appropriate solutions for addressing race conditions [Analyze]

3.10.4 OS/Protection and Safety (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:

Core

- Overview of operating system security mechanisms. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles
- Attacks and antagonism (scheduling, etc.). See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles
- Review of major vulnerabilities in real operating systems. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles
- Operating systems mitigation strategies such as backups. See also: System Reliability
- Policy/mechanism separation. See also: Risk and Data Management, Security Governance, Management and Policy
- Security methods and devices. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles 1. Rings of protection (history from Multics to virtualized x86) 2. x86_64 rings -1 and -2 (hypervisor and ME/PSP)
- Protection, access control, and authentication. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles, Protocols, Applications and Advanced Cryptography, Cryptographic Algorithms, Cryptography Fundamentals

Learning Outcomes:

Core:

1. Understand the requirement for protection and security mechanisms in operating systems [Explain]

2. List and describe the attack vectors that leverage OS vulnerabilities [List]
3. Understand the mechanisms available in an OS to control access to resources [Explain]
4. Summarize the features and limitations of an operating system that impact protection and security [Summarize]

3.10.5 OS/Scheduling (Core Tier2: 2 hrs)

Topics:
Core

- Preemptive and non-preemptive scheduling
- Schedulers and policies (e.g., first come, first serve, shortest job first, priority, round robin, multilevel). See also: Resource Management
- Concepts of Symmetric Multi-Processor (SMP) scheduling and cache coherence. See also: Memory Hierarchy
- Timers (e.g., building many timers out of finite hardware timers). See also: Assembly Level Machine Organization
- Fairness and starvation

Non Core

- Subtopics of operating systems such as energy-aware scheduling and real-time scheduling. See also: Performance and Energy Efficiency, Embedded Characteristics and Architectures, Real-Time, Memory and Verification, Systems, Programming and Sensors, Mobile Platforms: Capabilities and Asynchronous Computing, Mobile Platforms: Architecture and Constraints, Mobile Platforms: Development Fundamentals
- Cooperative scheduling, such as Linux futexes and userland scheduling.

Learning Outcomes:
Core:

1. Compare and contrast the common algorithms used for both preemptive and non-preemptive scheduling of tasks in operating systems, such as priority, performance comparison, and fair-share schemes [Compare]
2. Explain the relationships between scheduling algorithms and application domains [Explain]
3. Explain the distinctions among types of processor scheduler such as short-term, medium-term, long-term, and I/O [Explain]
4. Evaluate a problem or solution to determine appropriateness for asymmetric and/or symmetric multiprocessing [Evaluate]
5. Evaluate a problem or solution to determine appropriateness as a process vs threads [Evaluate]
6. List some contexts benefitting from preemption and deadline scheduling [List]

NonCore:

7. Explain the ways that the logic embodied in scheduling algorithms are applicable to other operating systems mechanisms, such as first come first serve or priority to disk I/O, network scheduling, project scheduling, and problems beyond computing [Explain]

3.10.6 OS/Process Model (Core Tier2: 2 hrs)

Topics:
Core

- Processes and threads relative to virtualization protected memory, process state, memory isolation, etc.
- Memory footprint/segmentation (e.g., stack, heap, etc.). See also: Assembly Level Machine Organization
- Creating and loading executables, shared libraries, and dynamic linking. See also: Language Translation and Execution
- Dispatching and context switching. See also: Assembly Level Machine Organization
- Interprocess communication (e.g., shared memory, message passing, signals, environment variables). See also: Communication

Learning Outcomes:
Core:

1. Understand how processes and threads use concurrency features to virtualize control [Explain]
2. Understand reasons for using interrupts, dispatching, and context switching to support concurrency and virtualization in an operating system [Explain]
3. Understand the different states that a task may pass through, and the data structures needed to support the management of many tasks [Explain]
4. Understand the different ways of allocating memory to tasks, citing the relative merits of each [Explain]
5. Apply the appropriate interprocess communication mechanism for a specific purpose in a programmed software artifact [Apply]

3.10.7 OS/Memory Management (Core Tier2: 1 hr)

Topics:
Core

- Review of physical memory, address translation and memory management hardware See also: Memory Hierarchy, Discrete Mathematics
- Impact of memory hierarchy including cache concept, cache lookup, and per-CPU caching on operating system mechanisms and policy See also: Memory Hierarchy, Latency, Cache and Memory Hierarchy, Virtualization and Isolation
- Logical and physical addressing, address space virtualization. See also: Memory Hierarchy, Discrete Mathematics
- Concepts of paging, page replacement, thrashing and allocation of pages and frames
- Allocation/deallocation/storage techniques (algorithms and data structure) performance and flexibility 1. Arenas, slab allocators, free lists, size classes, heterogeneously sized pages (huge pages)
- Memory caching and cache coherence and the effect of flushing the cache to avoid speculative execution vulnerabilities. See also: Functional Organization, Memory Hierarchy, Latency, Cache and Memory Hierarchy, Virtualization and Isolation
- Security mechanisms and concepts in memory management including sandboxing, protection, isolation, and relevant vectors of attack. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles

Non Core

- Virtual memory: leveraging virtual memory hardware for OS services and efficiency

Learning Outcomes:
Core:

1. Explain memory hierarchy and cost-performance tradeoffs [Explain]
2. Summarize the principles of virtual memory as applied to caching and paging [Summarize]
3. Evaluate the tradeoffs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed [Evaluate]
4. Describe the reason for and use of cache memory (performance and proximity, how caches complicate isolation and virtual machine abstraction) [Describe]
5. Code/Develop efficient programs that consider the effects of page replacement and frame allocation on the performance of a process and the system in which it executes [Create]

NonCore:

6. Explain how hardware is utilized for efficient virtualization [Explain]

3.10.8 OS/Device management (Core Tier2: 1 hr)

Topics:
Core

- Buffering strategies. See also: Interfacing and Communication
- Direct Memory Access (DMA) and polled I/O, Memory-mapped I/O. See also: Interfacing and Communication 1. DMA communication protocols (e.g., ring buffers etc.)
- Historical and contextual - Persistent storage device management (e.g., magnetic, Solid State Device (SSD)). See also: Foundations and Early History of Computing, Modern History: PC, Internet and AI

Non Core

- Device interface abstractions, hardware abstraction layer
- Device driver purpose, abstraction, implementation, and testing challenges
- High-level fault tolerance in device communication

Learning Outcomes:

Core:

1. Explain architecture level device control implementation and link relevant operating system mechanisms and policy (e.g., buffering strategies, direct memory access) [Explain]
2. Explain OS device management layers and the architecture (e.g., device controller, device driver, device abstraction) [Explain]
3. Explain the relationship between the physical hardware and the virtual devices maintained by the operating system [Explain]
4. Explain I/O data buffering and describe strategies for implementing it [Explain]
5. Describe the advantages and disadvantages of direct memory access and discuss the circumstances in which its use is warranted [Describe]

NonCore:

6. Describe the complexity and best practices for the creation of device drivers [Describe]

3.10.9 OS/File Systems API and Implementation (Core Tier2: 2 hrs)

Topics:

Core

- Concept of a file including data, metadata, operations, and access-mode
- File system mounting
- File access control
- File sharing
- Basic file allocation methods, including linked allocation table
- File system structures comprising file allocation including various directory structures and methods for uniquely identifying files (e.g., name, identified or metadata storage location)
- Allocation/deallocation/storage techniques (algorithms and data structure) impact on performance and flexibility (i.e., internal and external fragmentation and compaction)
- Free space management such as using bit tables vs linking
- Implementation of directories to segment and track file location

Learning Outcomes:

Core:

1. Explain the choices to be made in designing file systems [Explain]
2. Evaluate different approaches to file organization, recognizing the strengths and weaknesses of each [Evaluate]
3. Apply software constructs appropriately given knowledge of the file system implementation [Apply]

3.10.10 OS/Advanced File systems (Core Tier2: 1 hr)

Topics:

Core

- File systems: partitioning, mount/unmount, virtual file systems
- In-depth implementation techniques
- Memory-mapped files. See also: Interfacing and Communication
- Special-purpose file systems
- Naming, searching, access, backups
- Journaling and log-structured file systems. See also: System Reliability

Non Core

- Distributed file systems
- Encrypted file systems
- Fault tolerance

Learning Outcomes:
Core:
1. Explain how hardware developments have led to changes in the priorities for the design and the management of file systems [Explain]
2. Map file abstractions to a list of relevant devices and interfaces [Map]
3. Identify and categorize different mount types [Analyze]
4. Explain specific file systems requirements and the specialized file systems features that meet those requirements [Explain]
5. Explain the use of journaling and how log-structured file systems enhance fault tolerance [Explain]

NonCore:
6. Explain purpose and complexity of distributed file systems [Explain]
7. List examples of distributed file systems protocols [List]
8. Explain mechanisms in file systems to improve fault tolerance [Explain]

3.10.11 OS/Virtualization (Core Tier2: 1 hr)

Topics:
Core
- Using virtualization and isolation to achieve protection and predictable performance. See also: Latency, Cache and Memory Hierarchy, Virtualization and Isolation
- Advanced paging and virtual memory. See also: Latency, Cache and Memory Hierarchy, Virtualization and Isolation
- Virtual file systems and virtual devices.
- Containers and their comparison to virtual machines.
- Thrashing (e.g., Popek and Goldberg requirements for recursively virtualizable systems).

Non Core
- Types of virtualizations (including hardware/software, OS, server, service, network). See also: Latency, Cache and Memory Hierarchy, Virtualization and Isolation
- Portable virtualization; emulation vs isolation. See also: Latency, Cache and Memory Hierarchy, Virtualization and Isolation
- Cost of virtualization. See also: Latency, Cache and Memory Hierarchy, Virtualization and Isolation
- Virtual machines and container escapes, dangers from a security perspective. See also: Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering
- Hypervisors- hardware virtual machine extensions, hosts with kernel support, QEMU KVM

Learning Outcomes:
Core:
1. Explain how hardware architecture provides support and efficiencies for virtualization [Explain]
2. Explain the difference between emulation and isolation [Explain]
3. Evaluate virtualization tradeoffs [Evaluate]

NonCore:
4. Explain hypervisors and the need for them in conjunction with different types of hypervisors [Explain]

3.10.12 OS/Real-time and Embedded Systems (Core Tier2: 1 hr)

Topics:
Core
- Process and task scheduling.
- Deadlines and real-time issues. See also: Embedded Characteristics and Architectures, Real-Time, Memory and Verification, Systems, Programming and Sensors
- Low-latency vs ”soft real-time” vs ”hard real time”. See also: Embedded Characteristics and Architectures, Real-Time, Memory and Verification, Systems, Programming and Sensors, Event-Driven and Reactive Programming

Non Core
- Memory/disk management requirements in a real-time environment.
- Failures, risks, and recovery.
- Special concerns in real-time systems (safety).

Learning Outcomes:
Core:
1. Explain what makes a system a real-time system [Explain]
2. Explain latency and its sources in software systems and its characteristics [Explain]
3. Explain special concerns that real-time systems present, including risk, and how these concerns are addressed [Explain]

NonCore:
4. Explain specific real time operating systems features and mechanisms [Explain]

3.10.13 OS/Fault tolerance (Core Tier2: 1 hr)

Topics:
Core
- Reliable and available systems. See also: System Reliability
- Software and hardware approaches to address tolerance (RAID). See also: System Reliability

Non Core
- Spatial and temporal redundancy. See also: System Reliability
- Methods used to implement fault tolerance. See also: System Reliability
- Error identification and correction mechanisms, checksums of volatile memory in RAM. See also: Memory Hierarchy
- File system consistency check and recovery.
- Journaling and log-structured file systems. See also: System Reliability
- Use-cases for fault-tolerance (databases, safety-critical). See also: System Reliability
- Examples of OS mechanisms for detection, recovery, restart to implement fault tolerance, use of these techniques for the OS's own services. See also: System Reliability

Learning Outcomes:
Core:
1. Explain how operating systems can facilitate fault tolerance, reliability, and availability [Explain]
2. Explain the range of methods for implementing fault tolerance in an operating system [Explain]
3. Explain how an operating system can continue functioning after a fault occurs [Explain]
4. Explain the performance and flexibility tradeoffs that impact using fault tolerance [Explain]

NonCore:
5. Describe operating systems fault tolerance issues and mechanisms in detail [Describe]

3.10.14 OS/Society, Ethics, and the Profession

Topics:
Core
- Open source in operating systems. See also: Open Source, Piracy and DRM, Digital IP Rights and Copyright, Intellectual Property Foundations, Licensing, Plagiarism and Authorship 1. Identification of vulnerabilities in open-source kernels, 2. Open-source guest operating systems, 3. Open-source host operating systems, and 4. Changes in monetization (paid vs free upgrades).
- End-of-life issues with sunsetting operating systems. Example concept: Privacy implications of using proprietary operating systems/operating environments, including telemetry, automated scanning of personal data, built-in advertising, and automatic cloud integration.

Learning Outcomes:
Core:
1. Explain advantages and disadvantages of finding and addressing bugs in open-source kernels [Explain]
2. Contextualize history and positive and negative impact of Linux as an open-source product [Contextualize]

3. List complications with reliance on operating systems past end-of-life [List]
4. Understand differences in finding and addressing bugs for various operating systems payment models [Explain]

3.11 Parallel and Distributed Computing (PDC)

Parallel and distributed programming arranges, coordinates, and controls multiple computations occurring at the same time across different places. The ubiquity of parallelism and distribution are inevitable consequences of increasing numbers of gates in processors, processors in computers, and computers everywhere that may be used to improve performance compared to sequential programs, while also coping with the intrinsic interconnectedness of the world, and the possibility that some components or connections fail or behave maliciously. Parallel and distributed programming removes the restrictions of sequential programming that require computational steps to occur in a serial order in a single place, revealing further distinctions, techniques, and analyses applying at each layer of computing systems.

In most conventional usage, "parallel" programming focuses on establishing and coordinating multiple activities that may occur at the same time, "distributed" programming focuses on establishing and coordinating activities that may occur in different places, and "concurrent" programming focuses on interactions of ongoing activities with each other and the environment. However, all three terms may apply in most contexts. Parallelism generally implies some form of distribution because multiple activities occurring without sequential ordering constraints happen in multiple physical places (unless they rely on context-switching or quantum effects). Conversely, actions in different places need not bear any specific sequential ordering with respect to each other in the absence of communication constraints.

Parallel, distributed, and concurrent programming techniques form the core of High Performance Computing (HPC), distributed systems, and increasingly, nearly every computing application. The PDC knowledge area has evolved from a diverse set of advanced topics into a central body of knowledge and practice, permeating almost every other aspect of computing. Growth of the field has occurred irregularly across different subfields of computing, sometimes with different goals, terminology, and practices, masking the considerable overlap of basic ideas and skills that are the primary focus of this knowledge area. Nearly every problem with a sequential solution also admits parallel and/or distributed solutions; additional problems and solutions arise only in the context of concurrency. Nearly every application domain of parallel and distributed computing is a well-developed area of study and/or engineering too large to enumerate.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.11.1 Programs: Declarative Parallelism (Pg.. 105)	1	1
⭐ 3.11.2 Programs: Starting Activities (Pg.. 105)	1	1
⭐ 3.11.3 Programs: Execution Properties (Pg.. 105)	1	1
⭐ 3.11.4 Programs: Distribution (Pg.. 106)	1	1
⭐ 3.11.5 Programs: Implementation Mappings (Pg.. 106)	1	1
3.11.6 GPU Programming (Pg.. 106)	Elective	
⭐ 3.11.7 Communication (Pg.. 107)	2	4
⭐ 3.11.8 Communication: Properties and Extensions (Pg.. 107)	1	1
⭐ 3.11.9 Memory and Consistency (Pg.. 108)	1	1
⭐ 3.11.10 Coordination (Pg.. 108)	1	3
⭐ 3.11.11 Coordination: Synchronization and Atomicity (Pg.. 109)	1	2
⭐ 3.11.12 Coordination: Advanced Properties (Pg.. 109)	1	2
⭐ 3.11.13 Evaluation (Pg.. 109)	1	3
⭐ 3.11.14 Evaluation: Formal Methods (Pg.. 110)	1	1
⭐ 3.11.15 Evaluation: Testing and Measurement (Pg.. 110)	1	1
⭐ 3.11.16 Algorithms (Pg.. 111)	1	3
⭐ 3.11.17 Algorithms: Application Domains Survey (Pg.. 111)	1	3
⭐ 3.11.18 Algorithms: Algorithmic Domains (Pg.. 112)	1	3

Table 3.11: List of KUs in the Parallel and Distributed Computing area.

3.11.1 PDC/Programs: Declarative Parallelism (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Parallelism 1. Declarative parallelism: Determining which actions can, or must not, be performed in parallel, at the level of instructions, functions, closures, compound actions, sessions, tasks, and services is the core idea underlying PDC algorithms; failing to do so is the primary source of errors. See also: Algorithms 2. Defining order: for example, using "happens-before" relationships or series-parallel directed acyclic graphs (DAGs) representing programs. 3. Independence: determining when order does not matter, in terms of commutativity, dependencies, preconditions. 4. Guaranteeing order between otherwise parallel actions when necessary, including locks, safe publication; and enforcing communication - sending a message happens before receiving it; and relaxing it when not necessary.

Learning Outcomes:
Core:

1. Graphically represent (as a Directed Acyclic Graph - DAG) how to parallelize a compound numeric expression; for example, $a = (b + c) * (d + e)$ [Design]
2. Explain why consistency and fault tolerance concepts do not arise in purely sequential programs [Explain]

3.11.2 PDC/Programs: Starting Activities (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Starting activities 1. Options allowing actions to be performed (eventually) at locations range from hardwired to configuration scripts; also establishing communication and resource management; these are expressed differently in languages and contexts, typically relying on automated provisioning and management by platforms. See also: Resource Management 2. Procedural: Allowing multiple actions to start at a given program point; for example, starting new threads, possibly delimiting their scope or organizing them into hierarchical groups. 3. Reactive: Enabling upon event occurrence by installing an event handler, with less control over when actions start or finish, and may apply even on single processors. 4. Dependent: Enabling upon completion of others; for example, sequencing sets of parallel actions. See also: Coordination. 5. Granularity: The execution cost of action bodies should outweigh the overhead of organizing them.

Learning Outcomes:
Core:

1. Write a service that creates a thread (or other form of procedural activation) to return a requested web page to each new client [Write]

3.11.3 PDC/Programs: Execution Properties (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Execution Properties 1. Non-deterministic execution of actions without order. 2. Consistency: Ensuring agreement among parties on values and predicates when necessary to avoid races, maintain safety and atomicity, or reach consensus. 3. Fault tolerance: Handling failures in parts or communication, including (Byzantine) misbehavior due to unreliable parties and protocols, when necessary to maintain progress or availability. See also: System Reliability. 4. Trade-offs are a focus of evaluation. See also: Evaluation.

Learning Outcomes:
Core:

1. Write a function that counts events, such as network packet receptions, efficiently [Write]

3.11.4 PDC/Programs: Distribution (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Distribution 1. Defining locations as devices that execute actions, including hardware components, remote hosts; may also include external uncontrolled devices, hosts, and users. See also: Interfacing and Communication 2. A device may time-share or emulate multiple parallel actions using fewer processors through scheduling and virtualization. See also: Scheduling 3. Naming or identifying locations (e.g., device IDs) and actions as parts (e.g., thread IDs). 4. Activities across locations may communicate through media. See also: Communication

Learning Outcomes:
Core:

1. Write a filter/map/reduce program in multiple styles [Write]

3.11.5 PDC/Programs: Implementation Mappings (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- One or more of the following mappings and mechanisms across layered systems: 1. Instruction-level and data parallelism in the CPU. See also: Functional Organization. 2. SIMD and heterogeneous data parallelism. See also: Heterogeneous Architectures. 3. Scheduled concurrency on multi-core, tasks, actors. See also: Scheduling. 4. Clusters, clouds; elastic provisioning. See also: Common Aspects: Platforms, APIs and Constraints, Common Aspects: Platform Languages and Web Programming Patterns. 5. Networked distributed systems. See also: Networked Applications. 6. Emerging technologies such as quantum computing and molecular computing.

Learning Outcomes:
Core:

1. Explain the trade-offs between different mapping strategies in terms of performance, cost, and implementation complexity [Explain]

3.11.6 PDC/GPU Programming

Topics:
Non Core

- General-Purpose GPU Computing (GPGPU) 1. GPU architecture overview: streaming multiprocessors (SMs), CUDA cores, and warp execution model. 2. SIMT (Single Instruction, Multiple Threads) execution model and its differences from CPU SIMD. 3. Host-device interaction: data transfer between CPU (host) and GPU (device) memory via PCIe. 4. Use cases: scientific computing, machine learning, image processing, and large-scale simulation.
- CUDA Programming Model 1. CUDA execution hierarchy: grids, thread blocks, and individual threads; mapping to GPU hardware. 2. Kernel definition and launch syntax; configuration parameters (grid and block dimensions). 3. Thread indexing: threadIdx, blockIdx, blockDim, gridDim; computing global indices in 1D, 2D, and 3D. 4. Warp divergence: performance implications of branching within a warp. 5. Intra-block synchronization primitive: __syncthreads().
- CUDA Memory Hierarchy 1. Global memory: large, high-latency, accessible by all threads; coalesced access patterns for performance. 2. Shared memory: low-latency on-chip memory shared within a thread block; bank conflicts. 3. Registers and local memory: per-thread private storage. 4. Constant and texture memory: read-only caches optimized for specific access patterns. 5. Unified Memory: simplified programming model with automatic host-device data migration.

Learning Outcomes:
NonCore:

1. Write a CUDA kernel that performs a data-parallel operation (e.g., vector addition, matrix multiplication) and correctly configure grid and block dimensions [Write]

2. Analyze the impact of global memory access patterns (coalesced vs. strided) and shared memory usage on GPU kernel performance [Analyze]
3. Design a parallel algorithm using the CUDA programming model, including thread hierarchy, memory allocation strategy, and host-device data transfers [Design]

3.11.7 PDC/Communication (Core Tier1: 2 hrs, Core Tier2: 4 hrs)

Topics:
Core

- Media 1. Varieties: channels (message passing or I/O), shared memory, heterogeneous, data stores 2. Reliance on the availability and nature of underlying hardware, connectivity, and protocols; language support, emulation. See also: Interfacing and Communication
- Channels 1. Explicit (usually named) party-to-party communication media 2. APIs: Sockets, architectural, language-based, and toolkit constructs, such as Message Passing Interface (MPI), and layered constructs such as Remote Procedure Call (RPC). See also: Fundamentals of Networks and Communications 3. I/O channel APIs
- Memory 1. Shared memory architectures in which parties directly communicate only with memory at given addresses, with extensions to heterogeneous memory supporting multiple memory stores with explicit data transfer across them; for example, GPU local and shared memory, Direct Memory Access (DMA) 2. Memory hierarchies: Multiple layers of sharing domains, scopes, and caches; locality: latency, false-sharing 3. Consistency properties: Bitwise atomicity limits, coherence, local ordering
- Data Stores 1. Cooperatively maintained data structures implementing maps and related ADTs 2. Varieties: Owned, shared, sharded, replicated, immutable, versioned

Learning Outcomes:
Core:

1. Explain the similarities and differences among: (1) Party A sends a message on channel X with contents 1 received by party B (2) A sets shared variable X to 1, read by B (3) A sets "X=1" in a distributed shared map accessed by B [Explain]
2. Write a program that distributes different segments of a data set to multiple workers, and collects results (for the simplest example, summing segments of an array) [Write]
3. Write a parallel program that requests data from multiple sites and summarizes them using some form of reduction [Write]
4. Compare the performance of buffered versus unbuffered versions of a producer-consumer program [Compare]

3.11.8 PDC/Communication: Properties and Extensions (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- One or more of the following properties and extensions 1. Topologies: Unicast, Multicast, Mailboxes, Switches; Routing via hardware and software interconnection networks 2. Media concurrency properties: Ordering, consistency, idempotency, overlapping communication with computation 3. Media performance: Latency, bandwidth (throughput) contention (congestion), responsiveness (liveness), reliability (error and drop rates), protocol-based progress (acks, timeouts, mediation) 4. Media security properties: integrity, privacy, authentication, authorization. See also: Information Flow and Non-Interference, Injection and Input Validation, Memory Safety and Types, Malware Analysis and Advanced Security 5. Data formats: Marshaling, validation, encryption, compression 6. Channel policies: Endpoints, sessions, buffering, saturation response (waiting vs dropping), rate control 7. Multiplexing and demultiplexing many relatively slow I/O devices or parties; completion-based and scheduler-based techniques; async-await, select and polling APIs 8. Formalization and analysis of channel communication; for example, CSP 9. Applications of queuing theory to model and predict performance.

Learning Outcomes:
Core:

1. Determine whether a given communication scheme provides sufficient security properties for a given usage [Determine]
2. Give an example of a scenario in which blocking message sends can deadlock [Create]
3. Describe at least one design technique for avoiding liveness failures in programs using multiple locks [Describe]

3.11.9 PDC/Memory and Consistency (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Memory models, data consistency, and fault tolerance 1. Memory models: sequential and release/acquire consistency 2. Memory management; including reclamation of shared data; reference counts and alternatives 3. Bulk data placement and transfer; reducing message traffic and improving locality; overlapping data transfer and computation; impact of data layout such as array-of-structs vs struct-of-arrays 4. Emulating shared memory: distributed shared memory, Remote Direct Memory Access (RDMA) 5. Data store consistency: Atomicity, linearizability, transactionality, coherence, causal ordering, conflict resolution, eventual consistency, blockchains 6. Faults, partitioning, and partial failures; voting; protocols such as Paxos and Raft. 7. Design tradeoffs among consistency, availability, partition (fault) tolerance; impossibility of meeting all at once 8. Security and trust: Byzantine failures, proof of work and alternatives

Learning Outcomes:
Core:

1. Give an example of an ordering of accesses among concurrent activities (e.g., program with a data race) that is not sequentially consistent [Create]
2. Write a program that illustrates memory-access or message reordering [Write]
3. Describe the relative merits of optimistic versus conservative concurrency control under different rates of contention among updates [Describe]
4. Give an example of a scenario in which an attempted optimistic update may never complete [Create]
5. Modify a concurrent system to use a more scalable, reliable, or available data store [Create]
6. Using an existing platform supporting replicated data stores, write a program that maintains a key-value mapping even when one or more hosts fail [Write]

3.11.10 PDC/Coordination (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- Dependencies 1. Initiation or progress of one activity may be dependent on other activities, so as to avoid race conditions, ensure termination, or meet other requirements 2. Ensuring progress by avoiding dependency cycles, using monotonic conditions, removing inessential dependencies
- Control constructs and design patterns 1. Completion-based: Barriers, joins, including termination control 2. Data-enabled: Queues, producer-consumer designs 3. Condition-based: Polling, retrying, backoffs, helping, suspension, signaling, timeouts 4. Reactive: Enabling and triggering continuations

Learning Outcomes:
Core:

1. Show how to ensure that a program correctly terminates when all of a set of concurrent tasks have completed [Design]
2. Write a function that efficiently counts events such as sensor inputs or networking packet receptions [Write]
3. Write a filter/map/reduce program in multiple styles [Write]
4. Write a program in which the termination of one set of parallel actions is followed by another [Write]
5. Write a service that creates a thread (or other procedural form of activation) to return a requested web page to each new client [Write]

3.11.11 PDC/Coordination: Synchronization and Atomicity (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Atomicity 1. Atomic instructions, enforced local access orderings 2. Locks and mutual exclusion; lock granularity 3. Using locks in a specific language; maintaining liveness without introducing races 4. Deadlock avoidance: Ordering, coarsening, randomized retries; backoffs, encapsulation via lock managers 5. Common errors: Failing to lock or unlock when necessary, holding locks while invoking unknown operations 6. Avoiding locks: replication, read-only, ownership, and non-blocking constructions

Learning Outcomes:
Core:

1. Show how to avoid or repair a race error in a given program [Analyze]

3.11.12 PDC/Coordination: Advanced Properties (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- One or more of the following properties and extensions 1. Progress properties including lock-free, wait-free, fairness, priority scheduling, interactions with consistency, reliability 2. Performance with respect to contention, granularity, convoying, scaling 3. Non-blocking data structures and algorithms 4. Ownership and resource control 5. Lock variants and alternatives: sequence locks, read-write locks; Read-Copy-Update (RCU), reentrancy; tickets; controlling spinning versus blocking 6. Transaction-based control: Optimistic and conservative 7. Distributed locking: reliability 8. Alternatives to barriers: Clocks; counters, virtual clocks; dataflow and continuations; futures and RPC; consensus-based, gathering results with reducers and collectors 9. Speculation, selection, cancellation; observability and security consequences 10. Resource control using semaphores and condition variables 11. Control flow: Scheduling computations, series-parallel loops with (possibly elected) leaders, pipelines and streams, nested parallelism 12. Exceptions and failures. Handlers, detection, timeouts, fault tolerance, voting

Learning Outcomes:
Core:

1. Write a program that speculatively searches for a solution by multiple activities, terminating others when one is found [Write]
2. Write a program in which a numerical exception (such as divide by zero) in one activity causes termination of others [Write]
3. Write a program for multiple parties to agree upon the current time of day; discuss its limitations compared to protocols such as network transfer protocol (NTP) [Write]

3.11.13 PDC/Evaluation (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- Safety and liveness requirements in terms of temporal logic constructs to express ”always” and ”eventually” See also: Parallel and Distributed Computing
- Identifying, testing for, and repairing violations, including common forms of errors such as failure to ensure necessary ordering (race errors), atomicity (including check-then-act errors), and termination (livelock)
- Performance requirements metrics for throughput, responsiveness, latency, availability, energy consumption, scalability, resource usage, communication costs, waiting and rate control, fairness; service level agreements. See also: Latency, Cache and Memory Hierarchy, Virtualization and Isolation
- Performance impact of design and implementation choices, including granularity, overhead, consensus costs, and energy consumption. See also: Sustainable Design and Pervasive Computing,

Environmental Footprint of Computing Systems, Systemic Effects and Social Context
- Estimating scalability limitations, for example using Amdahl's Law or Universal Scalability Law. See also: Events, Tools and Experimentation, Performance Metrics and Benchmarks, Analytical Performance Models

Learning Outcomes:
Core:
1. Revise a specification to enable parallelism and distribution without violating other essential properties or features [Redesign]
2. Explain how concurrent notions of safety and liveness extend their sequential counterparts [Explain]
3. Specify a set of invariants that must hold at each bulk-parallel step of a computation [Analyze]
4. Write a test program that can reveal a data race error; for example, missing an update when two activities both try to increment a variable [Write]
5. In a given context, explain the extent to which introducing parallelism in an otherwise sequential program would be expected to improve throughput and/or reduce latency, and how it may impact energy efficiency [Explain]
6. Show how scaling and efficiency change for sample problems without and with the assumption of problem size changing with the number of processors; further explain whether and how scalability would change under relaxations of sequential dependencies [Design]

3.11.14 PDC/Evaluation: Formal Methods (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Formal verification and analysis methods 1. Extensions to formal sequential requirements such as linearizability 2. Protocol, session, and transactional specifications 3. Use of tools such as Unified Modelling Language (UML), Temporal Logic of Actions (TLA), program logics 4. Security analysis: safety and liveness in the presence of hostile or buggy behaviors by other parties; required properties of communication mechanisms (for example lack of cross-layer leakage), input screening, rate limiting. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles 5. Static analysis applied to correctness, throughput, latency, resources, energy. See also: Sustainable Design and Pervasive Computing, Environmental Footprint of Computing Systems, Systemic Effects and Social Context 6. Directed Acyclic Graph (DAG) model analysis of algorithmic efficiency (work, span, critical paths)

Learning Outcomes:
Core:
1. Specify and measure behavior when a service is requested by unexpectedly many clients [Analyze]
2. Identify and repair a performance problem due to sequential bottlenecks [Analyze]
3. Empirically compare throughput of two implementations of a common design (perhaps using an existing test harness framework) [Compare]

3.11.15 PDC/Evaluation: Testing and Measurement (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Testing tools and measurement techniques 1. Testing and debugging; tools such as race detectors, fuzzers, lock dependency checkers, unit/stress/torture tests, visualizations, continuous integration, continuous deployment, and test generators 2. Measuring and comparing throughput, overhead, waiting, contention, communication, data movement, locality, resource usage, behavior in the presence of excessive numbers of events, clients, or threads. See also: Events, Tools and Experimentation, Performance Metrics and Benchmarks, Analytical Performance Models 3. Application domain specific analyses and evaluation techniques

Learning Outcomes:
Core:

1. Identify and repair a performance problem due to communication or data latency [Analyze]
2. Identify and repair a performance problem due to resource management overhead [Analyze]
3. Identify and repair a reliability or availability problem [Analyze]

3.11.16 PDC/Algorithms (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- Expressing and implementing algorithms in given languages and frameworks, to initiate activities (for example threads), use shared memory constructs, and channel, socket, and/or remote procedure call APIs. See also: Parallel and Distributed Computing. 1. Data parallel examples including map/reduce. 2. Using channel, socket, and/or RPC APIs in a given language, with program control for sending (usually procedural) vs receiving. (usually reactive or RPC-based). 3. Using locks, barriers, and/or synchronizers to maintain liveness without introducing races.

Learning Outcomes:
Core:

1. Implement a parallel/distributed component based on a known algorithm [Implement]
2. Write a data-parallel program that for example computes the average of an array of numbers [Write]
3. Write a producer-consumer program in which one component generates numbers, and another computes their average. Measure speedups when the numbers are small scalars versus large multi-precision values [Write]

3.11.17 PDC/Algorithms: Application Domains Survey (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- Survey of common application domains across multicore, reactive, data parallel, cluster, cloud, open distributed systems, and frameworks (with reference to the following table). 1. Multicore: Typical Execution agents: Threads. Typical Communication mechanisms: Shared memory, Atomics, locks. Typical Algorithmic domains: Resource management, data processing. Typical Engineering goals: Throughput, latency, energy. 2. Reactive: Typical Execution agents: Handlers, threads. Typical Communication mechanisms: I/O Channels. Typical Algorithmic domains: Services, real-time. Typical Engineering goals: Latency. 3. Data parallel: Typical Execution agents: GPU, SIMD, accelerators, hybrid. Typical Communication mechanisms: Heterogeneous memory. Typical Algorithmic domains: Linear algebra, graphics, data analysis. Typical Engineering goals: Throughput, energy. 4. Cluster: Typical Execution agents: Managed hosts. Typical Communication mechanisms: Sockets, channels. Typical Algorithmic domains: Simulation, data analysis. Typical Engineering goals: Throughput. 5. Cloud: Typical Execution agents: Provisioned hosts. Typical Communication mechanisms: Service APIs. Typical Algorithmic domains: Web applications. Typical Engineering goals: Scalability. 6. Open distributed: Typical Execution agents: Autonomous hosts. Typical Communication mechanisms: Sockets, Data stores. Typical Algorithmic domains: Fault tolerant data stores and services. Typical Engineering goals: Reliability.

Learning Outcomes:
Core:

1. Extend an event-driven sequential program by establishing a new activity in an event handler (for example a new thread in a GUI action handler) [Design]
2. Improve the performance of a sequential component by introducing parallelism and/or distribution [Create]
3. Choose among different parallel/distributed designs for components of a given system [Assess]

3.11.18 PDC/Algorithms: Algorithmic Domains (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- One of more of the following algorithmic domains. See also: Algorithmic Strategies: 1. Linear algebra: Vector and matrix operations, numerical precision/stability, applications in data analytics and machine learning. 2. Data processing: sorting, searching and retrieval, concurrent data structures. 3. Graphs, search, and combinatorics: Marking, edge-parallelization, bounding, speculation, network-based analytics. 4. Modeling and simulation: differential equations; randomization, N-body problems, genetic algorithms. 5. Computational logic: satisfiability (SAT), concurrent logic programming. 6. Graphics and computational geometry: Transforms, rendering, ray-tracing. 7. Resource management: Allocating, placing, recycling and scheduling processors, memory, channels, and hosts; exclusive vs shared resources; static, dynamic and elastic algorithms; Real-time constraints; Batching, prioritization, partitioning; decentralization via work-stealing and related techniques. 8. Services: Implementing web APIs, electronic currency, transaction systems, multiplayer games.

Learning Outcomes:
Core:

1. Design, implement, analyze, and evaluate a component or application for X operating in a given context, where X is in one of the listed domains, for example, a genetic algorithm for factory floor design [Design]
2. Critique the design and implementation of an existing component or application, or one developed by classmates [Critique]
3. Compare the performance and energy efficiency of multiple implementations of a similar design, for example, multicore versus clustered versus GPU [Compare]

3.12 Software Development Fundamentals (SDF)

Fluency in the process of software development is fundamental to the study of computer science. To use computers to solve problems most effectively, students must be competent at reading and writing programs. Beyond programming skills, however, they must be able to select and use appropriate data structures and algorithms and use modern development and testing tools.

The SDF knowledge area brings together fundamental concepts and skills related to software development, focusing on concepts and skills that should be taught early in a computer science program, typically in the first year. This includes fundamental programming concepts and their effective use in writing programs, use of fundamental data structures which may be provided by the programming language, basics of programming practices for writing good quality programs, reading, and understanding programs, and some understanding of the impact of algorithms on the performance of the programs. The 43 hours of material in this knowledge area may be augmented with core material from other knowledge areas as students progress to mid- and upper-level courses.

This knowledge area assumes a contemporary programming language with built-in support for common data types including associative data types like dictionaries/maps as the vehicle for introducing students to programming (e.g., Python, Java). However, this is not to discourage the use of older or lower-level languages for SDF - the knowledge units below can be suitably adapted for the actual language used.

The emergence of generative AI and Large Language Models (LLMs), which can generate programs for many programming tasks, will undoubtedly affect the programming profession and consequently the teaching of many CS topics. However, to be able to effectively use generative AI in programming tasks, a programmer must have a good understanding of programs, and hence must still learn the foundations of programming and develop basic programming skills - which is the aim of SDF. Consequently, we feel that the desired outcomes for SDF should remain the same, though different instructors may now give more emphasis to program understanding, documenting, specifications, analysis, and testing. (This is like teaching students multiplication, addition, etc. even though calculators can be used to do them).

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.12.1 Basic Programming Concepts and Constructs (Pg.. 113)	10	
⭐ 3.12.2 Programming Practices: Testing, Debugging, and Documentation (Pg.. 113)	10	
⭐ 3.12.3 Fundamental Data Structures (Pg.. 114)	12	
⭐ 3.12.4 Algorithms (Pg.. 114)	6	
⭐ 3.12.5 Software Development Practices (Pg.. 114)	5	
3.12.6 Society, Ethics, and the Profession (Pg.. 115)	Elective	

Table 3.12: List of KUs in the Software Development Fundamentals area.

3.12.1 SDF/Basic Programming Concepts and Constructs (Core Tier1: 10 hrs)

Topics:
Core

- Basic concepts such as variables, primitive data types, expressions, and their evaluation
- How imperative programs work: state and state transitions on execution of statements, flow of control
- Basic constructs such as assignment statements, conditional and iterative statements, basic I/O
- Key modularity constructs such as functions (and methods and classes, if supported in the language) and related concepts like parameter passing, scope, abstraction, data encapsulation Object-Oriented Programming Fundamentals, Object-Oriented Programming: Encapsulation, Subtyping, and Reflection
- Input and output using files and APIs
- Structured data types available in the chosen programming language like sequences (e.g., arrays, lists), associative containers (e.g., dictionaries, maps), others (e.g., sets, tuples) and when and how to use them Foundational Data Structures, Foundational Algorithms, Advanced Algorithms

Learning Outcomes:
Core:

1. Develop programs that use the fundamental programming constructs: assignment and expressions, basic I/O, conditional and iterative statements [Create]
2. Develop programs using functions with parameter passing [Create]
3. Develop programs that effectively use the different structured data types provided in the language like arrays/lists, dictionaries, and sets [Create]
4. Develop programs that use file I/O to provide data persistence across multiple executions [Create]
5. Develop programs that use APIs to access or update data (e.g., from the web) [Create]
6. Develop programs that create simple classes and instantiate objects of those classes (if supported by the language) [Create]
7. Trace the flow of control during the execution of a program [Trace]
8. Use appropriate terminology to identify elements of a program (e.g., identifier, operator, operand) [Use]

3.12.2 SDF/Programming Practices: Testing, Debugging, and Documentation (Core Tier1: 10 hrs)

Topics:
Core

- Libraries and frameworks provided by the language (when/where applicable)
- Recursion
- Dealing with runtime errors in programs (e.g., exception handling).
- Basic concepts of programming errors, testing, and debugging Coding Practices, Large-Scale Construction and Process, Information Flow and Non-Interference, Injection and Input Valida-

tion, Memory Safety and Types, Malware Analysis and Advanced Security
- Documenting/commenting code at the program and module level. Coding Practices, Large-Scale Construction and Process
- Develop a security mindset. AI and Hardware Security, Access Control and Applications, Security Mindset and Principles

Learning Outcomes:
Core:
1. Develop programs that use language-provided libraries and frameworks (where applicable) [Create]
2. Explain the concept of recursion and identify when and how to use it effectively [Explain]
3. Develop recursive functions [Create]
4. Develop programs that can handle runtime errors [Create]
5. Read a given program and explain what it does [Explain]
6. Write comments for a program or a module specifying what it does [Write]

3.12.3 SDF/Fundamental Data Structures (Core Tier1: 12 hrs)

Topics:
Core
- Standard abstract data types such as lists, stacks, queues, sets, and maps/dictionaries, including operations on them.
- Selecting and using appropriate data structures.
- Performance implications of choice of data structure(s).
- Strings and string processing.

Learning Outcomes:
Core:
1. Write programs that use each of the key abstract data types provided in the language (e.g., arrays, tuples/records/structs, lists, stacks, queues, and associative data types like sets, dictionaries/maps) [Write]
2. Select the appropriate data structure for a given problem [Evaluate]
3. Explain how the performance of a program may change when using different data structures or operations [Explain]
4. Write programs that work with text by using string processing capabilities provided by the language [Write]

3.12.4 SDF/Algorithms (Core Tier1: 6 hrs)

Topics:
Core
- Concept of algorithm and notion of algorithm efficiency
- Some common algorithms (e.g., sorting, searching, tree traversal, graph traversal)
- Impact of algorithms on time-space efficiency of programs

Learning Outcomes:
Core:
1. Explain the role of algorithms for writing programs [Explain]
2. Demonstrate how a problem may be solved by different algorithms, each with different properties [Demonstrate]
3. Explain some common algorithms (e.g., sorting, searching, tree traversal, graph traversal) [Explain]
4. Explain the impact on space/time performance of some algorithms [Explain]

3.12.5 SDF/Software Development Practices (Core Tier1: 5 hrs)

Topics:
Core
- Basic testing, including test case design

- Use of a general-purpose IDE, including its debugger
- Programming style that improves readability
- Specifying functionality of a module in a natural language.

Learning Outcomes:
Core:

1. Develop tests for modules and apply a variety of strategies to design test cases [Create]
2. Explain some limitations of testing programs [Explain]
3. Build, execute, and debug programs using a modern IDE and associated tools such as visual debuggers [Analyze]
4. Apply basic programming style guidelines to aid readability of programs such as comments, indentation, proper naming of variables, etc [Apply]
5. Write specifications of a module as module comment describing its functionality [Write]

3.12.6 SDF/Society, Ethics, and the Profession

Topics:
Core

- Intellectual property rights of programmers for programs they develop.
- Plagiarism and academic integrity.
- Responsibility and liability of programmers regarding code they develop for solutions. AI and Hardware Security, Access Control and Applications, Security Mindset and Principles
- Basic professional work ethics of programmers.

Learning Outcomes:
Core:

1. Explain/understand some of the intellectual property issues relating to programs [Explain]
2. Explain/understand when code developed by others can be used and proper ways of disclosing their use [Explain]
3. Explain/understand the responsibility of programmers when developing code for an overall solution (which may be developed by a team) [Explain]
4. Explain/understand one or more codes of conduct applicable to programmers [Explain]

3.13 Software Engineering (SE)

As far back as the early 1970s, British computer scientist Brian Randell allegedly said, ”Software engineering is the multi-person construction of multi-version programs.” This is an essential insight: while programming is the skill that governs our ability to write a program, software engineering is distinct in two dimensions: time and people.

First, a software engineering project is a team endeavor; being a solitary programming expert is insufficient. Skilled software engineers must demonstrate expertise in communication and collaboration. Programming may be an individual activity, but software engineering is a collaborative one, deeply tied to issues of professionalism, teamwork, and communication.

Second, a software engineering project is usually ”multi-version.” It has an expected lifespan; it needs to function properly for months, years, or decades. Features may be added or removed to meet product requirements. The engineering team itself will likely change. The technological context will shift, as our computing platforms evolve, programming languages change, dependencies upgrade, etc. This exposure to matters of time and change is novel when compared to a programming project: it isn't enough to build a thing that works, instead it must work and stay working. Many of the most challenging topics in tech share ”time will lead to change” as a root cause: backward compatibility, version skew, dependency management, schema changes, protocol evolution.

Software engineering presents a particularly difficult challenge for learning in an academic setting. Given that the major differences between programming and software engineering are time and teamwork, it is hard to generate lessons that *require* successful teamwork and that faithfully present the challenges of time. Additionally, some topics in software engineering will be more authentic and more relevant if our learners experience collaborative and long-term software engineering projects *in vivo* rather than in the classroom. Regardless of whether that happens as an internship, involvement in an

open-source project, or full-time engineering role, a month of full-time hands-on experience has more available hours than the average software engineering course.

Thus, a software engineering curriculum must focus on concepts needed by most new-graduate hires, and that either are novel for those who are trained primarily as programmers, or that are abstract concepts that may not get explicitly stated/shared on the job. Such topics include, but are not limited to:

- Testing
- Teamwork, collaboration
- Communication
- Design
- Maintenance and evolution
- Software engineering tools

All evidence suggests that the role of software in our society will continue to grow for the foreseeable future. Additionally, the era of "two programmers in a garage" seems to have drawn to a close. Most important software these days is a team effort, building on existing code and leveraging existing functionality. The study of software engineering skills is a deeply important counterpoint to the everyday experience of computing students - we must impress on them the reality that few software projects are managed by writing from scratch as a solo endeavor. Communication, teamwork, planning, testing, and tooling are far more important as our students move on from the classroom and make their mark on the wider world.

Although most CS graduates will go on to an industry position that requires this material, the CS Core topics presented here are of value regardless of whether graduates go on to industry or academia.

Knowledge Area (KA)	**Core Tier1**	**Core Tier2**
⭐ 3.13.1 Team Collaboration and Dynamics (Pg.. 116)	2	2
⭐ 3.13.2 Diversity, Stakeholders and Distributed Teams (Pg.. 117)	1	1
⭐ 3.13.3 Version Control and CI/CD (Pg.. 117)	1	2
⭐ 3.13.4 Development Environment and IDE (Pg.. 118)	1	2
⭐ 3.13.5 Requirements Engineering (Pg.. 118)		2
⭐ 3.13.6 Requirements Evolution and Estimation (Pg.. 119)		1
⭐ 3.13.7 Principles and Software Architecture (Pg.. 119)	1	2
⭐ 3.13.8 API Design and Modeling (Pg.. 120)	1	1
⭐ 3.13.9 Software Design Quality and Evaluation (Pg.. 120)	1	2
⭐ 3.13.10 Coding Practices (Pg.. 121)	1	2
⭐ 3.13.11 Large-Scale Construction and Process (Pg.. 122)	1	2
⭐ 3.13.12 Verification and validation Concepts (Pg.. 122)	1	1
⭐ 3.13.13 Test Planning and Types (Pg.. 122)	1	1
⭐ 3.13.14 Advanced Testing Practices (Pg.. 123)	1	1
⭐ 3.13.15 Testing Tools and Analysis (Pg.. 123)	1	1
⭐ 3.13.16 Performance Testing and Benchmarking (Pg.. 124)	1	1
⭐ 3.13.17 API Compatibility and Versioning (Pg.. 124)		2
⭐ 3.13.18 Refactoring Techniques (Pg.. 125)		1
⭐ 3.13.19 Reliability Concepts (Pg.. 125)		2
⭐ 3.13.20 Reliability Engineering (Pg.. 125)		1
3.13.21 Formal Methods (Pg.. 126)	Elective	

Table 3.13: List of KUs in the Software Engineering area.

3.13.1 SE/Team Collaboration and Dynamics (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Topics:

Core

- Effective communication, including oral and written, as well as formal (email, docs, comments, presentations) and informal (team chat, meetings). Technical and Professional Communication,

Team Communication and Culture
- Common causes of team conflict, and approaches for conflict resolution.
- Cooperative programming: 1. Pair programming or Swarming 2. Code review 3. Collaboration through version control
- Roles and responsibilities in a software team: Ethics Codes and Legal Accountability, Professional Responsibility and Public Policy, Professional Values, Nature and Continuing Development, Ethical Dissent and Cultural Context, Workplace Ethics 1. Advantages of teamwork 2. Risks and complexity of such collaboration
- Team processes - responsibilities for tasks, effort estimation, meeting structure, work schedule

Learning Outcomes:

Core:

1. Follow effective team communication practices [Analyze]
2. Articulate the sources of, hazards of, and potential benefits of team conflict - especially focusing on the value of disagreeing about ideas or proposals without insulting people [Articulate]
3. Facilitate a conflict-resolution and problem-solving strategy in a team setting [Analyze]
4. Collaborate effectively in cooperative development/programming [Analyze]
5. Propose and delegate necessary roles and responsibilities in a software development team [Propose]
6. Compose and follow an agenda for a team meeting [Compose]
7. Facilitate through involvement in a team project, the central elements of team building, establishing healthy team culture, and team management including creating and executing a team work plan [Create]

3.13.2 SE/Diversity, Stakeholders and Distributed Teams (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Importance of team diversity and inclusivity. Technical and Professional Communication, Team Communication and Culture
- Interfacing with stakeholders, as a team: 1. Management & other non-technical teams 2. Customers 3. Users
- Risks associated with physical, distributed, hybrid, and virtual teams - including communication, perception, structure, points of failure, mitigation, and recovery, etc.

Learning Outcomes:

Core:

1. Promote the importance of and benefits that diversity and inclusivity brings to a software development team [Defend]
2. Reference, as a team, the importance of, and strategies to interface with stakeholders outside the team on both technical and non-technical levels [Reference]
3. Enumerate the risks associated with physical, distributed, hybrid, and virtual teams and possible points of failure and how to mitigate against and recover/learn from failures [Enumerate]

3.13.3 SE/Version Control and CI/CD (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:

Core

- Software configuration management and version control: Software Development Practices 1. Configuration in version control, reproducible builds/configuration. 2. Version control branching strategies. Development branches vs release branches. Trunk-based development. 3. Merging/rebasing strategies, when relevant.
- Release management.
- Testing tools including static and dynamic analysis tools. Software Development Practices, Information Flow and Non-Interference, Injection and Input Validation, Memory Safety and Types, Malware Analysis and Advanced Security

- Software process automation: 1. Build systems - the value of fast, hermetic, reproducible builds, compare/contrast approaches to building a project. 2. Continuous Integration (CI) - the use of automation and automated tests to do preliminary validation that the current head/trunk revision builds and passes (basic) tests. 3. Continuous Deployment (CD) - the use of automation to automatically release every change that passes the automated tests to the production environment, ensuring frequent and reliable deliveries. 4. Dependency management - updating external/upstream dependencies, package management, SemVer.
- Collaborative review workflow as a quality gate 1. Mandatory pull-request review before merging 2. Code ownership via CODEOWNERS-style rules 3. Conventional commit message conventions 4. Branch-protection rules enforcing checks before merge
- Dependency and supply-chain vulnerability scanning integrated into CI pipelines

Learning Outcomes:

Core:

1. Describe the difference between centralized and distributed software configuration management [Describe]
2. Describe how version control can be used to help manage software release management [Describe]
3. Identify configuration items and use a source code control tool in a small team-based project [Analyze]
4. Describe how available static and dynamic test tools can be integrated into the software development environment [Describe]
5. Understand the use of CI/CD systems as a ground-truth for the state of the team's shared code (build and test success) [Explain]
6. Apply a pull-request review workflow with code-ownership rules and branch protection to enforce quality gates on a team project [Apply]
7. Use a dependency/supply-chain vulnerability scanner as part of a CI pipeline to detect vulnerable dependencies [Use]

3.13.4 SE/Development Environment and IDE (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:

Core

- Design and communication tools (docs, diagrams, common forms of design diagrams like UML).
- Tool integration concepts and mechanisms. Software Development Practices
- Use of modern IDE facilities - debugging, refactoring, searching/indexing, ML-powered code assistants, etc. Software Development Practices

Learning Outcomes:

Core:

1. Describe the issues that are important in selecting a set of tools for the development of a specific software system, including tools for requirements tracking, design modeling, implementation, build automation, and testing [Describe]
2. Demonstrate the capability to use software tools in support of the development of a software product of medium size [Demonstrate]

3.13.5 SE/Requirements Engineering (Core Tier2: 2 hrs)

Topics:

Core

- Describe functional requirements using, for example, use cases or user stories. 1. Using at least one method of documenting and structuring functional requirements. 2. Understanding how the method supports design and implementation. 3. Strengths and weaknesses of using a specific approach.
- Properties of requirements including consistency, validity, completeness, and feasibility.
- Requirements elicitation. 1. Sources of requirements, for example, users, administrators, or support personnel. 2. Methods of requirement gathering, for example, surveys, interviews, or

behavioral analysis.
- Non-functional requirements, for example, security, usability, or performance, also called as Quality Attributes. Sustainable Design and Pervasive Computing, Environmental Footprint of Computing Systems, Systemic Effects and Social Context
- Risk identification and management, including ethical considerations surrounding the proposed product. Ethics Codes and Legal Accountability, Professional Responsibility and Public Policy, Professional Values, Nature and Continuing Development, Ethical Dissent and Cultural Context, Workplace Ethics
- Communicating and/or formalizing requirement specifications.

Learning Outcomes:
Core:

1. Compare different methods of eliciting requirements along multiple axes [Compare]
2. Identify differences between two methods of describing functional requirements (e.g., customer interviews, user studies) and the situations where each would be preferred [Analyze]
3. Identify which behaviors are required, allowed, or barred from a given set of requirements and a list of candidate behaviors [Analyze]
4. Collect a set of requirements for a simple software system [Analyze]
5. Identify areas of a software system that need to be changed, given a description of the system and a set of new requirements to be implemented [Analyze]
6. Identify the functional and non-functional requirements in a set of requirements [Analyze]

3.13.6 SE/Requirements Evolution and Estimation (Core Tier2: 1 hr)

Topics:
Non Core

- Prototyping a tool for both eliciting and validating/confirming requirements.
- Product evolution: when requirements change, how to understand what effect that has and what changes need to be made.
- Effort estimation: 1. Learning techniques for better estimating the effort required to complete a task; 2. Practicing estimation and comparing it to how long tasks take; 3. Effort estimation is quite difficult, so students are likely to be way off in many cases, but seeing the process play out with their own work is valuable.

Learning Outcomes:
NonCore:

1. Create a prototype of a software system to validate a set of requirements - building a mock-up, MVP, etc [Create]
2. Estimate the time to complete a set of tasks, then compare estimates to the actual time taken [Estimate]
3. Determine an implementation sequence for a set of tasks, adhering to dependencies between them, with a goal to retire risk as early as possible [Determine]
4. Write a requirement specification for a simple software system [Write]

3.13.7 SE/Principles and Software Architecture (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- System design principles. System Reliability 1. Levels of abstraction (e.g., architectural design and detailed design) 2. Separation of concerns 3. Information hiding 4. Coupling and cohesion
- Software architecture. System Reliability 1. Design paradigms (a) Top-down functional decomposition/layered design (b) Data-oriented architecture (c) Object-oriented analysis and design (d) Event-driven design 2. Standard architectures (e.g., client-server and microservice architectures including REST discussions, n-layer, pipes-and-filters, Model View Controller) 3. Identifying component boundaries and dependencies
- Programming in the large vs programming in the small. System Reliability

- Code smells and other indications of code quality, distinct from correctness. Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering

Learning Outcomes:

Core:

1. Identify the standard software architecture of a given high-level design [Analyze]
2. Select and use an appropriate design paradigm to design a simple software system and explain how system design principles have been applied in this design [Evaluate]
3. Adapt a flawed system design to better follow principles such as separation of concerns or information hiding [Create]
4. Identify the dependencies among a set of software components in an architectural design [Analyze]

3.13.8 SE/API Design and Modeling (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- API design principles 1. Consistency (a) Consistent APIs are easier to learn and less error-prone (b) Consistency is both internal (between different portions of the API) and external (following common API patterns) 2. Composability 3. Documenting contracts (a) API operations should describe their effect on the system, but not generally their implementation (b) Preconditions, postconditions, and invariants 4. Expandability 5. Error reporting (a) Errors should be clear, predictable, and actionable (b) Input that does not match the contract should produce an error (c) Errors that can be reliably managed without reporting should be managed
- Identifying and codifying data invariants and time invariants
- Structural and behavioral models of software designs

Learning Outcomes:

Core:

1. Design an API for a single component of a large software system, including identifying and documenting each operation's invariants, contract, and error conditions [Design]
2. Evaluate an API description in terms of consistency, composability, and expandability [Evaluate]
3. Expand an existing design to include a new piece of functionality [Create]

3.13.9 SE/Software Design Quality and Evaluation (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:

Core

- Data design Data Modeling 1. Data structures 2. Storage systems
- Requirement traceability 1. Understanding which requirements are satisfied by a design

Non Core

- Design modeling, for instance with class diagrams, entity relationship diagrams, or sequence diagrams
- Measurement and analysis of design quality
- Principles of secure design and coding Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering 1. Principle of least privilege 2. Principle of fail-safe defaults 3. Principle of psychological acceptability
- Evaluating design tradeoffs (e.g., efficiency vs reliability, security vs usability)
- ISO/IEC 25010 quality characteristics model and cost of poor quality 1. Quality characteristics: functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, portability 2. Cost of poor quality and technical debt as a business/economic concern, not just a technical one

Learning Outcomes:

Core:

1. Design a set of data structures to implement a provided API surface [Design]
2. Identify which requirements are satisfied by a provided software design [Analyze]

NonCore:

3. Translate a natural language software design into class diagrams [Translate]
4. Adapt a flawed system design to better follow the principles of least privilege and fail-safe defaults [Create]
5. Contrast two software designs across different qualities, such as efficiency or usability [Contrast]
6. Evaluate a software system against the ISO/IEC 25010 quality characteristics and estimate the business cost of poor quality/technical debt [Evaluate]

3.13.10 SE/Coding Practices (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Practical small-scale testing Software Development Practices 1. Unit testing 2. Test-driven development - This is particularly valuable for students psychologically, as it is far easier to engage constructively with the challenge of identifying challenging inputs for a given API (edge cases, corner cases) a priori. If they implement first, the instinct is often to avoid trying to crash their new creation, while a test-first approach gives them the intellectual satisfaction of spotting the problem cases and then watching as more tests pass during the development process.
- Documentation Software Development Practices 1. Interface documentation - describe interface requirements, potentially including (formal or informal) contracts, pre and post conditions, invariants. 2. Implementation documentation should focus on tricky and non-obvious pieces of code, whether because the code is using advanced language features, or the behavior of the code is complex. (Do not add comments that re-state common/obvious operations and simple language features.) (a) Clarify dataflow, computation, etc., focusing on what the code is. (b) Identify subtle/tricky pieces of code and refactor to be self-explanatory if possible or provide appropriate comments to clarify.
- Coding style Software Development Practices 1. Style guides 2. Commenting 3. Naming
- "Best Practices" for coding: techniques, idioms/patterns, mechanisms for building quality programs Software Development Practices, Information Flow and Non-Interference, Injection and Input Validation, Memory Safety and Types, Malware Analysis and Advanced Security 1. Defensive coding practices 2. Secure coding practices and principles 3. Using exception handling mechanisms to make programs more robust, fault-tolerant
- Debugging Software Development Practices
- Logging
- Use of libraries and frameworks developed by others Software Development Practices

Learning Outcomes:
Core:

1. Write appropriate unit tests for a small component (several functions, a single type, etc.) [Write]
2. Write appropriate interface and (if needed) implementation comments for a small component [Write]
3. Describe techniques, coding idioms and mechanisms for implementing designs to achieve desired properties such as reliability, efficiency, and robustness [Describe]
4. Write robust code using exception handling mechanisms [Write]
5. Describe secure coding and defensive coding practices [Describe]
6. Select and use a defined coding standard in a small software project [Use]
7. Compare and contrast integration strategies including top-down, bottom-up, and sandwich integration [Compare]
8. Describe the process of analyzing and implementing changes to code base developed for a specific project [Describe]
9. Describe the process of analyzing and implementing changes to a large existing code base [Describe]

3.13.11 SE/Large-Scale Construction and Process (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Non Core

- Larger-scale testing 1. Test doubles (stubs, mocks, fakes) 2. Dependency injection
- Work sequencing, including dependency identification, milestones, and risk retirement 1. Dependency identification: Identifying the dependencies between different tasks 2. Milestones: A collection of tasks that serve as a marker of progress when completed. Ideally, the milestone encompasses a useful unit of functionality. 3. Risk retirement: Identifying what elements of a project are risky and prioritizing completing tasks that address those risks.
- Potential security problems in programs Information Flow and Non-Interference, Injection and Input Validation, Memory Safety and Types, Malware Analysis and Advanced Security 1. Buffer and other types of overflows 2. Race conditions 3. Improper initialization, including choice of privileges 4. Input validation
- Documentation (autogenerated)
- Development context: "green field" vs existing code base 1. Change impact analysis 2. Change actualization
- Release management
- DevOps practices 1. Definition of Done as a shared quality bar for completed work 2. Blameless postmortems for learning from incidents/failures 3. Team quality culture: shared ownership, feedback, and continuous learning

Learning Outcomes:
NonCore:

1. Rewrite a simple program to remove common vulnerabilities, such as buffer overflows, integer overflows and race conditions [Apply]
2. Write a software component that performs some non-trivial task and is resilient to input and run-time errors [Write]

3.13.12 SE/Verification and validation Concepts (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Verification and validation concepts 1. Verification: Are we building the thing right? 2. Validation: Did we build the right thing?
- Why testing matters: Does the component remain functional as the code evolves?
- Testing objectives 1. Usability 2. Reliability 3. Conformance to specification 4. Performance 5. Security

Learning Outcomes:
Core:

1. Explain why testing is important [Explain]
2. Distinguish between program validation and verification [Distinguish]
3. Describe different objectives of testing [Describe]

3.13.13 SE/Test Planning and Types (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Test kinds 1. Unit 2. Integration 3. Validation 4. System
- Stylistic differences between tests and production code: DAMP vs DRY - more duplication is warranted in test code.
- Test planning and generation 1. Test case generation, from formal models, specifications, etc. 2. Test coverage (a) Test matrices (b) Code coverage - how much of the code is tested? (c) Environment coverage - how many hardware architectures, operating systems, browsers, etc. are tested? 3. Test data and inputs

Learning Outcomes:
Core:

1. Compare and contrast the different types and levels of testing (regression, unit, integration, systems, and acceptance) [Compare]
2. Describe techniques for creating a test plan and generating test cases [Describe]
3. Create a test plan for a medium-size code segment which includes a test matrix and generation of test data and inputs [Create]
4. Implement a test plan for a medium-size code segment [Implement]

3.13.14 SE/Advanced Testing Practices (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Test development 1. Test-driven development 2. Object oriented testing, mocking, and dependency injection 3. Opaque-box (previously, black-box) and transparent-box (previously, white-box) testing techniques 4. Test tooling, including code coverage, static analysis, and fuzzing
- Verification and validation in the development cycle 1. Code reviews 2. Test automation, including automation of tooling 3. Pre-commit and post-commit testing 4. Tradeoffs between test coverage and throughput/latency of testing 5. Defect tracking and prioritization: reproducibility of reported defects
- Domain specific verification and validation challenges 1. Performance testing and benchmarking 2. Asynchrony, parallelism, and concurrency 3. Safety-critical 4. Numeric

Learning Outcomes:
Core:

1. Identify the fundamental principles of test-driven development methods and explain the role of automated testing in these methods [Analyze]
2. Discuss issues involving the testing of object-oriented software [Debate]
3. Describe mocking and dependency injection and their application [Describe]
4. Undertake, as part of a team activity, a code review of a medium-size code segment [Evaluate]
5. Describe the role that tools can play in the validation of software [Describe]
6. Automate the testing in a small software project [Design]
7. Explain the roles, pros, and cons of pre-commit and post-commit testing [Explain]
8. Discuss the tradeoffs between test coverage and test throughput/latency and how this can impact verification [Debate]
9. Use a defect tracking tool to manage software defects in a small software project [Use]
10. Discuss the limitations of testing in certain domains [Debate]

3.13.15 SE/Testing Tools and Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Non Core

- Verification and validation tooling and automation 1. Static analysis 2. Code coverage 3. Fuzzing 4. Dynamic analysis and fault containment (sanitizers, etc.) 5. Fault logging and fault tracking
- Test planning and generation 1. Fault estimation and testing termination including defect seeding 2. Use of random and pseudo random numbers in testing
- Testing asynchronous, parallel, and concurrent systems
- Verification and validation of non-code artifacts (documentation, training materials)
- Behavior-driven development (BDD) 1. Expressing acceptance criteria as Gherkin Given/When/Then scenarios 2. Executing BDD scenarios as automated end-to-end tests
- Mutation testing and static code-quality analysis 1. Mutation testing to assess test-suite effectiveness 2. Static detection of code smells and excessive cyclomatic complexity

Learning Outcomes:
NonCore:

1. Describe and compare different tools for verification and validation [Describe]

2. Automate the use of different tools in a small software project [Design]
3. Explain how and when random numbers should be used in testing [Explain]
4. Describe approaches for fault estimation [Describe]
5. Estimate the number of faults in a small software application based on fault density and fault seeding [Estimate]
6. Describe techniques and issues with testing asynchronous, concurrent, and parallel software [Describe]
7. Create a test plan for a medium-size code segment which contains asynchronous, concurrent, and/or parallel code, including a test matrix and generation of test data and inputs [Create]
8. Describe techniques for the verification and validation of non-code artifacts [Describe]
9. Write Gherkin Given/When/Then scenarios for a feature's acceptance criteria and execute them as automated end-to-end tests [Write]
10. Analyze a test suite's mutation score and refactor tests to kill surviving mutants [Analyze]

3.13.16 SE/Performance Testing and Benchmarking (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Non Core

- Performance testing and benchmarking 1. Throughput and latency 2. Degradation under load (stress testing, FIFO vs LIFO handling of requests) 3. Speedup and scaling (a) Amdahl's law (b) Gustafson's law (c) Soft and weak scaling 4. Identifying and measuring figures of merits 5. Common performance bottlenecks (a) Compute-bound (b) Memory-bandwidth bound (c) Latency-bound 6. Statistical methods and best practices for benchmarking (a) Estimation of uncertainty (b) Confidence intervals 7. Analysis and presentation (graphs, etc.) 8. Timing techniques

Learning Outcomes:
NonCore:

1. Describe throughput and latency and provide examples of each [Describe]
2. Explain speedup and the different forms of scaling and how they are computed [Explain]
3. Describe common performance bottlenecks [Describe]
4. Describe statistical methods and best practices for benchmarking software [Describe]
5. Explain techniques for and challenges with measuring time when constructing a benchmark [Explain]
6. Identify the figures of merit, construct and run a benchmark, and statistically analyze and visualize the results for a small software project [Analyze]

3.13.17 SE/API Compatibility and Versioning (Core Tier2: 2 hrs)

Topics:
Core

- Hyrum's Law/The Law of Implicit Interfaces
- Backward compatibility 1. Compatibility is not a property of a single entity, it's a property of a relationship. 2. Backward compatibility needs to be evaluated in terms of provider + consumer(s) or with a well-specified model of what forms of compatibility a provider aspires to/promises.
- Versioning 1. Semantic Versioning (SemVer) 2. Trunk-based development

Learning Outcomes:
Core:

1. Identify both explicit and implicit behavior of an interface and identify potential risks from Hyrum's Law [Analyze]
2. Identify changes that can be broadly considered "backward compatible," potentially with explicit statements about what usage is or is not supported [Analyze]
3. Evaluate whether a proposed change is sufficiently safe given the versioning methodology in use for a given project [Evaluate]

3.13.18 SE/Refactoring Techniques (Core Tier2: 1 hr)

Topics:
Core

- Refactoring 1. Standard refactoring patterns (rename, inline, outline, etc.) 2. Use of refactoring tools in IDE 3. Application of static-analysis tools (to identify code in need of refactoring, generate changes, etc.) 4. Value of refactoring as a remedy for technical debt

Non Core

- "Large Scale" Refactoring - techniques when a refactoring change is too large to commit safely (large projects), or when it is impossible to synchronize change between provider + all consumers (multiple repositories, consumers with private code). 1. Express both old and new APIs so that they can co-exist. 2. Minimize the size of behavior changes. 3. Why these techniques are required, (e.g., "API consumers I can see" vs "consumers I can't see").

Learning Outcomes:
Core:

1. Consider inputs from static analysis tools and/or Software Design principles to identify code in need of refactoring [Analyze]
2. Refactor the implementation of an interface to improve design, clarity, etc. with minimal/zero impact on existing users [Create]

NonCore:

3. Plan a complex multi-step refactoring to change default behavior of an API safely [Plan]

3.13.19 SE/Reliability Concepts (Core Tier2: 2 hrs)

Topics:
Core

- Concept of reliability as probability of failure or mean time between failures, and faults as cause of failures
- Identifying reliability requirements for different kinds of software
- Software failures caused by defects/bugs, and so for high reliability the goal is to have minimum defects - by injecting fewer defects (better training, education, planning), and by removing most of the injected defects (testing, code review, etc.)
- Software reliability, system reliability and failure behavior
- Defect injection and removal cycle, and different approaches for defect removal
- Compare the "error budget" approach to reliability with the "error-free" approach and identify domains where each is relevant.

Learning Outcomes:
Core:

1. Describe how to determine the level of reliability required by a software system [Describe]
2. Explain the problems that exist in achieving very high levels of reliability [Explain]
3. Understand approaches to minimizing faults that can be applied at each stage of the software lifecycle [Explain]

3.13.20 SE/Reliability Engineering (Core Tier2: 1 hr)

Topics:
Non Core

- Software reliability models
- Software fault tolerance techniques and models 1. Contextual differences in fault tolerance (e.g., crashing a flight critical system is strongly avoided, crashing a data processing system before corrupt data is written to storage is highly valuable)
- Software reliability engineering practices - including reviews, testing, practical model checking
- Identification of dependent and independent failure domains, and their impact on system reliability
- Measurement-based analysis of software reliability - telemetry, monitoring and alerting, dashboards, release qualification metrics, etc.

Learning Outcomes:
NonCore:

1. Demonstrate the ability to apply multiple methods to develop reliability estimates for a software system [Demonstrate]
2. Identify methods that will lead to the realization of a software architecture that achieves a specified level of reliability [Analyze]
3. Identify ways to apply redundancy to achieve fault tolerance [Analyze]
4. Identify single-point-of-failure (SPF) dependencies in a system design [Analyze]

3.13.21 SE/Formal Methods

Topics:
Non Core

- Formal specification of interfaces 1. Specification of pre- and post- conditions 2. Formal languages for writing and analyzing pre- and post-conditions.
- Problem areas well served by formal methods 1. Lock-free programming, data races 2. Asynchronous and distributed systems, deadlock, livelock, etc.
- Comparison to other tools and techniques for defect detection 1. Testing 2. Fuzzing
- Formal approaches to software modeling and analysis 1. Model checkers 2. Model finders

Learning Outcomes:
NonCore:

1. Describe the role formal specification and analysis techniques can play in the development of complex software and compare their use as validation and verification techniques with testing [Describe]
2. Apply formal specification and analysis techniques to software designs and programs with low complexity [Apply]
3. Explain the potential benefits and drawbacks of using formal specification languages [Explain]

3.14 Security (SEC)

Computing supports nearly every facet of modern critical infrastructure: transportation, communication, healthcare, education, energy generation and distribution, to name a few. With rampant attacks on and breaches of this infrastructure, computer science graduates have an important role in designing, implementing, and operating software systems that are robust, safe, and secure.

The Security (SEC) knowledge area focuses on developing a *security mindset* into the overall ethos of computer science graduates so that security is embedded in all their work products. Computer science students need to learn about system vulnerabilities and understand threats against computer systems. The *Security* title choice was intentional to serve as a one-word umbrella term for this knowledge area, which also includes concepts to support privacy, cryptography, secure systems, secure data, and secure code.

The SEC knowledge area relies on shared concepts pervasive in all the other areas of CS2023. It identifies seven crosscutting concepts of cybersecurity: *confidentiality*, *integrity*, *availability*, *risk assessment*, *systems thinking*, *adversarial thinking*, and *human-centered thinking*. The seventh concept, *human-centered thinking*, is additional to the six crosscutting concepts originally defined in the Cybersecurity Curricula 2017 (CSEC2017). This addition reinforces to students that humans are also a link in the overall chain of security. Principles of protecting systems include security-by-design, privacy-by-design, defense-in-depth, and zero-trust.

Another concept is the notion of assurance, which is an attestation that security mechanisms need to comply with the security policies that have been defined for data, processes, and systems. Assurance is tied in with the concepts of verification and validation. Considerations of data privacy and security are shared with the DM (technical aspects) and SEP knowledge areas.

The SEC knowledge area thus sits atop several of the other CS2023 knowledge areas, while including additional concepts that are not present in those knowledge areas. CS2023 treats security as a crucial component of the skillset of any CS graduate, and the hours needed for security preparation come from all the other 16 CS2023 knowledge areas.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.14.1 Security Mindset and Principles (Pg.. 127)	1	3
⭐ 3.14.2 Access Control and Applications (Pg.. 128)	1	3
⭐ 3.14.3 AI and Hardware Security (Pg.. 128)	1	2
⭐ 3.14.4 Security Ethics, Law and Society (Pg.. 128)	1	2
⭐ 3.14.5 Privacy and Security by Design (Pg.. 129)	1	1
⭐ 3.14.6 Injection and Input Validation (Pg.. 129)	1	2
⭐ 3.14.7 Memory Safety and Types (Pg.. 130)	1	1
⭐ 3.14.8 Information Flow and Non-Interference (Pg.. 130)	1	1
⭐ 3.14.9 Malware Analysis and Advanced Security (Pg.. 130)	1	3
⭐ 3.14.10 Cryptography Fundamentals (Pg.. 131)	1	2
⭐ 3.14.11 Cryptographic Algorithms (Pg.. 131)	1	2
⭐ 3.14.12 Protocols, Applications and Advanced Cryptography (Pg.. 132)	1	2
⭐ 3.14.13 Threat Analysis and Security Engineering (Pg.. 132)	1	4
⭐ 3.14.14 Security Design and Controls Engineering (Pg.. 133)	1	3
⭐ 3.14.15 Trusted Computing and Privacy Engineering (Pg.. 133)	1	3
⭐ 3.14.16 Digital Forensics Techniques (Pg.. 134)		4
⭐ 3.14.17 Legal Aspects and Incident Handling (Pg.. 134)		3
⭐ 3.14.18 Security Governance, Management and Policy (Pg.. 135)		2
⭐ 3.14.19 Risk and Data Management (Pg.. 135)		1

Table 3.14: List of KUs in the Security area.

3.14.1 SEC/Security Mindset and Principles (Core Tier1: 1 hr, Core Tier2: 3 hrs)

This knowledge unit introduces fundamental security concepts, crosscutting principles, and the development of a security mindset. It covers confidentiality, integrity, availability, risk assessment, systems thinking, adversarial thinking, and human-centered thinking. It also includes basic authentication, authorization, vulnerabilities, threats, attack surfaces, denial of service, protection principles, and the impact of AI on security and privacy.

Topics:

Core

- Developing a security mindset incorporating crosscutting concepts: confidentiality, integrity, availability, risk assessment, systems thinking, adversarial thinking, human-centered thinking.
- Vulnerabilities, threats, attack surfaces, and attack vectors Protection and Safety.
- Denial of Service (DoS) and Distributed Denial of Service (DDoS) Protection and Safety.
- Principles and practices of protection, e.g., least privilege, open design, fail-safe defaults, defense in depth, and zero trust; and how they can be implemented Principles of Operating System, Protection and Safety, Coding Practices, Large-Scale Construction and Process, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Optimization considerations between security, privacy, performance, and other design goals System Design, Software Development Practices, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Concepts of trust and trustworthiness.

Learning Outcomes:

Core:

1. Evaluate a system for possible attacks that can be launched by an adversary [Evaluate]
2. Design and develop approaches to protect a system from a set of identified threats [Design]
3. Describe how harm to user privacy can be avoided [Describe]
4. Develop a system that incorporates various principles of security and privacy [Create]

3.14.2 SEC/Access Control and Applications (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- Basic concepts of authentication and authorization/access control.
- Access control models (e.g., discretionary, mandatory, role-based, and attribute-based).
- Security controls.
- Applications of a security mindset: web, cloud, and mobile devices System Design, Common Aspects: Platforms, APIs and Constraints, Common Aspects: Platform Languages and Web Programming Patterns.
- Protecting embedded and cyber-physical systems Embedded Characteristics and Architectures, Real-Time, Memory and Verification, Systems, Programming and Sensors.
- Principles of usable security and human-centered computing System Design, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.

Learning Outcomes:
Core:

1. Compare the different access control models in terms of functionality and performance [Compare]

3.14.3 SEC/AI and Hardware Security (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Impact of AI on security and privacy: using AI to bolster defenses as well as address increased adversarial capabilities due to AI AI Applications and Societal Impact, Additional AI Applications and Societal Impact, Society, Ethics, and the Profession, System Design.
- Security and trust in AI/machine learning systems, e.g., fit for purpose, ethical operating boundaries, authoritative knowledge sources, verified training data, repeatable system evaluation tests, system attestation, independent validation/certification; unintended consequences from: adverse effect AI Applications and Societal Impact, Additional AI Applications and Societal Impact, Machine Learning Formulation and Evaluation, Deep Learning, RL and Ethics, Introduction to AI: Concepts and Intelligent Behavior, Problem Characteristics and Agent Nature, Machine Learning Fundamentals, Machine Learning Data, Representations and Evaluation, Neural Networks and Ethics, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Security risks in building and operating AI/machine learning systems (e.g., algorithm bias, knowledge corpus bias, training corpus bias, copyright violation) AI Applications and Societal Impact, Additional AI Applications and Societal Impact, Machine Learning Formulation and Evaluation, Deep Learning, RL and Ethics, Introduction to AI: Concepts and Intelligent Behavior, Problem Characteristics and Agent Nature, Machine Learning Fundamentals, Machine Learning Data, Representations and Evaluation, Neural Networks and Ethics.
- Hardware considerations in security, e.g., principles of secure hardware, secure processor architectures, cryptographic acceleration, compartmentalization, software-hardware interaction Assembly Level Machine Organization, Machine-Level Data Representation, Role and Purpose of Operating Systems.

Learning Outcomes:
Core:

1. Show how an adversary could use machine learning algorithms to reduce the security of a system [Analyze]
2. Show how a developer could improve the security of a system using machine learning algorithms [Analyze]
3. Describe hardware (especially CPU) vulnerabilities that can impact software [Describe]

3.14.4 SEC/Security Ethics, Law and Society (Core Tier1: 1 hr, Core Tier2: 2 hrs)

This unit covers societal, ethical, legal, and professional aspects of security and privacy, including security-by-design, privacy-by-design, ethical hacking, and the implications of emerging technologies.

Topics:
Core

- Societal impacts on breakdowns in security and privacy Digital Equity and Accessibility, Social Impact and AI, Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Applicability of laws and regulations on security and privacy Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Professional ethical considerations when designing secure systems and maintaining privacy; ethical hacking Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Ethics Codes and Legal Accountability, Professional Responsibility and Public Policy, Professional Values, Nature and Continuing Development, Ethical Dissent and Cultural Context, Workplace Ethics, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Security and privacy implications of malicious AI/machine learning actors, e.g., identifying deep fakes Machine Learning Formulation and Evaluation, Deep Learning, RL and Ethics, Introduction to AI: Concepts and Intelligent Behavior, Problem Characteristics and Agent Nature, Machine Learning Fundamentals, Machine Learning Data, Representations and Evaluation, Neural Networks and Ethics, Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Societal impacts of Internet of Things (IoT) devices and other emerging technologies on security and privacy Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.

Learning Outcomes:
Core:

1. Calculate the impact of a breakdown in security of a given system [Calculate]
2. Construct a system that conforms to security laws [Create]
3. Apply a set of privacy regulations to design a system that protects privacy [Apply]
4. Evaluate the legal ramifications of a system not corresponding to applicable laws and regulations [Evaluate]

3.14.5 SEC/Privacy and Security by Design (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Principles and practices of privacy Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Security by design System Design, System Security.
- Privacy by design and privacy engineering Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.

Learning Outcomes:
Core:

1. Construct a system that is designed to avoid harm to user privacy [Create]

3.14.6 SEC/Injection and Input Validation (Core Tier1: 1 hr, Core Tier2: 2 hrs)

This unit covers common software vulnerabilities, secure coding practices, defensive programming, and tools for static and dynamic analysis to prevent and mitigate security flaws.
Topics:
Core

- Common vulnerabilities and weaknesses.
- SQL injection and other injection attacks.
- Cross-site scripting techniques and mitigations.

- Input validation and data sanitization Protection and Safety, Basic Programming Concepts and Constructs, Programming Practices: Testing, Debugging, and Documentation, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.

Learning Outcomes:

Core:

1. Identify underlying problems in given examples of an enumeration of common weaknesses and explain how they can be circumvented [Analyze]
2. Apply input validation and data sanitization techniques to enhance security of a program [Apply]
3. Evaluate and prevent SQL injections attacks on a database application [Evaluate]
4. Evaluate and prevent cross-site scripting attacks against a website [Evaluate]
5. Construct a program that tests for all input handling errors [Create]

3.14.7 SEC/Memory Safety and Types (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Type safety and type-safe languages Advanced Type Theory, Systems Execution and Memory Model, Type Systems Fundamentals, Type Systems: Polymorphism and Complementary Typing, Protection and Safety, Basic Programming Concepts and Constructs, Programming Practices: Testing, Debugging, and Documentation, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Buffer overflows, stack smashing, and integer overflows Assembly Level Machine Organization, Systems Execution and Memory Model, Protection and Safety.
- Security issues due to race conditions Parallel and Distributed Computing, Evaluation.

Learning Outcomes:

Core:

1. Describe how the selection of a programming language can impact the security of the system being constructed [Describe]
2. Rewrite a program in a type-safe language (e.g., Java or Rust) originally written in an unsafe programming language (e.g., C/C++) [Redesign]
3. Evaluate a program for possible buffer overflow attacks and rewrite to prevent such attacks [Evaluate]
4. Evaluate a set of related programs for possible race conditions and prevent an adversary from exploiting them [Evaluate]

3.14.8 SEC/Information Flow and Non-Interference (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Principles of noninterference and nondeducibility.
- Preventing information flow attacks.

Learning Outcomes:

Core:

1. Construct a system that is protected from unauthorized information flows [Create]

3.14.9 SEC/Malware Analysis and Advanced Security (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:

Core

- Offensive security techniques as a defense.
- AI-assisted malware detection techniques.
- Ransomware: creation, prevention, and mitigation.

- Secure use of third-party components Coding Practices, Large-Scale Construction and Process, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Malware: varieties, creation, reverse engineering, and defense against them Language Translation and Execution, Systems Execution and Memory Model.
- Assurance: testing (including fuzzing and penetration testing), verification, and validation Protection and Safety, Basic Programming Concepts and Constructs, Programming Practices: Testing, Debugging, and Documentation, Coding Practices, Large-Scale Construction and Process, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Static and dynamic analyses Program Analysis and Analyzers, Protection and Safety, Evaluation, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Secure compilers and secure code generation Language Translation and Execution, Run-time Behavior and Systems.

Learning Outcomes:
Core:

1. Describe different kinds of malicious software [Describe]
2. Explain the risks of misusing interfaces with third-party code and how to correctly use third-party code [Explain]
3. Discuss the need to update software to fix security vulnerabilities and the lifecycle management of the fix [Debate]
4. Apply static and dynamic tools to identify programming faults [Apply]
5. Evaluate a system for the existence of malware and remove it [Evaluate]
6. Implement preventive techniques to reduce the occurrence of ransomware [Implement]

3.14.10 SEC/Cryptography Fundamentals (Core Tier1: 1 hr, Core Tier2: 2 hrs)

This unit introduces cryptographic principles, algorithms, protocols, and their applications in securing data and communications, including symmetric/asymmetric encryption, hashing, digital signatures, and emerging topics like quantum and post-quantum cryptography.
Topics:
Core

- Differences between algorithmic, applied, and mathematical views of cryptography.
- Mathematical preliminaries: modular arithmetic, Euclidean algorithm, probabilistic independence, linear algebra basics, number theory, finite fields, complexity, asymptotic analysis Discrete Mathematics, Linear Algebra.
- Basic cryptography: symmetric key and public key cryptography Foundational Data Structures, Foundational Algorithms, Advanced Algorithms, Discrete Mathematics.
- Classical cryptosystems, such as shift, substitution, transposition ciphers, code books, and machines Discrete Mathematics.
- Kerckhoff's principle and use of vetted libraries Coding Practices, Large-Scale Construction and Process.

Learning Outcomes:
Core:

1. Explain the role of cryptography in supporting security and privacy [Explain]
2. Discuss the risks of inventing one's own cryptographic methods [Debate]
3. Discuss the importance of prime numbers in cryptography and explain their use in cryptographic algorithms [Debate]
4. Implement and cryptanalyze classical ciphers [Implement]

3.14.11 SEC/Cryptographic Algorithms (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Basic cryptographic building blocks, including symmetric encryption, asymmetric encryption, hashing, and message authentication Discrete Mathematics.
- Additional mathematics: primality, factoring, and elliptic curve cryptography Discrete Mathematics.
- Private-key cryptosystems: substitution-permutation networks, linear cryptanalysis, differential cryptanalysis, DES, and AES Discrete Mathematics, Network Security.
- Public-key cryptosystems: Diffie-Hellman and RSA Discrete Mathematics.
- Data integrity and authentication: hashing, and digital signatures Data Security and Privacy, Discrete Mathematics.
- Attacker capabilities: chosen-message attack (for signatures), birthday attacks, side channel attacks, and fault injection attacks Network Security.

Learning Outcomes:
Core:

1. Describe how crypto keys can be managed securely [Describe]
2. Compare the space and time performance of a given set of cryptographic methods [Compare]
3. Discuss how modern private-key cryptosystems work and ways to cryptanalyze them [Debate]
4. Discuss how modern public-key cryptosystems work and ways to cryptanalyze them [Debate]
5. Compare different cryptographic algorithms in terms of security [Compare]
6. Explain key exchange protocols and show approaches to reduce their failure [Explain]

3.14.12 SEC/Protocols, Applications and Advanced Cryptography (Core Tier1: 1 hr, Core Tier2: 2 hrs)

Topics:
Core

- Usage of cryptography in real-world applications, e.g., electronic cash, secure channels between clients and servers, secure electronic mail, entity authentication, device pairing, steganography, and voting systems Image Processing, Network Security.
- Cryptographic protocols: challenge-response authentication, zero-knowledge protocols, commitment, oblivious transfer, secure two- or multi-party computation, hash functions, secret sharing, and applications Discrete Mathematics.
- Quantum cryptography; Post Quantum/Quantum resistant cryptography Foundational Data Structures, Foundational Algorithms, Advanced Algorithms, Discrete Mathematics.
- Blockchain and cryptocurrencies Discrete Mathematics, Communication.

Learning Outcomes:
Core:

1. Describe real-world applications of cryptographic primitives and protocols [Describe]
2. Discuss how quantum cryptography works and the impact of quantum computing on cryptographic algorithms [Debate]

3.14.13 SEC/Threat Analysis and Security Engineering (Core Tier1: 1 hr, Core Tier2: 4 hrs)

This unit covers systematic approaches to security engineering, including threat modeling, risk analysis, security design, attack domains, and trustworthy computing.
Topics:
Core

- Security engineering goals: building systems that remain dependable despite errors, accidents, or malicious adversaries Coding Practices, Large-Scale Construction and Process, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Problem analysis and situational analysis to address system security Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.

- Security analysis, covering security requirements analysis; security controls analysis; threat analysis; and vulnerability analysis Program Analysis and Analyzers, Evaluation.
- Security attack domains and attack surfaces, e.g., communications and networking, hardware, physical, social engineering, software, and supply chain Network Security.
- Security attack modes, techniques, and tactics, e.g., authentication abuse; brute force; buffer manipulation; code injection; content insertion; denial of service; eavesdropping; function bypass; impersonation; integrity attack; interception; phishing; protocol analysis; privilege abuse; spoofing; and traffic injection Network Security, Protection and Safety, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.

Learning Outcomes:
Core:

1. Create a threat model for a system or system design [Create]
2. Apply situational analysis to develop secure solutions under a specified scenario [Apply]
3. Identify and mitigate security vulnerabilities and weaknesses in a system [Analyze]

3.14.14 SEC/Security Design and Controls Engineering (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- Engineering tradeoff analysis based on time, cost, risk tolerance, risk acceptance, return on investment, and so on Evaluation, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Security design and engineering, including functional requirements, security subsystems, information protection, security testing, security assessment, and evaluation Evaluation, Performance Testing and Benchmarking, Requirements Engineering, Requirements Evolution and Estimation, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Design and development of cyber-physical systems.

Learning Outcomes:
Core:

1. Evaluate a given scenario for tradeoff analysis for system performance, risk assessment, and costs [Evaluate]
2. Design a set of technical security controls, countermeasures, and information protections to meet the security requirements and security objectives for a system [Design]
3. Evaluate the effectiveness of security functions, technical controls, and componentry for a system [Evaluate]

3.14.15 SEC/Trusted Computing and Privacy Engineering (Core Tier1: 1 hr, Core Tier2: 3 hrs)

Topics:
Core

- Privacy engineering goals: building systems that design, implement, and deploy privacy features and controls Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation.
- Attestation of software products with respect to their specification and adaptiveness Performance Testing and Benchmarking, Requirements Engineering, Requirements Evolution and Estimation, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Considerations for trustworthy computing, e.g., tamper resistant packaging, trusted boot, trusted kernel, hardware root of trust, software signing and verification, hardware-based cryptography, virtualization, and containers Coding Practices, Large-Scale Construction and Process, Per-

formance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.

Learning Outcomes:

Core:

1. Evaluate and predict emergent behavior in areas such as Data Science, AI, and Machine Learning [Evaluate]

3.14.16 SEC/Digital Forensics Techniques (Core Tier2: 4 hrs)

This unit covers principles, methodologies, and tools for digital forensics, including evidence collection, analysis, incident response, and legal considerations.

Topics:

Core

- Basic principles and methodologies for digital forensics.
- System design for forensics.
- Forensics in different situations: operating systems, file systems, application forensics, web forensics, network forensics, mobile device forensics, use of database auditing Network Security.
- Attacks on forensics and preventing such attacks.

Learning Outcomes:

Core:

1. Explain what a digital investigation is and how it can be implemented Computer Crimes and Attacks, Security Laws, Policies and Responsibility [Explain]
2. Design and implement software to support forensics [Design]
3. Extract data from a hard drive to comply with the law Computer Crimes and Attacks, Security Laws, Policies and Responsibility [Extract]
4. Recover data based on a given search term from an imaged system [Analyze]
5. Reconstruct data and events from an application history, or a web artifact, or a cloud database, or a mobile device. Mobile Platforms: Capabilities and Asynchronous Computing, Mobile Platforms: Architecture and Constraints, Mobile Platforms: Development Fundamentals, Cloud Architecture and Data Management, Frontend, Frameworks and Web Standards [Apply]
6. Capture and analyze network traffic. Network Security [Analyze]
7. Develop approaches to address the challenges associated with mobile device forensics [Create]
8. Apply forensics tools to investigate security breaches [Apply]
9. Identify and mitigate anti-forensic methods [Analyze]

3.14.17 SEC/Legal Aspects and Incident Handling (Core Tier2: 3 hrs)

Topics:

Core

- Incident handling processes.
- Rules of evidence - general concepts and differences between jurisdictions Computer Crimes and Attacks, Security Laws, Policies and Responsibility.
- Legal issues: digital evidence protection and management, chains of custody, reporting, serving as an expert witness Computer Crimes and Attacks, Security Laws, Policies and Responsibility.

Learning Outcomes:

Core:

1. Describe legal requirements for using seized data and its usage. Computer Crimes and Attacks, Security Laws, Policies and Responsibility [Describe]
2. Describe and implement an end-to-end chain of custody from initial digital evidence seizure to evidence disposal. Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Computer Crimes and Attacks, Security Laws, Policies and Responsibility [Describe]
3. Discuss a person's professional responsibilities and liabilities when testifying as a forensics expert Ethics Codes and Legal Accountability, Professional Responsibility and Public Policy, Professional Values, Nature and Continuing Development, Ethical Dissent and Cultural Context,

Workplace Ethics [Debate]

3.14.18 SEC/Security Governance, Management and Policy (Core Tier2: 2 hrs)

This unit covers organizational security governance, risk management, policy development, data lifecycle management, and protection of critical assets.

Topics:

Core

- Protecting critical assets from threats.
- Security governance: organizational objectives and general risk assessment.
- Security management: achieve and maintain appropriate levels of confidentiality, integrity, availability, accountability, authenticity, and reliability Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.
- Security policy: organizational policies, issue-specific policies, system-specific policies.

Learning Outcomes:

Core:

1. Describe critical assets and how they can be protected [Describe]
2. Differentiate between security governance, management, and controls, giving examples of each [Differentiate]
3. Describe a technical control and implement it to mitigate specific threats [Describe]

3.14.19 SEC/Risk and Data Management (Core Tier2: 1 hr)

Topics:

Core

- Approaches to identifying and mitigating risks to computing infrastructure.
- Data lifecycle management policies: data collection, backups, and retention; cloud storage and services; breach disclosure Data Security and Privacy.

Learning Outcomes:

Core:

1. Identify and assess risk of programs and database applications causing breaches [Analyze]
2. Design and implement appropriate backup strategies conforming to a given policy [Design]
3. Discuss a breach disclosure policy based on legal requirements and implement the policy [Debate]
4. Identify the risks and benefits of outsourcing to the cloud [Analyze]

3.15 Society, Ethics, and the Profession (SEP)

The ACM Code of Ethics and Professional Conduct states: "Computing professionals' actions change the world. To act responsibly, they should reflect upon the wider impacts of their work, consistently supporting the public good." The IEEE Code of Ethics starts by recognizing "the importance of our technologies in affecting the quality of life throughout the world." The AAAI Code of Professional Ethics and Conduct begins with "Computing professionals, and in particular, AI professionals' actions change the world. To act responsibly, they should reflect upon the wider impacts of their work, consistently supporting the public good."

While technical issues dominate the computing curriculum, they do not constitute a complete educational program in the broader context. It is more evident today than ever that students must also be exposed to the larger societal context of computing to develop an understanding of the critical and relevant social, ethical, legal, and professional issues and responsibilities at hand. This need to incorporate the study of these non-technical issues into the ACM curriculum was formally recognized in 1991.

Computer science educators may opt to deliver the material in this knowledge area within the contexts of traditional technical and theoretical courses, in dedicated courses, and as part of capstone, project, and professional practice courses. The material in this knowledge area is best covered through a combination of all the above.

The ACM Code of Ethics and Professional Conduct, the IEEE Code of Ethics, and the AAAI Code of Professional Ethics and Conduct provide guidance that serve as the basis for the conduct of all computing professionals in their work. The ACM Code emphasizes that ethical reasoning is not an algorithm to be followed, and computer professionals are expected to consider how their work impacts the public good as the primary consideration.

A computer science student must not graduate without understanding how society and ethics influence the computing profession. Nor should it be possible to complete a computer science degree without learning how computing professionals influence society, the ethical considerations involved in shaping that impact, and the student-turned-graduate's role in these relationships, as both computing professionals and members of society.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.15.1 Social Impact and AI (Pg.. 136)	2	1
⭐ 3.15.2 Digital Equity and Accessibility (Pg.. 137)	2	1
⭐ 3.15.3 Ethical Foundations and Frameworks (Pg.. 137)	2	1
⭐ 3.15.4 Ethical Analysis Methods and Tools (Pg.. 138)	1	1
⭐ 3.15.5 Professional Values, Nature and Continuing Development (Pg.. 138)	1	1
⭐ 3.15.6 Ethics Codes and Legal Accountability (Pg.. 139)	1	1
⭐ 3.15.7 Professional Responsibility and Public Policy (Pg.. 139)	1	1
⭐ 3.15.8 Ethical Dissent and Cultural Context (Pg.. 140)	1	1
⭐ 3.15.9 Workplace Ethics (Pg.. 140)	1	1
⭐ 3.15.10 Intellectual Property Foundations (Pg.. 140)	1	1
⭐ 3.15.11 Licensing, Plagiarism and Authorship (Pg.. 141)	1	1
⭐ 3.15.12 Digital IP Rights and Copyright (Pg.. 141)	1	1
⭐ 3.15.13 Open Source, Piracy and DRM (Pg.. 141)	1	1
⭐ 3.15.14 Data, Surveillance and Privacy Technology (Pg.. 142)	1	1
⭐ 3.15.15 Privacy Philosophical Foundations and Legislation (Pg.. 142)	1	1
⭐ 3.15.16 Civil Liberties and Expression (Pg.. 142)	1	1
⭐ 3.15.17 Technical and Professional Communication (Pg.. 143)	1	1
⭐ 3.15.18 Team Communication and Culture (Pg.. 143)	1	1
⭐ 3.15.19 Environmental Footprint of Computing Systems (Pg.. 144)	1	1
⭐ 3.15.20 Systemic Effects and Social Context (Pg.. 144)	1	1
⭐ 3.15.21 Sustainable Design and Pervasive Computing (Pg.. 144)	1	1
⭐ 3.15.22 Foundations and Early History of Computing (Pg.. 145)	1	1
⭐ 3.15.23 Modern History: PC, Internet and AI (Pg.. 145)	1	1
⭐ 3.15.24 Economic Models and Strategies (Pg.. 146)		1
⭐ 3.15.25 Automation, AI and Economic Ethics (Pg.. 146)		1
⭐ 3.15.26 Computer Crimes and Attacks (Pg.. 146)	1	1
⭐ 3.15.27 Security Laws, Policies and Responsibility (Pg.. 147)	1	1
⭐ 3.15.28 Identity and Diversity in Computing (Pg.. 147)	1	1
⭐ 3.15.29 DEIA Historical and Cross-Cultural Context (Pg.. 148)	1	1
⭐ 3.15.30 Inclusion and Computing Practices (Pg.. 148)	1	1
⭐ 3.15.31 Accessible Design (Pg.. 148)	1	1

Table 3.15: List of KUs in the Society, Ethics, and the Profession area.

3.15.1 SEP/Social Impact and AI (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Social Context provides the foundation for all other knowledge units in SEP, particularly Professional Ethics. It covers the social implications of computing in a hyper-networked world, the impact of computing applications on individual well-being and safety, the consequences of involving computing technologies in civic life, and how deficits in diversity and accessibility in computing affect society.

Topics:

Core

- Social implications (e.g., political and cultural ideologies) in a hyper-networked world where the capabilities and impact of social media, artificial intelligence, and computing in general are rapidly evolving.
- Impact of computing applications (e.g., social media, artificial intelligence applications) on individual well-being, and safety of all kinds (e.g., physical, emotional, economic).
- Consequences of involving computing technologies, particularly artificial intelligence, biometric technologies, and algorithmic decision-making systems, in civic life (e.g., facial recognition technology, biometric tags, resource distribution algorithms, policing software) and how human agency and oversight is crucial.
- Growth and control of the internet, data, computing, and artificial intelligence.

Learning Outcomes:

Core:

1. Describe the different ways that computer technology (networks, mobile computing, artificial intelligence) mediates social interaction at the personal and collective levels [Describe]
2. Interpret the social context of a given design and its implementation [Interpret]
3. Analyze the efficacy of a given design and implementation using empirical data [Analyze]
4. Understand the implications of technology use (e.g., social media) for different identities, cultures, and communities [Explain]
5. Describe the internet's role in facilitating communication between citizens, governments, and each other [Describe]
6. Analyze the effects of reliance on computing in the implementation of democracy (e.g., delivery of social services, electronic voting) [Analyze]
7. Describe the impact of a lack of appropriate representation of people from historically minoritized populations in the computing profession (e.g., industry culture, product diversity) [Describe]

3.15.2 SEP/Digital Equity and Accessibility (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:

Core

- How deficits in diversity and accessibility in computing affect society and what steps can be taken to improve equity in computing.
- Often referred to as the digital divide, differences in access to digital technology resources and its resulting ramifications for gender, class, ethnicity, geography, and/or developing countries, including consideration of responsibility to those who might be less wealthy, under threat, or who would struggle to have their voices heard.
- Accessibility issues, including legal requirements such as Web Content Accessibility Guidelines (www.w3.org/TR/WCAG21).
- Context-aware computing.

Learning Outcomes:

Core:

1. Identify developers' assumptions and values embedded in hardware and software design, especially as they pertain to usability for diverse populations including under-served and those with disabilities [Analyze]
2. Discuss the implications of context awareness in ubiquitous computing systems [Debate]
3. Express how access to the internet and computing technologies affect different societies [Explain]
4. Identify why/how internet access can be viewed as a human right [Analyze]

3.15.3 SEP/Ethical Foundations and Frameworks (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Ethical theories and principles are the foundations of ethical analysis because they are the viewpoints which can provide guidance along the pathway to a decision. Each theory emphasizes different assumptions and methods for determining the ethicality of a given action. It is important for students to recognize that decisions in different contexts may require different ethical theories (including combinations) to arrive at ethically acceptable outcomes, and what constitutes 'acceptable' depends on a variety of factors such as cultural context. Applying methods for ethical analysis requires both an

understanding of the underlying principles and assumptions guiding a given tool and an awareness of the social context for that decision. Traditional ethical frameworks as provided by western philosophy can be useful, but they are not all-inclusive. Effort must be taken to include decolonial, indigenous, and historically marginalized ethical perspectives whenever possible.

Topics:

Core

- Avoiding fallacies and misrepresentation in argumentation.
- Ethical theories and decision-making (philosophical and social frameworks, e.g. [1]).
- Recognition of the role culture plays in our understanding, adoption, design, and use of computing technology.
- Why ethics is important in computing, and how ethics is similar to, and different from, laws and social norms.
- Introduction to ethical frameworks (e.g., consequentialism such as utilitarianism, non-consequentialism such as duty, rights, or justice, agent-centered such as virtue or feminism, contractarianism, ethics of care) and their use for analyzing an ethical dilemma.

Learning Outcomes:

Core:

1. Describe how a given cultural context impacts decision making [Describe]
2. Express the use of example and analogy in ethical argument [Analyze]
3. Analyze (and avoid) basic logical fallacies in an argument [Analyze]
4. Analyze an argument to identify premises and conclusion [Analyze]
5. Evaluate how and why ethics is so important in computing and how it relates to cultural norms, values, and law [Evaluate]
6. Justify a decision made on ethical grounds [Justify]
7. Discuss the advantages and disadvantages of traditional ethical frameworks [Debate]
8. Analyze ethical dilemmas related to the creation and use of technology from multiple perspectives using ethical frameworks [Analyze]

3.15.4 SEP/Ethical Analysis Methods and Tools (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Professional checklists.
- Evaluation rubrics.
- Stakeholder analysis.
- Standpoint theory.

Learning Outcomes:

Core:

1. Distinguish all stakeholder positions in relation to their cultural context in a given situation [Distinguish]
2. Analyze the potential for introducing or perpetuating ethical debt (deferred consideration of ethical impacts or implications) in technical decisions [Analyze]

3.15.5 SEP/Professional Values, Nature and Continuing Development (Core Tier1: 1 hr, Core Tier2: 1 hr)

Computing professionals have responsibilities beyond their technical skills. This unit explores the community values and legal frameworks that guide the profession, the nature of professionalism (including care, discipline, and mentoring), and the professional's responsibility to remain current in a constantly evolving field.

Topics:

Core

- Community values and the laws by which we live.
- The nature of being a professional including care, attention, discipline, fiduciary responsibility, and mentoring.

- Keeping up to date as a computing professional in terms of familiarity, tools, skills, legal and professional frameworks as well as the ability and responsibility to self-assess and progress in the computing field.

Learning Outcomes:
Core:

1. Identify ethical issues that arise in software design, development practices, and software deployment [Analyze]
2. Discuss how to address ethical issues in specific situations [Debate]
3. Describe the mechanisms that typically exist for a professional to keep up to date in ethical matters [Describe]

3.15.6 SEP/Ethics Codes and Legal Accountability (Core Tier1: 1 hr, Core Tier2: 1 hr)

Professional ethics codes from organizations such as ACM, IEEE, and AAAI define standards of conduct and responsibility in computing. This unit examines professional certification, relevant ethics codes, and the legal and ethical dimensions of software accountability, including correctness, reliability, and security.

Topics:
Core

- Professional certification, codes of ethics, conduct, and practice, such as the ACM, IEEE, AAAI, and other international societies.
- Accountability, responsibility, and liability (e.g., software correctness, reliability and safety, warranty, negligence, strict liability, ethical approaches to security vulnerability disclosures) including whether a product/service should be built, not just doing so because it is technically possible.

Learning Outcomes:
Core:

1. Express the ethical responsibility of ensuring software correctness, reliability and safety including from where this responsibility arises (e.g., ACM/IEEE/AAAI Codes of Ethics, laws and regulations, organizational policies) [Analyze]
2. Describe the strengths and weaknesses of relevant professional codes as expressions of being a professional and guides to decision-making [Describe]

3.15.7 SEP/Professional Responsibility and Public Policy (Core Tier1: 1 hr, Core Tier2: 1 hr)

Computing professionals operate within complex social, legal, and cultural frameworks. This unit examines theories that explain how humans create and use technology—including decolonial and social justice approaches—and explores professionals' responsibilities to recognize, report, and maintain awareness of the ethical consequences of their work, including their participation in public policy.

Topics:
Core

- Introduction to theories describing the human creation and use of technology including instrumentalism, sociology of technological systems, disability justice, neutrality thesis, pragmatism, and decolonial models, including developing and using technology to right wrongs and do good.
- Strategies for recognizing and reporting designs, systems, software, and professional conduct (or their outcomes) that may violate law or professional codes of ethics.
- The role of the computing professional and professional societies in public policy.
- Maintaining awareness of consequences.

Learning Outcomes:
Core:

1. Analyze a global computing issue, observing the role of professionals and government officials in managing this problem [Analyze]

2. Describe ways in which professionals and professional organizations may contribute to public policy [Describe]
3. Describe the consequences of inappropriate professional behavior [Describe]

3.15.8 SEP/Ethical Dissent and Cultural Context (Core Tier1: 1 hr, Core Tier2: 1 hr)

Ethical dissent is a critical component of responsible professional conduct. This unit addresses whistleblowing as an accountability mechanism, and explores how regional cultural context shapes understanding and response to ethical dilemmas in the computing profession.

Topics:

Core

- Ethical dissent and whistleblowing.
- The relationship between regional culture and ethical dilemmas.

Learning Outcomes:

Core:

1. Be familiar with whistleblowing and have access to the knowledge to guide one through an incident [Assess]
2. Identify examples of how regional culture interplays with ethical dilemmas [Analyze]

3.15.9 SEP/Workplace Ethics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Dealing with harassment and discrimination.
- Forms of professional credentialing.
- Ergonomics and healthy computing environments.
- Time-to-market and cost considerations versus quality professional standards.

Learning Outcomes:

Core:

1. Describe forms of harassment and discrimination and avenues of assistance [Describe]
2. Assess various forms of professional credentialing [Assess]
3. State the relationship between ergonomics in computing environments and people's health [State]
4. Describe issues associated with industries' push to focus on time-to-market versus enforcing quality professional standards [Describe]

3.15.10 SEP/Intellectual Property Foundations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Intellectual property refers to a range of intangible rights of ownership in any product of human intellect, such as a software program. Laws, which vary by locality, provide different methods for protecting these rights of ownership based on their type. Ideally, intellectual property laws balance the interests of creators and users of the property. There are four types of intellectual property rights relevant to software: patents, copyrights, trade secrets, and trademarks. Moreover, property rights are often protected by user licenses. Each affords a different type of legal protection.

Topics:

Core

- Intellectual property rights.
- Legal foundations for intellectual property protection.
- Philosophical foundations of intellectual property.
- Forms of intellectual property (e.g., copyrights, patents, trade secrets, trademarks) and the rights they protect.

Learning Outcomes:

Core:

1. Defend legal and ethical uses of copyrighted materials [Defend]

2. Discuss the philosophical bases of intellectual property in an appropriate context (e.g., country) [Debate]
3. Contrast the protections and obligations of copyright, patent, trade secret, and trademarks [Contrast]
4. Describe the rationale for the legal protection of intellectual property in the appropriate context (e.g., country) [Describe]

3.15.11 SEP/Licensing, Plagiarism and Authorship (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Common software licenses (e.g., MIT, GPL and its variants, Apache, Mozilla, Creative Commons).
- Plagiarism and authorship.

Learning Outcomes:
Core:

1. Select an appropriate software license for a given project [Analyze]
2. Interpret the intent and implementation of software licensing [Interpret]
3. Analyze the ethical issues inherent in various plagiarism detection mechanisms [Analyze]
4. Identify multiple forms of plagiarism beyond verbatim copying of text or software (e.g., intentional paraphrasing, authorship misrepresentation, and improper attribution) [Analyze]

3.15.12 SEP/Digital IP Rights and Copyright (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Intangible digital intellectual property (IDIP).
- Limitations on copyright protections, including fair use and the first sale doctrine.
- Intellectual property laws and treaties that impact the enforcement of copyrights.

Learning Outcomes:
Core:

1. Describe and critique legislation and precedent aimed at digital copyright infringements [Describe]
2. Identify contemporary examples of intangible digital intellectual property [Analyze]
3. Discuss whether a use of copyrighted material is likely to be fair use [Debate]
4. Analyze the use of copyrighted work under the concepts of fair use and the first sale doctrine [Analyze]
5. Discuss the nature of anti-circumvention laws in the context of copyright protection [Debate]

3.15.13 SEP/Open Source, Piracy and DRM (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Software piracy and technical methods for enforcing intellectual property rights, such as digital rights management and closed source software as a trade secret.
- Moral and legal foundations of the open-source movement.
- Systems that use others' data (e.g., large language models).

Learning Outcomes:
Core:

1. Distinguish the conflicting issues involved in securing software patents [Distinguish]
2. Identify the goals of the open-source movement and its impact on fields beyond computing, such as the right-to-repair movement [Analyze]
3. Summarize the global nature of software piracy [Summarize]
4. Criticize the use of technical measures of digital rights management (e.g., encryption, watermarking, copy restrictions, and region lockouts) from multiple stakeholder perspectives [Criticize]

3.15.14 SEP/Data, Surveillance and Privacy Technology (Core Tier1: 1 hr, Core Tier2: 1 hr)

The massive collection of data in transactional systems, surveillance platforms, cloud computing, and artificial intelligence poses fundamental challenges for privacy. This unit examines the implications of these systems, conceptions of anonymity and pseudonymity, and technological solutions for protecting privacy, such as end-to-end encryption and differential privacy.

Topics:

Core

- Privacy implications of widespread data collection including but not limited to transactional databases, data warehouses, surveillance systems, cloud computing, and artificial intelligence.
- Conceptions of anonymity, pseudonymity, and identity.
- Technology-based solutions for privacy protection (e.g., end-to-end encryption and differential privacy).

Learning Outcomes:

Core:

1. Evaluate solutions to privacy threats in transactional databases and data warehouses [Evaluate]
2. Describe the role of data collection in the implementation of pervasive surveillance systems (e.g., RFID, face recognition, toll collection, mobile computing) [Describe]
3. Distinguish the concepts and goals of anonymity and pseudonymity [Distinguish]
4. Describe the ramifications of technology-based privacy protections, including differential privacy and end-to-end encryption [Describe]

3.15.15 SEP/Privacy Philosophical Foundations and Legislation (Core Tier1: 1 hr, Core Tier2: 1 hr)

The right to privacy has deep philosophical and legal roots that vary by cultural and legal context. This unit examines philosophical and legal conceptions of privacy, international legal frameworks such as GDPR, and sector-specific legislation such as HIPAA and the EU AI Act, critically evaluating their intent, value, and implementation.

Topics:

Core

- Philosophical and legal conceptions of the nature of privacy including the right to privacy.
- Legal foundations of privacy protection in relevant jurisdictions (e.g., GDPR in the EU).
- Privacy legislation in areas of practice (e.g., HIPAA in the US, AI Act in the EU).

Learning Outcomes:

Core:

1. Discuss the philosophical basis for the legal protection of personal privacy in an appropriate context (e.g., country) [Debate]
2. Critique the intent, potential value, and implementation of various forms of privacy legislation and principles beyond what the law requires [Critique]

3.15.16 SEP/Civil Liberties and Expression (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Civil liberties, privacy rights, and cultural differences.
- Basic Principles of human-subjects research and principles beyond what the law requires (e.g., Belmont Report, UN Universal Declaration on Human Rights and how this relates to technology).
- Freedom of expression and its limitations.
- User-generated content, content moderation, and liability.

Learning Outcomes:

Core:

1. Identify cultural differences regarding the nature and necessity of privacy and other civil liberties [Analyze]

2. Identify strategies to enable appropriate freedom of expression [Analyze]

3.15.17 SEP/Technical and Professional Communication (Core Tier1: 1 hr, Core Tier2: 1 hr)

Computing is an inherently collaborative and social discipline making communication an essential aspect of the profession. Much but not all of this communication occurs in a professional setting where communication styles, expectations, and norms differ from other contexts where similar technology might be used. Both professional and informal communication conveys information to various audiences who may have different goals and needs for that information. Good communication is also necessary for transparency and trustworthiness. It is also important to note that computing professionals are not just communicators but are also listeners who must be able to hear and thoughtfully make use of feedback received from various stakeholders. Effective communication skills are not something one 'just knows' - they are developed and can be learned. Communication skills are best taught in context throughout the undergraduate curriculum.

Topics:

Core

- Oral, written, and electronic team and group communication.
- Technical communication materials (e.g., source code, and documentation, tutorials, reference materials, API documentation).
- Communicating with different stakeholders such as customers, leadership, or the public.
- Tradeoffs in competing factors that affect communication channels and choices.

Learning Outcomes:

Core:

1. Understand the importance of writing concise and accurate technical documents following well-defined standards for format and for including appropriate tables, figures, and references [Produce]
2. Analyze written technical documentation for technical accuracy, concision, lack of ambiguity, and awareness of audience [Analyze]
3. Compose and deliver an audience-aware, accessible, and organized formal presentation [Compose]
4. Identify and describe qualities of effective communication (e.g., virtual, face-to-face, intragroup, shared documents) [Analyze]
5. Choose an appropriate way to communicate delicate ethical concerns [Assess]

3.15.18 SEP/Team Communication and Culture (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Team collaboration (including tools) and conflict resolution.
- Accessibility and inclusivity requirements for addressing professional audiences.
- Cultural competence in communication including considering the impact of difference in natural language.
- Communicating to solve problems or make recommendations in the workplace, such as raising ethical concerns or addressing accessibility issues.

Learning Outcomes:

Core:

1. Plan interactions (e.g., virtual, face-to-face, shared documents) with others in ways that invite inclusive participation, model respectful consideration of others' contributions, and explicitly value diversity of ideas [Plan]
2. Understand how to communicate effectively and appropriately as a member of a team including conflict resolution techniques [Apply]
3. Discuss ways to influence performance and results in diverse and cross-cultural teams [Debate]
4. Assess personal strengths and weaknesses to work remotely as part of a team drawing from diverse backgrounds and experiences [Assess]

3.15.19 SEP/Environmental Footprint of Computing Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

The lifecycle of hardware and software carries significant environmental costs. This unit examines the environmental and social impacts of computing systems—from e-waste and carbon footprint to energy consumption in data centers and AI models—evaluating their effect on design decisions and global society.

Topics:

Core

- Environmental, social, and cultural impacts of implementation decisions (e.g., sustainability goals, algorithmic bias/outcomes, economic viability, and resource consumption).
- Local/regional/global social and environmental impacts of computing systems and their use (e.g., carbon footprints, resource usage, e-waste) due to hardware (e.g., e-waste, data centers, rare element and resource utilization, recycling) and software (e.g., cloud-based services, blockchain, AI model training and use). This includes everyday use of hardware (cheap hardware replaced frequently) and software (web-browsing, email, and other services with hidden/remote computational demands).

Learning Outcomes:

Core:

1. Assess the environmental impacts of a given project's deployment (e.g., energy consumption, contribution to e-waste, impact of manufacturing) [Assess]
2. Describe global social and environmental impacts of computer use and disposal [Describe]
3. Describe the environmental impacts of design choices within the field of computing that relate to algorithm design, operating system design, networking design, database design, etc [Describe]
4. Assess computing applications in respect to environmental issues (e.g., energy, pollution, resource usage, recycling and reuse, food management and production) [Assess]

3.15.20 SEP/Systemic Effects and Social Context (Core Tier1: 1 hr, Core Tier2: 1 hr)

Complex computing technologies such as generative AI, social media, and remote work generate systemic effects that are difficult to predict. This unit analyzes how software sustainability depends on surrounding social systems—users, organizations, markets, and policies—and how technological designs in turn transform those systems.

Topics:

Core

- Systemic effects of complex computing technologies and phenomena (e.g., generative AI, data centers, social media, offshoring, remote work).
- How the sustainability of software systems is interdependent with social systems, including the knowledge and skills of its users, organizational processes and policies, and its societal context (e.g., market forces, government policies).

Learning Outcomes:

Core:

1. List the sustainable effects of modern practices and activities (e.g., remote work, e-commerce, cryptocurrencies, AI models, data centers) [List]
2. Analyze the social and environmental impacts of new system designs [Analyze]

3.15.21 SEP/Sustainable Design and Pervasive Computing (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Guidelines for sustainable design standards.
- Pervasive computing: Information processing that has been integrated into everyday objects and activities, such as smart energy systems, social networking, and feedback systems to promote sustainable behavior, transportation, environmental monitoring, citizen science and activism.

Learning Outcomes:
Core:

1. Identify ways to be a sustainable practitioner in a specific area or with a specific project [Analyze]
2. Design guidelines for sustainable IT design or deployment in areas such as smart energy systems, social networking, transportation, agriculture, supply-chain systems, environmental monitoring, and citizen activism [Design]

3.15.22 SEP/Foundations and Early History of Computing (Core Tier1: 1 hr, Core Tier2: 1 hr)

History is important because it provides a mechanism for understanding why our computing systems operate the way they do, the societal contexts in which current approaches arose, and how those continue to echo through the discipline today. Not only does computing affect society but vice-versa, resulting in a complex socio-technical context that is constantly changing, requiring the perspective of history to put the present, as well as possible futures, into appropriate perspective. It also informs decisions based on successes and failures of the past including harm done and how to not repeat them. The history of computing is often taught in context with foundational concepts, such as system fundamentals and software development fundamentals. A focus should be placed on those who, due to marginalization, have not historically featured as prominently as they should.
Topics:
Core

- The history of computing: hardware, software, and human/organizational.
- The role of history in the present including within different social contexts, and the relevance of this history on the future.
- Age I (Pre-digital): Ancient analog computing (Stonehenge, Antikythera mechanism, Salisbury Cathedral clock, etc.), human-calculated number tables, Euclid, Lovelace, Babbage, Gödel, Church, Turing, pre-electronic (electro-mechanical and mechanical) hardware.
- Age II (Early modern computing): ENIAC, UNIVAC, Bombes (Bletchley Park and codebreakers), computer companies (e.g., IBM), mainframes, etc.

Learning Outcomes:
Core:

1. Understand the relevance and impact of computing history on recent events, present context, and possible future outcomes, from more than one cultural perspective [Assess]
2. Discuss how perspectives held today have been shaped by history, and that alternative perspectives exist (e.g., fears of AI replacing human workers vs AI augmenting human work, various views on Moore's Law) [Debate]
3. Identify formative and consequential trends in the history of the computing field [Analyze]
4. Identify the contributions of pioneering individuals or organizations (research labs, computer companies, government offices) in the computing field [Analyze]

3.15.23 SEP/Modern History: PC, Internet and AI (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Age III (PC era): PCs, modern computer hardware and software, Moore's Law.
- Age IV (Internet): Networking, internet architecture, browsers and their evolution, standards, born-on-the-internet companies, and services (e.g., Google, Amazon, Microsoft, etc.), distributed computing.
- Age V (Mobile & Cloud): Mobile computing and smartphones, cloud computing and models thereof (e.g., SaaS), remote servers, security and privacy, social media.
- Age VI (AI): Decision making systems, recommender systems, generative AI and other machine learning driven tools and technologies.

Learning Outcomes:
Core:

1. Discuss the historical context for important moments in history of computing, such as the move from vacuum tubes to transistors (TRADIC), early seminal operating systems (e.g., OS 360), Xerox PARC and the first Apple computer with a GUI, the creation of specific programming language paradigms, the first computer virus, the creation of the internet, the creation of the WWW, the dot com bubble, Y2K, the introduction of smartphones, etc [Debate]
2. Compare daily life before and after the advent of milestone developments (e.g., personal computers or the internet) [Compare]

3.15.24 SEP/Economic Models and Strategies (Core Tier2: 1 hr)

The economies of computing are important to those who develop and provide computing resources and services to others as well as society in general. They impact users of these resources and services, both professional and non-professional. Computing professionals have a duty to know the impact of these topics on their own roles and activities and how choices made will impact users and society.

Topics:

Core

- Economic models: regulated and unregulated, monopolies, network effects, and open market; knowledge and attention economies.
- Pricing and deployment strategies: planned obsolescence, subscriptions, freemium, software licensing, open-source, free software, adware.
- Impacts of differences in access to computing resources, and the effect of skilled labor supply and demand on the quality of computing products.

Learning Outcomes:

Core:

1. Summarize the social effects of economic models (e.g., the knowledge and attention economies) [Summarize]
2. Describe the differences and similarities of competing strategies (e.g., subscription vs freemium vs free) [Describe]
3. Discuss examples of digital divides [Debate]

3.15.25 SEP/Automation, AI and Economic Ethics (Core Tier2: 1 hr)

Topics:

Core

- Automation, AI, and their effects on job markets, developers, and users.
- Ethical concerns surrounding the attention economy and other economies of computing (e.g. informed consent, data collection, use of verbose legalese in user agreements).

Learning Outcomes:

Core:

1. Understand the effects of automation and AI on society [Assess]
2. Understand the ethical implications of computing economies that rely on attention and data [Assess]

3.15.26 SEP/Computer Crimes and Attacks (Core Tier1: 1 hr, Core Tier2: 1 hr)

While security policies, laws and computer crimes are important topics, it is essential they are viewed with the foundation of other social and professional knowledge units, such as Intellectual Property, Privacy and Civil Liberties, Social Context, and Professional Ethics. Computers, the internet, and artificial intelligence, perhaps more than any other technologies, have transformed society over the past 75 years. At the same time, these technologies have contributed to unprecedented threats to privacy; new categories of computer crime and antisocial behavior; major disruptions to organizations; and the large-scale concentration of risk in information systems.

Topics:

Core

- Computer crimes, legal redress for computer criminals and impact on victims and society.
- Social engineering, computing-enabled fraud, identity theft and recovery from these.

- Cyber terrorism, criminal hacking, and hacktivism.
- Malware, viruses, worms.
- Attacks on critical infrastructure such as electrical grids and pipelines.

Learning Outcomes:
Core:

1. List classic examples of computer crimes and social engineering incidents with societal impact [List]
2. Identify issues with laws that apply to computer crimes [Analyze]
3. Describe the motivation and ramifications of cyber terrorism, data theft, hacktivism, ransomware, and other attacks [Describe]
4. Examine the ethical and legal issues surrounding the misuse of access and various breaches of security [Examine]

3.15.27 SEP/Security Laws, Policies and Responsibility (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core

- Non-technical fundamentals of security (e.g., human engineering, policy, confidentiality).
- Benefits and challenges of existing and proposed computer crime laws.
- Security policies and the challenges of change and compliance.
- Responsibility for security throughout the computing life cycle.
- International and local laws and how they intersect.

Learning Outcomes:
Core:

1. Discuss the professional's role in security and the tradeoffs and challenges involved [Debate]
2. Investigate measures that can be taken by both individuals and organizations including governments to prevent or mitigate the undesirable effects of computer crimes [Investigate]
3. Design a company-wide security policy, which includes procedures for managing passwords and employee monitoring [Design]
4. Understand how legislation from one region may affect activities in another (e.g., how EU GDPR applies globally, when EU persons are involved) [Assess]

3.15.28 SEP/Identity and Diversity in Computing (Core Tier1: 1 hr, Core Tier2: 1 hr)

Identity—including race, gender, background, and ability—shapes both the experience of computing professionals and the products they create. This unit examines how identity interacts with computing technologies and environments, and the demonstrable benefits of diverse development teams, as well as the costs of a lack of diversity.

Topics:
Core

- How identity impacts and is impacted by computing technologies and environments (academic and professional).
- The benefits of diverse development teams and the impacts of teams that are not diverse.

Learning Outcomes:
Core:

1. Define and distinguish equity, equality, diversity, inclusion, and accessibility [Define]
2. Identify examples of the benefits that diverse teams can bring to software products, and how a lack of diversity has costs [Analyze]
3. Assess the impact of power and privilege in the computing profession as it relates to culture, industry, products, and society [Assess]

3.15.29 SEP/DEIA Historical and Cross-Cultural Context (Core Tier1: 1 hr, Core Tier2: 1 hr)

Current inequalities in computing have deep historical roots in systemic social mechanisms. This unit examines historical marginalization in computing, global infrastructure challenges, and cross-cultural differences in DEIA, exploring how systemic changes can improve equity in both familiar and unfamiliar contexts.

Topics:

Core

- Historic marginalization due to systemic social mechanisms, technological supremacy and global infrastructure challenges to diversity, equity, inclusion, and accessibility.
- Cross-cultural differences in, and needs for, diversity, equity, inclusion, and accessibility.

Learning Outcomes:

Core:

1. Compare the demographics of your institution's computer science and STEM majors to the overall institutional demographics. If they differ, identify factors that contribute to inequitable access, engagement, and achievement in computer science among marginalized groups. If they do not, assess why not [Compare]
2. Develop examples of systemic changes that could positively address diversity, equity, inclusion, and accessibility in a familiar context (i.e., in an introductory computing course) and an unfamiliar context and when these might be different, or the same [Create]
3. Compare the demographics of your institution to the overall community demographics. If they differ, identify factors that contribute to inequitable access, engagement, and achievement among marginalized groups. If they do not, assess why not [Compare]

3.15.30 SEP/Inclusion and Computing Practices (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Inclusive language and charged terminology, and why their use matters.
- Inclusive behaviors and why they matter.
- How computing professionals can influence and impact diversity, equity, inclusion and accessibility, including but not only through the software they create.
- Experts and their practices that reflect the identities of the classroom and the world through practical DEIA principles.

Learning Outcomes:

Core:

1. Identify language, practices, and behaviors that may make someone feel included in a workplace and/or a team, and why is it relevant. Avoid charged terminology - see Words Matter (www.acm.org/diversity-inclusion/words-matter) - this includes identifying and accommodating users who are often excluded without thought and not considered at all [Analyze]
2. Demonstrate collegiality and respect when working with team members who do not share your identity. It is not enough to merely assign team projects. Faculty should prepare students for teamwork and monitor, mentor, and assess the effectiveness of their student teams throughout a project [Demonstrate]
3. Identify developers' assumptions and values embedded in hardware and software design, especially as they pertain to usability by diverse populations [Analyze]
4. Analyze the work of experts who reflect the identities of the classroom and the world [Analyze]

3.15.31 SEP/Accessible Design (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:

Core

- Designing and developing technology with accessibility in mind.

Learning Outcomes:
Core:

1. Evaluate the accessibility of your classroom or lab. Evaluate the accessibility of your webpage. (See www.w3.org/WAI and www.w3.org/TR/WCAG21) [Evaluate]

3.16 System Fundamentals (SF)

A computer system is a set of hardware and software infrastructures upon which applications are constructed. Computer systems have become a pillar of people's daily life. As such, it is essential for students to learn knowledge about computer systems, grasp the skills to use and design these systems, and understand the fundamental rationale and principles in computer systems. It could equip students with the necessary competence for a career related to computer science.

In the curriculum of computer science, the study of computer systems typically spans multiple knowledge areas, including, but not limited to, operating systems, parallel and distributed systems, communications networks, computer architecture and organization, and software engineering. The System Fundamentals knowledge area, as suggested by its name, focuses on the fundamental concepts and design principles in computer systems that are shared by these courses within their respective cores. The goal of this knowledge area is to present an integrative view of these fundamental concepts and design principles in a unified albeit simplified fashion, providing a common foundation for the different specialized mechanisms and policies appropriate to the specific domain area. The fundamental concepts in this knowledge area include an overview of computer systems, basic concepts such as state and state transition, resource allocation and scheduling, and so on. Moreover, this knowledge area introduces basic design principles to improve the reliability, availability, efficiency, and security of computer systems.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.16.1 Hardware Components and Logic (Pg.. 149)	2	
⭐ 3.16.2 Layers, Abstractions and Scalability (Pg.. 150)	2	
⭐ 3.16.3 Digital Systems and Logic (Pg.. 150)	2	
⭐ 3.16.4 State Machines (Pg.. 150)	2	
⭐ 3.16.5 Sequential and Parallel Processing (Pg.. 151)	2	
⭐ 3.16.6 Resource Management (Pg.. 151)	1	1
⭐ 3.16.7 Latency, Cache and Memory Hierarchy (Pg.. 152)	2	2
⭐ 3.16.8 Virtualization and Isolation (Pg.. 152)	1	1
⭐ 3.16.9 Performance Metrics and Benchmarks (Pg.. 153)	1	1
⭐ 3.16.10 Analytical Performance Models (Pg.. 153)	1	1
⭐ 3.16.11 Events, Tools and Experimentation (Pg.. 153)	1	1
⭐ 3.16.12 System Reliability (Pg.. 154)	2	1
⭐ 3.16.13 System Security (Pg.. 154)	2	1
⭐ 3.16.14 System Design (Pg.. 155)	2	1
3.16.15 Society, Ethics, and the Profession (Pg.. 155)	Elective	

Table 3.16: List of KUs in the System Fundamentals area.

3.16.1 SF/Hardware Components and Logic (Core Tier1: 2 hrs)

Introduces the building blocks and basic components of a computer, hardware as a computational paradigm, and the concept of pipelining.
Topics:
Core

- Basic building blocks and components of a computer (gates, flip-flops, registers, interconnections; datapath + control + memory).
- Hardware as a computational paradigm: Fundamental logic building blocks; Logic expressions, minimization, sum of product forms Digital Logic and Digital Systems.
- Basic concept of pipelining, overlapped processing stages Functional Organization.

Learning Outcomes:
Core:

1. Describe the basic building blocks of computers and their role in the historical development of computer architecture [Describe]
2. Design a simple logic circuit using the fundamental building blocks of logic design to solve a simple problem (e.g., adder) [Design]

3.16.2 SF/Layers, Abstractions and Scalability (Core Tier1: 2 hrs)

Covers programming abstractions, the distinction between application and operating system services, the concept of scaling, and the layered organization of computer systems.
Topics:
Core

- Programming abstractions, interfaces, use of libraries Programs: Distribution, Programs: Execution Properties, GPU Programming, Programs: Implementation Mappings, Programs: Declarative Parallelism, Programs: Starting Activities.
- Distinction and interaction between application and OS services, remote procedure call Role and Purpose of Operating Systems.
- Basic concept of scaling: performance vs problem size.

Learning Outcomes:
Core:

1. Describe how computing systems are constructed of layers upon layers, based on separation of concerns, with well-defined interfaces, hiding details of low layers from the higher layers [Describe]
2. Describe that hardware, OS, VM, and application are additional layers of interpretation/processing [Describe]
3. Describe the mechanisms of how errors are detected, signaled back, and handled through the layers [Describe]
4. Construct a simple program (e.g., a TCP client/server) using methods of layering, error detection and recovery, and reflection of error status across layers [Create]
5. Identify bugs in a layered program by using tools for program tracing, single stepping, and debugging [Analyze]
6. Understand the concept of strong vs weak scaling, i.e., how performance is affected by the scale of the problem vs the scale of resources to solve the problem. This can be motivated by simple, real-world examples [Assess]

3.16.3 SF/Digital Systems and Logic (Core Tier1: 2 hrs)

Covers digital vs. analog systems, logic gates, Boolean expressions, clocks, and combinational and sequential logic, as the foundation for understanding computer systems operation.
Topics:
Core

- Digital vs Analog/Discrete vs Continuous Systems.
- Simple logic gates, logical expressions, Boolean logic simplification.
- Clocks, State, Sequencing.
- Combinational Logic, Sequential Logic, Registers, Memories Digital Logic and Digital Systems.

Learning Outcomes:
Core:

1. Describe the differences between digital and analog systems, and between discrete and continuous systems. Can give real-world examples of these systems [Describe]

3.16.4 SF/State Machines (Core Tier1: 2 hrs)

Introduces the concept of state and state transitions, finite state machines (FSMs), and their application to computers and network protocols as examples of state machines.

Topics:
Core

- State and state transition (e.g., starting state, final state, life cycle of states) Formal Languages and Automata.
- Finite state machines (e.g., NFA, DFA) Formal Languages and Automata.
- Computers and Network Protocols as examples of State Machines Fundamentals of Networks and Communications.

Learning Outcomes:
Core:

1. Describe computations as a system characterized by a known set of configurations with transitions from one unique configuration (state) to another (state) [Describe]
2. Describe the distinction between systems whose output is only a function of their input (stateless) and those with memory/history (stateful) [Describe]
3. Develop state machine descriptions for simple problem statement solutions (e.g., traffic light sequencing, pattern recognizers) [Create]
4. Describe a computer as a state machine that interprets machine instructions [Describe]
5. Explain how a program or network protocol can also be expressed as a state machine and that alternative representations for the same computation can exist [Explain]
6. Derive the time-series behavior of a state machine from its state machine representation (e.g., TCP connection management state machine) [Analyze]

3.16.5 SF/Sequential and Parallel Processing (Core Tier1: 2 hrs)

Covers sequential and parallel processing, including single-thread and multi-thread models at the application level, and the evaluation of performance of parallel program versions.
Topics:
Core

- Sequential vs parallel processing. Concurrency, Programs: Distribution, Programs: Execution Properties, GPU Programming, Programs: Implementation Mappings, Programs: Declarative Parallelism, Programs: Starting Activities.
- Application-level sequential processing: single thread Concurrency, Programs: Distribution, Programs: Execution Properties, GPU Programming, Programs: Implementation Mappings, Programs: Declarative Parallelism, Programs: Starting Activities.
- Simple application-level parallel processing: request level (web services/client-server/distributed), single thread per server, multiple threads with multiple servers, pipelining Concurrency, Programs: Distribution, Programs: Execution Properties, GPU Programming, Programs: Implementation Mappings, Programs: Declarative Parallelism, Programs: Starting Activities.

Learning Outcomes:
Core:

1. Write a simple sequential problem and a simple parallel version of the same program [Write]
2. Evaluate the performance of simple sequential and parallel versions of a program with different problem sizes and be able to describe the speed-ups achieved [Evaluate]
3. Describe on an execution timeline how parallelism events and operations can take place simultaneously (i.e., at the same time). Explain how work can be performed in less elapsed time if this can be exploited [Describe]

3.16.6 SF/Resource Management (Core Tier1: 1 hr, Core Tier2: 1 hr)

This knowledge unit covers different types of resources in computer systems and common resource allocation and scheduling algorithms.
Topics:
Core

- Different types of resources (e.g., processor share, memory, disk, net bandwidth) Fundamentals of Networks and Communications, File Systems API and Implementation, Memory Management, Scheduling.

- Common resource allocation/scheduling algorithms (e.g., first-come-first-serve, priority-based scheduling, fair scheduling, and preemptive scheduling) Scheduling.
- Advantages and disadvantages of common scheduling algorithms Scheduling.

Learning Outcomes:

Core:

1. Define how finite computer resources (e.g., processor share, memory, storage, and network bandwidth) are managed by their careful allocation to existing entities [Define]
2. Describe how common resource allocation/scheduling algorithms work [Describe]
3. Develop common scheduling algorithms and evaluate their performances [Create]
4. Describe the pros and cons of common scheduling algorithms [Describe]

3.16.7 SF/Latency, Cache and Memory Hierarchy (Core Tier1: 2 hrs, Core Tier2: 2 hrs)

Covers latencies in computer systems, spatial and temporal locality in caches, cache coherency, and the processor memory hierarchy with its average memory access time formula.

Topics:

Core

- Latencies in computer systems: 1. Speed of light and computers (one foot per nanosecond vs one GHz clocks) Functional Organization 2. Memory vs disk latencies vs across-the-network memory Memory Hierarchy, Performance and Energy Efficiency
- Caches and the effects of spatial and temporal locality on performance in processors and systems Memory Hierarchy, Performance and Energy Efficiency, Memory Management.
- Caches and cache coherency in databases, operating systems, distributed systems, and computer architecture Memory Hierarchy, DBMS Internals, Memory Management.
- Introduction to the processor memory hierarchy Memory Hierarchy, Performance and Energy Efficiency.
- The formula for average memory access time Memory Hierarchy.

Learning Outcomes:

Core:

1. Describe the breakdown of the latency of computer systems in terms of memory, disk, and network [Describe]
2. Explain the importance of locality in determining system performance [Explain]
3. Calculate average memory access time and describe the tradeoffs in memory hierarchy performance in terms of capacity, miss/hit rate, and access time [Calculate]

3.16.8 SF/Virtualization and Isolation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Covers the rationale for isolation and virtualization for protection and predictable performance, levels of indirection via virtual memory, and methods for implementing virtual memory and virtual machines.

Topics:

Core

- Rationale of virtualization and isolation: protection and predictable performance Virtualization.
- Levels of indirection, illustrated by virtual memory for managing physical memory resources Virtualization.
- Methods for implementing virtual memory and virtual machines Virtualization.

Learning Outcomes:

Core:

1. Explain why it is important to isolate and protect the execution of individual programs and environments that share common underlying resources [Explain]
2. Describe how the concept of indirection can create the illusion of a dedicated machine and its resources even when physically shared among multiple programs and environments [Describe]
3. Evaluate the performance of two application instances running on separate virtual machines and determine the effect of performance isolation [Evaluate]

3.16.9 SF/Performance Metrics and Benchmarks (Core Tier1: 1 hr, Core Tier2: 1 hr)

Covers performance figures of merit, representative workloads and benchmarks, and collection and analysis methods for the quantitative evaluation of computer system performance.

Topics:

Core

- Performance figures of merit Performance and Energy Efficiency, Evaluation.
- Workloads and representative benchmarks, and methods of collecting and analyzing performance figures of merit Performance and Energy Efficiency, Evaluation.

Learning Outcomes:

Core:

1. Explain how the components of system architecture contribute to improving its performance [Explain]
2. Explain the circumstances in which a given figure of a system performance metric is useful [Explain]
3. Explain the usage and inadequacies of benchmarks as a measure of system performance [Explain]

3.16.10 SF/Analytical Performance Models (Core Tier1: 1 hr, Core Tier2: 1 hr)

Covers the CPI equation as a design tool, Amdahl's Law and its limitations, order of magnitude analysis (Big O), critical path analysis, and analytical tools for quantitative evaluation.

Topics:

Core

- CPI (Cycles per Instruction) equation as a tool for understanding tradeoffs in the design of instruction sets, processor pipelines, and memory system organizations Performance and Energy Efficiency, Evaluation.
- Amdahl's Law: the part of the computation that cannot be sped up limits the effect of the parts that can Performance and Energy Efficiency, Evaluation.
- Order of magnitude analysis (Big O notation) Asymptotic Notation and Complexity Classes, Complexity Analysis: Recursion, Amortization, and Tight Bounds, Complexity Analysis Framework, Computational Complexity Theory.
- Analysis of slow and fast paths of a system Functional Organization.
- Analytical tools to guide quantitative evaluation.

Learning Outcomes:

Core:

1. Describe Amdahl's law and discuss its limitations [Describe]
2. Apply limit studies or simple calculations to produce order-of-magnitude estimates for a given performance metric in a given context [Apply]

3.16.11 SF/Events, Tools and Experimentation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Covers events and their effect on performance (stalls, cache misses, page faults), understanding layered systems as evaluation platforms, microbenchmark pitfalls, and tools and experiments for measuring system performance.

Topics:

Core

- Events on their effect on performance (e.g., instruction stalls, cache misses, page faults) Functional Organization, Memory Management.
- Understanding layered systems, workloads, and platforms, their implications for performance, and the challenges they represent for evaluation.
- Microbenchmark pitfalls.

Learning Outcomes:

Core:

1. Apply software tools to profile and measure program performance [Apply]

2. Design and conduct a performance-oriented experiment of a common system (e.g., an OS and Spark) [Design]
3. Design a performance experiment on a layered system to determine the effect of a system parameter on system performance [Design]

3.16.12 SF/System Reliability (Core Tier1: 2 hrs, Core Tier2: 1 hr)

This knowledge unit covers the distinction between bugs, faults, and failures; reliability vs availability; and reliability through redundancy including check and retry, redundant encoding, and duplication.

Topics:

Core

- Distinction between bugs, faults, and failures Coordination, Reliability Concepts, Reliability Engineering.
- Reliability vs availability.
- Reliability through redundancy: 1. check and retry Reliability Support, Fault tolerance 2. redundant encoding (error correction codes, CRC, FEC, RAID) Memory Hierarchy, Distributed Databases/Cloud Computing, Reliability Support 3. duplication/mirroring/replicas Distributed Databases/Cloud Computing
- Other approaches to reliability (e.g., journaling) Reliability Support, Fault tolerance, Reliability Concepts, Reliability Engineering.

Learning Outcomes:

Core:

1. Explain the distinction between program errors, system errors, and hardware faults (e.g., corrupted memory) and exceptions (e.g., attempt to divide by zero) [Explain]
2. Articulate the distinction between detecting, handling, and recovering from faults and the methods for their implementation [Articulate]
3. Describe the role of error correction codes in providing error checking and correction techniques in memories, storage, and networks [Describe]
4. Apply simple algorithms for exploiting redundant information for the purposes of data correction [Apply]
5. Compare different error detection and correction methods for their data overhead, implementation complexity, and relative execution time for encoding, detecting, and correcting errors [Compare]

3.16.13 SF/System Security (Core Tier1: 2 hrs, Core Tier2: 1 hr)

This knowledge unit covers common system security issues and countermeasures, including cryptography and security architecture, as well as representative countermeasure systems such as intrusion detection systems and antivirus systems.

Topics:

Core

- Common system security issues (e.g., viruses, denial-of-service attacks, and eavesdropping) Network Security, Protection and Safety, Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering, AI and Hardware Security, Access Control and Applications, Security Mindset and Principles.
- Countermeasures Network Security, Principles of Operating System, Protection and Safety: 1. Cryptography Protocols, Applications and Advanced Cryptography, Cryptographic Algorithms, Cryptography Fundamentals 2. Security architecture Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering
- Representative countermeasure systems: 1. Intrusion detection systems, firewalls Network Security 2. Antivirus systems

Learning Outcomes:

Core:

1. Describe some common system security issues and give examples [Describe]
2. Describe some countermeasures against system security issues [Describe]

3. Describe representative countermeasure systems [Describe]

3.16.14 SF/System Design (Core Tier1: 2 hrs, Core Tier2: 1 hr)

This knowledge unit covers common criteria of system design (e.g., liveness, safety, robustness, scalability, and security) and the designs of representative systems.

Topics:

Core

- Common criteria of system design (e.g., liveness, safety, robustness, scalability, and security) Evaluation.
- Designs of representative systems (e.g., Apache web server, Spark, and Linux).

Learning Outcomes:

Core:

1. Describe common criteria of system design [Describe]
2. Given the functionality requirements of a system and its key design criteria, provide a high-level design of this system [Generate]
3. Describe the design of some representative systems [Describe]

3.16.15 SF/Society, Ethics, and the Profession

This knowledge unit covers intellectual property rights of computer systems, common software licenses, and computer crimes.

Topics:

Core

- Intellectual property rights of computer systems Open Source, Piracy and DRM, Digital IP Rights and Copyright, Intellectual Property Foundations, Licensing, Plagiarism and Authorship.
- Common software licenses Open Source, Piracy and DRM, Digital IP Rights and Copyright, Intellectual Property Foundations, Licensing, Plagiarism and Authorship.
- Computer crimes Computer Crimes and Attacks, Security Laws, Policies and Responsibility.

Learning Outcomes:

Core:

1. Describe the intellectual property rights of computer systems [Describe]
2. List representative software licenses and compare their differences [List]
3. List representative computer crimes [List]

3.17 Specialized Platform Development (SPD)

The Specialized Platform Development (SPD) knowledge area refers to attributes involving the creation of software targeting non-traditional hardware platforms. Developing for each specialized platform, for example, robots, mobile systems, web-based systems, and embedded systems, typically involves unique considerations.

Societal and industry needs have created a high demand for developers on specialized platforms, such as mobile applications, web platforms, robotic platforms, and embedded systems. Some unique professional abilities relevant to this knowledge area include the following:

- Creating applications that provide a consistent user experience across various devices, screen sizes, and operating systems.
- Developing application programming interfaces (APIs) to support the functionality of each specialized platform.
- Managing challenges related to resource constraints such as computation, memory, storage, and networking and communication.
- Applying cross-cutting concerns such as optimization, security, better development practices, etc.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 3.17.1 Common Aspects: Platforms, APIs and Constraints (Pg.. 156)	3	2
⭐ 3.17.2 Common Aspects: Platform Languages and Web Programming Patterns (Pg.. 157)	2	1
⭐ 3.17.3 Frontend, Frameworks and Web Standards (Pg.. 157)		3
⭐ 3.17.4 Cloud Architecture and Data Management (Pg.. 157)		3
⭐ 3.17.5 Mobile Platforms: Development Fundamentals (Pg.. 158)		2
⭐ 3.17.6 Mobile Platforms: Architecture and Constraints (Pg.. 158)		2
⭐ 3.17.7 Mobile Platforms: Capabilities and Asynchronous Computing (Pg.. 158)		1
⭐ 3.17.8 Robot Platforms (Pg.. 158)		4
⭐ 3.17.9 Embedded Characteristics and Architectures (Pg.. 159)		1
⭐ 3.17.10 Systems, Programming and Sensors (Pg.. 159)		2
⭐ 3.17.11 Real-Time, Memory and Verification (Pg.. 160)		3
⭐ 3.17.12 Platform History, Types and Characteristics (Pg.. 160)		1
⭐ 3.17.13 Sensors, Controllers and eSports Ecosystems (Pg.. 161)		1
⭐ 3.17.14 Architectures and Rendering (Pg.. 161)		1
⭐ 3.17.15 Algorithms, Media and Virtual Worlds (Pg.. 161)		1
⭐ 3.17.16 Languages, APIs and Development Tools (Pg.. 161)		1
⭐ 3.17.17 Game Engines (Pg.. 162)		1
⭐ 3.17.18 Game Design Foundations (Pg.. 162)		1
⭐ 3.17.19 Genres, Audiences and Social Reach (Pg.. 162)		1
⭐ 3.17.20 Interactive Data and Analytics Platforms (Pg.. 163)		2
⭐ 3.17.21 AI Programming and Quantum Platforms (Pg.. 163)		1
3.17.22 SPD-SEP/Mobile (Pg.. 164)	Elective	
3.17.23 SPD-SEP/Web (Pg.. 164)	Elective	
3.17.24 Intellectual Property in Games (Pg.. 164)	Elective	
3.17.25 Game Interface and Accessibility (Pg.. 165)	Elective	
3.17.26 Health, Safety and Sustainability (Pg.. 165)	Elective	
3.17.27 Representation, Inclusion and Equality (Pg.. 165)	Elective	
3.17.28 SPD-SEP/Robotics (Pg.. 166)	Elective	
3.17.29 SPD-SEP/Interactive (Pg.. 166)	Elective	

Table 3.17: List of KUs in the Specialized Platform Development area.

3.17.1 SPD/Common Aspects: Platforms, APIs and Constraints (Core Tier1: 3 hrs, Core Tier2: 2 hrs)

Topics:
Core

- Overview of development platforms (i.e., web, mobile, game, robotics, embedded, and interactive). 1. Input/sensors/control devices/haptic devices 2. Resource constraints (a) Computational (b) Data storage (c) Memory (d) Communication 3. Requirements - security, uptime availability, fault tolerance Reliability Concepts, Reliability Engineering, Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering 4. Output/actuators/haptic devices
- Programming via platform-specific Application Programming Interface (API) vs traditional application construction
- Programming under platform constraints and requirements (e.g., available development tools, development, security considerations) AI and Hardware Security, Access Control and Applications, Security Mindset and Principles

Learning Outcomes:
Core:

1. List the constraints of mobile programming [List]

3.17.2 SPD/Common Aspects: Platform Languages and Web Programming Patterns (Core Tier1: 2 hrs, Core Tier2: 1 hr)

Topics:
Core

- Overview of platform Languages (e.g., Python, Swift, Lua, Kotlin)
- Techniques for learning and mastering a platform-specific programming language

Learning Outcomes:
Core:

1. List the characteristics of scripting languages [List]
2. Describe the three-tier model of web programming [Describe]
3. Describe how the state is maintained in web programming [Describe]

3.17.3 SPD/Frontend, Frameworks and Web Standards (Core Tier2: 3 hrs)

Covers web programming languages, frameworks, the SaaS model, web standards (DOM, accessibility), and security and privacy considerations.
Topics:
Core

- Web programming languages (e.g., HTML5, JavaScript, Typescript, CSS)
- Web platforms, frameworks, or meta-frameworks 1. Cloud services 2. API, Web Components
- Software as a Service (SaaS)
- Web standards such as document object model, accessibility Accessibility and Inclusive Design
- Security and Privacy Considerations Computer Crimes and Attacks, Security Laws, Policies and Responsibility

Learning Outcomes:
Core:

1. Compare and contrast web programming with general-purpose programming [Compare]
2. Describe the differences between Software-as-a-Service (SaaS) and traditional software products [Describe]
3. Discuss how web standards impact software development [Debate]
4. Evaluate an existing web application against current web standards [Evaluate]

3.17.4 SPD/Cloud Architecture and Data Management (Core Tier2: 3 hrs)

Covers requirements analysis for web applications, cloud computing services, data management, architectures (microservices, serverless, event-driven), and storage solutions.
Topics:
Non Core

- Analyzing requirements for web applications
- Computing services NoSQL Systems 1. Cloud Hosting 2. Scalability (e.g., Autoscaling, Clusters) 3. Cost estimation for services
- Data management Core Database System Concepts 1. Data residency: where the data is located and what paths can be taken to access it 2. Data integrity: guaranteeing data is accessible and that data is deleted when required
- Architecture 1. Monoliths vs Microservices 2. Micro-frontends 3. Event-Driven vs RESTful architectures: advantages and disadvantages 4. Serverless, cloud computing on demand
- Storage solutions NoSQL Systems, Relational Databases 1. Relational Databases 2. NoSQL databases

Learning Outcomes:
Core:

1. Design and implement a web-based application using a microservice architecture design [Design]
2. Describe the constraints, such as hosting, services, and scalability, related to web platforms [Describe]

3.17.5 SPD/Mobile Platforms: Development Fundamentals (Core Tier2: 2 hrs)

Topics:
Core

- Development with 1. Mobile programming languages 2. Mobile programming environments
- Mobile platform constraints 1. User interface design Understanding the User: Individual goals and interactions with others 2. Security
- Access 1. Accessing data through APIs Query Construction 2. Designing API endpoints for mobile apps: pitfalls and design considerations 3. Network and web interfaces Data Modeling, Fundamentals of Networks and Communications

Learning Outcomes:
Core:

1. Compare mobile programming with general-purpose programming [Compare]
2. Evaluate the pros and cons of native and cross-platform mobile application development [Evaluate]

3.17.6 SPD/Mobile Platforms: Architecture and Constraints (Core Tier2: 2 hrs)

Topics:
Non Core

- Development 1. Native versus cross-platform development 2. Software design/architecture patterns for mobile applications API Design and Modeling, Principles and Software Architecture, Software Design Quality and Evaluation
- Mobile platform constraints 1. Responsive user interface design Accessibility and Inclusive Design 2. Heterogeneity and mobility of devices 3. Differences in user experiences (e.g., between mobile and web-based applications) 4. Power and performance tradeoff
- Specification and testing Software Development Practices, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices

Learning Outcomes:
NonCore:

1. Design the architecture of a mobile application applying appropriate software design patterns, justifying the choice between native and cross-platform development [Design]
2. Evaluate the constraints of a given mobile platform (responsive UI, device heterogeneity, energy-performance tradeoffs) when specifying and testing an application [Evaluate]

3.17.7 SPD/Mobile Platforms: Capabilities and Asynchronous Computing (Core Tier2: 1 hr)

Topics:
Non Core

- Mobile computing affordances 1. Location-aware applications 2. Sensor-driven computing (e.g., gyroscope, accelerometer, health data from a watch) 3. Telephony and instant messaging 4. Augmented reality Immersion
- Asynchronous computing Algorithms 1. Difference from traditional synchronous programming 2. Handling success via callbacks 3. Handling errors asynchronously 4. Testing asynchronous code and typical problems in testing

Learning Outcomes:
Core:

1. Develop a location-aware mobile application with data API integration [Create]
2. Build a sensor-driven mobile application capable of logging data on a remote server [Create]
3. Create a communication app incorporating telephony and instant messaging [Create]

3.17.8 SPD/Robot Platforms (Core Tier2: 4 hrs)

Topics:
Core

- Types of robotic platforms and devices Robotics
- Sensors, embedded computation, and effectors (actuators) Tangible/Physical Computing
- Robot-specific languages and libraries Robotics
- Robotic software architecture (e.g., using the Robot Operating System (ROS))
- Robotic platform constraints and design considerations Robotics
- Interconnections with physical or simulated systems Simulation, Tangible/Physical Computing
- Robotic Algorithms Robotics, Computer Animation 1. Forward kinematics 2. Inverse kinematics 3. Dynamics 4. Navigation and path planning 5. Grasping and manipulation
- Safety and interaction considerations Digital Equity and Accessibility, Social Impact and AI, Ethics Codes and Legal Accountability, Professional Responsibility and Public Policy, Professional Values, Nature and Continuing Development, Ethical Dissent and Cultural Context, Workplace Ethics

Learning Outcomes:

Core:

1. Design and implement an application on a given robotic platform [Design]
2. Integrate an Arduino-based robot kit and program it to navigate a maze [Integrate]
3. Compare robot-specific languages and techniques with those used for general-purpose software development [Compare]
4. Explain the rationale behind the design of the robotic platform and its interconnections with physical or simulated systems [Explain]
5. Given a high-level application, design a robot software architecture using ROS specifying all components and interconnections (ROS topics) to accomplish that application [Design]
6. Discuss the constraints a given robotic platform imposes on developers [Debate]

3.17.9 SPD/Embedded Characteristics and Architectures (Core Tier2: 1 hr)

Covers the unique characteristics of embedded systems compared to other systems, APIs for custom architectures (GPU, FPGA), and the evaluation of architectures according to requirements.

Topics:

Core

- Introduction to the unique characteristics of embedded systems 1. Real-time vs soft real-time and non-real-time systems 2. Resource constraints, such as memory profiles and deadlines Memory Hierarchy
- API for custom architectures 1. GPU technology Heterogeneous Architectures, Shading and Advanced Rendering 2. Field Programmable Gate Arrays (FPGA) Digital Logic and Digital Systems 3. Cross-platform systems

Learning Outcomes:

Core:

1. Describe the unique characteristics of embedded systems versus other systems [Describe]
2. Identify different types of embedded architectures [Analyze]
3. Evaluate which architecture is best for a given set of requirements [Evaluate]

3.17.10 SPD/Systems, Programming and Sensors (Core Tier2: 2 hrs)

Covers embedded systems with microcontrollers, interrupts and energy efficiency, embedded programming, and interfacing with sensors and actuators.

Topics:

Core

- Embedded Systems 1. Microcontrollers 2. Interrupts and feedback 3. Interrupt handlers in high-level languages Hardware Components and Logic, Layers, Abstractions and Scalability 4. Hard and soft interrupts and trap-exits Principles of Operating System 5. Interacting with hardware, actuators, and sensors 6. Energy efficiency 7. Loosely timed coding and synchronization 8. Software adapters
- Embedded programming
- Sensors and actuators

Learning Outcomes:
Core:
1. Interface with sensors/actuators [Program]
2. Debug a problem with an existing embedded platform [Analyze]
3. Design and develop software to interact with and control hardware [Design]

3.17.11 SPD/Real-Time, Memory and Verification (Core Tier2: 3 hrs)

Covers hard and soft real-time systems, real-time and memory resource management, safety considerations, analysis and verification, and embedded application design.
Topics:
Core
- Hard real-time systems vs soft real-time systems Real-time and Embedded Systems 1. Timeliness 2. Time synchronization/scheduling 3. Prioritization 4. Latency 5. Compute jitter
- Real-time resource management
- Memory management 1. Mapping programming construct (variable) to a memory location Memory Hierarchy 2. Shared memory Memory Management 3. Manual memory management. 4. Garbage collection Language Translation and Execution
- Safety considerations and safety analysis Digital Equity and Accessibility, Social Impact and AI, Ethics Codes and Legal Accountability, Professional Responsibility and Public Policy, Professional Values, Nature and Continuing Development, Ethical Dissent and Cultural Context, Workplace Ethics
- Analysis and verification
- Application design

Learning Outcomes:
Core:
1. Design and implement a small embedded system for a given platform (e.g., a smart alarm clock or a drone) [Design]
2. Design methods for real-time systems [Design]
3. Evaluate real-time scheduling and schedulability analysis [Evaluate]
4. Evaluate formal specification and verification of timing constraints and properties [Evaluate]

3.17.12 SPD/Platform History, Types and Characteristics (Core Tier2: 1 hr)

Covers the historical evolution of game platforms, typical platform types, and their technical characteristics and constraints.
Topics:
Core
- Evolution of Game Platforms (e.g., Brown Box to Metaverse and beyond; improvement in computing architectures (CPU and GPU); platform convergence and mobility) Digital Logic and Digital Systems
- Typical Game Platforms (e.g., Personal Computer; Home Console; Handheld Console; Arcade Machine; Interactive Television; Mobile Phone; Tablet; Integrated Head-Mounted Display; Immersive Installations and Simulators; Internet of Things enabled Devices; CAVE Systems; Web Browsers; Cloud-based Streaming Systems)
- Characteristics and Constraints of Different Game Platforms (e.g., features: local storage, internetworking, peripherals; run-time performance: GPU/CPU frequency, number of cores; chipsets: physics processing units, vector co-processors; expansion bandwidth (PCIe); network throughput (Ethernet); memory types and capacities (DDR/GDDR); maximum stack depth; power consumption; thermal design; Endian)

Learning Outcomes:
Core:
1. Describe the key constraints a specific game platform will likely impose on developers [Describe]
2. Compare the interfaces of different game platforms, highlighting their respective implications for human-computer interaction [Compare]

3.17.13 SPD/Sensors, Controllers and eSports Ecosystems (Core Tier2: 1 hr)

Covers typical sensors, controllers and actuators in game platforms, and eSports ecosystems including events, streamed media and professional play.

Topics:

Core

- Typical Sensors, Controllers, and Actuators (e.g., distinctive control system designs - peripherals (mouse, keypad, joystick), game controllers, wearables, interactive surfaces; electronics and bespoke hardware; computer vision, inside-out tracking, and outside-in tracking; IoT-enabled electronics and I/O) Interaction
- eSports Ecosystems (e.g., evolution of gameplay across platforms; games and eSports; game events such as LAN/arcade tournaments and international events such as the Olympic eSports Series; streamed media and spectatorship; multimedia technologies and broadcast management; professional play; data and machine learning for coaching and training)

Learning Outcomes:

Core:

1. Explain how eSports are streamed to large audiences over the internet [Explain]

3.17.14 SPD/Architectures and Rendering (Core Tier2: 1 hr)

Covers CPU and GPU architectures for games and pipelines for graphical rendering and physical simulation.

Topics:

Core

- CPU and GPU Architectures (e.g., Flynn's taxonomy; parallelization; instruction sets; standard components - graphics compute array, graphics memory controller, video graphics array basic input/output system; bus interface; power management unit; video processing unit; display interface) Heterogeneous Architectures
- Pipelines for Graphical Rendering and Physical Simulation (e.g., tile-based, immediate-mode) Applied Rendering and Techniques

Learning Outcomes:

Core:

1. Recall the characteristics of common general-purpose graphics processing architectures [Recall]
2. Identify the key stages of the immediate-mode rendering pipeline [Analyze]

3.17.15 SPD/Algorithms, Media and Virtual Worlds (Core Tier2: 1 hr)

Covers common algorithms, data structures and mathematical functions in games, and media representations in virtual worlds.

Topics:

Core

- Common Contexts for Algorithms, Data Structures, and Mathematical Functions (e.g., game loops; spatial partitioning, viewport culling, and level of detail; collision detection and resolution; physical simulation; behavior for intelligent agents; procedural content generation) Foundational Data Structures, Foundational Algorithms, Advanced Algorithms, Discrete Mathematics
- Media Representations in Virtual Worlds (e.g., I/O and computation techniques for virtual worlds: audio; music; sprites; models and textures; text; dialogue; multimedia (e.g., olfaction, tactile)) Fundamental Concepts of Graphics and Interactive Techniques

Learning Outcomes:

Core:

1. Translate complex mathematical functions into performant source code [Translate]
2. Optimize a function for processing collision detection in a simulated environment [Create]

3.17.16 SPD/Languages, APIs and Development Tools (Core Tier2: 1 hr)

Covers programming and shading languages, graphics APIs and libraries, and development tools and environments for games.

Topics:
Core

- Game Development Tools and Techniques 1. Programming Languages (e.g., C++; C#; Lua; Python; JavaScript) 2. Shader Languages (e.g., HLSL, GLSL; Shader Graph) 3. Graphics Libraries and APIs (e.g., DirectX; SDL; OpenGL; Metal; Vulkan; WebGL). Applied Rendering and Techniques, System Design 4. Common Development Tools and Environments (e.g., IDEs; Debuggers; Profilers; Version Control Systems including those handling binary assets; Development Kits and Production/Consumer Kits; Emulators) Software Development Practices, Development Environment and IDE, Version Control and CI/CD

Learning Outcomes:
Core:

1. Use an industry-standard graphics API to render a 3D model in a virtual scene [Use]
2. Modify a shader to change a visual effect according to stated requirements [Create]
3. Recommend an appropriate set of development tools and techniques for implementing a game of a particular genre for a given platform [Recommend]

3.17.17 SPD/Game Engines (Core Tier2: 1 hr)

Covers open-source game engines and production techniques, quality assurance, optimization, porting and internationalization.

Topics:
Core

- Game Engines 1. Open Game Engines (e.g., Unreal; Unity; Godot; CryEngine; Phyre; Source 2; Pygame and Ren'Py; Phaser; Twine; Spring RTS) 2. Techniques (e.g., Ideation, Prototyping, Iterative Design and Implementation, Compiling Executable Builds, Development Operations and Quality Assurance - Play Testing and Technical Testing, Profiling; Optimization, Porting; Internationalization and Localization, Networking) Performance and Energy Efficiency, Requirements Engineering, Requirements Evolution and Estimation

Learning Outcomes:
Core:

1. Implement a game for a particular platform according to the specification [Implement]
2. Assess a game's run-time and memory performance using an industry-standard tool and development environment [Assess]
3. Discuss the key challenges in making a digital game that is cross-platform compatible [Debate]

3.17.18 SPD/Game Design Foundations (Core Tier2: 1 hr)

Covers game design vocabulary and models, design thinking and user-centred experience design.

Topics:
Core

- Game Design Vocabulary (e.g., game definitions; mechanics-dynamics-aesthetics model; industry terminology; experience design; models of experience and emotion)
- Design Thinking and User-Centered Experience Design (e.g., methods of designing games; iteration, incrementing, and the double-diamond; phases of pre- and post-production; quality assurance, including alpha and beta testing; stakeholder and customer involvement; community management) API Design and Modeling, Principles and Software Architecture, Software Design Quality and Evaluation

Learning Outcomes:
Core:

1. Express how game developers can enhance the accessibility of a game interface [Demonstrate]

3.17.19 SPD/Genres, Audiences and Social Reach (Core Tier2: 1 hr)

Covers game genres, audiences and player taxonomies, and the proliferation of game technologies to domains beyond entertainment.

Topics:
Core
- Game Genres (e.g., adventure; walking simulator; first-person shooter; real-time strategy; multiplayer online battle arena (MOBA); role-playing game (rpg))
- Audiences and Player Taxonomies (e.g., people who play games; diversity and broadening participation; pleasures, player types, and preferences; Bartle, Yee) Understanding the User: Individual goals and interactions with others
- Proliferation of Digital Game Technologies to Domains Beyond Entertainment (e.g., Education and Training; Serious Games; Virtual Production; eSports; Gamification; Immersive Experience Design; Creative Industry Practice; Artistic Practice; Procedural Rhetoric) AI Applications and Societal Impact, Additional AI Applications and Societal Impact

Learning Outcomes:
Core:
1. Create novel forms of gameplay using frontier game platforms [Create]

3.17.20 SPD/Interactive Data and Analytics Platforms (Core Tier2: 2 hrs)

Covers interactive data analysis platforms such as Jupyter notebooks, Google Colab, R, cloud SQL platforms, and design tools with low-latency feedback loops.
Topics:
Non Core
- Computational Notebook Platforms (e.g., Jupyter notebooks; Google Colab; R; SPSS; Observable)
- Cloud SQL and Data Analysis Platforms (e.g., BigQuery) Query Construction 1. Apache Spark 2. Data Visualizations Visualization
- Interactive Presentations Backed by Data
- Design Tools Requiring Low-Latency Feedback Loops 1. Rendering tools 2. Graphic design tools

Learning Outcomes:
NonCore:
1. Analyze large datasets interactively [Analyze]
2. Create a backing track for a musical performance, such as live coding [Create]
3. Create compelling computational notebooks that construct a narrative for a given journalistic goal/story [Create]
4. Implement interactive code that uses a dataset and generates exploratory graphics [Implement]
5. Contextualize the attributes of different data analysis styles, such as interactive vs engineered pipeline [Contextualize]
6. Write a program using a notebook computing platform (e.g., searching, sorting, or graph manipulation) [Write]

3.17.21 SPD/AI Programming and Quantum Platforms (Core Tier2: 1 hr)

Covers prompt programming with large language models (LLMs) and generative AI, and quantum platforms for programming quantum logic operators and available quantum services.
Topics:
Non Core
- Prompt programming 1. Generative AI (e.g., OpenAI's ChatGPT, OpenAI's Codex, GitHub's Copilot) and LLMs are accessed/interacted
- Quantum Platforms Quantum Architectures 1. Program quantum logic operators in quantum machines. 2. Use API for available quantum services 3. Signal analysis/Fourier analysis/Signal processing (for music composition, audio/RF analysis) Image Processing

Learning Outcomes:
NonCore:
1. Create a program that performs a task using LLM systems [Create]
2. Contrast a program developed by an AI platform and by a human [Contrast]
3. Implement a system that interacts with a human without using a screen [Implement]

4. Demonstrate a quantum gate outcome using a quantum platform [Demonstrate]

3.17.22 SPD/SPD-SEP/Mobile

Topics:
Non Core
- Privacy and data protection
- Accessibility in mobile design
- Security and cybersecurity
- Social impacts of mobile technology
- Ethical use of AI and algorithms

Learning Outcomes:
NonCore:
1. Understand and uphold ethical responsibilities for safeguarding user privacy and data protection in mobile applications [Assess]
2. Design mobile applications with accessibility in mind, ensuring effective use by people with disabilities [Design]
3. Demonstrate proficiency in secure coding practices to mitigate risks associated with various security threats in mobile development [Demonstrate]
4. Analyze the broader social impacts of mobile technology, including its influence on communication patterns, relationships, and mental health [Analyze]
5. Comprehend the ethical considerations of using AI in mobile applications, ensuring unbiased and fair algorithms [Explain]

3.17.23 SPD/SPD-SEP/Web

Topics:
Non Core
- Privacy concerns with mobile apps
- Designing for inclusivity and accessibility
- Ethical use of AI in mobile apps
- Sustainable app development and server hosting
- Avoiding spam or intrusive notifications
- Addressing cyberbullying and harassment
- Promoting positive online communities
- Monetization and advertising
- Ethical use of gamification

Learning Outcomes:
NonCore:
1. Understand how mobile computing impacts communications and the flow of information within society [Analyze]
2. Design mobile apps that have made daily tasks easier/faster [Design]
3. Recognize how the ubiquity of mobile computing has affected work-life balance [Recognize]
4. Understand how mobile computing impacts health monitoring and healthcare services [Analyze]
5. Define how mobile apps are used to educate about and help achieve UN sustainability goals [Define]

3.17.24 SPD/Intellectual Property in Games

Covers intellectual property rights in the creative industries, including copyright, trademarks, licensing and enforcement.
Topics:
Non Core
- Intellectual Property Rights in Creative Industries 1. Intellectual Property Ownership: copyright, trademark; design right, patent, trade secret, civil versus criminal law; international

agreements; procedural content generation and the implications of generative artificial intelligence 2. Licensing: Usage and fair usage exceptions; open-source license agreements; proprietary and bespoke licensing; enforcement

Learning Outcomes:

NonCore:

1. Discuss how creators can protect their intellectual property [Debate]
2. Compare options for open-source licensing of a game development tool [Compare]

3.17.25 SPD/Game Interface and Accessibility

Covers the usability and accessibility of game interfaces, including heuristic evaluation methods and design for players with disabilities.

Topics:

Non Core

- Fair Access to Play 1. Game Interface Usability: user requirements, affordances, ergonomic design, user research, experience measurement, and heuristic evaluation methods for games 2. Game Interface Accessibility: forms of impairment and disability; means to facilitate game access; universal design; legislated requirements for game platforms; compliance evaluation; challenging game mechanics and access

Learning Outcomes:

NonCore:

1. Identify common pitfalls in game interfaces that exclude players with impaired or non-functional vision [Analyze]
2. Describe how heuristic evaluation can be used to identify usability problems in game interfaces [Describe]
3. Recommend changes to a specific game interface to improve access to players who are deaf or whose hearing is otherwise impaired [Recommend]

3.17.26 SPD/Health, Safety and Sustainability

Covers game-related injuries and their mitigation, risk assessment for events and manufacturing, player mental health, and the supply chain and sustainability of platform hardware.

Topics:

Non Core

- Game-Related Health and Safety 1. Injuries in Play: ways of mitigating common upper body injuries, such as repetitive strain injury; exercise psychology and physiotherapy in eSports 2. Risk Assessment for Events and Manufacturing: control of substances hazardous to health (COSHH); fire safety; electrical and electronics safety; risk assessment for games and game events; risk assessment for manufacturing 3. Mental Health: motivation to play; gamification and gameful design; game psychology - internet gaming disorder
- Platform Hardware Supply Chain and Sustainability 1. Platform Lifecycle: platform composition - materials, assembly; mineral excavation and processing; power usage; recycling; planned obsolescence. 2. Modern Slavery: supply chains; forced labor and civil rights; working conditions; detection and remission; certification bodies and charitable endeavors.

Learning Outcomes:

NonCore:

1. Explain why upper body injuries are common in eSports [Explain]
2. Modify a policy for a LAN party event to include mitigations that lower the risk of fire [Design]
3. Design a gamification strategy to motivate serious play for an awareness-raising game [Design]
4. Assess the appropriateness of two manufacturers for producing a new game console [Assess]
5. Discuss whether games are addictive [Debate]

3.17.27 SPD/Representation, Inclusion and Equality

Covers representation in games media and industry, inclusion of diverse audiences, equality and social mobility in the industry, and the use of games to advocate for social causes.

Topics:
Non Core
- Representation in the Media and Industry 1. Inclusion: identity and identification; inclusion of a broad range of characters for diverse audiences; media representation and its effects; media literacy; content analysis; stereotyping; sexualization 2. Equality: histories and controversies, such as gamergate, quality of life in the industry, professional discourse and conduct in business contexts, pathways to game development careers, social mobility, the experience of developers from different backgrounds and identities, gender, and technology

Learning Outcomes:
NonCore:
1. Discuss how to reform characters and dialogues in a scene to reduce stereotype threat [Debate]
2. Illustrate how the portrayal of race in a game can influence the risk of social exclusion in the associated online community around the game [Apply]
3. Analyze the role of company hiring policies and advocacy on social mobility [Analyze]
4. Suggest how the portrayal of women in video games influences how players perceive members of those groups [Evaluate]
5. Create a video game that successfully advocates for climate science [Create]

3.17.28 SPD/SPD-SEP/Robotics

Topics:
Non Core
- Fairness, transparency, and accountability in robotic algorithms
- Mitigating biases in robot decision-making
- Public safety in shared spaces with robots
- Compliance with data protection laws
- Patient consent and trust in medical robots

Learning Outcomes:
NonCore:
1. Identify instances of bias in robotic algorithms and propose strategies to mitigate them [Analyze]
2. Evaluate and critique robotic systems for ethical and fairness considerations, suggesting improvements where necessary [Evaluate]
3. Analyze real-world examples of biases in robot decision-making and develop strategies to reduce bias in robotic systems [Analyze]
4. Assess the potential risks associated with robots in public spaces and propose safety measures to mitigate those risks [Assess]
5. Evaluate the impact of patient consent and trust on the effectiveness of medical robot deployments in healthcare contexts [Evaluate]

3.17.29 SPD/SPD-SEP/Interactive

Topics:
Non Core
- Ethical guidelines when using AI models to assist in journalism and content creation
- Accountability for AI-generated outputs
- Behavior among prompt programmers and AI developers
- Trust with the public when using AI models

Learning Outcomes:
NonCore:
1. Indicate a framework for accountability in AI model deployment, including clear documentation and attribution [Evaluate]
2. Discuss ethical codes of conduct and professional standards relevant to prompt programming and AI development [Debate]
3. Create communication plans and materials to educate the public about AI capabilities, limitations, and ethical safeguards [Create]

Chapter 4

Body of knowledge of Mathematics

The 12 knowledge areas in Mathematics are:

4.1 Calculus and Analysis (CAN) (Pg.. 168)
- 4.1.1 Limits and Continuity (Pg.. 168)
- 4.1.2 Differential Calculus (Pg.. 169)
- 4.1.3 Antiderivatives and Integration Techniques (Pg.. 169)
- 4.1.4 Improper Integrals (Pg.. 169)
- 4.1.5 Sequences, Series, and Power Series (Pg.. 169)
- 4.1.6 Applications of the Definite Integral (Pg.. 170)
- 4.1.7 Vectors and the Geometry of Space (Pg.. 170)
- 4.1.8 Quadric Surfaces (Pg.. 171)
- 4.1.9 Parametric Equations and Polar Coordinates (Pg.. 171)
- 4.1.10 Multivariable Differential Calculus (Pg.. 171)
- 4.1.11 Multiple Integrals (Pg.. 172)
- 4.1.12 Line Integrals and Vector Fields (Pg.. 172)
- 4.1.13 Surface Integrals and Fundamental Theorems (Pg.. 172)
- 4.1.14 Vector Functions of a Real Variable (Pg.. 173)
- 4.1.15 Multivariable Transformations (Pg.. 173)
- 4.1.16 Real Analysis and Lebesgue Integration (Pg.. 173)
- 4.1.17 Complex Analysis and Conformal Mapping (Pg.. 174)
- 4.1.18 Functional Analysis and Banach/Hilbert Spaces (Pg.. 174)
- 4.1.19 Harmonic Analysis and Fourier Theory (Pg.. 174)
- 4.1.20 Calculus of Variations (Pg.. 175)
- 4.1.21 Measure Theory (Pg.. 175)

4.2 Algebra and Number Theory (ANT) (Pg.. 175)
- 4.2.1 Matrices and Systems of Linear Equations (Pg.. 176)
- 4.2.2 Vector Spaces and Linear Transformations (Pg.. 176)
- 4.2.3 Eigenvalues, Inner Product Spaces, and Quadratic Forms (Pg.. 177)
- 4.2.4 Group Theory (Pg.. 177)
- 4.2.5 Rings, Fields, and Galois Theory (Pg.. 177)
- 4.2.6 Module Theory and Commutative Algebra (Pg.. 178)
- 4.2.7 Analytic Number Theory (Pg.. 178)
- 4.2.8 Algebraic Number Theory (Pg.. 178)
- 4.2.9 Representation Theory (Pg.. 179)
- 4.2.10 Cryptography and Coding Theory (Pg.. 179)

4.3 Geometry and Topology (GTO) (Pg.. 179)
- 4.3.1 Euclidean and Projective Geometry (Pg.. 180)
- 4.3.2 General Topology (Pg.. 180)
- 4.3.3 Differential Geometry of Curves and Surfaces (Pg.. 181)
- 4.3.4 Smooth Manifolds and Differential Forms (Pg.. 181)
- 4.3.5 Riemannian Geometry (Pg.. 181)
- 4.3.6 Algebraic Topology (Pg.. 182)
- 4.3.7 Symplectic Geometry (Pg.. 182)
- 4.3.8 Geometric Group Theory (Pg.. 182)

4.4 Discrete Mathematics and Combinatorics (DMC) (Pg.. 183)
- 4.4.1 Enumerative Combinatorics (Pg.. 183)
- 4.4.2 Graph Theory (Pg.. 183)
- 4.4.3 Extremal and Probabilistic Combinatorics (Pg.. 184)
- 4.4.4 Discrete Optimization (Pg.. 184)
- 4.4.5 Design Theory and Combinatorial Codes (Pg.. 184)
- 4.4.6 Mathematical Logic and Set Theory (Pg.. 185)

4.5 Probability and Statistics (PST) (Pg.. 185)
- 4.5.1 Probability Axioms and Distributions (Pg.. 185)
- 4.5.2 Limit Theorems (Pg.. 186)
- 4.5.3 Stochastic Processes (Pg.. 186)
- 4.5.4 Estimation Theory (Pg.. 186)
- 4.5.5 Hypothesis Testing (Pg.. 187)
- 4.5.6 Bayesian Inference (Pg.. 187)
- 4.5.7 High-Dimensional Statistics (Pg.. 187)
- 4.5.8 Time Series Analysis (Pg.. 188)
- 4.5.9 Nonparametric Methods (Pg.. 188)

4.6 Numerical and Scientific Analysis (NSA) (Pg.. 188)
- 4.6.1 Error Analysis and Floating-Point Arithmetic (Pg.. 189)
- 4.6.2 Root-Finding for Nonlinear Equations (Pg.. 189)
- 4.6.3 Numerical Linear Algebra (Pg.. 190)
- 4.6.4 Approximation Theory and Interpolation (Pg.. 190)
- 4.6.5 Numerical Integration and Quadrature (Pg.. 190)
- 4.6.6 Numerical Methods for Differential Equations (Pg.. 191)
- 4.6.7 Finite Element Method (Pg.. 191)
- 4.6.8 Optimization Algorithms (Pg.. 191)
- 4.6.9 Parallel and High-Performance Computing (Pg.. 192)

4.7 Mathematical Modeling and Simulation (MMS) (Pg.. 192)
- 4.7.1 First-Order Ordinary Differential Equations (Pg.. 193)
- 4.7.2 Higher-Order Differential Equations and Systems (Pg.. 193)
- 4.7.3 Partial Differential Equations (Pg.. 193)
- 4.7.4 Dynamical Systems and Chaos (Pg.. 194)
- 4.7.5 Asymptotic Analysis and Perturbation Methods (Pg.. 194)
- 4.7.6 Mathematical Fluid Dynamics (Pg.. 194)
- 4.7.7 Mathematical Control Theory (Pg.. 195)
- 4.7.8 Stochastic Differential Equations (Pg.. 195)
- 4.7.9 Agent-Based and Complex Systems Modeling (Pg.. 195)

4.8 Mathematical Foundations of Data Science (FDS) (Pg.. 196)
- 4.8.1 Statistical Learning Theory (Pg.. 196)
- 4.8.2 Optimization for Machine Learning (Pg.. 196)
- 4.8.3 Topological Data Analysis (Pg.. 197)
- 4.8.4 Dimensionality Reduction (Pg.. 197)
- 4.8.5 Kernel Methods and Reproducing Kernel Hilbert Spaces (Pg.. 198)
- 4.8.6 Information Theory (Pg.. 198)
- 4.8.7 Mathematical Foundations of Deep Learning (Pg.. 198)

4.9 Computational Geometry and Topology (CGT) (Pg.. 199)
- 4.9.1 Voronoi Diagrams and Delaunay Triangulations (Pg.. 199)
- 4.9.2 Convex Hulls and Polytopes (Pg.. 199)

- 4.9.3 Digital Topology and Image Analysis (Pg.. 200)
- 4.9.4 Persistent Homology and Applied Topology (Pg.. 200)
- 4.9.5 Mesh Generation and Geometric Algorithms (Pg.. 201)

4.10 Quantum Information and Computing (QIC) (Pg.. 201)

- 4.10.1 Quantum Mechanics Foundations for Computing (Pg.. 201)
- 4.10.2 Quantum Circuits and Gates (Pg.. 202)
- 4.10.3 Quantum Algorithms (Pg.. 202)
- 4.10.4 Entanglement and Bell Inequalities (Pg.. 202)
- 4.10.5 Quantum Error Correction (Pg.. 203)
- 4.10.6 Tensor Networks and Quantum Simulation (Pg.. 203)

4.11 Mathematical Biology and Life Sciences (MBL) (Pg.. 204)

- 4.11.1 Population Dynamics (Pg.. 204)
- 4.11.2 Mathematical Epidemiology (Pg.. 204)
- 4.11.3 Systems Biology and Network Dynamics (Pg.. 205)
- 4.11.4 Bioinformatics Algorithms (Pg.. 205)
- 4.11.5 Mathematical Neuroscience (Pg.. 205)

4.12 Financial and Actuarial Mathematics (FAM) (Pg.. 206)

- 4.12.1 Interest Theory and Time Value of Money (Pg.. 206)
- 4.12.2 Arbitrage-Free Pricing and Market Models (Pg.. 207)
- 4.12.3 Black-Scholes and Option Pricing (Pg.. 207)
- 4.12.4 Risk Measures and Portfolio Theory (Pg.. 207)
- 4.12.5 Actuarial Models and Life Tables (Pg.. 208)
- 4.12.6 Computational Finance and Monte Carlo Methods (Pg.. 208)

4.1 Calculus and Analysis (CAN)

This area covers the fundamental study of change, limits, and the rigorous properties of functions. It spans from basic differentiation and integration to advanced measure theory and functional spaces, providing the essential toolkit for all mathematical modeling.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.1.1 Limits and Continuity (Pg.. 168)	1	1
⭐ 4.1.2 Differential Calculus (Pg.. 169)	1	1
⭐ 4.1.3 Antiderivatives and Integration Techniques (Pg.. 169)	1	1
⭐ 4.1.4 Improper Integrals (Pg.. 169)	1	1
⭐ 4.1.5 Sequences, Series, and Power Series (Pg.. 169)	1	1
⭐ 4.1.6 Applications of the Definite Integral (Pg.. 170)	1	1
⭐ 4.1.7 Vectors and the Geometry of Space (Pg.. 170)	1	1
⭐ 4.1.8 Quadric Surfaces (Pg.. 171)	1	1
⭐ 4.1.9 Parametric Equations and Polar Coordinates (Pg.. 171)	1	1
⭐ 4.1.10 Multivariable Differential Calculus (Pg.. 171)	1	1
⭐ 4.1.11 Multiple Integrals (Pg.. 172)	1	1
⭐ 4.1.12 Line Integrals and Vector Fields (Pg.. 172)	1	1
⭐ 4.1.13 Surface Integrals and Fundamental Theorems (Pg.. 172)	1	1
⭐ 4.1.14 Vector Functions of a Real Variable (Pg.. 173)	1	1
⭐ 4.1.15 Multivariable Transformations (Pg.. 173)	1	1
⭐ 4.1.16 Real Analysis and Lebesgue Integration (Pg.. 173)	1	1
⭐ 4.1.17 Complex Analysis and Conformal Mapping (Pg.. 174)	1	1
4.1.18 Functional Analysis and Banach/Hilbert Spaces (Pg.. 174)	Elective	
4.1.19 Harmonic Analysis and Fourier Theory (Pg.. 174)	Elective	
4.1.20 Calculus of Variations (Pg.. 175)	Elective	
⭐ 4.1.21 Measure Theory (Pg.. 175)	1	1

Table 4.1: List of KUs in the Calculus and Analysis area.

4.1.1 CAN/Limits and Continuity (Core Tier1: 1 hr, Core Tier2: 1 hr)

Rigorous study of limits and continuity of single-variable functions, including the epsilon-delta definition, limit laws, and fundamental theorems.

Topics:

Core

- Epsilon-delta definition of a limit and formal proof techniques
- Limit laws, one-sided limits, and limits at infinity
- Continuity: types, properties, and the Intermediate Value Theorem
- L'Hôpital's rule and indeterminate forms
- Limits of sequences: convergence, Squeeze Theorem, and Bolzano-Weierstrass

Learning Outcomes:
Core:
1. State the epsilon-delta definition of a limit and explain its geometric meaning [Familiarity]
2. Evaluate limits using limit laws, L'Hôpital's rule, and asymptotic analysis [Usage]
3. Prove continuity or discontinuity of a function and apply the Intermediate Value Theorem [Assessment]

4.1.2 CAN/Differential Calculus (Core Tier1: 1 hr, Core Tier2: 1 hr)

Derivatives of single-variable functions: definition, rules, higher-order derivatives, and applications to optimization and curve analysis.
Topics:
Core
- Derivative as a limit; geometric and physical interpretations; differentiability
- Product, quotient, and chain rules; implicit differentiation; related rates
- Rolle's theorem, Mean Value Theorem, and monotonicity criteria
- Higher-order derivatives, Taylor polynomials, and remainder estimation
- Extrema: first and second derivative tests, curve sketching, and applied optimization

Learning Outcomes:
Core:
1. Interpret the derivative as a rate of change and as the slope of the tangent line [Familiarity]
2. Apply differentiation rules and implicit differentiation to compute derivatives of composite and implicit functions [Usage]
3. Solve optimization problems and sketch curves using the Mean Value Theorem and derivative tests [Assessment]

4.1.3 CAN/Antiderivatives and Integration Techniques (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- The Fundamental Theorem of Calculus and the antiderivative
- Integration techniques: substitution, parts, partial fractions, and trigonometric substitution

Learning Outcomes:
Core:
1. State the Fundamental Theorem of Calculus and explain the relationship between differentiation and integration [Familiarity]
2. Compute definite and indefinite integrals using standard techniques including substitution and integration by parts [Usage]

4.1.4 CAN/Improper Integrals (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Improper integrals: convergence criteria and comparison tests

Learning Outcomes:
Core:
1. Determine convergence of improper integrals using comparison and other convergence tests [Assessment]

4.1.5 CAN/Sequences, Series, and Power Series (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topics:
Core
- Sequences and series: convergence tests (ratio, root, comparison, integral)

- Power series, radius of convergence, Taylor and Maclaurin series

Learning Outcomes:

Core:

1. Determine convergence of power series and construct Taylor series for standard functions [Assessment]

4.1.6 CAN/Applications of the Definite Integral (Core Tier1: 1 hr, Core Tier2: 1 hr)

Geometric and physical applications of the definite integral: areas, volumes of revolution, arc length, surface area, work, and centroids.

Topics:

Core

- Area between curves by integrating with respect to x or y
- Volume of solids of revolution by the disk and washer method
- Volume of solids of revolution by the cylindrical shell method
- Arc length of a plane curve
- Surface area of solids of revolution
- Work via integration (springs, pumping fluids, cables)

Elective

- Moments and centroid of a plane region or solid
- Hydrostatic force and pressure via integration

Learning Outcomes:

Core:

1. Calculate the area between two curves by integrating with respect to x or y, as appropriate [Usage]
2. Calculate the volume of a solid of revolution using the disk or washer method [Usage]
3. Calculate the volume of a solid of revolution using the cylindrical shell method [Usage]
4. Calculate the arc length of a curve given by $y = f(x)$ [Usage]
5. Calculate the work done by a variable force using integration [Usage]

Elective:

6. Calculate the centroid of a plane region using moments and integration [Usage]
7. Calculate the hydrostatic force on a submerged surface using integration [Assessment]

4.1.7 CAN/Vectors and the Geometry of Space (Core Tier1: 1 hr, Core Tier2: 1 hr)

Vectors in three-dimensional space, the dot and cross products, and their application to lines and planes in $\mathbb{R}^3$.

Topics:

Core

- Vectors in $\mathbb{R}^3$: operations, magnitude, and unit vectors
- Dot (scalar) product: definition, properties, and angle between vectors
- Cross (vector) product: definition, properties, and geometric interpretation
- Scalar triple product and its interpretation as volume
- Vector, parametric, and symmetric equations of a line in space
- Equation of a plane: normal vector, general form, and point-to-plane distance

Learning Outcomes:

Core:

1. Compute the dot and cross products of vectors in $\mathbb{R}^3$ and interpret the results geometrically [Usage]
2. Determine the vector, parametric, and symmetric equations of a line given a point and a direction vector [Usage]
3. Determine the equation of a plane given a point and a normal vector, or three non-collinear points [Usage]
4. Calculate distances between points, lines, and planes in space using vector products [Assessment]

4.1.8 CAN/Quadric Surfaces (Core Tier1: 1 hr, Core Tier2: 1 hr)

Classification and graphing of quadric surfaces in $\mathbb{R}^3$ from their general equation.

Topics:

Core

- Cylindrical surfaces and traces in the coordinate planes
- Ellipsoids and spheres: canonical equation and traces
- Elliptic and hyperbolic paraboloids
- Hyperboloids of one and two sheets
- Elliptic cones

Elective

- Cylindrical and spherical coordinates for describing surfaces

Learning Outcomes:

Core:

1. Classify a quadric surface from its general equation using the trace method [Usage]
2. Sketch a quadric surface by identifying its traces in the coordinate planes [Usage]

Elective:

3. Convert points and equations among Cartesian, cylindrical, and spherical coordinates [Usage]

4.1.9 CAN/Parametric Equations and Polar Coordinates (Core Tier1: 1 hr, Core Tier2: 1 hr)

Parametric curves and their differential and integral calculus, and polar coordinates with their associated area and arc length calculations.

Topics:

Core

- Curves defined parametrically: $x = f(t)$, $y = g(t)$
- Derivatives and tangent lines to parametric curves; parametric arc length
- Polar coordinate system and conversion with Cartesian coordinates
- Graphing polar curves (cardioids, roses, spirals, circles)
- Area and arc length in polar coordinates

Elective

- Conic sections in polar coordinates (eccentricity and directrix)

Learning Outcomes:

Core:

1. Compute the derivative dy/dx and find tangent lines to a parametric curve [Usage]
2. Calculate the arc length of a parametric curve [Usage]
3. Convert points and equations between polar and Cartesian coordinates [Usage]
4. Graph common polar curves, identifying symmetries and special values [Usage]
5. Calculate the area of a region bounded by a polar curve [Assessment]

Elective:

6. Identify and classify conic sections given in polar form [Familiarity]

4.1.10 CAN/Multivariable Differential Calculus (Core Tier1: 1 hr, Core Tier2: 1 hr)

Differential calculus of scalar functions of several variables: limits, partial derivatives, gradient, tangent plane, optimization, and Taylor's theorem.

Topics:

Core

- Limits and continuity of functions of several variables in $\mathbb{R}^n$
- Partial derivatives, directional derivatives, and the gradient vector
- Tangent plane, total differential, and linear approximation
- Extrema of multivariable functions and Lagrange multipliers
- Taylor's theorem for functions of several variables

Learning Outcomes:
Core:
1. Evaluate limits and determine continuity of functions of several variables [Familiarity]
2. Compute gradients, directional derivatives, and tangent planes for multivariable functions [Usage]
3. Apply Lagrange multipliers to solve constrained optimization problems [Usage]
4. Expand functions of several variables using Taylor's theorem and estimate error bounds [Assessment]

4.1.11 CAN/Multiple Integrals (Core Tier1: 1 hr, Core Tier2: 1 hr)

Integration of functions of several variables: double and triple integrals in Cartesian, polar, cylindrical, and spherical coordinates, with applications.
Topics:
Core
- Double integrals over rectangles and general regions; iterated integrals (Fubini's theorem)
- Double integrals in polar coordinates and change-of-variables formula
- Triple integrals in Cartesian coordinates and applications to volume and mass
- Triple integrals in cylindrical and spherical coordinates
- Applications: area, volume, centers of mass, and moments of inertia

Learning Outcomes:
Core:
1. Evaluate double and triple integrals over general regions using iterated integrals [Usage]
2. Apply coordinate changes in polar, cylindrical, and spherical systems to simplify multiple integrals [Usage]
3. Compute areas, volumes, and centers of mass using multiple integrals [Assessment]

4.1.12 CAN/Line Integrals and Vector Fields (Core Tier1: 1 hr, Core Tier2: 1 hr)

Scalar and vector line integrals, conservative fields, potential functions, path independence, and Green's Theorem.
Topics:
Core
- Vector fields: definition, visualization, divergence, and curl
- Line integrals of scalar functions: arc length and mass
- Line integrals of vector fields: work and circulation
- Conservative fields, potential functions, and path independence
- Green's Theorem and its applications in the plane

Learning Outcomes:
Core:
1. Compute scalar and vector line integrals over oriented curves [Usage]
2. Determine whether a vector field is conservative and construct its potential function [Usage]
3. Apply Green's Theorem to evaluate line integrals via double integrals [Assessment]

4.1.13 CAN/Surface Integrals and Fundamental Theorems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Parametric surfaces, scalar and flux surface integrals, curl, divergence, and the fundamental theorems of Stokes and Gauss.
Topics:
Core
- Parametric surfaces, tangent plane, and surface area
- Surface integrals of scalar functions
- Flux integrals: surface integrals of vector fields
- Stokes' Theorem: relation between line integrals and surface integrals
- Divergence Theorem (Gauss): relation between flux and volume integral

Learning Outcomes:
Core:
1. Parametrize surfaces and evaluate scalar and flux surface integrals [Usage]
2. Apply Stokes' and Divergence Theorems to reduce complex integrals [Assessment]
3. Relate local differential properties (curl, divergence) to global behavior via integral theorems [Assessment]

4.1.14 CAN/Vector Functions of a Real Variable (Core Tier1: 1 hr, Core Tier2: 1 hr)

Space curves as vector functions of a real variable, including differential geometry of curves: Frenet frame, curvature, and torsion.
Topics:
Core
- Definition, domain, and range of vector functions of a real variable
- Limits, continuity, and derivatives of vector functions
- Arc length and natural parametrization of a curve
- Tangent, normal, and binormal vectors; moving Frenet frame
- Curvature, torsion, and Frenet-Serret formulas

Learning Outcomes:
Core:
1. Identify limits, continuity, and derivatives of vector functions of a real variable [Familiarity]
2. Compute tangent, normal, and binormal vectors and reparametrize a curve by arc length [Usage]
3. Calculate curvature and torsion using the Frenet-Serret formulas [Assessment]

4.1.15 CAN/Multivariable Transformations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Vector functions of vector variables: differentiability, the Jacobian matrix, change of variables, and the inverse and implicit function theorems.
Topics:
Core
- Transformations from $\mathbb{R}^n$ to $\mathbb{R}^m$ and their differentiability
- Jacobian matrix and Jacobian determinant
- Change of variables in polar, cylindrical, and spherical coordinates
- Inverse function theorem
- Implicit function theorem in multivariable contexts

Learning Outcomes:
Core:
1. Describe transformations between coordinate systems using the Jacobian matrix [Familiarity]
2. Compute the Jacobian and apply change of variables to evaluate multiple integrals [Usage]
3. Apply the inverse and implicit function theorems to analyze local invertibility [Assessment]

4.1.16 CAN/Real Analysis and Lebesgue Integration (Core Tier1: 1 hr, Core Tier2: 1 hr)

Rigorous treatment of metric spaces, measure theory, and the Lebesgue integral, including Lp spaces and convergence theorems.
Topics:
Core
- Metric spaces: completeness, compactness, and the Baire category theorem
- Lebesgue measure on $\mathbb{R}^n$: measurable sets and null sets
- Convergence theorems: Monotone Convergence, Dominated Convergence, and Fatou's Lemma
- L^p spaces: definition, Hölder and Minkowski inequalities, and duality
- Uniform convergence and interchange of limit, differentiation, and integration

Learning Outcomes:
Core:

1. Contrast the Lebesgue integral with the Riemann integral and identify their relative scope [Familiarity]
2. Prove basic properties of measurable functions using the convergence theorems [Usage]
3. Verify completeness and compactness criteria in metric and L^p spaces [Assessment]

4.1.17 CAN/Complex Analysis and Conformal Mapping (Core Tier1: 1 hr, Core Tier2: 1 hr)

Functions of a complex variable: analyticity, contour integration, Laurent series, residues, and conformal mappings.

Topics:

Core

- Holomorphic functions, Cauchy-Riemann equations, and harmonic functions
- Cauchy's integral theorem and formula; contour integration techniques
- Laurent series and classification of isolated singularities (removable, pole, essential)
- Residue theorem and evaluation of real integrals

Elective

- Conformal mappings, Möbius transformations, and the Riemann mapping theorem

Learning Outcomes:

Core:

1. Determine analyticity of complex functions using the Cauchy-Riemann equations [Familiarity]
2. Evaluate real integrals and sum series using the Residue Theorem [Usage]

Elective:

3. Construct conformal maps between standard domains using Möbius transformations [Assessment]

4.1.18 CAN/Functional Analysis and Banach/Hilbert Spaces

Infinite-dimensional normed spaces, bounded linear operators, and spectral theory, with applications to PDEs and quantum mechanics.

Topics:

Elective

- Normed vector spaces, Banach spaces, and the Hahn-Banach theorem
- Hilbert spaces, inner products, orthonormal bases, and the Riesz Representation Theorem
- Bounded and compact linear operators; dual spaces
- Open mapping theorem, closed graph theorem, and uniform boundedness principle
- Spectral theory of compact self-adjoint operators and the Fredholm alternative

Learning Outcomes:

Elective:

1. Explain completeness in Banach and Hilbert spaces and give canonical examples of each [Familiarity]
2. Apply the Riesz Representation Theorem to identify dual spaces of Hilbert spaces [Usage]
3. Analyze the spectrum of a compact operator and characterize its eigenvalue structure [Assessment]

4.1.19 CAN/Harmonic Analysis and Fourier Theory

Fourier series and transforms, L^2 theory, distributions, and wavelets, with applications to signal processing and PDEs.

Topics:

Elective

- Fourier series: convergence, Parseval's theorem, and L^2 completeness
- The Fourier transform on L^1 and L^2: properties, inversion, and convolution
- Plancherel theorem and tempered distributions (Schwartz space)
- Wavelet theory: multiresolution analysis and the discrete wavelet transform
- Applications to signal processing, PDE solving, and data compression

Learning Outcomes:
Elective:

1. Decompose periodic functions into Fourier series and assess pointwise and L^2 convergence [Familiarity]
2. Apply the Fourier transform to solve linear PDEs and compute convolutions [Usage]
3. Analyze the decay rate of Fourier coefficients in relation to the smoothness of a function [Assessment]

4.1.20 CAN/Calculus of Variations

Optimization of functionals: Euler-Lagrange equations, second-order conditions, isoperimetric constraints, and direct methods.
Topics:
Elective

- Euler-Lagrange equations and classical extremal problems (brachistochrone, geodesics)
- Isoperimetric constraints and Lagrange multipliers in function spaces
- Second variation and sufficient conditions for local minima
- Weak formulations and Sobolev spaces in variational problems
- Direct methods in the calculus of variations: existence via coercivity and lower semicontinuity

Learning Outcomes:
Elective:

1. State the Euler-Lagrange equation and derive it for standard functionals [Familiarity]
2. Derive equations of motion for mechanical systems using the principle of least action [Usage]
3. Solve classical variational problems (brachistochrone, isoperimetric) using Euler-Lagrange theory [Assessment]

4.1.21 CAN/Measure Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Abstract measure and integration theory: sigma-algebras, abstract measures, product spaces, and the Radon-Nikodym theorem.
Topics:
Core

- Sigma-algebras, measurable spaces, and Borel sets
- Abstract measures, outer measures, and the Carathéodory extension theorem
- Product measures and the Fubini-Tonelli theorem
- Radon-Nikodym theorem, absolute continuity, and decomposition of measures
- Signed measures, the Hahn decomposition, and the Jordan decomposition

Learning Outcomes:
Core:

1. Define sigma-algebras and state the conditions for a set function to be a measure [Familiarity]
2. Apply the Fubini-Tonelli theorem to evaluate iterated integrals on product spaces [Usage]
3. Prove the Radon-Nikodym theorem and apply it to derive chain rules for densities [Assessment]

4.2 Algebra and Number Theory (ANT)

This area covers the algebraic structures underlying modern mathematics and their applications, from the foundations of linear algebra and group theory to the arithmetic of number fields, algebraic coding, and cryptography.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.2.1 Matrices and Systems of Linear Equations (Pg.. 176)	1	1
⭐ 4.2.2 Vector Spaces and Linear Transformations (Pg.. 176)	1	1
⭐ 4.2.3 Eigenvalues, Inner Product Spaces, and Quadratic Forms (Pg.. 177)	1	1
⭐ 4.2.4 Group Theory (Pg.. 177)	1	1
⭐ 4.2.5 Rings, Fields, and Galois Theory (Pg.. 177)	1	1
4.2.6 Module Theory and Commutative Algebra (Pg.. 178)	Elective	
4.2.7 Analytic Number Theory (Pg.. 178)	Elective	
4.2.8 Algebraic Number Theory (Pg.. 178)	Elective	
4.2.9 Representation Theory (Pg.. 179)	Elective	
⭐ 4.2.10 Cryptography and Coding Theory (Pg.. 179)	1	1

Table 4.2: List of KUs in the Algebra and Number Theory area.

4.2.1 ANT/Matrices and Systems of Linear Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Matrix algebra, systems of linear equations and their solution methods, determinants, and matrix decompositions.

Topics:

Core

- Matrix algebra: addition, scalar multiplication, matrix multiplication, transpose, types of matrices, and the inverse of a matrix
- Solving systems of linear equations via Gaussian elimination and Gauss-Jordan; row echelon form, rank, and consistency of systems
- Determinants: definition, properties, Cramer's rule, and geometric interpretation
- Matrix decompositions: LU, QR, and singular value decomposition (SVD)

Learning Outcomes:

Core:

1. Identify types of matrices and perform basic matrix algebra operations, including computing the inverse of a matrix [Familiarity]
2. Solve systems of linear equations using Gaussian elimination or Gauss-Jordan, determining the rank and consistency of the system [Usage]
3. Apply Cramer's rule and the LU, QR, and SVD decompositions to solve linear systems and factor matrices [Assessment]

4.2.2 ANT/Vector Spaces and Linear Transformations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Vector spaces, subspaces, bases, and linear maps between vector spaces.

Topics:

Core

- Vector spaces and subspaces; bases, dimension, and linear independence
- Linear maps, kernel and image, rank-nullity theorem, and matrix representations

Learning Outcomes:

Core:

1. Identify whether a set with given operations forms a vector space and determine a basis [Familiarity]
2. Determine the kernel, image, and matrix representation of a linear map, verifying the rank-nullity theorem [Usage]
3. Apply linear transformations to model and solve problems involving change of basis and composition of maps [Assessment]

4.2.3 ANT/Eigenvalues, Inner Product Spaces, and Quadratic Forms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Eigenvalues, diagonalization, inner product spaces, and quadratic forms, with applications to optimization and data approximation.

Topics:

Core

- Eigenvalues and eigenvectors; diagonalization and the spectral theorem
- Inner product spaces, orthogonality, Gram-Schmidt process, and least squares
- Quadratic forms and definiteness: classification as positive, negative, or indefinite via eigenvalues

Learning Outcomes:

Core:

1. Identify the eigenvalues and eigenvectors of a matrix and determine whether it is diagonalizable [Familiarity]
2. Compute the diagonalization of matrices via the spectral theorem and apply the Gram-Schmidt process to obtain orthogonal bases [Usage]
3. Apply least squares methods and the classification of quadratic forms via eigenvalues to solve approximation and optimization problems [Assessment]

4.2.4 ANT/Group Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Groups, subgroups, homomorphisms, quotient groups, and the classification of finite groups.

Topics:

Core

- Groups, subgroups, cyclic groups, and order of elements
- Cosets, Lagrange's theorem, and normal subgroups
- Group homomorphisms, isomorphisms, and the isomorphism theorems
- Group actions, orbits, stabilizers, and Burnside's lemma
- Sylow theorems and classification of finite groups of small order

Learning Outcomes:

Core:

1. State Lagrange's theorem and explain the relationship between subgroups and cosets [Familiarity]
2. Apply the isomorphism theorems to analyze homomorphic images and quotient groups [Usage]
3. Use Sylow theorems to determine the structure and number of subgroups of a finite group [Assessment]

4.2.5 ANT/Rings, Fields, and Galois Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Rings, ideals, polynomial rings, field extensions, and the Galois correspondence between field extensions and permutation groups.

Topics:

Core

- Rings, ideals, quotient rings, and the ring isomorphism theorems
- Polynomial rings: irreducibility, factorization, and unique factorization domains
- Field extensions: algebraic and transcendental elements, degree, and splitting fields
- Galois groups, the Galois correspondence, and solvability by radicals
- Finite fields: structure, existence, and applications in coding theory

Learning Outcomes:

Core:

1. Classify rings as integral domains, PIDs, or UFDs and justify the classification [Familiarity]
2. Determine the Galois group of a polynomial and apply the Galois correspondence [Usage]
3. Prove that general quintic equations are not solvable by radicals using Galois theory [Assessment]

4.2.6 ANT/Module Theory and Commutative Algebra

Modules over rings, free and projective modules, the structure theorem for finitely generated modules, and commutative algebra.

Topics:

Non Core

- Modules over rings: definitions, submodules, quotients, and homomorphisms
- Free and projective modules; exact sequences and split sequences
- Structure theorem for finitely generated modules over a PID (Jordan normal form)
- Tensor products of modules: definition, properties, and applications
- Localization, prime and maximal ideals, Noetherian rings, and Hilbert's basis theorem

Learning Outcomes:

NonCore:

1. Explain the structure theorem for finitely generated modules over a PID and derive the Jordan normal form [Familiarity]
2. Construct tensor products and apply them to multilinear algebra problems [Usage]
3. Analyze a commutative ring by identifying its prime spectrum and applying localization [Assessment]

4.2.7 ANT/Analytic Number Theory

Distribution of prime numbers, Dirichlet series, the Riemann zeta function, and L-functions.

Topics:

Non Core

- Arithmetic functions, the prime counting function, and the Prime Number Theorem
- Dirichlet series: convergence, the Euler product, and multiplicative functions
- The Riemann zeta function: analytic continuation, functional equation, and the critical strip
- Dirichlet characters, L-functions, and primes in arithmetic progressions
- Sieve methods: Eratosthenes, Legendre, and the large sieve

Learning Outcomes:

NonCore:

1. State the Prime Number Theorem and describe the role of the Riemann zeta function in its proof [Familiarity]
2. Apply Dirichlet series and multiplicative functions to derive asymptotic estimates [Usage]
3. Analyze the distribution of primes in arithmetic progressions using Dirichlet's theorem [Assessment]

4.2.8 ANT/Algebraic Number Theory

Number fields, rings of integers, ideal factorization, class groups, and the geometry of numbers.

Topics:

Non Core

- Number fields: algebraic integers, rings of integers, and their basic properties
- Unique factorization of ideals in Dedekind domains
- Ideal class group, Minkowski's theorem, and finiteness of the class number
- Units, Dirichlet's unit theorem, and ramification of primes
- p-adic numbers and valuations; local fields

Learning Outcomes:

NonCore:

1. Describe the ring of integers of a number field and explain why unique prime factorization of elements may fail [Familiarity]
2. Compute the class group and unit group of quadratic number fields using Minkowski's bound [Usage]
3. Apply the factorization of ideals in Dedekind domains to solve Diophantine equations [Assessment]

4.2.9 ANT/Representation Theory

Representations of finite groups, characters, Schur's lemma, the character table, and applications to physics and combinatorics.

Topics:

Non Core

- Linear representations of groups: definitions, subrepresentations, and direct sums
- Schur's lemma, irreducible representations, and Maschke's theorem
- Characters, character tables, orthogonality relations, and the number of irreducible representations
- Induced representations, Frobenius reciprocity, and Mackey's theorem
- Applications to physics (molecular symmetry), combinatorics, and modular representation theory

Learning Outcomes:

NonCore:

1. Explain Schur's lemma and Maschke's theorem and their role in decomposing representations [Familiarity]
2. Construct the character table of a finite group and verify orthogonality relations [Usage]
3. Apply representation theory to determine the vibrational modes of a symmetric molecule [Assessment]

4.2.10 ANT/Cryptography and Coding Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Mathematical foundations of modern cryptography and error-correcting codes, including RSA, elliptic curves, and linear codes.

Topics:

Core

- Modular arithmetic, the RSA cryptosystem, and discrete logarithm problem
- Elliptic curves over finite fields and elliptic curve cryptography (ECC)
- Linear codes: generator and parity-check matrices, Hamming distance, and error correction
- Cyclic codes: polynomial representation, BCH codes, and Reed-Solomon codes

Non Core

- Post-quantum cryptography: lattice-based schemes and the Learning With Errors (LWE) problem

Learning Outcomes:

Core:

1. Explain the mathematical hardness assumptions underlying RSA and elliptic curve cryptography [Familiarity]
2. Construct a linear code, compute its parameters, and determine its error-correction capability [Usage]

NonCore:

3. Analyze the security assumptions of lattice-based cryptographic schemes in the post-quantum context [Assessment]

4.3 Geometry and Topology (GTO)

This area covers the study of shape, space, and continuity, ranging from classical Euclidean geometry to the modern theory of manifolds, algebraic topology, and symplectic geometry, with deep connections to physics and data science.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.3.1 Euclidean and Projective Geometry (Pg.. 180)	1	1
⭐ 4.3.2 General Topology (Pg.. 180)	1	1
⭐ 4.3.3 Differential Geometry of Curves and Surfaces (Pg.. 181)	1	1
⭐ 4.3.4 Smooth Manifolds and Differential Forms (Pg.. 181)	1	1
4.3.5 Riemannian Geometry (Pg.. 181)	Elective	
4.3.6 Algebraic Topology (Pg.. 182)	Elective	
4.3.7 Symplectic Geometry (Pg.. 182)	Elective	
4.3.8 Geometric Group Theory (Pg.. 182)	Elective	

Table 4.3: List of KUs in the Geometry and Topology area.

4.3.1 GTO/Euclidean and Projective Geometry (Core Tier1: 1 hr, Core Tier2: 1 hr)

Axiomatic Euclidean geometry, isometries, symmetry groups, and the projective plane.
Topics:
Core

- Euclidean geometry: axioms, congruence, similarity, and classical theorems
- Isometries of the plane and space: reflections, rotations, translations, and glide reflections
- Symmetry groups: frieze groups, wallpaper groups, and the classification of finite isometry groups
- Projective plane: homogeneous coordinates, duality, and cross-ratio
- Conic sections in projective geometry: classification and projective transformations

Learning Outcomes:
Core:

1. Describe the axiomatic foundation of Euclidean geometry and identify the role of the parallel postulate [Familiarity]
2. Classify isometries of the plane and determine the symmetry group of a given geometric figure [Usage]
3. Apply projective coordinates and duality to prove classical theorems (Pappus, Desargues) [Assessment]

4.3.2 GTO/General Topology (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topological spaces, continuity, compactness, connectedness, separation axioms, and quotient spaces.
Topics:
Core

- Topological spaces: open and closed sets, bases, and subbases
- Continuous functions, homeomorphisms, and topological equivalence
- Compactness (Heine-Borel), connectedness, and path-connectedness
- Separation axioms (T_1, Hausdorff, normal) and Urysohn's lemma
- Quotient spaces, product topology, and the Tychonoff theorem

Learning Outcomes:
Core:

1. Identify whether a given space is compact, connected, or Hausdorff using standard criteria [Familiarity]
2. Construct new topological spaces via quotients and products and determine their topological properties [Usage]
3. Prove basic theorems on continuous functions, including the extreme value and intermediate value theorems in topological settings [Assessment]

4.3.3 GTO/Differential Geometry of Curves and Surfaces (Core Tier1: 1 hr, Core Tier2: 1 hr)

Local and global theory of curves and surfaces in $\mathbb{R}^3$: curvature, Gauss map, and the Gauss-Bonnet theorem.

Topics:

Core

- Space curves: curvature, torsion, Frenet-Serret frame, and the fundamental theorem of curves
- Regular surfaces in $\mathbb{R}^3$: parametrizations, tangent planes, and first fundamental form
- Second fundamental form, principal curvatures, Gaussian and mean curvature, and the Gauss map
- Geodesics on surfaces and the Gauss-Bonnet theorem
- Gauss's Theorema Egregium: intrinsic nature of Gaussian curvature

Learning Outcomes:

Core:

1. Compute the curvature and torsion of a space curve from its parametric equations [Familiarity]
2. Calculate the Gaussian and mean curvature of parametric surfaces using the first and second fundamental forms [Usage]
3. Apply the Gauss-Bonnet theorem to relate the curvature of a surface to its topological Euler characteristic [Assessment]

4.3.4 GTO/Smooth Manifolds and Differential Forms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Smooth manifolds, tangent bundles, differential forms, exterior calculus, and de Rham cohomology.

Topics:

Core

- Smooth manifolds: charts, atlases, smooth maps, and the implicit function theorem
- Tangent and cotangent bundles; vector fields and their flows
- Differential forms, wedge product, exterior derivative, and Stokes' theorem on manifolds
- De Rham cohomology and the Poincaré lemma
- Lie groups and Lie algebras: definitions and fundamental examples

Learning Outcomes:

Core:

1. Define smooth manifolds and smooth maps using charts and atlases [Familiarity]
2. Apply Stokes' theorem on manifolds to unify the classical integral theorems of vector calculus [Usage]
3. Compute de Rham cohomology groups of standard manifolds and interpret their topological meaning [Assessment]

4.3.5 GTO/Riemannian Geometry

Riemannian metrics, connections, geodesics, curvature tensors, and comparison geometry.

Topics:

Non Core

- Riemannian metrics: length, distance, and volume on manifolds
- Levi-Civita connection, parallel transport, and covariant derivative
- Geodesics: existence, minimality, and the exponential map
- Riemann curvature tensor, Ricci curvature, and scalar curvature
- Comparison theorems: Bonnet-Myers, Hadamard-Cartan, and their topological consequences

Learning Outcomes:

NonCore:

1. Describe the Levi-Civita connection and explain what distinguishes it from an arbitrary connection [Familiarity]
2. Compute the Riemann curvature tensor for standard spaces (sphere, hyperbolic space) [Usage]

3. Apply comparison theorems to deduce topological consequences from curvature bounds [Assessment]

4.3.6 GTO/Algebraic Topology

Homotopy, fundamental group, covering spaces, singular homology and cohomology, and the Mayer-Vietoris sequence.

Topics:

Non Core

- Fundamental group, homotopy equivalence, and Van Kampen's theorem
- Covering spaces and their classification via the fundamental group
- Singular homology: chain complexes, exact sequences, and the Mayer-Vietoris sequence
- Singular cohomology, the Universal Coefficient Theorem, and cup products
- Higher homotopy groups and the Hurewicz theorem

Learning Outcomes:

NonCore:

1. Compute the fundamental group of standard spaces using Van Kampen's theorem [Familiarity]
2. Apply the Mayer-Vietoris sequence to compute singular homology groups [Usage]
3. Distinguish topological spaces by computing and comparing their homology and cohomology rings [Assessment]

4.3.7 GTO/Symplectic Geometry

Symplectic manifolds, Hamiltonian systems, Lagrangian submanifolds, and connections to classical mechanics and mirror symmetry.

Topics:

Elective

- Symplectic manifolds and the symplectic form; Darboux's theorem
- Hamiltonian vector fields, Poisson brackets, and symplectic flows
- Lagrangian submanifolds and their role in quantization and mirror symmetry
- Moment maps and symplectic reduction (Marsden-Weinstein)
- Contact geometry: contact manifolds, Reeb vector fields, and connections to dynamics

Learning Outcomes:

Elective:

1. State Darboux's theorem and explain why all symplectic manifolds of the same dimension are locally equivalent [Familiarity]
2. Formulate Hamiltonian systems on symplectic manifolds and compute symplectic flows [Usage]
3. Apply symplectic reduction to analyze mechanical systems with symmetry [Assessment]

4.3.8 GTO/Geometric Group Theory

Groups as geometric objects: Cayley graphs, quasi-isometry, word metrics, hyperbolic groups, and growth.

Topics:

Elective

- Cayley graphs, word metric, and the coarse geometry of groups
- Quasi-isometry: definition, examples, and the Švarc-Milnor lemma
- Hyperbolic groups (Gromov): definition, thin triangles, and boundary at infinity
- Growth functions of groups: polynomial growth (Gromov's theorem) and exponential growth
- CAT(0) spaces and groups acting on them

Learning Outcomes:

Elective:

1. Construct the Cayley graph of a finitely generated group and compute its word metric [Familiarity]
2. Verify quasi-isometry between groups acting on the same metric space via the Švarc-Milnor lemma [Usage]

3. Analyze the geometry of a hyperbolic group using thin triangles and the boundary at infinity [Assessment]

4.4 Discrete Mathematics and Combinatorics (DMC)

This area covers the mathematics of finite and countable structures, including combinatorial enumeration, graph theory, discrete optimization, and the logical foundations of mathematics. It underpins theoretical computer science, operations research, and modern data science.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.4.1 Enumerative Combinatorics (Pg.. 183)	1	1
⭐ 4.4.2 Graph Theory (Pg.. 183)	1	1
4.4.3 Extremal and Probabilistic Combinatorics (Pg.. 184)	Elective	
⭐ 4.4.4 Discrete Optimization (Pg.. 184)	1	1
4.4.5 Design Theory and Combinatorial Codes (Pg.. 184)	Elective	
⭐ 4.4.6 Mathematical Logic and Set Theory (Pg.. 185)	1	1

Table 4.4: List of KUs in the Discrete Mathematics and Combinatorics area.

4.4.1 DMC/Enumerative Combinatorics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Systematic counting of discrete structures using bijections, recursions, generating functions, and the principle of inclusion-exclusion.

Topics:

Core

- Basic counting principles: permutations, combinations, and the binomial theorem
- Principle of inclusion-exclusion and applications to derangements and Euler's totient function
- Ordinary and exponential generating functions; solving recurrences
- Integer partitions, Young tableaux, and the hook-length formula
- Bijective proofs and Möbius inversion on posets

Learning Outcomes:

Core:

1. Apply basic counting principles and inclusion-exclusion to enumeration problems [Familiarity]
2. Construct ordinary and exponential generating functions to enumerate combinatorial structures [Usage]
3. Derive closed-form formulas for combinatorial sequences using bijections and Möbius inversion [Assessment]

4.4.2 DMC/Graph Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Structure and properties of graphs: connectivity, colorings, matchings, planarity, and spectral graph theory.

Topics:

Core

- Graphs: definitions, isomorphism, degree sequences, trees, and spanning trees
- Connectivity, Menger's theorem, and network flows (max-flow min-cut)
- Matchings (Hall's theorem), graph colorings, and the chromatic polynomial
- Planar graphs, Euler's formula, Kuratowski's theorem, and the four-color theorem
- Adjacency and Laplacian matrices, eigenvalues, and expander graphs

Learning Outcomes:

Core:

1. Identify structural properties of graphs (connectivity, planarity, bipartiteness) and apply Euler's formula [Familiarity]
2. Apply Hall's marriage theorem and network flow algorithms to matching and routing problems [Usage]

3. Analyze a graph's spectrum to bound its chromatic number and connectivity properties [Assessment]

4.4.3 DMC/Extremal and Probabilistic Combinatorics

Extremal problems on graphs and set systems, the probabilistic method, and random graphs.
Topics:
Non Core

- Turán-type problems: Turán's theorem and the Kruskal-Katona theorem
- The probabilistic method: first and second moment methods, Lovász Local Lemma
- Erdős-Rényi random graphs: threshold phenomena and component structure
- Ramsey theory: Ramsey numbers, van der Waerden's theorem, and Hales-Jewett
- Algebraic methods in combinatorics: polynomial method and Chevalley-Warning theorem

Learning Outcomes:
NonCore:

1. State Turán's theorem and apply it to bound the number of edges in a K_r-free graph [Familiarity]
2. Apply the probabilistic method and the Lovász Local Lemma to prove existence of combinatorial structures [Usage]
3. Analyze threshold phenomena in Erdős-Rényi random graphs using first and second moment methods [Assessment]

4.4.4 DMC/Discrete Optimization (Core Tier1: 1 hr, Core Tier2: 1 hr)

Integer programming, network flows, matroids, matching algorithms, and the theory of NP-completeness.
Topics:
Core

- Integer linear programming: branch and bound, cutting planes, and totally unimodular matrices
- Network flow algorithms: shortest paths, max-flow, min-cost flow
- Matroid theory: independent sets, greedy algorithm, and matroid intersection
- Complexity theory: P vs. NP, NP-completeness, and reductions
- Approximation algorithms and the polynomial-time approximation scheme (PTAS)

Learning Outcomes:
Core:

1. Explain the relationship between linear programming relaxations and integer programming bounds [Familiarity]
2. Apply network flow and matroid algorithms to solve combinatorial optimization problems [Usage]
3. Prove NP-completeness of a combinatorial problem via polynomial reduction [Assessment]

4.4.5 DMC/Design Theory and Combinatorial Codes

Combinatorial designs, Latin squares, finite geometries, and their connections to error-correcting codes.
Topics:
Non Core

- Block designs: balanced incomplete block designs (BIBDs), Fisher's inequality, and symmetric designs
- Latin squares, orthogonality, and applications to experimental design
- Finite projective and affine planes; connection to combinatorial designs
- Hadamard matrices and their applications in coding and signal processing
- Connections between combinatorial designs and linear codes (Hamming, Golay)

Learning Outcomes:
NonCore:

1. Define a BIBD and verify that a given configuration satisfies its parameters [Familiarity]
2. Construct a balanced design or orthogonal Latin squares for a given parameter set [Usage]

3. Relate the structure of a combinatorial design to the parity-check matrix of a corresponding linear code [Assessment]

4.4.6 DMC/Mathematical Logic and Set Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Propositional and first-order logic, proof systems, Gödel's completeness and incompleteness theorems, and axiomatic set theory.

Topics:

Core

- Propositional logic: syntax, semantics, satisfiability, and resolution
- First-order logic: models, soundness, and the Gödel completeness theorem
- Gödel's incompleteness theorems and their implications for formal systems
- Axiomatic set theory (ZFC): ordinals, cardinals, and the axiom of choice
- Computability: Turing machines, decidability, and the halting problem

Learning Outcomes:

Core:

1. State Gödel's completeness and first incompleteness theorems and explain their significance [Familiarity]
2. Construct formal proofs in first-order logic and determine the satisfiability of formulas [Usage]
3. Analyze the undecidability of problems by reduction to the halting problem [Assessment]

4.5 Probability and Statistics (PST)

This area covers the mathematical theory of uncertainty and inference, from the axiomatic foundations of probability to modern statistical methodology including Bayesian inference, high-dimensional statistics, and time series analysis.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.5.1 Probability Axioms and Distributions (Pg.. 185)	1	1
⭐ 4.5.2 Limit Theorems (Pg.. 186)	1	1
⭐ 4.5.3 Stochastic Processes (Pg.. 186)	1	1
⭐ 4.5.4 Estimation Theory (Pg.. 186)	1	1
⭐ 4.5.5 Hypothesis Testing (Pg.. 187)	1	1
⭐ 4.5.6 Bayesian Inference (Pg.. 187)	1	1
4.5.7 High-Dimensional Statistics (Pg.. 187)	Elective	
⭐ 4.5.8 Time Series Analysis (Pg.. 188)	1	1
⭐ 4.5.9 Nonparametric Methods (Pg.. 188)	1	1

Table 4.5: List of KUs in the Probability and Statistics area.

4.5.1 PST/Probability Axioms and Distributions (Core Tier1: 1 hr, Core Tier2: 1 hr)

Kolmogorov's axiomatic foundation of probability, random variables, distributions, and expectations.

Topics:

Core

- Kolmogorov axioms, sigma-algebras, and probability spaces
- Random variables: discrete and continuous distributions, CDF, PDF, and PMF
- Expectation, variance, moments, and moment generating functions
- Joint, marginal, and conditional distributions; independence and covariance
- Standard families: Binomial, Poisson, Normal, Exponential, Gamma, and Beta

Learning Outcomes:

Core:

1. State Kolmogorov's axioms and construct a probability space for a standard experiment [Familiarity]
2. Compute expectations, variances, and moment generating functions for standard distributions [Usage]
3. Determine joint and conditional distributions and assess independence of random variables [Assessment]

4.5.2 PST/Limit Theorems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Laws of large numbers, central limit theorem, modes of convergence, and large deviations.
Topics:
Core

- Modes of convergence: almost sure, in probability, in L^p, and in distribution
- Weak and strong laws of large numbers and their applications
- Central Limit Theorem: statement, proof via characteristic functions, and Berry-Esseen bound
- Characteristic functions and the continuity theorem (Lévy)
- Large deviations: Cramér's theorem and the Gärtner-Ellis theorem

Learning Outcomes:
Core:

1. Distinguish between modes of convergence of random variables and provide counterexamples showing their differences [Familiarity]
2. Apply the Central Limit Theorem to approximate distributions of sums and means [Usage]
3. Prove the strong law of large numbers and estimate large deviation probabilities using Cramér's theorem [Assessment]

4.5.3 PST/Stochastic Processes (Core Tier1: 1 hr, Core Tier2: 1 hr)

Markov chains, Poisson processes, Brownian motion, martingales, and stochastic calculus.
Topics:
Core

- Discrete-time Markov chains: classification of states, stationary distributions, and ergodic theorem
- Poisson processes: definition, properties, and generalizations (compound, inhomogeneous)
- Brownian motion: definition, path properties, and the reflection principle
- Martingales, stopping times, optional sampling theorem, and Doob's inequalities
- Itô integral, Itô's formula, and stochastic differential equations

Learning Outcomes:
Core:

1. Classify states of a Markov chain and determine its stationary distribution [Familiarity]
2. Apply martingale theory and the optional sampling theorem to solve stopping time problems [Usage]
3. Formulate and solve a stochastic differential equation using Itô's formula [Assessment]

4.5.4 PST/Estimation Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Point and interval estimation: sufficiency, unbiasedness, efficiency, maximum likelihood, and asymptotic theory.
Topics:
Core

- Sufficient statistics, the factorization theorem, and completeness
- Unbiased estimators, the Cramér-Rao lower bound, and UMVUE
- Maximum likelihood estimation: properties, consistency, and asymptotic normality
- Confidence intervals: pivotal quantities, likelihood ratio, and bootstrap methods
- Asymptotic theory: delta method, M-estimators, and the Fisher information matrix

Learning Outcomes:
Core:

1. Explain the Cramér-Rao bound and identify when a UMVUE exists using the Lehmann-Scheffé theorem [Familiarity]
2. Derive maximum likelihood estimators and compute their asymptotic variance [Usage]
3. Construct confidence intervals using pivotal quantities and bootstrap resampling [Assessment]

4.5.5 PST/Hypothesis Testing (Core Tier1: 1 hr, Core Tier2: 1 hr)

Neyman-Pearson framework, uniformly most powerful tests, likelihood ratio tests, and multiple testing.
Topics:
Core

- Neyman-Pearson lemma: most powerful tests for simple hypotheses
- Uniformly most powerful (UMP) tests and monotone likelihood ratio families
- Generalized likelihood ratio tests and Wilks' theorem
- p-values, multiple testing, and false discovery rate (FDR) control
- Nonparametric tests: sign test, Wilcoxon rank-sum, and Kolmogorov-Smirnov

Learning Outcomes:
Core:

1. State the Neyman-Pearson lemma and explain the trade-off between type I and type II errors [Familiarity]
2. Apply the likelihood ratio test and Wilks' theorem to composite hypothesis problems [Usage]
3. Control the false discovery rate in a multiple testing scenario using the Benjamini-Hochberg procedure [Assessment]

4.5.6 PST/Bayesian Inference (Core Tier1: 1 hr, Core Tier2: 1 hr)

Bayesian paradigm: prior and posterior distributions, conjugate families, hierarchical models, and computational methods.
Topics:
Core

- Bayes' theorem in continuous settings; prior and posterior distributions
- Conjugate prior families: Beta-Binomial, Dirichlet-Multinomial, Normal-Normal
- Hierarchical Bayesian models and empirical Bayes
- Markov chain Monte Carlo (MCMC): Metropolis-Hastings and Gibbs sampling
- Variational inference and the evidence lower bound (ELBO)

Learning Outcomes:
Core:

1. Apply Bayes' theorem to update a prior with observed data and obtain a posterior distribution [Familiarity]
2. Construct a hierarchical Bayesian model and derive its joint and conditional distributions [Usage]
3. Implement Gibbs sampling or Metropolis-Hastings to approximate a posterior distribution [Assessment]

4.5.7 PST/High-Dimensional Statistics

Statistical theory in high dimensions: concentration inequalities, sparsity, LASSO, and random matrix theory.
Topics:
Non Core

- Concentration inequalities: Markov, Chebyshev, Chernoff bounds, and sub-Gaussian variables
- Sparse estimation: LASSO, ridge regression, and the restricted isometry property (RIP)
- Random matrix theory: Wigner semicircle law and the Marchenko-Pastur distribution
- Compressed sensing: exact recovery conditions and convex relaxation
- Minimax lower bounds and the bias-variance trade-off in high dimensions

Learning Outcomes:
NonCore:

1. Apply concentration inequalities to bound the deviation of sums of random variables [Familiarity]
2. Analyze the statistical properties of LASSO estimators under sparsity and RIP conditions [Usage]
3. Derive minimax lower bounds for estimation problems using information-theoretic methods [Assessment]

4.5.8 PST/Time Series Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr)

Stationary processes, ARMA models, spectral analysis, state-space models, and forecasting.
Topics:
Core

- Stationarity, autocovariance function, and the autocorrelation function (ACF)
- ARMA models: identification, estimation (Yule-Walker, MLE), and diagnostics
- Spectral density, the periodogram, and Wiener-Khinchin theorem
- State-space models and the Kalman filter
- ARIMA models, seasonal adjustment, and multi-step forecasting

Learning Outcomes:
Core:

1. Identify stationarity and determine the order of an ARMA model from ACF and PACF plots [Familiarity]
2. Fit ARIMA models to time series data, validate residuals, and produce forecasts [Usage]
3. Apply the Kalman filter to estimate hidden states in a linear Gaussian state-space model [Assessment]

4.5.9 PST/Nonparametric Methods (Core Tier1: 1 hr, Core Tier2: 1 hr)

Distribution-free inference, rank-based tests, kernel density estimation, and nonparametric regression.
Topics:
Core

- Empirical distribution function, the Glivenko-Cantelli theorem, and the bootstrap
- Rank-based tests: Wilcoxon, Kruskal-Wallis, and Spearman correlation
- Kernel density estimation: bandwidth selection, bias-variance trade-off
- Nonparametric regression: Nadaraya-Watson estimator and smoothing splines
- Permutation tests and randomization inference

Learning Outcomes:
Core:

1. Explain the advantages and limitations of nonparametric methods compared to parametric alternatives [Familiarity]
2. Apply kernel density estimation and select bandwidth using cross-validation [Usage]
3. Conduct permutation and rank-based tests and interpret their results for inference without distributional assumptions [Assessment]

4.6 Numerical and Scientific Analysis (NSA)

This area covers the design, analysis, and implementation of algorithms for numerical computation, including floating-point arithmetic, linear systems, approximation, integration, differential equations, optimization, and high-performance computing.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.6.1 Error Analysis and Floating-Point Arithmetic (Pg.. 189)	1	1
⭐ 4.6.2 Root-Finding for Nonlinear Equations (Pg.. 189)	1	1
⭐ 4.6.3 Numerical Linear Algebra (Pg.. 190)	1	1
⭐ 4.6.4 Approximation Theory and Interpolation (Pg.. 190)	1	1
⭐ 4.6.5 Numerical Integration and Quadrature (Pg.. 190)	1	1
⭐ 4.6.6 Numerical Methods for Differential Equations (Pg.. 191)	1	1
⭐ 4.6.7 Finite Element Method (Pg.. 191)	1	1
⭐ 4.6.8 Optimization Algorithms (Pg.. 191)	1	1
⭐ 4.6.9 Parallel and High-Performance Computing (Pg.. 192)	1	1

Table 4.6: List of KUs in the Numerical and Scientific Analysis area.

4.6.1 NSA/Error Analysis and Floating-Point Arithmetic (Core Tier1: 1 hr, Core Tier2: 1 hr)

Sources of numerical error, IEEE 754 floating-point arithmetic, rounding, cancellation, and forward and backward error analysis.

Topics:

Core

- IEEE 754 floating-point representation: machine epsilon, overflow, underflow, and rounding modes
- Rounding errors, catastrophic cancellation, and loss of significance
- Forward and backward error analysis; condition number of a problem
- Numerical stability: stable vs. unstable algorithms; Wilkinson's backward error analysis
- Error propagation in arithmetic operations and function evaluation

Learning Outcomes:

Core:

1. Explain the IEEE 754 floating-point standard and identify the sources of rounding and cancellation errors [Familiarity]
2. Estimate the condition number of a problem and predict the accuracy of a numerical result [Usage]
3. Analyze an algorithm for numerical stability using forward and backward error bounds [Assessment]

4.6.2 NSA/Root-Finding for Nonlinear Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Iterative methods for approximating roots of single-variable nonlinear equations, their convergence analysis, and their extension to systems of nonlinear equations.

Topics:

Core

- Bisection method: interval bracketing, stopping criteria, and error bound
- Fixed-point iteration and sufficient conditions for convergence
- Newton-Raphson method: derivation, quadratic convergence, and failure cases
- Secant method as an alternative to Newton's method without an explicit derivative
- Order of convergence and efficiency comparison among iterative methods
- Extension of Newton's method to systems of nonlinear equations via the Jacobian matrix

Learning Outcomes:

Core:

1. Compare the assumptions and convergence guarantees of the bisection, fixed-point, Newton-Raphson, and secant methods [Familiarity]
2. Implement the Newton-Raphson method and the secant method to approximate roots of a nonlinear function [Usage]

3. Analyze the order of convergence of a given iterative method and justify the choice of method based on computational cost and derivative availability [Assessment]

4.6.3 NSA/Numerical Linear Algebra (Core Tier1: 1 hr, Core Tier2: 1 hr)

Algorithms for solving linear systems, least squares, eigenvalue problems, and matrix factorizations, with attention to stability and complexity.

Topics:

Core

- Gaussian elimination with partial pivoting, LU factorization, and complexity analysis
- QR factorization via Householder and Gram-Schmidt; least squares problems
- Eigenvalue algorithms: power iteration, QR algorithm, and the Lanczos method
- Singular value decomposition: computation, truncation, and applications (PCA, pseudoinverse)
- Iterative methods for large sparse systems: CG, GMRES, and preconditioning

Learning Outcomes:

Core:

1. Compare direct and iterative solvers and select the appropriate method for a given matrix structure [Familiarity]
2. Apply QR factorization and SVD to solve least squares problems and compute low-rank approximations [Usage]
3. Analyze the numerical stability and computational cost of eigenvalue algorithms [Assessment]

4.6.4 NSA/Approximation Theory and Interpolation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Polynomial and spline interpolation, best approximation, Chebyshev polynomials, and rational approximation.

Topics:

Core

- Polynomial interpolation: Lagrange, Newton divided differences, and Runge's phenomenon
- Spline interpolation: cubic splines, B-splines, and piecewise polynomial methods
- Best approximation in normed spaces: Chebyshev (minimax) and least squares
- Chebyshev polynomials: properties, orthogonality, and spectral convergence
- Trigonometric approximation and the fast Fourier transform (FFT)

Learning Outcomes:

Core:

1. Explain Runge's phenomenon and justify the choice of Chebyshev nodes to mitigate it [Familiarity]
2. Construct cubic spline and Chebyshev interpolants for given data and estimate the interpolation error [Usage]
3. Apply the FFT to efficiently compute trigonometric approximations of a sampled function [Assessment]

4.6.5 NSA/Numerical Integration and Quadrature (Core Tier1: 1 hr, Core Tier2: 1 hr)

Quadrature rules, adaptive integration, Gaussian quadrature, and multi-dimensional integration.

Topics:

Core

- Newton-Cotes rules: trapezoidal, Simpson's, and composite rules; error analysis
- Adaptive quadrature and automatic error control
- Gaussian quadrature: optimal nodes and weights, orthogonal polynomial connection
- Numerical treatment of improper and singular integrals
- Monte Carlo integration and quasi-Monte Carlo methods for high-dimensional integrals

Learning Outcomes:
Core:

1. Derive the error formula for composite Simpson's rule and identify its order of accuracy [Familiarity]
2. Select and apply an appropriate quadrature rule (Gaussian, adaptive) based on integrand regularity [Usage]
3. Apply Monte Carlo integration to estimate high-dimensional integrals and quantify the statistical error [Assessment]

4.6.6 NSA/Numerical Methods for Differential Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Finite difference and Runge-Kutta methods for ODEs, stability analysis, and finite difference methods for PDEs.
Topics:
Core

- Runge-Kutta methods: Euler, RK4, and embedded methods for error control
- Linear multistep methods: Adams-Bashforth, Adams-Moulton, and BDF methods
- Stability analysis: zero-stability, absolute stability regions, and stiff ODEs
- Finite difference methods for parabolic and elliptic PDEs: stability and convergence (Lax equivalence theorem)
- Boundary value problems: shooting method and finite difference discretization

Learning Outcomes:
Core:

1. Compare explicit and implicit ODE solvers and explain their relative stability for stiff problems [Familiarity]
2. Implement a Runge-Kutta method with adaptive step control and apply it to a system of ODEs [Usage]
3. Analyze the stability and convergence of a finite difference scheme for a parabolic PDE using the Lax equivalence theorem [Assessment]

4.6.7 NSA/Finite Element Method (Core Tier1: 1 hr, Core Tier2: 1 hr)

Variational formulation, Galerkin methods, finite element spaces, error estimates, and applications to elliptic PDEs.
Topics:
Core

- Weak (variational) formulation of boundary value problems and Sobolev spaces
- Galerkin method: finite-dimensional approximation spaces and the stiffness matrix
- Triangular and quadrilateral finite element spaces: Lagrange elements and conformity
- A priori error estimates: Céa's lemma and interpolation error bounds

Non Core

- A posteriori error estimates and adaptive mesh refinement

Learning Outcomes:
Core:

1. Derive the weak formulation of an elliptic BVP and show its equivalence to the strong form [Familiarity]
2. Assemble the global stiffness matrix and load vector for a linear finite element discretization [Usage]
3. Estimate the H^1 error of a finite element solution using Céa's lemma and interpolation theory [Assessment]

4.6.8 NSA/Optimization Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Gradient methods, Newton and quasi-Newton methods, constrained optimization, and convex optimization algorithms.

Topics:
Core
- Gradient descent and line search methods: Armijo-Wolfe conditions and convergence rates
- Newton's method and quasi-Newton methods (BFGS, L-BFGS)
- Constrained optimization: KKT conditions, penalty methods, and sequential quadratic programming
- Convex optimization algorithms: interior-point methods and the alternating direction method of multipliers (ADMM)
- Stochastic gradient descent (SGD), variance reduction, and Adam optimizer

Learning Outcomes:
Core:
1. Explain the convergence guarantees of gradient descent for smooth convex functions [Familiarity]
2. Apply Newton and quasi-Newton methods to unconstrained optimization problems [Usage]
3. Formulate a constrained optimization problem as a KKT system and apply an interior-point method [Assessment]

4.6.9 NSA/Parallel and High-Performance Computing (Core Tier1: 1 hr, Core Tier2: 1 hr)

Parallel architectures, performance models, parallel algorithms for linear algebra, and GPU computing.
Topics:
Core
- Parallel architectures: shared memory (OpenMP), distributed memory (MPI), and GPU (CUDA)
- Performance models: Amdahl's law, roofline model, and communication complexity
- Parallel algorithms for dense and sparse linear algebra (ScaLAPACK, PETSc)
- Domain decomposition and parallel PDE solvers

Non Core
- Automatic differentiation (forward and reverse mode) and its role in scientific ML

Learning Outcomes:
Core:
1. Explain Amdahl's law and identify the bottlenecks limiting parallel speedup in a given algorithm [Familiarity]
2. Implement a parallel linear algebra routine using MPI or OpenMP and measure parallel efficiency [Usage]

NonCore:
3. Apply automatic differentiation to compute gradients of complex scientific functions for optimization [Assessment]

4.7 Mathematical Modeling and Simulation (MMS)

This area covers the mathematical tools for describing and simulating real-world phenomena, spanning ordinary and partial differential equations, dynamical systems, perturbation methods, fluid dynamics, control theory, and agent-based modeling.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.7.1 First-Order Ordinary Differential Equations (Pg.. 193)	1	1
⭐ 4.7.2 Higher-Order Differential Equations and Systems (Pg.. 193)	1	1
⭐ 4.7.3 Partial Differential Equations (Pg.. 193)	1	1
⭐ 4.7.4 Dynamical Systems and Chaos (Pg.. 194)	1	1
⭐ 4.7.5 Asymptotic Analysis and Perturbation Methods (Pg.. 194)	1	1
4.7.6 Mathematical Fluid Dynamics (Pg.. 194)	Elective	
⭐ 4.7.7 Mathematical Control Theory (Pg.. 195)	1	1
⭐ 4.7.8 Stochastic Differential Equations (Pg.. 195)	1	1
4.7.9 Agent-Based and Complex Systems Modeling (Pg.. 195)	Elective	

Table 4.7: List of KUs in the Mathematical Modeling and Simulation area.

4.7.1 MMS/First-Order Ordinary Differential Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Solution methods, applications, and qualitative analysis of first-order ordinary differential equations: exact solution techniques, modeling of physical phenomena, and qualitative behavior of solutions.

Topics:

Core

- First-order ODEs: separable, linear, exact equations, and integrating factors
- Applications and modeling with first-order ODEs: growth and decay, mixing problems, Newton's law of cooling, and Torricelli's law
- Qualitative analysis of autonomous first-order ODEs: direction fields, equilibrium solutions, and stability

Learning Outcomes:

Core:

1. Classify first-order ODEs and select an appropriate solution method (separation of variables, linear equation, exact equation, or integrating factor) [Familiarity]
2. Formulate and solve first-order models for growth and decay, mixing problems, Newton's law of cooling, and Torricelli's law [Usage]
3. Analyze autonomous first-order ODEs qualitatively using direction fields, and identify equilibrium points and their stability [Assessment]

4.7.2 MMS/Higher-Order Differential Equations and Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Theory and solution methods for higher-order ODEs and linear systems: characteristic equations, eigenvalue systems, existence and uniqueness theorems, and Laplace transform methods.

Topics:

Core

- Higher-order linear ODEs: characteristic equations, variation of parameters, and undetermined coefficients
- Systems of linear ODEs: matrix exponential, eigenvalue methods, and phase portraits
- Existence and uniqueness theorems (Picard-Lindelöf) and continuation of solutions
- Laplace transform methods for solving ODEs with discontinuous forcing

Learning Outcomes:

Core:

1. State the Picard-Lindelöf existence and uniqueness theorem and interpret its implications for the continuation of solutions of a higher-order ODE [Familiarity]
2. Solve linear systems of ODEs using eigenvalue methods and sketch their phase portraits [Usage]
3. Apply the Laplace transform to solve ODEs with piecewise and impulsive forcing functions [Assessment]

4.7.3 MMS/Partial Differential Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Classification and solution of PDEs: heat, wave, and Laplace equations; characteristics; Fourier methods; and weak solutions.

Topics:

Core

- Classification of second-order linear PDEs: elliptic, parabolic, and hyperbolic
- Fourier series and separation of variables for the heat and wave equations
- Green's functions for the Laplace and Poisson equations
- Method of characteristics for first-order and hyperbolic PDEs; shock formation
- Weak solutions, Sobolev spaces, and variational formulations of elliptic problems

Learning Outcomes:

Core:

1. Classify a second-order PDE and identify the appropriate solution strategy [Familiarity]
2. Apply separation of variables and Fourier series to solve initial-boundary value problems [Usage]
3. Derive weak formulations of elliptic PDEs and prove existence of solutions using the Lax-Milgram theorem [Assessment]

4.7.4 MMS/Dynamical Systems and Chaos (Core Tier1: 1 hr, Core Tier2: 1 hr)

Phase plane analysis, stability, bifurcations, limit cycles, strange attractors, and chaos in continuous and discrete systems.

Topics:

Core

- Phase portraits of 2D systems: equilibria, stability, and limit cycles
- Lyapunov stability theory and the linearization (Hartman-Grobman) theorem
- Bifurcation theory: saddle-node, pitchfork, Hopf, and global bifurcations
- Chaos: Lorenz system, sensitivity to initial conditions, and Lyapunov exponents
- Discrete-time maps: fixed points, period-doubling, and the logistic map

Learning Outcomes:

Core:

1. Sketch phase portraits of 2D autonomous systems and classify equilibria by stability [Familiarity]
2. Apply Lyapunov's method and the Hartman-Grobman theorem to assess stability of equilibria [Usage]
3. Identify bifurcation types in parameterized systems and trace the evolution of invariant sets [Assessment]

4.7.5 MMS/Asymptotic Analysis and Perturbation Methods (Core Tier1: 1 hr, Core Tier2: 1 hr)

Asymptotic expansions, regular and singular perturbation theory, boundary layers, and multiple scales.

Topics:

Core

- Asymptotic expansions and the Landau order symbols (O, o, $\sim$)
- Regular perturbation theory for algebraic equations and ODEs
- Singular perturbation theory and matched asymptotic expansions (boundary layers)
- Method of multiple scales and the WKB approximation
- Laplace's method and stationary phase for asymptotic evaluation of integrals

Learning Outcomes:

Core:

1. Construct a regular perturbation expansion for an ODE with a small parameter [Familiarity]
2. Apply matched asymptotic expansions to solve singular perturbation problems with boundary layers [Usage]
3. Use the method of multiple scales to analyze slowly varying oscillations and secular terms [Assessment]

4.7.6 MMS/Mathematical Fluid Dynamics

Continuum mechanics, Euler and Navier-Stokes equations, vorticity, potential flow, and turbulence modeling.

Topics:

Non Core

- Continuum hypothesis, material derivative, and the Reynolds transport theorem
- Euler equations for inviscid flow: conservation form, vorticity, and potential flow
- Navier-Stokes equations: derivation, exact solutions, and dimensional analysis
- Boundary layer theory: Prandtl equations and the Blasius solution
- Turbulence: Reynolds averaging, the k-ε model, and direct numerical simulation

Learning Outcomes:

NonCore:

1. Derive the Navier-Stokes equations from conservation principles and identify each term's physical meaning [Familiarity]
2. Analyze inviscid flows using the Euler equations and apply Bernoulli's principle [Usage]
3. Apply boundary layer theory to estimate drag on a flat plate and compare with the Blasius solution [Assessment]

4.7.7 MMS/Mathematical Control Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

State-space models, controllability and observability, feedback design, stability (Lyapunov), and optimal control.

Topics:

Core

- State-space representation of linear systems: transfer functions and realizations
- Controllability and observability: rank conditions and Kalman decomposition
- Feedback stabilization: pole placement, LQR design, and the separation principle
- Lyapunov stability for nonlinear systems and input-output stability
- Optimal control: the Pontryagin Maximum Principle and Hamilton-Jacobi-Bellman equation

Learning Outcomes:

Core:

1. Determine the controllability and observability of a linear system from its state-space matrices [Familiarity]
2. Design a state-feedback controller using pole placement or LQR optimization [Usage]
3. Apply the Pontryagin Maximum Principle to solve a basic optimal control problem [Assessment]

4.7.8 MMS/Stochastic Differential Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Itô calculus, SDEs driven by Brownian motion, the Fokker-Planck equation, and numerical simulation of SDEs.

Topics:

Core

- Itô integral, quadratic variation, and Itô's formula
- Existence and uniqueness of solutions to SDEs under Lipschitz conditions
- Fokker-Planck equation: derivation and connection to SDEs
- Feynman-Kac formula and connections to parabolic PDEs
- Euler-Maruyama and Milstein schemes for numerical simulation of SDEs

Learning Outcomes:

Core:

1. State Itô's formula and apply it to compute the differential of a function of a stochastic process [Familiarity]
2. Derive the Fokker-Planck equation from a given SDE and interpret it as a probability flow [Usage]
3. Implement the Euler-Maruyama scheme to simulate trajectories of a stochastic differential equation [Assessment]

4.7.9 MMS/Agent-Based and Complex Systems Modeling

Agent-based modeling, emergence, network models, complex adaptive systems, and simulation methodology.

Topics:

Non Core

- Agent-based models (ABMs): design principles, rules, and emergence of macro-level behavior
- Complex network models: Erdős-Rényi, Barabási-Albert, and small-world networks
- Cellular automata: Conway's Game of Life, lattice Boltzmann methods, and universality
- System dynamics: stock-and-flow models and feedback loops (Forrester)
- Simulation methodology: calibration, sensitivity analysis, and model validation

Learning Outcomes:
NonCore:

1. Describe the principles of agent-based modeling and explain how macro-level patterns emerge from local rules [Familiarity]
2. Construct an agent-based model of a social or biological system and analyze its emergent behavior [Usage]
3. Validate a simulation model through sensitivity analysis and comparison with empirical data [Assessment]

4.8 Mathematical Foundations of Data Science (FDS)

This area covers the mathematical theory underlying modern data science and machine learning, including statistical learning, optimization, kernel methods, dimensionality reduction, topological data analysis, information theory, and the mathematics of deep learning.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.8.1 Statistical Learning Theory (Pg.. 196)	1	1
⭐ 4.8.2 Optimization for Machine Learning (Pg.. 196)	1	1
⭐ 4.8.3 Topological Data Analysis (Pg.. 197)	1	1
⭐ 4.8.4 Dimensionality Reduction (Pg.. 197)	1	1
⭐ 4.8.5 Kernel Methods and Reproducing Kernel Hilbert Spaces (Pg.. 198)	1	1
⭐ 4.8.6 Information Theory (Pg.. 198)	1	1
⭐ 4.8.7 Mathematical Foundations of Deep Learning (Pg.. 198)	1	1

Table 4.8: List of KUs in the Mathematical Foundations of Data Science area.

4.8.1 FDS/Statistical Learning Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

PAC learning framework, VC dimension, Rademacher complexity, generalization bounds, and the bias-variance trade-off.
Topics:
Core

- PAC learning framework: sample complexity, hypothesis classes, and learnability
- VC dimension: definition, examples, and the fundamental theorem of learning
- Rademacher complexity and uniform convergence bounds
- Bias-variance decomposition and the double descent phenomenon
- Regularization: Tikhonov regularization and structural risk minimization

Learning Outcomes:
Core:

1. State the PAC learning definition and determine whether a hypothesis class is PAC-learnable [Familiarity]
2. Compute the VC dimension of standard hypothesis classes and derive sample complexity bounds [Usage]
3. Analyze generalization error using Rademacher complexity and explain the double descent phenomenon [Assessment]

4.8.2 FDS/Optimization for Machine Learning (Core Tier1: 1 hr, Core Tier2: 1 hr)

Convex optimization theory, SGD and its variants, adaptive methods, and non-convex optimization landscapes.
Topics:
Core

- Convexity, strong convexity, and Fenchel duality in optimization
- Stochastic gradient descent: convergence rates for convex and non-convex objectives

- Adaptive gradient methods: AdaGrad, RMSProp, and Adam
- Non-convex optimization: saddle points, local minima, and loss landscape geometry of neural networks
- Proximal gradient methods and coordinate descent for structured problems

Learning Outcomes:
Core:

1. Explain the convergence guarantees of SGD for smooth convex functions and identify the role of the learning rate [Familiarity]
2. Apply Adam and proximal gradient methods to train regularized machine learning models [Usage]
3. Analyze the loss landscape of an overparameterized network and explain why SGD finds good minima [Assessment]

4.8.3 FDS/Topological Data Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr)

Persistent homology, Vietoris-Rips complexes, the mapper algorithm, and applications to data shape analysis.

Topics:
Core

- Simplicial complexes: Čech, Vietoris-Rips, and witness complexes
- Persistent homology: filtrations, persistence diagrams, and barcodes
- Stability theorem for persistence diagrams and the bottleneck distance
- Mapper algorithm for topological summarization of high-dimensional data

Non Core

- Applications of TDA to materials science, genomics, and neural network analysis (2020-2025)

Learning Outcomes:
Core:

1. Construct a Vietoris-Rips filtration from a point cloud and compute its persistence diagram [Familiarity]
2. Apply the stability theorem to bound the distance between persistence diagrams of similar data sets [Usage]

NonCore:

3. Interpret topological features of a data set using the mapper algorithm and explain their scientific significance [Assessment]

4.8.4 FDS/Dimensionality Reduction (Core Tier1: 1 hr, Core Tier2: 1 hr)

Linear and nonlinear dimensionality reduction: PCA, random projections, manifold learning, and modern methods (t-SNE, UMAP).

Topics:
Core

- PCA: singular value decomposition, explained variance, and optimal low-rank approximation
- Johnson-Lindenstrauss lemma and random projections for dimensionality reduction
- Manifold learning: Isomap, locally linear embedding (LLE), and spectral embedding
- Modern nonlinear methods: t-SNE and UMAP; theory, parameters, and limitations
- Autoencoders and variational autoencoders (VAEs) as nonlinear dimensionality reduction

Learning Outcomes:
Core:

1. Explain the Johnson-Lindenstrauss lemma and justify why random projections preserve distances [Familiarity]
2. Apply PCA and t-SNE to high-dimensional data and interpret the resulting low-dimensional embeddings [Usage]
3. Compare linear and nonlinear dimensionality reduction methods with respect to geometry preservation and computational cost [Assessment]

4.8.5 FDS/Kernel Methods and Reproducing Kernel Hilbert Spaces (Core Tier1: 1 hr, Core Tier2: 1 hr)

Positive definite kernels, RKHS theory, support vector machines, kernel PCA, and the kernel trick.
Topics:
Core

- Positive definite kernels: Mercer's theorem and common kernel functions (RBF, polynomial, Matérn)
- Reproducing kernel Hilbert spaces: the representer theorem and regularization
- Support vector machines: large-margin classification and the dual formulation
- Kernel PCA and kernel regression via the kernel trick
- Gaussian processes as distributions over functions: posterior inference and hyperparameter learning

Learning Outcomes:
Core:

1. State Mercer's theorem and verify that a given function is a positive definite kernel [Familiarity]
2. Derive the dual SVM formulation and apply the kernel trick to kernelize a linear algorithm [Usage]
3. Apply Gaussian process regression to a prediction problem and calibrate uncertainty estimates [Assessment]

4.8.6 FDS/Information Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Shannon entropy, mutual information, channel capacity, rate-distortion theory, and applications to machine learning.
Topics:
Core

- Shannon entropy, joint and conditional entropy, and the chain rule
- Mutual information, KL divergence, and the data processing inequality
- Channel capacity: Shannon's noisy channel coding theorem and the binary symmetric channel
- Rate-distortion theory and the trade-off between compression and fidelity
- Information-theoretic foundations of ML: MDL, variational inference (ELBO), and the information bottleneck

Learning Outcomes:
Core:

1. Compute Shannon entropy and mutual information for discrete distributions and interpret their meaning [Familiarity]
2. Apply Shannon's channel capacity theorem to determine the maximum reliable transmission rate [Usage]
3. Analyze a learning algorithm using the information bottleneck principle and relate compression to generalization [Assessment]

4.8.7 FDS/Mathematical Foundations of Deep Learning (Core Tier1: 1 hr, Core Tier2: 1 hr)

Universal approximation, neural tangent kernel, implicit bias of SGD, overparameterization, and geometric deep learning.
Topics:
Core

- Universal approximation theorems: width and depth trade-offs for ReLU networks
- Neural tangent kernel (NTK) and the infinite-width limit of neural networks
- Implicit regularization and implicit bias of SGD toward minimum-norm solutions
- Benign overfitting and double descent in overparameterized models
- Geometric deep learning: equivariance, group symmetry, and graph neural networks

Learning Outcomes:
Core:

1. State the universal approximation theorem and identify its assumptions and limitations [Familiarity]
2. Explain the neural tangent kernel and its role in understanding training dynamics of wide networks [Usage]
3. Analyze the implicit bias of gradient descent and relate it to generalization in overparameterized regimes [Assessment]

4.9 Computational Geometry and Topology (CGT)

This area covers geometric algorithms and topological methods for processing discrete spatial data, including Voronoi structures, convex hulls, mesh generation, digital topology, and applied persistent homology, with applications to computer graphics, robotics, and scientific computing.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.9.1 Voronoi Diagrams and Delaunay Triangulations (Pg.. 199)	1	1
⭐ 4.9.2 Convex Hulls and Polytopes (Pg.. 199)	1	1
⭐ 4.9.3 Digital Topology and Image Analysis (Pg.. 200)	1	1
⭐ 4.9.4 Persistent Homology and Applied Topology (Pg.. 200)	1	1
⭐ 4.9.5 Mesh Generation and Geometric Algorithms (Pg.. 201)	1	1

Table 4.9: List of KUs in the Computational Geometry and Topology area.

4.9.1 CGT/Voronoi Diagrams and Delaunay Triangulations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Voronoi diagrams, Delaunay triangulations, duality, algorithms, and applications in spatial computation.

Topics:

Core

- Voronoi diagram: definition, combinatorial complexity, and properties in $\mathbb{R}^d$
- Delaunay triangulation: empty circumcircle property and duality with Voronoi
- Algorithms: Fortune's sweep-line algorithm and randomized incremental construction
- Weighted Voronoi diagrams and power diagrams
- Applications: nearest-neighbor search, path planning, and mesh generation

Learning Outcomes:

Core:

1. Describe the duality between the Voronoi diagram and the Delaunay triangulation and prove the empty circumcircle property [Familiarity]
2. Construct Voronoi diagrams and Delaunay triangulations using Fortune's algorithm and analyze their complexity [Usage]
3. Apply Delaunay triangulations to mesh generation and nearest-neighbor queries in spatial datasets [Assessment]

4.9.2 CGT/Convex Hulls and Polytopes (Core Tier1: 1 hr, Core Tier2: 1 hr)

Convex hull algorithms in 2D and 3D, convex polytopes, extreme point queries, and half-space intersection.

Topics:

Core

- 2D convex hull algorithms: Graham scan, Jarvis march, and Chan's algorithm
- 3D convex hull: incremental construction and randomized algorithms
- Convex polytopes: face lattice, Euler's relation, and the upper bound theorem
- Half-space intersection and duality between convex hull and half-space problems
- Applications: linear programming, collision detection, and shape analysis

Learning Outcomes:
Core:

1. Compare the time complexity of Graham scan, Jarvis march, and Chan's algorithm and identify optimal cases [Familiarity]
2. Implement a 2D convex hull algorithm and extend it to the 3D incremental construction [Usage]
3. Apply duality between convex hull and half-space intersection to solve linear programming problems geometrically [Assessment]

4.9.3 CGT/Digital Topology and Image Analysis (Core Tier1: 1 hr, Core Tier2: 1 hr)

Topological properties of digital images, connected components, thinning, skeletonization, and topological persistence in image data.
Topics:
Core

- Digital adjacency, connectivity, and the Jordan curve theorem for digital images
- Connected component labeling and topological invariants of binary images
- Thinning algorithms and medial axis skeletonization
- Euler characteristic of digital objects and its computation from local patterns

Non Core

- Topological methods for image segmentation and surface reconstruction from point clouds

Learning Outcomes:
Core:

1. Explain digital adjacency and why inconsistent connectivity leads to topological paradoxes [Familiarity]
2. Apply thinning and skeletonization algorithms to extract structural features from binary images [Usage]

NonCore:

3. Compute topological invariants (Euler characteristic, Betti numbers) of a digital object and use them for shape classification [Assessment]

4.9.4 CGT/Persistent Homology and Applied Topology (Core Tier1: 1 hr, Core Tier2: 1 hr)

Simplicial homology, filtrations, persistence modules, barcodes, and applications to shape analysis and data.
Topics:
Core

- Simplicial homology: chain groups, boundary operators, and Betti numbers
- Persistence modules and the structure theorem for decomposition into interval modules
- Barcodes and persistence diagrams: birth-death pairs and topological lifetime
- Algorithms for persistent homology: the boundary matrix reduction algorithm

Non Core

- Applications: protein folding, materials science, and neuroscience (2020-2025)

Learning Outcomes:
Core:

1. Compute simplicial homology groups and Betti numbers from a simplicial complex [Familiarity]
2. Construct a filtration from a point cloud and compute its persistence diagram using boundary matrix reduction [Usage]

NonCore:

3. Interpret barcodes of real data and explain the topological features they reveal in a scientific context [Assessment]

4.9.5 CGT/Mesh Generation and Geometric Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Triangular and tetrahedral mesh generation, Delaunay refinement, quality metrics, and algorithms for spatial subdivision.

Topics:

Core

- Mesh quality metrics: aspect ratio, minimum angle, and circumradius-to-shortest-edge ratio
- Delaunay refinement: Ruppert's algorithm and Shewchuk's algorithm for quality mesh generation
- Tetrahedral mesh generation: constrained Delaunay tetrahedralization and Steiner points
- Quad and hex-dominant meshing: cross-field methods and polyhedral meshes

Non Core

- Adaptive mesh refinement driven by error estimators in finite element simulations

Learning Outcomes:

Core:

1. Explain the Delaunay refinement strategy and the role of Steiner points in achieving quality guarantees [Familiarity]
2. Generate a quality constrained Delaunay triangulation for a given planar domain using Ruppert's algorithm [Usage]

NonCore:

3. Design an adaptive mesh refinement strategy for a PDE problem and justify refinement criteria based on a posteriori error estimates [Assessment]

4.10 Quantum Information and Computing (QIC)

This area covers the mathematical theory of quantum information and computation, including the linear algebraic formalism of quantum mechanics, quantum algorithms, entanglement theory, error correction, and tensor network methods for quantum simulation.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.10.1 Quantum Mechanics Foundations for Computing (Pg.. 201)	1	1
⭐ 4.10.2 Quantum Circuits and Gates (Pg.. 202)	1	1
⭐ 4.10.3 Quantum Algorithms (Pg.. 202)	1	1
⭐ 4.10.4 Entanglement and Bell Inequalities (Pg.. 202)	1	1
⭐ 4.10.5 Quantum Error Correction (Pg.. 203)	1	1
4.10.6 Tensor Networks and Quantum Simulation (Pg.. 203)	Elective	

Table 4.10: List of KUs in the Quantum Information and Computing area.

4.10.1 QIC/Quantum Mechanics Foundations for Computing (Core Tier1: 1 hr, Core Tier2: 1 hr)

Postulates of quantum mechanics formulated for computation: state spaces, observables, measurement, and unitary evolution.

Topics:

Core

- Hilbert space formalism: state vectors, Dirac notation, and inner products
- Observables as Hermitian operators: eigenvalues, eigenstates, and the spectral theorem
- Measurement postulate: Born rule, projective measurements, and collapse of the state
- Unitary evolution: the Schrödinger equation and the time evolution operator
- Density matrices: mixed states, partial trace, and the Bloch sphere representation

Learning Outcomes:

Core:

1. State the postulates of quantum mechanics and represent single-qubit states on the Bloch sphere [Familiarity]
2. Compute measurement probabilities and post-measurement states using the Born rule and projection operators [Usage]
3. Distinguish pure from mixed states using density matrices and the partial trace over a subsystem [Assessment]

4.10.2 QIC/Quantum Circuits and Gates (Core Tier1: 1 hr, Core Tier2: 1 hr)

Qubits, quantum gates, quantum circuits, universality, and the circuit model of quantum computation.
Topics:
Core
- Single-qubit gates: Pauli, Hadamard, phase, and rotation gates
- Multi-qubit gates: CNOT, Toffoli, and controlled-U gates
- Quantum circuits: circuit diagrams, depth, and reversibility
- Universality: the Solovay-Kitaev theorem and universal gate sets
- Quantum parallelism, interference, and the role of superposition in computation

Learning Outcomes:
Core:
1. Construct quantum circuits using standard gate sets and represent their action on basis states [Familiarity]
2. Analyze the action of a quantum circuit on an input state using matrix multiplication and Dirac notation [Usage]
3. Justify the universality of a gate set using the Solovay-Kitaev theorem and approximate a target unitary [Assessment]

4.10.3 QIC/Quantum Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Quantum Fourier transform, phase estimation, Shor's factoring algorithm, Grover's search, and HHL for linear systems.
Topics:
Core
- Quantum Fourier transform: circuit construction and complexity analysis
- Quantum phase estimation algorithm and its role as a subroutine
- Shor's factoring algorithm: period finding and reduction to factoring
- Grover's search algorithm: quadratic speedup and amplitude amplification

Non Core
- HHL algorithm for linear systems and quantum speedup for machine learning

Learning Outcomes:
Core:
1. Describe the quantum Fourier transform and explain how phase estimation extracts eigenvalue information [Familiarity]
2. Trace through Shor's and Grover's algorithms step by step and identify the source of quantum speedup [Usage]

NonCore:
3. Analyze the resource requirements of the HHL algorithm and identify the conditions under which it achieves exponential speedup [Assessment]

4.10.4 QIC/Entanglement and Bell Inequalities (Core Tier1: 1 hr, Core Tier2: 1 hr)

Quantum entanglement: Bell states, Bell inequalities, CHSH inequality, no-cloning, and entanglement measures.
Topics:
Core
- Bell states, entangled versus separable states, and the Schmidt decomposition

- Bell inequalities and the CHSH inequality: local hidden variable theories vs. quantum mechanics
- No-cloning theorem and no-communication theorem
- Entanglement measures: entanglement entropy, concurrence, and negativity
- Quantum teleportation and superdense coding protocols

Learning Outcomes:
Core:

1. Verify that Bell states are maximally entangled and compute their Schmidt decomposition [Familiarity]
2. Derive the CHSH inequality and show that quantum mechanics violates it for the singlet state [Usage]
3. Quantify the entanglement of a bipartite state using entanglement entropy and concurrence [Assessment]

4.10.5 QIC/Quantum Error Correction (Core Tier1: 1 hr, Core Tier2: 1 hr)

Decoherence, quantum error models, stabilizer codes, the surface code, and fault-tolerant quantum computation.
Topics:
Core

- Decoherence and quantum noise: bit-flip, phase-flip, and depolarizing channels
- Quantum error-correcting codes: the Shor code and the 7-qubit Steane code
- Stabilizer formalism: Pauli group, stabilizer codes, and syndrome measurement
- Surface code: threshold theorem, logical qubits, and recent experimental realizations (2023-2025)
- Fault-tolerant quantum computation: transversal gates and the threshold theorem

Learning Outcomes:
Core:

1. Explain the Shor code construction and describe how it corrects arbitrary single-qubit errors [Familiarity]
2. Apply the stabilizer formalism to encode a logical qubit and determine the syndrome for a given error [Usage]
3. Analyze the surface code threshold and explain why it is the leading candidate for fault-tolerant quantum hardware [Assessment]

4.10.6 QIC/Tensor Networks and Quantum Simulation

Tensor network representations of quantum states (MPS, PEPS, MERA), contraction algorithms, and applications to quantum many-body simulation.
Topics:
Non Core

- Tensor contractions: index notation, contraction order, and computational complexity
- Matrix product states (MPS) and the density matrix renormalization group (DMRG)
- Projected entangled pair states (PEPS) for 2D and higher-dimensional systems
- Multi-scale entanglement renormalization ansatz (MERA) and holographic connections
- Tensor networks in machine learning: tensor decompositions and quantum-inspired algorithms

Learning Outcomes:
NonCore:

1. Describe the MPS representation of a many-body quantum state and explain how bond dimension controls approximation quality [Familiarity]
2. Apply DMRG to compute the ground state of a 1D quantum spin chain and analyze convergence with bond dimension [Usage]
3. Evaluate the computational cost of tensor network contraction and compare classical tensor network simulation with quantum hardware [Assessment]

4.11 Mathematical Biology and Life Sciences (MBL)

This area covers the application of mathematical methods to biological systems, spanning population dynamics, epidemiology, systems biology, bioinformatics, and computational neuroscience, with emphasis on models grounded in modern experimental data.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.11.1 Population Dynamics (Pg.. 204)	1	1
⭐ 4.11.2 Mathematical Epidemiology (Pg.. 204)	1	1
⭐ 4.11.3 Systems Biology and Network Dynamics (Pg.. 205)	1	1
⭐ 4.11.4 Bioinformatics Algorithms (Pg.. 205)	1	1
⭐ 4.11.5 Mathematical Neuroscience (Pg.. 205)	1	1

Table 4.11: List of KUs in the Mathematical Biology and Life Sciences area.

4.11.1 MBL/Population Dynamics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Mathematical models of single and interacting populations: logistic growth, Lotka-Volterra systems, spatial ecology, and evolutionary dynamics.

Topics:

Core

- Exponential and logistic growth models; carrying capacity and the Allee effect
- Lotka-Volterra predator-prey and competition models; equilibria and stability analysis
- Age-structured models: Leslie matrices and the Euler-Lotka equation
- Reaction-diffusion models and traveling waves in spatial ecology (Fisher-KPP equation)
- Evolutionary dynamics: replicator equations, game theory, and the Price equation

Learning Outcomes:

Core:

1. Analyze the stability of equilibria in Lotka-Volterra systems using linearization and phase portraits [Familiarity]
2. Construct age-structured population models using Leslie matrices and compute long-term growth rates [Usage]
3. Apply reaction-diffusion theory to model the spatial spread of an invasive species using the Fisher-KPP equation [Assessment]

4.11.2 MBL/Mathematical Epidemiology (Core Tier1: 1 hr, Core Tier2: 1 hr)

Compartmental models (SIR, SEIR), the basic reproduction number, herd immunity, network epidemiology, and real-time epidemic forecasting.

Topics:

Core

- SIR and SIS compartmental models: derivation, equilibria, and the epidemic threshold
- Basic reproduction number $\mathcal{R}_0$: definition, computation, and its role in herd immunity
- SEIR and SEIRS extensions: latency, waning immunity, and age-structured epidemics
- Network epidemiology: heterogeneous contact structures and superspreader events
- Real-time epidemic forecasting: data assimilation, uncertainty quantification, and lessons from COVID-19 (2020-2025)

Learning Outcomes:

Core:

1. Derive the epidemic threshold condition from the SIR model and interpret $\mathcal{R}_0$ in terms of herd immunity [Familiarity]
2. Extend the basic SIR model to incorporate vaccination, waning immunity, or age structure [Usage]
3. Calibrate an SEIR model to epidemic data and quantify forecast uncertainty using ensemble methods [Assessment]

4.11.3 MBL/Systems Biology and Network Dynamics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Gene regulatory networks, biochemical oscillators, signal transduction, biological network topology, and multiscale modeling.

Topics:

Core

- Gene regulatory networks: Boolean models, Hill functions, and feedback motifs
- Biochemical oscillators: the repressilator, Goodwin oscillator, and Hopf bifurcation in biology
- Signal transduction: ultrasensitivity, bistability, and the MAPK cascade
- Biological network topology: scale-free networks, network motifs, and robustness

Non Core

- Multiscale modeling: from molecular to tissue scale; constraint-based metabolic models (FBA)

Learning Outcomes:

Core:

1. Identify feedback motifs in gene regulatory networks and predict their role in adaptation and bistability [Familiarity]
2. Analyze a biochemical oscillator model using bifurcation theory and determine conditions for limit cycle oscillations [Usage]

NonCore:

3. Construct a flux balance analysis model of a metabolic network and predict growth rates under nutrient constraints [Assessment]

4.11.4 MBL/Bioinformatics Algorithms (Core Tier1: 1 hr, Core Tier2: 1 hr)

Sequence alignment, hidden Markov models, phylogenetics, genome assembly, and machine learning in genomics.

Topics:

Core

- Sequence alignment: dynamic programming (Needleman-Wunsch, Smith-Waterman) and BLAST heuristics
- Hidden Markov models for sequence analysis: the Viterbi and Baum-Welch algorithms
- Phylogenetic inference: distance methods (UPGMA, neighbor-joining) and maximum likelihood
- Genome assembly: de Bruijn graph approach, overlap-layout-consensus, and long-read assembly

Non Core

- Machine learning in genomics: deep learning for gene expression, variant calling, and protein structure (AlphaFold, 2021-2025)

Learning Outcomes:

Core:

1. Apply dynamic programming to perform global and local sequence alignment and interpret alignment scores [Familiarity]
2. Implement an HMM for gene finding and apply the Viterbi algorithm to decode the most likely hidden state sequence [Usage]

NonCore:

3. Evaluate deep learning approaches for protein structure prediction and explain the key innovations of AlphaFold [Assessment]

4.11.5 MBL/Mathematical Neuroscience (Core Tier1: 1 hr, Core Tier2: 1 hr)

Biophysical neuron models, neural coding, network dynamics, dimensionality reduction in neural data, and computational psychiatry.

Topics:

Core

- Hodgkin-Huxley model: ion channel dynamics, action potential generation, and bifurcation analysis

- Integrate-and-fire models: leaky IF, exponential IF, and adaptive spike-frequency adaptation
- Neural coding: rate coding, spike timing, population codes, and the neural information bottleneck
- Neural population dynamics: ring attractors, low-dimensional manifolds, and latent variable models (2020-2025)

Non Core

- Computational psychiatry: Bayesian brain hypothesis, predictive coding, and reinforcement learning in decision-making

Learning Outcomes:

Core:

1. Derive the action potential from the Hodgkin-Huxley equations and identify the role of each ion current [Familiarity]
2. Apply dimensionality reduction to neural population recordings and identify low-dimensional activity manifolds [Usage]

NonCore:

3. Formulate a predictive coding model of perception and relate it to hierarchical Bayesian inference [Assessment]

4.12 Financial and Actuarial Mathematics (FAM)

This area covers the mathematical foundations of finance and actuarial science, from interest theory and arbitrage pricing to stochastic models for derivatives, risk measurement, actuarial life tables, and computational methods in quantitative finance.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 4.12.1 Interest Theory and Time Value of Money (Pg.. 206)	1	1
⭐ 4.12.2 Arbitrage-Free Pricing and Market Models (Pg.. 207)	1	1
⭐ 4.12.3 Black-Scholes and Option Pricing (Pg.. 207)	1	1
⭐ 4.12.4 Risk Measures and Portfolio Theory (Pg.. 207)	1	1
⭐ 4.12.5 Actuarial Models and Life Tables (Pg.. 208)	1	1
⭐ 4.12.6 Computational Finance and Monte Carlo Methods (Pg.. 208)	1	1

Table 4.12: List of KUs in the Financial and Actuarial Mathematics area.

4.12.1 FAM/Interest Theory and Time Value of Money (Core Tier1: 1 hr, Core Tier2: 1 hr)

Time value of money, compound interest, annuities, yield curves, bond pricing, and immunization strategies.

Topics:

Core

- Compound interest: accumulation functions, effective and nominal rates, and force of interest
- Annuities-immediate and annuities-due: present and accumulated values, perpetuities
- Bond pricing: price-yield relationship, duration, convexity, and amortization schedules
- Yield curves: spot rates, forward rates, and bootstrap methods for term structure
- Portfolio immunization: Redington's conditions and cash-flow matching

Learning Outcomes:

Core:

1. Compute the present value and accumulated value of annuities under various interest rate assumptions [Familiarity]
2. Price bonds using the price-yield relationship and calculate duration and convexity for interest rate sensitivity [Usage]
3. Construct a portfolio satisfying Redington's immunization conditions against parallel yield curve shifts [Assessment]

4.12.2 FAM/Arbitrage-Free Pricing and Market Models (Core Tier1: 1 hr, Core Tier2: 1 hr)

Fundamental theorems of asset pricing, risk-neutral measures, complete markets, and discrete and continuous market models.

Topics:

Core

- Arbitrage: definition, the law of one price, and static replication arguments
- Risk-neutral probability measures and the first fundamental theorem of asset pricing
- Complete markets and the second fundamental theorem of asset pricing
- Binomial tree model: replication, risk-neutral pricing, and convergence to Black-Scholes
- Martingale representation theorem and hedging in continuous markets

Learning Outcomes:

Core:

1. State the first and second fundamental theorems of asset pricing and explain their financial interpretation [Familiarity]
2. Price European and American derivatives using the binomial tree model under risk-neutral valuation [Usage]
3. Verify market completeness and construct a replicating portfolio for a derivative using the martingale representation theorem [Assessment]

4.12.3 FAM/Black-Scholes and Option Pricing (Core Tier1: 1 hr, Core Tier2: 1 hr)

Black-Scholes model: derivation, PDE, Greeks, implied volatility, and extensions to stochastic volatility and jump models.

Topics:

Core

- Black-Scholes PDE: derivation via delta-hedging and risk-neutral pricing
- Closed-form Black-Scholes formula for European calls and puts; put-call parity
- Option Greeks: delta, gamma, vega, theta, and rho; dynamic hedging strategies
- Implied volatility surface, the volatility smile, and the Dupire local volatility model

Non Core

- Stochastic volatility models: Heston model and variance swaps (2020-2025 calibration methods)

Learning Outcomes:

Core:

1. Derive the Black-Scholes PDE from the delta-hedging argument and identify its boundary conditions [Familiarity]
2. Compute option prices and Greeks using the Black-Scholes formula and interpret their hedging significance [Usage]

NonCore:

3. Calibrate a stochastic volatility model to market option prices and reproduce the implied volatility smile [Assessment]

4.12.4 FAM/Risk Measures and Portfolio Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Mean-variance optimization, coherent risk measures, Value-at-Risk, CVaR, copulas, and extreme value theory.

Topics:

Core

- Mean-variance portfolio theory: efficient frontier, the CAPM, and the Sharpe ratio
- Value-at-Risk (VaR): definition, historical simulation, and parametric estimation
- Coherent risk measures: axioms, Conditional Value-at-Risk (CVaR), and Expected Shortfall
- Copulas: Sklar's theorem, Gaussian and t-copulas, and tail dependence

- Extreme value theory: Generalized Extreme Value distribution and the Peaks-over-Threshold method

Learning Outcomes:

Core:

1. Construct the mean-variance efficient frontier and identify the tangency portfolio under the CAPM [Familiarity]
2. Compute VaR and CVaR for a portfolio using historical simulation and parametric methods [Usage]
3. Apply copula models to capture tail dependence in a multivariate risk analysis [Assessment]

4.12.5 FAM/Actuarial Models and Life Tables (Core Tier1: 1 hr, Core Tier2: 1 hr)

Survival models, life tables, net premiums, policy reserves, loss models, and credibility theory.

Topics:

Core

- Survival functions, hazard rates, and the force of mortality; Gompertz and Makeham laws
- Life tables: construction, select and ultimate tables, and interpolation methods
- Life insurance and annuity valuation: net premiums, gross premiums, and the equivalence principle
- Policy reserves: prospective and retrospective methods, Thiele's differential equation
- Loss models: frequency-severity framework, aggregate distributions, and credibility theory

Learning Outcomes:

Core:

1. Compute life table quantities (mortality rates, life expectancy) from raw survival data [Familiarity]
2. Price life insurance and annuity products using net premium valuation under standard mortality tables [Usage]
3. Calculate policy reserves prospectively and retrospectively and verify their equivalence under constant interest [Assessment]

4.12.6 FAM/Computational Finance and Monte Carlo Methods (Core Tier1: 1 hr, Core Tier2: 1 hr)

Monte Carlo simulation for derivatives pricing, variance reduction, finite difference methods for the Black-Scholes PDE, and machine learning in quantitative finance.

Topics:

Core

- Monte Carlo simulation for pricing European and path-dependent derivatives
- Variance reduction techniques: antithetic variables, control variates, and importance sampling
- Finite difference methods for the Black-Scholes PDE: explicit, implicit, and Crank-Nicolson schemes
- American option pricing: least-squares Monte Carlo (Longstaff-Schwartz) and optimal stopping

Non Core

- Machine learning in quantitative finance: deep hedging, reinforcement learning for execution, and factor models (2020-2025)

Learning Outcomes:

Core:

1. Implement a Monte Carlo pricer for a European option and estimate its standard error [Familiarity]
2. Apply variance reduction techniques to accelerate Monte Carlo pricing of path-dependent derivatives [Usage]

NonCore:

3. Evaluate deep hedging and reinforcement learning approaches to derivatives pricing and compare them with classical Black-Scholes hedging [Assessment]

Chapter 5

Body of knowledge of Physics

The 12 knowledge areas in Physics are:

5.1 Classical Mechanics (CME) (Pg.. 210)
- 5.1.1 Kinematics and Dynamics of Particles (Pg.. 211)
- 5.1.2 Work, Energy, and Conservative Forces (Pg.. 211)
- 5.1.3 Systems of Particles and Linear Momentum (Pg.. 212)
- 5.1.4 Rotational Motion and Angular Momentum (Pg.. 213)
- 5.1.5 Rigid Body Dynamics in 3D (Pg.. 213)
- 5.1.6 Non-Inertial Reference Frames and Fictitious Forces (Pg.. 214)
- 5.1.7 Lagrangian and Hamiltonian Mechanics (Pg.. 214)
- 5.1.8 Central Force Motion and Orbital Mechanics (Pg.. 215)
- 5.1.9 Theory of Small Oscillations and Coupled Oscillators (Pg.. 215)
- 5.1.10 Continuum Mechanics Foundations (Pg.. 216)
- 5.1.11 Elasticity (Pg.. 216)
- 5.1.12 Fluids: Hydrostatics, Hydrodynamics, and Viscosity (Pg.. 217)
- 5.1.13 Vibrations and Waves (Pg.. 217)
- 5.1.14 Transport Phenomena (Pg.. 218)

5.2 Electromagnetism (ELM) (Pg.. 218)
- 5.2.1 Electrostatics: Electric Field and Gauss's Law (Pg.. 219)
- 5.2.2 Electric Potential and Poisson/Laplace Equations (Pg.. 219)
- 5.2.3 Electrostatic Energy and Dielectrics (Pg.. 220)
- 5.2.4 Magnetostatics: Biot-Savart and Ampere's Laws (Pg.. 220)
- 5.2.5 Magnetic Properties of Matter (Pg.. 221)
- 5.2.6 Electrodynamics: Faraday's Law and Induction (Pg.. 221)
- 5.2.7 Maxwell's Equations and Gauge Transformations (Pg.. 222)
- 5.2.8 Electromagnetic Waves and Propagation in Media (Pg.. 222)
- 5.2.9 Radiation Theory and Dipole Radiation (Pg.. 223)
- 5.2.10 Special Relativity in Electromagnetism (Pg.. 223)
- 5.2.11 Electric Current, Circuits, and Kirchhoff's Laws (Pg.. 223)

5.3 Thermodynamics and Statistical Mechanics (TSM) (Pg.. 224)
- 5.3.1 Laws of Thermodynamics and State Functions (Pg.. 224)
- 5.3.2 Kinetic Theory of Gases (Pg.. 224)
- 5.3.3 Heat Engines, Entropy, and the Second Law (Pg.. 225)
- 5.3.4 Thermodynamic Potentials and Maxwell Relations (Pg.. 225)
- 5.3.5 Phase Transitions and Critical Phenomena (Pg.. 226)
- 5.3.6 Microcanonical, Canonical, and Grand Canonical Ensembles (Pg.. 226)
- 5.3.7 Quantum Statistics: Fermi-Dirac and Bose-Einstein (Pg.. 226)
- 5.3.8 Blackbody Radiation and Phonons (Pg.. 227)
- 5.3.9 Non-Equilibrium Statistical Mechanics (Pg.. 227)
- 5.3.10 Applications to Biological and Complex Systems (Pg.. 227)

5.4 Quantum Mechanics (QME) (Pg.. 228)
- 5.4.1 Origins of Quantum Theory and Wave-Particle Duality (Pg.. 228)
- 5.4.2 The Schrödinger Equation and Wave Functions (Pg.. 228)
- 5.4.3 Formalism: Hilbert Space and Dirac Notation (Pg.. 229)
- 5.4.4 One-Dimensional Potentials (Wells, Barriers, Tunneling) (Pg.. 229)
- 5.4.5 The Quantum Harmonic Oscillator (Pg.. 229)
- 5.4.6 Angular Momentum and Spin Physics (Pg.. 230)
- 5.4.7 The Hydrogen Atom and Central Potentials (Pg.. 230)
- 5.4.8 Time-Independent and Time-Dependent Perturbation Theory (Pg.. 230)
- 5.4.9 Identical Particles and Exchange Symmetry (Pg.. 231)
- 5.4.10 Scattering Theory and Cross Sections (Pg.. 231)

5.5 Relativity and Cosmology (RCO) (Pg.. 232)
- 5.5.1 Special Relativity: Lorentz Transformations and 4-Vectors (Pg.. 232)
- 5.5.2 Relativistic Dynamics and Energy-Momentum (Pg.. 232)
- 5.5.3 General Relativity Foundations: Equivalence Principle (Pg.. 233)
- 5.5.4 Spacetime Curvature and Einstein Field Equations (Pg.. 233)
- 5.5.5 Schwarzschild Geometry and Black Holes (Pg.. 233)
- 5.5.6 Gravitational Waves (Pg.. 233)
- 5.5.7 Big Bang Theory and Friedmann Equations (Pg.. 234)
- 5.5.8 Cosmic Microwave Background (CMB) (Pg.. 234)
- 5.5.9 Dark Matter and Dark Energy (Pg.. 234)
- 5.5.10 Early Universe Inflation and Nucleosynthesis (Pg.. 235)

5.6 Condensed Matter Physics (SSP) (Pg.. 235)
- 5.6.1 Crystal Structures and Reciprocal Lattices (Pg.. 235)
- 5.6.2 X-Ray Diffraction and Lattice Dynamics (Pg.. 236)
- 5.6.3 Free Electron Model and Band Theory of Solids (Pg.. 236)
- 5.6.4 Semiconductors, Insulators, and Metals (Pg.. 236)
- 5.6.5 Magnetic Properties: Para-, Dia-, and Ferromagnetism (Pg.. 237)
- 5.6.6 Superconductivity: BCS Theory and High-Tc (Pg.. 237)
- 5.6.7 Soft Matter Physics and Polymers (Pg.. 237)
- 5.6.8 Nanostructures and Low-Dimensional Systems (Pg.. 238)
- 5.6.9 Spintronics and Topological Insulators (Pg.. 238)
- 5.6.10 Strongly Correlated Electron Systems (Pg.. 238)

5.7 Nuclear and Particle Physics (NPP) (Pg.. 239)
- 5.7.1 Nuclear Structure, Binding Energy, and Shell Model (Pg.. 239)
- 5.7.2 Radioactivity and Nuclear Decays (Alpha, Beta,

Gamma) (Pg.. 239)
- 5.7.3 Nuclear Reactions, Fission, and Fusion (Pg.. 240)
- 5.7.4 Particle Accelerators and Detectors (Pg.. 240)
- 5.7.5 The Standard Model of Particle Physics (Pg.. 240)
- 5.7.6 Quarks, Leptons, Gauge Bosons (Pg.. 241)
- 5.7.7 Symmetries and Conservation Laws (CPT) (Pg.. 241)
- 5.7.8 Fundamental Interactions and Feynman Diagrams (Pg.. 241)
- 5.7.9 Higgs Mechanism and Beyond the Standard Model (Pg.. 242)
- 5.7.10 Neutrino Physics and Oscillation (Pg.. 242)

5.8 Quantum Information and Computing (QIC) (Pg.. 242)
- 5.8.1 Qubits, Superposition, and Entanglement (Pg.. 243)
- 5.8.2 Quantum Logic Gates and Circuits (Pg.. 243)
- 5.8.3 Quantum Algorithms (Shor, Grover) (Pg.. 243)
- 5.8.4 Quantum Teleportation and Cryptography (Pg.. 244)
- 5.8.5 Decoherence and Quantum Error Correction (Pg.. 244)
- 5.8.6 Physical Realizations of Quantum Computers (Pg.. 244)
- 5.8.7 Quantum Sensing and Metrology (Pg.. 245)
- 5.8.8 Quantum Networks and Communication (Pg.. 245)

5.9 Optics and Photonics (OPH) (Pg.. 245)
- 5.9.1 Geometrical Optics and Optical Instruments (Pg.. 246)
- 5.9.2 Wave Optics: Interference and Diffraction (Pg.. 246)
- 5.9.3 Polarization and Crystal Optics (Pg.. 247)
- 5.9.4 Lasers: Principles and Applications (Pg.. 247)
- 5.9.5 Nonlinear Optics and Frequency Conversion (Pg.. 247)
- 5.9.6 Fiber Optics and Photonic Crystals (Pg.. 248)
- 5.9.7 Ultrafast Optics and Spectroscopy (Pg.. 248)
- 5.9.8 Nano-optics and Plasmonics (Pg.. 248)

5.10 Computational and Data-Driven Physics (CDP) (Pg.. 249)
- 5.10.1 Numerical Methods for Differential Equations in Physics (Pg.. 249)
- 5.10.2 Monte Carlo Methods and Simulations (Pg.. 249)
- 5.10.3 High-Performance Computing (HPC) for Large-Scale Simulations (Pg.. 250)
- 5.10.4 Machine Learning for Physical Discovery (Pg.. 250)
- 5.10.5 Data Analysis and Uncertainty Quantification (Pg.. 250)
- 5.10.6 Modeling Complex Adaptive Systems (Pg.. 251)
- 5.10.7 Neural Networks for Quantum Systems (Pg.. 251)
- 5.10.8 Physics-Informed Neural Networks (PINNs) (Pg.. 251)

5.11 Biophysics and Medical Physics (BMP) (Pg.. 252)
- 5.11.1 Physics of Biological Macromolecules (Pg.. 252)
- 5.11.2 Biomechanics and Cellular Forces (Pg.. 252)
- 5.11.3 Neural Physics and Information Processing in the Brain (Pg.. 253)
- 5.11.4 Imaging Physics (MRI, CT, PET, Ultrasound) (Pg.. 253)
- 5.11.5 Radiation Therapy and Dosimetry (Pg.. 253)
- 5.11.6 Statistical Mechanics of Biological Networks (Pg.. 254)
- 5.11.7 Microfluidics and Lab-on-a-Chip (Pg.. 254)

5.12 Astrophysics and Space Science (ASS) (Pg.. 254)
- 5.12.1 Stellar Structure and Evolution (Pg.. 255)
- 5.12.2 Planetary Science and Exoplanets (Pg.. 255)
- 5.12.3 Galactic Dynamics and Structure (Pg.. 255)
- 5.12.4 High-Energy Astrophysics (X-rays, Gamma-rays) (Pg.. 256)
- 5.12.5 Observational Techniques and Telescopes (Pg.. 256)
- 5.12.6 Solar Physics and Space Weather (Pg.. 256)
- 5.12.7 Multi-messenger Astronomy (Gravitational Waves + Neutrinos) (Pg.. 257)

5.1 Classical Mechanics (CME)

Classical Mechanics provides the foundational principles governing the motion of macroscopic objects, from particles to rigid bodies and continuous media. It encompasses Newtonian dynamics, conservation laws, Lagrangian and Hamiltonian formalisms, oscillatory systems, and celestial mechanics.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.1.1 Kinematics and Dynamics of Particles (Pg.. 211)	1	1
⭐ 5.1.2 Work, Energy, and Conservative Forces (Pg.. 211)	1	1
⭐ 5.1.3 Systems of Particles and Linear Momentum (Pg.. 212)	1	1
⭐ 5.1.4 Rotational Motion and Angular Momentum (Pg.. 213)	1	1
⭐ 5.1.5 Rigid Body Dynamics in 3D (Pg.. 213)	1	1
⭐ 5.1.6 Non-Inertial Reference Frames and Fictitious Forces (Pg.. 214)	1	1
⭐ 5.1.7 Lagrangian and Hamiltonian Mechanics (Pg.. 214)	1	1
⭐ 5.1.8 Central Force Motion and Orbital Mechanics (Pg.. 215)	1	1
⭐ 5.1.9 Theory of Small Oscillations and Coupled Oscillators (Pg.. 215)	1	1
5.1.10 Continuum Mechanics Foundations (Pg.. 216)	Elective	
⭐ 5.1.11 Elasticity (Pg.. 216)	1	1
⭐ 5.1.12 Fluids: Hydrostatics, Hydrodynamics, and Viscosity (Pg.. 217)	1	1
⭐ 5.1.13 Vibrations and Waves (Pg.. 217)	1	1
⭐ 5.1.14 Transport Phenomena (Pg.. 218)	1	1

Table 5.1: List of KUs in the Classical Mechanics area.

5.1.1 CME/Kinematics and Dynamics of Particles (Core Tier1: 1 hr, Core Tier2: 1 hr)

Mathematical description of motion (kinematics) and the study of forces that cause or modify motion (dynamics) for point particles in various coordinate systems and force fields.

Topics:

Core

- Vector description of position, velocity, and acceleration in Cartesian, polar, and spherical coordinates
- Newton's three laws of motion and inertial reference frames
- Formulation and solution of equations of motion for constant and variable forces
- Analysis of projectile motion with and without air resistance ($F_{drag} = -bv$ or $-cv^2$)
- Static and kinetic friction forces, including angle of repose
- Motion in resistive media including linear and quadratic drag models
- Motion under constraints such as a pendulum or beads on wires
- Introduction to nonlinear oscillators and phase space analysis
- Numerical integration of equations of motion using Euler and Runge-Kutta methods

Elective

- Motion of charged particles in electromagnetic fields under the Lorentz force: $F = q(E + v \times B)$

Learning Outcomes:

Core:

1. Formulate the equations of motion for a particle in One, Two, and Three dimensions given a force law [Usage]
2. Solve analytically for the trajectory of a projectile under constant gravity neglecting air resistance [Usage]
3. Calculate the terminal velocity of an object falling through a fluid with quadratic drag [Assessment]
4. Determine equilibrium points and stability for a particle in a One-dimensional potential [Usage]
5. Identify all forces acting on a particle in a given mechanical scenario and draw a correct free-body diagram [Familiarity]
6. Implement a simple numerical algorithm such as the Euler method to compute the motion of a damped harmonic oscillator [Usage]
7. Construct and interpret phase space diagrams (x vs. $\dot{x}$) for simple oscillatory systems [Assessment]

Elective:

8. Analyze the motion of a charged particle in uniform electric and magnetic fields [Usage]

5.1.2 CME/Work, Energy, and Conservative Forces (Core Tier1: 1 hr, Core Tier2: 1 hr)

Energy concepts in mechanics: work-energy theorem, kinetic and potential energy, conservation of mechanical energy, and properties of conservative force fields.

Topics:

Core

- Definition of work as a line integral $W = \int \vec{F} \cdot d\vec{r}$
- Work-energy theorem relating total work to change in kinetic energy: $W_{total} = \Delta K$
- Kinetic energy for single particles and systems of particles
- Conservative forces and the path independence of work
- Potential energy function $U(\vec{r})$ and the force relation $\vec{F} = -\nabla U$
- Conservation of total mechanical energy $E = K + U$
- Energy diagrams and interpretation of turning points and equilibrium stability
- Work done by non-conservative forces and mechanisms of energy dissipation
- Potential energy surfaces in Two and Three dimensions

Elective

- Energy considerations for velocity-dependent forces such as magnetic and drag forces

Learning Outcomes:
Core:

1. Calculate the work done by a force along a specified path using line integrals [Usage]
2. Determine whether a given force field is conservative and find its potential energy function [Usage]
3. Apply conservation of mechanical energy to solve for speeds and positions in One-dimensional motion [Usage]
4. Analyze an energy diagram to identify turning points, allowed regions, and equilibrium stability [Assessment]
5. Compute the instantaneous power delivered by a force: $P = \vec{F} \cdot \vec{v}$ [Usage]
6. Estimate energy dissipated by non-conservative forces in a mechanical process [Assessment]
7. Locate and classify critical points such as maxima, minima, and saddle points on a Two-dimensional potential surface [Usage]

Elective:

8. Explain why magnetic forces do no work on charged particles despite changing their velocity direction [Familiarity]

5.1.3 CME/Systems of Particles and Linear Momentum (Core Tier1: 1 hr, Core Tier2: 1 hr)

Dynamics of multi-particle systems: center of mass motion, linear momentum conservation, internal and external forces, and variable mass systems.
Topics:
Core

- Definition of center of mass: $\vec{R}_{CM} = \frac{1}{M} \sum m_i \vec{r}_i$
- Linear momentum of a particle and system: $\vec{p} = m\vec{v}$ and $\vec{P} = M\vec{V}_{CM}$
- Newton's second law for systems: $\vec{F}_{ext} = \frac{d\vec{P}}{dt}$
- Conservation of linear momentum for isolated systems
- Distinction between internal and external forces in a system
- Two-body collisions including elastic, inelastic, and completely inelastic cases
- Coefficient of restitution and its physical meaning in collisions
- Systems with variable mass such as the rocket equation: $M \frac{dV}{dt} = v_{ex} \frac{dM}{dt}$
- Many-particle systems and their center of mass dynamics

Elective

- Continuous mass distributions and their center of mass calculation using integration

Learning Outcomes:
Core:

1. Locate the center of mass for a discrete set of particles in One, Two, and Three dimensions [Usage]
2. Determine the total linear momentum of a system from the motion of its center of mass [Usage]
3. Apply conservation of linear momentum to solve collision problems in One and Two dimensions [Usage]
4. Classify collisions as elastic, inelastic, or completely inelastic based on kinetic energy conservation [Familiarity]
5. Calculate the coefficient of restitution from pre-collision and post-collision velocities [Usage]
6. Solve the rocket equation to determine velocity as a function of ejected mass [Usage]
7. Analyze how internal forces affect individual particles but not the total momentum of a system [Assessment]

Elective:

8. Integrate to find the center of mass of symmetric continuous bodies such as rods, disks, and spheres [Usage]

5.1.4 CME/Rotational Motion and Angular Momentum (Core Tier1: 1 hr, Core Tier2: 1 hr)

Rotational kinematics and dynamics, torque, moment of inertia, rotational energy, and conservation of angular momentum for particles and systems.

Topics:

Core

- Rotational kinematics including angular displacement, velocity, and acceleration (θ, ω, α)
- Torque definition: $\vec{\tau} = \vec{r} \times \vec{F}$ and its magnitude calculation
- Angular momentum of a particle: $\vec{L} = \vec{r} \times \vec{p}$
- Rotational form of Newton's second law: $\vec{\tau}_{net} = \frac{d\vec{L}}{dt}$
- Moment of inertia $I = \sum m_i r_i^2$ and the parallel axis theorem
- Rotational kinetic energy: $K_{rot} = \frac{1}{2} I \omega^2$
- Conservation of angular momentum for isolated systems
- Gyroscopic motion and the phenomenon of precession
- Rolling without slipping dynamics: $v_{CM} = R\omega$

Elective

- Coriolis effect as a manifestation of rotational dynamics in non-inertial frames

Learning Outcomes:

Core:

1. Relate linear and angular kinematic quantities for circular motion [Usage]
2. Calculate torque about a given axis for a force applied at a specific point [Usage]
3. Determine the moment of inertia for simple geometries such as a rod, hoop, disk, or sphere [Usage]
4. Apply conservation of angular momentum to analyze changes in rotational speed when the moment of inertia changes [Assessment]
5. Compute rotational kinetic energy for a rotating rigid body [Usage]
6. Analyze the motion of objects rolling without slipping down an incline [Assessment]
7. Predict the direction of precession for a spinning top or gyroscope [Assessment]

Elective:

8. Explain the Coriolis effect in terms of rotating reference frames and its effect on projectiles [Familiarity]

5.1.5 CME/Rigid Body Dynamics in 3D (Core Tier1: 1 hr, Core Tier2: 1 hr)

Three-dimensional motion of rigid bodies: inertia tensor, principal axes, Euler's equations, and complex rotational motion.

Topics:

Core

- Inertia tensor I_{ij} and its matrix representation
- Principal axes and the diagonalization of the inertia tensor
- Euler's equations for rigid body rotation: $I_1\dot{\omega}_1 = (I_2 - I_3)\omega_2\omega_3 + \tau_1$
- Free rotation of asymmetric tops and stability analysis of rotation axes
- Cases where the angular momentum vector is not parallel to the angular velocity vector
- Torque-free motion and polhodes in rigid body dynamics
- Euler angles (ϕ, θ, ψ) and the description of orientation in space

Elective

- Forced rotation and the phenomenon of nutation
- Collisions between rigid bodies in Three dimensions
- Numerical simulation of complex rigid body dynamics

Learning Outcomes:

Core:

1. Construct the inertia tensor for a system of discrete masses [Usage]
2. Diagonalize a given inertia tensor to find principal moments and principal axes [Usage]
3. Apply Euler's equations to analyze torque-free rotation of symmetric tops [Usage]

4. Interpret why angular momentum and angular velocity vectors are not parallel for asymmetric bodies [Assessment]
5. Compute Euler angles to describe the orientation of a rigid body in space [Usage]

Elective:

6. Simulate numerically the Three-dimensional motion of a rigid body under given torques [Usage]
7. Analyze nutation in a heavy symmetric top [Assessment]

5.1.6 CME/Non-Inertial Reference Frames and Fictitious Forces (Core Tier1: 1 hr, Core Tier2: 1 hr)

Physics in accelerating reference frames: centrifugal, Coriolis, and Euler forces, with applications to Earth's rotation.

Topics:

Core

- Transformations between inertial and accelerating reference frames
- Centrifugal force defined as $F_{cent} = -m\vec{\omega} \times (\vec{\omega} \times \vec{r}')$
- Coriolis force defined as $F_{Cor} = -2m\vec{\omega} \times \vec{v}'$
- Euler force for non-constant rotation: $F_{Euler} = -m\frac{d\vec{\omega}}{dt} \times \vec{r}'$
- Effective gravity on the rotating Earth: $g_{eff} = g - \vec{\omega} \times (\vec{\omega} \times \vec{R})$
- Foucault pendulum as a demonstration of the Earth's rotation
- Coriolis effect on global weather patterns and ocean currents

Elective

- Larmor precession in magnetic fields viewed as a rotating frame effect
- Tidal forces as differential gravitational effects in co-rotating frames
- Connection to General Relativity's equivalence principle

Learning Outcomes:

Core:

1. Transform Newton's second law to a uniformly rotating reference frame [Usage]
2. Calculate centrifugal and Coriolis forces for given positions and velocities in a rotating frame [Usage]
3. Predict the deflection direction of projectiles due to the Earth's rotation through the Coriolis effect [Assessment]
4. Explain the precession of a Foucault pendulum in terms of the Earth's rotation [Familiarity]
5. Compute effective gravitational acceleration at different latitudes on Earth [Usage]

Elective:

6. Analyze Larmor precession using the concept of rotating reference frames [Assessment]
7. Compare tidal forces to centrifugal forces in co-rotating systems [Assessment]

5.1.7 CME/Lagrangian and Hamiltonian Mechanics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Advanced formulations of mechanics: principle of least action, Lagrangian formulation, Hamiltonian mechanics, and canonical transformations.

Topics:

Core

- Principle of least action and Hamilton's principle of stationary action
- Lagrangian definition: $L = T - V$ for conservative systems
- Euler-Lagrange equations of motion: $\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) = \frac{\partial L}{\partial q_i}$
- Generalized coordinates and the treatment of holonomic constraints
- Cyclic coordinates and the associated conservation laws
- Legendre transform from the Lagrangian to the Hamiltonian: $H = \sum p_i \dot{q}_i - L$
- Hamilton's equations: $\dot{q}_i = \frac{\partial H}{\partial p_i}$ and $\dot{p}_i = -\frac{\partial H}{\partial q_i}$

Elective

- Poisson brackets: $\{f, g\} = \sum \left(\frac{\partial f}{\partial q_i}\frac{\partial g}{\partial p_i} - \frac{\partial f}{\partial p_i}\frac{\partial g}{\partial q_i}\right)$
- Canonical transformations and their generating functions

- Action-angle variables for periodic Hamiltonian systems

Learning Outcomes:

Core:

1. Construct the Lagrangian for simple mechanical systems in appropriate generalized coordinates [Usage]
2. Derive equations of motion from the Euler-Lagrange equations [Usage]
3. Identify cyclic coordinates and their associated conserved quantities in a given Lagrangian [Assessment]
4. Formulate the Hamiltonian from a given Lagrangian system [Usage]
5. Solve Hamilton's equations for simple systems such as the harmonic oscillator [Usage]

Elective:

6. Compute Poisson brackets for pairs of dynamical variables [Usage]
7. Apply canonical transformations to simplify the description of Hamiltonian systems [Usage]

5.1.8 CME/Central Force Motion and Orbital Mechanics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Motion under central forces: Kepler's laws, orbital dynamics, gravitational potential, and the two-body problem.

Topics:

Core

- Central forces $\vec{F} = F(r)\hat{r}$ and the conservation of angular momentum
- Reduced mass and the reduction of the two-body problem to a one-body problem
- Effective potential: $U_{eff}(r) = U(r) + \frac{L^2}{2\mu r^2}$ and turning points
- Kepler's Three laws of planetary motion
- Orbit equation: $\frac{1}{r} = \frac{GM\mu^2}{L^2}(1 + e\cos\theta)$ for inverse-square laws
- Orbital elements including semi-major axis, eccentricity, and inclination
- Energy classification of orbits into bound ($E < 0$) and unbound ($E \geq 0$) states

Elective

- Rutherford scattering theory and the concept of cross section
- Precession of orbits in non-Newtonian gravitational potentials
- Restricted Three-body problem and the five Lagrange points

Learning Outcomes:

Core:

1. Reduce a two-body central force problem to an equivalent One-body problem using reduced mass [Usage]
2. Analyze the effective potential to determine turning points and allowed orbit types [Assessment]
3. Derive Kepler's Third law $T^2 \propto a^3$ from the principles of Newtonian gravitation [Usage]
4. Calculate orbital parameters such as period and energy from initial conditions [Usage]
5. Classify an orbit as circular, elliptical, parabolic, or hyperbolic based on total energy [Assessment]

Elective:

6. Compute scattering angles in Rutherford scattering experiments [Usage]
7. Locate the five Lagrange points in the circular restricted Three-body problem [Usage]

5.1.9 CME/Theory of Small Oscillations and Coupled Oscillators (Core Tier1: 1 hr, Core Tier2: 1 hr)

Small oscillations about equilibrium: normal modes, eigenfrequencies, damping, forcing, and coupled systems.

Topics:

Core

- Stability analysis of equilibrium points using potential energy second derivatives
- Small oscillation approximation via Taylor expansion about a stable equilibrium
- Coupled oscillators and the formulation of equations of motion in matrix form
- Normal modes and eigenfrequencies derived from the eigenvalue problem

- Normal coordinates that diagonalize the mechanical system
- Damped oscillators: underdamped, critically damped, and overdamped regimes
- Forced oscillations and the physics of resonance phenomena

Elective

- Continuous systems and the wave equation as a limit of many coupled oscillators
- Nonlinear oscillations and the study of anharmonic effects
- Parametric resonance and the introduction to the Mathieu equation

Learning Outcomes:

Core:

1. Linearize equations of motion about a stable equilibrium point [Usage]
2. Find normal mode frequencies and eigenvectors for systems of Two coupled oscillators [Usage]
3. Solve the damped harmonic oscillator equation for all Three damping regimes [Usage]
4. Analyze the amplitude and phase response of a driven harmonic oscillator near resonance [Assessment]
5. Transform to normal coordinates to decouple the equations of motion for a system [Usage]

Elective:

6. Derive the wave equation from the continuum limit of many coupled oscillators [Usage]
7. Investigate anharmonic corrections to the simple harmonic oscillator potential [Assessment]

5.1.10 CME/Continuum Mechanics Foundations

Introduction to continuous media: stress and strain tensors, elasticity, fluid mechanics, and conservation laws.

Topics:

Elective

- Continuum hypothesis and the definition of density fields
- Stress tensor σ_{ij} and the concept of force per unit area
- Strain tensor ϵ_{ij} as a measure of local deformation
- Hooke's law for isotropic materials: $\sigma_{ij} = \lambda\epsilon_{kk}\delta_{ij} + 2\mu\epsilon_{ij}$
- Navier-Stokes equation for viscous fluids: $\rho\left(\frac{\partial\vec{v}}{\partial t} + \vec{v}\cdot\nabla\vec{v}\right) = -\nabla p + \mu\nabla^2\vec{v} + \vec{f}$
- Bernoulli's principle for inviscid and incompressible fluid flow
- Conservation of mass and the continuity equation: $\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\vec{v}) = 0$
- Conservation of momentum expressed in continuum mechanics form
- Wave propagation in elastic media including P-waves and S-waves
- Viscosity and the rheology of non-Newtonian fluids

Learning Outcomes:

Elective:

1. Interpret the physical meaning of the components of the stress tensor [Familiarity]
2. Calculate strain tensor components for simple deformations [Usage]
3. Apply Hooke's law to relate stress and strain for isotropic elastic materials [Usage]
4. Simplify the Navier-Stokes equation for special cases such as steady or inviscid flow [Usage]
5. Apply Bernoulli's principle to calculate pressure variations in fluid flow [Usage]
6. Derive the continuity equation from the principle of mass conservation [Usage]
7. Analyze different wave propagation modes in elastic solids [Assessment]

5.1.11 CME/Elasticity (Core Tier1: 1 hr, Core Tier2: 1 hr)

Elastic properties of solids under axial, shear, and volumetric load: stress, strain, and scalar elastic moduli.

Topics:

Core

- Definition of stress (force/area) and strain ($\Delta L/L$)
- Young's modulus and Hooke's law for longitudinal strain: $\sigma = E\epsilon$
- Shear modulus (rigidity) and shear strain
- Bulk modulus (compressibility) and volumetric strain
- Elastic limit, plastic region, and fracture point on the stress-strain diagram

Elective

- Elastic energy stored in a deformed material

Learning Outcomes:

Core:

1. Apply the scalar Hooke's law to calculate the deformation of a rod under axial load [Usage]
2. Calculate Young's, shear, or bulk modulus from experimental stress-strain data [Usage]
3. Identify the elastic region, elastic limit, and fracture point on a stress-strain curve [Familiarity]

Elective:

4. Calculate the elastic potential energy stored in a deformed spring or rod [Usage]

5.1.12 CME/Fluids: Hydrostatics, Hydrodynamics, and Viscosity (Core Tier1: 1 hr, Core Tier2: 1 hr)

Statics and dynamics of incompressible fluids: pressure, buoyancy, the continuity and Bernoulli equations, and an introduction to viscous flow.

Topics:

Core

- Density, pressure, and its variation with depth in a fluid at rest: $p = p_0 + \rho g h$
- Pascal's principle and its applications (hydraulic press)
- Archimedes' principle, buoyant force, and flotation
- Pressure measurement with manometers and barometers
- Continuity equation for incompressible flow: $A_1 v_1 = A_2 v_2$
- Bernoulli's equation and energy conservation in ideal flow
- Applications: Venturi tube, Torricelli's theorem, Pitot tube

Elective

- Viscosity and Poiseuille's law for laminar flow in pipes
- Reynolds number and the transition to turbulent flow
- Stokes' law for the drag force on a sphere in a viscous fluid

Learning Outcomes:

Core:

1. Calculate the pressure at a point in a fluid at rest given depth and density [Usage]
2. Apply Archimedes' principle to determine whether an object floats, sinks, or remains in equilibrium [Usage]
3. Apply the continuity and Bernoulli equations to solve flow problems in pipes of varying cross-section [Usage]
4. Interpret manometer and barometer readings to determine absolute and gauge pressures [Assessment]

Elective:

5. Apply Poiseuille's law to calculate the flow rate of a viscous fluid in a cylindrical tube [Usage]
6. Estimate the Reynolds number of a flow and predict whether it is laminar or turbulent [Assessment]

5.1.13 CME/Vibrations and Waves (Core Tier1: 1 hr, Core Tier2: 1 hr)

Simple harmonic motion, damped and forced oscillations, and the propagation, superposition, and resonance of mechanical waves.

Topics:

Core

- Simple harmonic motion (SHM): equation of motion, position, velocity, and acceleration
- Oscillating systems: spring-mass and simple/physical pendulum
- Energy in simple harmonic motion
- Damped and forced oscillations; introduction to resonance
- Transverse and longitudinal waves; propagation speed
- Wave equation and speed of a wave on a taut string: $v = \sqrt{T/\mu}$
- Superposition principle and standing waves
- Sound waves, intensity, and sound intensity level (decibels)

Elective

- Doppler effect for sound waves
- Beats and interference of waves of nearby frequencies

Learning Outcomes:
Core:

1. Derive the equation of motion and the period of a spring-mass system and a simple pendulum [Usage]
2. Calculate the kinetic, potential, and total energy of a simple harmonic oscillator at any instant [Usage]
3. Analyze the amplitude response of a forced oscillator near the resonance frequency [Assessment]
4. Calculate the propagation speed of a wave on a string from the tension and linear density [Usage]
5. Determine the frequencies and wavelengths of the normal modes of a standing wave on a fixed string [Usage]
6. Calculate the sound intensity level in decibels from the intensity of a sound wave [Usage]

Elective:

7. Apply the Doppler effect formula for a moving source and/or observer [Usage]

5.1.14 CME/Transport Phenomena (Core Tier1: 1 hr, Core Tier2: 1 hr)

Introduction to transport phenomena in continuous media: heat conduction and mass diffusion as gradient-driven flux processes.
Topics:
Core

- Heat conduction and Fourier's law: $\vec{q} = -k\nabla T$
- Thermal conductivity of materials and thermal resistance in composite walls
- Molecular diffusion and Fick's first law: $\vec{J} = -D\nabla C$

Elective

- Analogy between heat, mass, and momentum transport (Prandtl and Schmidt numbers)

Learning Outcomes:
Core:

1. Apply Fourier's law to calculate the heat flux through a plane or composite wall [Usage]
2. Apply Fick's first law to calculate the diffusive flux of a species in a concentration gradient [Usage]

Elective:

3. Identify mathematical analogies between heat, mass, and momentum transport processes [Familiarity]

5.2 Electromagnetism (ELM)

Electromagnetism studies the behavior of electric charges at rest and in motion, the properties of electric and magnetic fields, and their interaction as described by Maxwell's equations. It covers electrostatics, magnetostatics, electrodynamics, and the propagation of electromagnetic waves in vacuum and matter.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.2.1 Electrostatics: Electric Field and Gauss's Law (Pg.. 219)	1	1
⭐ 5.2.2 Electric Potential and Poisson/Laplace Equations (Pg.. 219)	1	1
⭐ 5.2.3 Electrostatic Energy and Dielectrics (Pg.. 220)	1	1
⭐ 5.2.4 Magnetostatics: Biot-Savart and Ampere's Laws (Pg.. 220)	1	1
⭐ 5.2.5 Magnetic Properties of Matter (Pg.. 221)	1	1
⭐ 5.2.6 Electrodynamics: Faraday's Law and Induction (Pg.. 221)	1	1
⭐ 5.2.7 Maxwell's Equations and Gauge Transformations (Pg.. 222)	1	1
⭐ 5.2.8 Electromagnetic Waves and Propagation in Media (Pg.. 222)	1	1
⭐ 5.2.9 Radiation Theory and Dipole Radiation (Pg.. 223)	1	1
⭐ 5.2.10 Special Relativity in Electromagnetism (Pg.. 223)	1	1
⭐ 5.2.11 Electric Current, Circuits, and Kirchhoff's Laws (Pg.. 223)	1	1

Table 5.2: List of KUs in the Electromagnetism area.

5.2.1 ELM/Electrostatics: Electric Field and Gauss's Law (Core Tier1: 1 hr, Core Tier2: 1 hr)

Analysis of stationary charges, the vector nature of the electric field, and the application of Gauss's Law to symmetric systems.

Topics:

Core

- Coulomb's law for point charges and the principle of linear superposition
- The electric field vector $\vec{E}$ and calculation for continuous charge distributions
- Gauss's Law in integral form and its relation to enclosed charge
- Differential form of Gauss's Law: $\nabla \cdot \vec{E} = \rho/\epsilon_0$
- Definition of electric flux through open and closed surfaces
- Applications of Gauss's Law to spheres and shells
- Applications of Gauss's Law to infinite lines and cylinders
- Applications of Gauss's Law to infinite sheets and parallel plates
- The Divergence Theorem applied to electrostatic field vectors
- Point charges represented by the Dirac delta function in 3D

Learning Outcomes:

Core:

1. Calculate the net electrostatic force on a charge from a discrete distribution of charges [Usage]
2. Apply Gauss's Law to find the electric field for systems with spherical or cylindrical symmetry [Usage]
3. Evaluate the electric flux through a given surface for a non-uniform electric field [Assessment]
4. Determine the volume charge density from a given expression of the electric field using the differential form of Gauss's Law [Usage]
5. Use the Dirac delta function to formally express the charge density of a point charge at the origin [Usage]

5.2.2 ELM/Electric Potential and Poisson/Laplace Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

The scalar potential formulation of electrostatics and the solution of boundary value problems.

Topics:

Core

- Definition of the scalar potential V and the relation $\vec{E} = -\nabla V$
- The conservative nature of the electrostatic field: $\nabla \times \vec{E} = 0$
- Calculating potential from continuous line, surface, and volume charge distributions
- Properties of equipotential surfaces and their relation to field lines
- Derivation of Poisson's equation $\nabla^2 V = -\rho/\epsilon_0$
- Laplace's equation $\nabla^2 V = 0$ in charge-free regions

- The first and second uniqueness theorems for electrostatic solutions
- The method of images for planar and spherical grounded conductors
- Solution of Laplace's equation via separation of variables in Cartesian and Spherical coordinates

- Expansion of potential in terms of Monopole, Dipole, and Quadrupole moments

Learning Outcomes:

Core:

1. Compute the potential energy of a configuration of point charges [Usage]
2. Derive the electric field vector from a given scalar potential function [Usage]
3. Solve Laplace's equation for a rectangular pipe with given boundary conditions [Assessment]
4. Apply the method of images to find the force between a point charge and an infinite conducting plane [Usage]
5. Analyze the dominant term of the multipole expansion for a charge distribution at large distances [Assessment]

5.2.3 ELM/Electrostatic Energy and Dielectrics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Study of energy storage in fields, capacitors, and the macroscopic behavior of dielectric materials.

Topics:

Core

- Work required to assemble a distribution of charges
- Energy density in the electric field: $u = \frac{1}{2}\epsilon_0 E^2$
- Definition of capacitance $C = Q/V$ and calculation for standard geometries
- Atomic polarization and the polarization vector $\vec{P}$
- Bound surface and volume charges in polarized media
- The electric displacement vector $\vec{D}$ and Gauss's Law in matter
- Linear dielectrics, electric susceptibility χ_e, and relative permittivity ϵ_r
- Interface boundary conditions for $\vec{E}$ and $\vec{D}$ vectors
- Energy in dielectric systems: $W = \frac{1}{2}\int \vec{D} \cdot \vec{E} d\tau$

Elective

- Microscopic description of dielectrics and the Clausius-Mossotti relation

Learning Outcomes:

Core:

1. Calculate the capacitance of a parallel-plate capacitor filled with a linear dielectric [Usage]
2. Evaluate the bound charge distribution for a dielectric sphere with uniform polarization [Assessment]
3. Solve for the electric field on both sides of a dielectric interface using boundary conditions [Usage]
4. Analyze the force exerted on a dielectric slab being pulled into a capacitor [Assessment]

5.2.4 ELM/Magnetostatics: Biot-Savart and Ampere's Laws (Core Tier1: 1 hr, Core Tier2: 1 hr)

The physics of steady electric currents, magnetic fields, and the forces on moving charges.

Topics:

Core

- Lorentz force law for point charges: $\vec{F} = q(\vec{E} + \vec{v} \times \vec{B})$
- Magnetic force on current-carrying wires and loops
- Biot-Savart Law for the magnetic field of a current element
- Ampère's Law in integral form for high-symmetry currents
- Differential form of Ampère's Law: $\nabla \times \vec{B} = \mu_0 \vec{J}$
- The solenoidal nature of $\vec{B}$: $\nabla \cdot \vec{B} = 0$
- The magnetic vector potential $\vec{A}$ and its relation $\vec{B} = \nabla \times \vec{A}$
- The Coulomb gauge $\nabla \cdot \vec{A} = 0$ in magnetostatics
- Magnetic dipole moment $\vec{m}$ of a current loop

- Multipole expansion of the vector potential and the dipole term

Learning Outcomes:

Core:

1. Calculate the magnetic field of an infinite straight wire using Ampère's Law [Usage]
2. Compute the magnetic field on the axis of a circular current loop using the Biot-Savart Law [Usage]
3. Verify that a given magnetic field satisfies the solenoidal condition $\nabla \cdot \vec{B} = 0$ [Assessment]
4. Determine the magnetic vector potential for a uniform magnetic field [Usage]
5. Analyze the torque and force on a magnetic dipole in a non-uniform magnetic field [Assessment]

5.2.5 ELM/Magnetic Properties of Matter (Core Tier1: 1 hr, Core Tier2: 1 hr)

Magnetization of materials, bound currents, and the classification of magnetic media.

Topics:

Core

- The magnetization vector $\vec{M}$ as magnetic dipole moment per unit volume
- Bound surface currents $\vec{K}_b$ and volume currents $\vec{J}_b$
- The magnetic field intensity $\vec{H}$ and Ampère's Law in matter
- Physical principles of diamagnetism and paramagnetism
- Ferromagnetism, magnetic domains, and the hysteresis loop
- Interface boundary conditions for $\vec{B}$ and $\vec{H}$ vectors
- Linear magnetic media and magnetic permeability μ

Elective

- The magnetic scalar potential in current-free regions
- Introduction to perfect diamagnetism and the Meissner effect

Learning Outcomes:

Core:

1. Find the bound current distribution for a long cylinder with uniform longitudinal magnetization [Usage]
2. Classify materials as diamagnetic, paramagnetic, or ferromagnetic based on their susceptibility [Assessment]
3. Solve for the $\vec{B}$ and $\vec{H}$ fields in a linear magnetic sphere placed in a uniform field [Usage]
4. Analyze the energy loss per cycle in a ferromagnetic material from its hysteresis curve [Assessment]

5.2.6 ELM/Electrodynamics: Faraday's Law and Induction (Core Tier1: 1 hr, Core Tier2: 1 hr)

Time-dependent magnetic fields and the generation of electromotive forces.

Topics:

Core

- Faraday's Law of induction and Lenz's Law for direction
- The flux rule for electromotive force (EMF) in moving circuits
- The non-conservative nature of the induced electric field: $\nabla \times \vec{E} = -\partial \vec{B}/\partial t$
- Mutual inductance M between two circuits
- Self-inductance L and the back-EMF in inductors
- Energy stored in the magnetic field: $W = \frac{1}{2\mu_0} \int B^2 d\tau$
- Formation of eddy currents and magnetic braking
- Formalism of the inductance matrix for multi-circuit systems
- The Neumann formula for calculating mutual inductance

Learning Outcomes:

Core:

1. Calculate the induced EMF in a rotating loop within a constant magnetic field [Usage]
2. Apply Lenz's Law to determine the direction of the induced current in a given physical scenario [Assessment]
3. Compute the total magnetic energy stored in a long solenoid [Usage]

4. Evaluate the mutual inductance between a long wire and a neighboring rectangular loop [Assessment]

5.2.7 ELM/Maxwell's Equations and Gauge Transformations (Core Tier1: 1 hr, Core Tier2: 1 hr)

The unification of electricity and magnetism and the mathematical structure of the field equations.
Topics:
Core

- Maxwell's correction to Ampère's Law and the displacement current $\vec{J}_d$
- The complete set of four Maxwell's equations in vacuum and matter
- Conservation of charge and the continuity equation: $\nabla \cdot \vec{J} + \partial\rho/\partial t = 0$
- Formulation of Maxwell's equations in terms of V and $\vec{A}$
- Concept of gauge invariance in electrodynamics
- The Lorentz gauge condition: $\nabla \cdot \vec{A} + \mu_0\epsilon_0\partial V/\partial t = 0$
- Poynting's theorem and the conservation of energy in fields
- The Poynting vector $\vec{S} = \vec{E} \times \vec{H}$ as energy flux density
- Conservation of momentum and the Maxwell stress tensor $\vec{T}$

Elective

- Retarded potentials and Jefimenko's equations

Learning Outcomes:
Core:

1. Calculate the displacement current in the gap of a charging capacitor [Usage]
2. State all four Maxwell equations in both integral and differential forms [Familiarity]
3. Apply a gauge transformation to a given set of potentials while keeping the fields invariant [Usage]
4. Analyze the energy flow and power dissipation in a resistor using the Poynting vector [Assessment]
5. Compute the radiation pressure exerted by a plane wave on a perfectly reflecting surface [Usage]

5.2.8 ELM/Electromagnetic Waves and Propagation in Media (Core Tier1: 1 hr, Core Tier2: 1 hr)

The propagation of electromagnetic disturbances in vacuum, dielectrics, and conductors.
Topics:
Core

- Derivation of the electromagnetic wave equation in vacuum
- Monochromatic plane wave solutions and their properties
- Polarization states: linear, circular, and elliptical
- Energy density, intensity, and momentum of plane waves
- Reflection and refraction at normal incidence on an interface
- Fresnel equations and Brewster's angle for oblique incidence
- Wave propagation in conducting media and the skin depth δ
- Frequency dependence of permittivity and the physics of dispersion
- Phase velocity versus group velocity in dispersive media

Elective

- TE and TM modes in rectangular waveguides

Learning Outcomes:
Core:

1. Derive the dispersion relation and the speed of light from Maxwell's equations [Usage]
2. Determine the polarization state of a wave from its vector components [Assessment]
3. Apply Fresnel's equations to find the reflection coefficient for TE and TM polarizations [Usage]
4. Analyze how the skin depth varies with frequency for a given metallic conductor [Assessment]
5. Calculate the Brewster angle for an interface between air and glass [Usage]

5.2.9 ELM/Radiation Theory and Dipole Radiation (Core Tier1: 1 hr, Core Tier2: 1 hr)

Generation of electromagnetic radiation by accelerating charges and oscillating dipoles.
Topics:
Core
- Radiation from an oscillating electric dipole
- Radiation from an oscillating magnetic dipole
- Concept of radiation resistance in antennas
- The Larmor formula for the total power radiated by an accelerating point charge

Elective
- The Liénard-Wiechert potentials for a moving point charge
- Introduction to Bremsstrahlung and Synchrotron radiation

Learning Outcomes:
Core:
1. Calculate the total power radiated by an oscillating electric dipole [Usage]
2. Analyze the angular distribution of radiation from an electric dipole [Assessment]
3. Apply the Larmor formula to determine the power loss of an electron in a circular orbit [Usage]

5.2.10 ELM/Special Relativity in Electromagnetism (Core Tier1: 1 hr, Core Tier2: 1 hr)

The relativistic covariance of Maxwell's equations and the transformation of fields.
Topics:
Core
- The four-current density J^μ
- The electromagnetic field tensor $F^{\mu\nu}$ and its properties
- Lorentz transformation of the $\vec{E}$ and $\vec{B}$ fields
- Relativistic invariants of the electromagnetic field

Elective
- The dual field tensor and its role in Maxwell's equations

Learning Outcomes:
Core:
1. Transform the electric and magnetic fields from one inertial frame to another [Usage]
2. Verify the covariance of Maxwell's equations using four-vector notation [Assessment]
3. Explain how a purely electric field in one frame can appear as a magnetic field in another [Familiarity]

5.2.11 ELM/Electric Current, Circuits, and Kirchhoff's Laws (Core Tier1: 1 hr, Core Tier2: 1 hr)

Study of electric charge flow, resistance, dissipated power, and DC circuit analysis using Kirchhoff's laws.
Topics:
Core
- Definition of electric current, current density, and the continuity equation
- Ohm's law, resistivity, and temperature dependence of resistance
- Power dissipated in a resistor: Joule heating and $P = I^2R$
- Kirchhoff's Current Law (KCL): charge conservation at a node
- Kirchhoff's Voltage Law (KVL): energy conservation around a loop
- Series and parallel circuit analysis: equivalent resistances
- Electromotive force (EMF) sources and internal resistance of batteries
- RC circuits: charging and discharging, time constant $\tau = RC$

Learning Outcomes:
Core:
1. Apply Kirchhoff's laws to solve DC circuits with multiple loops and nodes [Usage]

2. Calculate power dissipated and energy consumed in resistive circuit elements [Assessment]
3. Analyze the transient behavior of an RC circuit and interpret its time constant [Usage]

5.3 Thermodynamics and Statistical Mechanics (TSM)

Thermodynamics and Statistical Mechanics bridge the gap between microscopic properties of individual particles and macroscopic observables of bulk matter. This area covers the laws of energy transfer, entropy, equilibrium states, and the statistical ensembles used to describe many-body systems in both classical and quantum regimes.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.3.1 Laws of Thermodynamics and State Functions (Pg.. 224)	1	1
⭐ 5.3.2 Kinetic Theory of Gases (Pg.. 224)	1	1
⭐ 5.3.3 Heat Engines, Entropy, and the Second Law (Pg.. 225)	1	1
⭐ 5.3.4 Thermodynamic Potentials and Maxwell Relations (Pg.. 225)	1	1
⭐ 5.3.5 Phase Transitions and Critical Phenomena (Pg.. 226)	1	1
⭐ 5.3.6 Microcanonical, Canonical, and Grand Canonical Ensembles (Pg.. 226)	1	1
⭐ 5.3.7 Quantum Statistics: Fermi-Dirac and Bose-Einstein (Pg.. 226)	1	1
⭐ 5.3.8 Blackbody Radiation and Phonons (Pg.. 227)	1	1
⭐ 5.3.9 Non-Equilibrium Statistical Mechanics (Pg.. 227)	1	1
⭐ 5.3.10 Applications to Biological and Complex Systems (Pg.. 227)	1	1

Table 5.3: List of KUs in the Thermodynamics and Statistical Mechanics area.

5.3.1 TSM/Laws of Thermodynamics and State Functions (Core Tier1: 1 hr, Core Tier2: 1 hr)

Fundamental principles governing energy, heat, and work, and the mathematical definition of state variables.

Topics:

Core

- Zeroth Law of Thermodynamics and the definition of temperature
- First Law of Thermodynamics: $dU = dQ - dW$ and internal energy conservation
- Distinction between state functions and path-dependent variables
- Mathematical properties of exact and inexact differentials in thermodynamics
- Internal energy of ideal gases and its dependence on temperature
- Definition of enthalpy $H = U + PV$ and its physical significance
- Heat capacities at constant volume C_V and constant pressure C_P
- Adiabatic processes and the relation $PV^\gamma = \text{constant}$
- Real gas expansion and the Joule-Thomson coefficient

Learning Outcomes:

Core:

1. Calculate the work done by a gas during isothermal and adiabatic expansions [Usage]
2. Evaluate the change in internal energy for a cyclic process using the First Law [Assessment]
3. Apply Mayer's relation to find the difference between C_P and C_V for an ideal gas [Usage]

5.3.2 TSM/Kinetic Theory of Gases (Core Tier1: 1 hr, Core Tier2: 1 hr)

Microscopic model of gases based on molecular motion and collisions.

Topics:

Core

- Derivation of macroscopic pressure from molecular momentum transfer
- Maxwell-Boltzmann speed distribution and its temperature dependence
- Equipartition theorem and degrees of freedom for monatomic and diatomic gases
- Root mean square, average, and most probable molecular speeds

- Mean free path λ and its relation to molecular cross-section and density
- Transport phenomena: viscosity, thermal conductivity, and diffusion in gases
- Molecular effusion and Graham's law of effusion
- Van der Waals equation of state for real gases: $(P + a/v^2)(v - b) = RT$

Learning Outcomes:
Core:

1. Compute the rms velocity of nitrogen molecules at room temperature [Usage]
2. Interpret the shift in the Maxwell-Boltzmann distribution peak as temperature increases [Assessment]
3. Calculate the mean free path of particles in a high-vacuum chamber [Usage]

5.3.3 TSM/Heat Engines, Entropy, and the Second Law (Core Tier1: 1 hr, Core Tier2: 1 hr)

Analysis of thermal efficiency and the statistical/thermodynamic definition of entropy.
Topics:
Core

- Kelvin-Planck and Clausius statements of the Second Law
- The Carnot cycle and the maximum theoretical efficiency $\eta = 1 - T_c/T_h$
- Definition of the absolute thermodynamic temperature scale
- Clausius inequality and the definition of entropy $dS \geq dQ/T$
- Principle of the increase of entropy in isolated systems
- Analysis of thermodynamic cycles using Temperature-Entropy $(T - S)$ diagrams
- Boltzmann's entropy formula: $S = k_B \ln \Omega$
- Entropy of mixing and the Gibbs paradox
- Third Law of Thermodynamics and the unattainability of absolute zero

Learning Outcomes:
Core:

1. Calculate the maximum efficiency of a heat engine operating between two given temperatures [Usage]
2. Evaluate the change in entropy for an irreversible process like the mixing of two liquids [Assessment]
3. Compute the entropy of a system of ten non-interacting spins [Usage]

5.3.4 TSM/Thermodynamic Potentials and Maxwell Relations (Core Tier1: 1 hr, Core Tier2: 1 hr)

The Legendre transformations and symmetry relations between thermodynamic variables.
Topics:
Core

- Mathematical basis of Legendre transformations for thermodynamic potentials
- Gibbs free energy $G = H - TS$ and its role in chemical equilibrium
- Helmholtz free energy $A = U - TS$ and its relation to maximum work
- Derivation of the four primary Maxwell relations from exact differentials
- The Gibbs-Duhem equation and its constraints on intensive variables
- Use of thermodynamic squares to recall relationships between potentials
- Definition of chemical potential μ and particle exchange equilibrium

Learning Outcomes:
Core:

1. Apply a Maxwell relation to simplify the calculation of internal energy changes [Usage]
2. Determine the appropriate thermodynamic potential to minimize for a system at constant P and T [Assessment]
3. Derive a Maxwell relation from the Helmholtz free energy differential [Usage]

5.3.5 TSM/Phase Transitions and Critical Phenomena (Core Tier1: 1 hr, Core Tier2: 1 hr)

Study of matter changing states and the behavior near critical points.
Topics:
Core

- $P-T$ and $P-V$ phase diagrams for single-component systems
- The Clapeyron equation for phase boundary slopes
- Clausius-Clapeyron equation for liquid-vapor transitions
- Ehrenfest classification of first-order and second-order phase transitions
- Latent heat of fusion, vaporization, and sublimation
- Critical temperature, pressure, and opalescence

Elective

- Introduction to Landau theory of phase transitions and order parameters

Learning Outcomes:
Core:

1. Estimate the change in boiling point of water at high altitudes using the Clausius-Clapeyron equation [Usage]
2. Classify a given transition as first-order or second-order based on heat capacity discontinuities [Assessment]
3. Determine critical pressure and temperature from the van der Waals equation [Usage]

5.3.6 TSM/Microcanonical, Canonical, and Grand Canonical Ensembles (Core Tier1: 1 hr, Core Tier2: 1 hr)

Statistical frameworks for calculating macroscopic properties from microstate probabilities.
Topics:
Core

- Phase space, cells, and the counting of microstates
- The Microcanonical Ensemble (N, V, E): Equal a priori probability postulate
- The Canonical Ensemble (N, V, T) and the Boltzmann factor $e^{-\beta E}$
- The partition function Z and its relation to Helmholtz free energy $A = -k_B T \ln Z$
- The Grand Canonical Ensemble (μ, V, T) and the Gibbs factor
- Energy and particle number fluctuations in ensembles
- Gibbs entropy formula and the relationship to information theory

Learning Outcomes:
Core:

1. Calculate the partition function for a two-level system and derive its average energy [Usage]
2. Evaluate pressure and entropy of a system starting from its partition function [Assessment]
3. Compute the grand partition function for a system of non-interacting particles [Usage]

5.3.7 TSM/Quantum Statistics: Fermi-Dirac and Bose-Einstein (Core Tier1: 1 hr, Core Tier2: 1 hr)

Statistical behavior of identical particles governed by the Pauli exclusion principle or Bose symmetry.
Topics:
Core

- Distinguishable vs. indistinguishable particles and wave function symmetry
- Fermi-Dirac distribution for fermions and the Fermi level ϵ_F
- Bose-Einstein distribution for bosons
- Density of states $g(\epsilon)$ in ThreeD momentum space
- Properties of the degenerate Fermi gas and electron degeneracy pressure
- Phenomenon of Bose-Einstein Condensation (BEC) at low temperatures
- The Maxwell-Boltzmann limit of quantum statistics at high temperatures

Learning Outcomes:
Core:

1. Calculate the Fermi energy for a given density of free electrons in a metal [Usage]
2. Analyze the conditions of temperature and density required to achieve Bose-Einstein condensation [Assessment]
3. Determine the pressure of a completely degenerate Fermi gas at $T = 0$ [Usage]

5.3.8 TSM/Blackbody Radiation and Phonons (Core Tier1: 1 hr, Core Tier2: 1 hr)

Application of quantum statistics to photon and lattice vibration gases.
Topics:
Core
- Derivation of Planck's law for the spectral energy density of blackbody radiation
- Stefan-Boltzmann law for total radiated power: $P \propto T^4$
- Wien's displacement law for the peak wavelength of emission
- Einstein model of specific heat based on independent oscillators
- Debye model of solids and the T^3 law for low-temperature heat capacity
- Phonons as quantized lattice vibrations and their density of states

Learning Outcomes:
Core:
1. Calculate the surface temperature of a star given its peak emission wavelength [Usage]
2. Compare the Einstein and Debye models in explaining the experimental specific heat of solids [Assessment]
3. Estimate the Debye frequency from the sound speed of a solid [Usage]

5.3.9 TSM/Non-Equilibrium Statistical Mechanics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Study of systems approaching equilibrium and transport coefficients.
Topics:
Core
- Einstein's theory of Brownian motion and the diffusion coefficient
- Langevin equation for stochastic dynamics
- Introduction to the Fokker-Planck equation for probability density evolution
- The Fluctuation-Dissipation Theorem and its significance

Elective
- The Boltzmann transport equation and the relaxation time approximation

Learning Outcomes:
Core:
1. Calculate the diffusion constant for a spherical particle in a viscous fluid [Usage]
2. Analyze the effect of random forces on the mean square displacement of a particle [Assessment]
3. Apply the Einstein relation to find the mobility from the diffusion coefficient [Usage]

5.3.10 TSM/Applications to Biological and Complex Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Thermodynamic modeling of macromolecules, membranes, and information-rich systems.
Topics:
Core
- Statistical mechanics of polymer chains: Freely jointed and worm-like models
- Thermodynamics of protein folding and DNA denaturation
- Statistical mechanics of ion channels and membrane potentials

Elective
- Shannon entropy and its connection to physical entropy
- Introduction to self-organization in systems far from equilibrium

Learning Outcomes:
Core:

1. Model the force-extension curve of a DNA molecule using polymer statistics [Usage]
2. Evaluate the stability of a protein fold based on the free energy landscape [Assessment]
3. Estimate the surface tension of a lipid bilayer from its elastic moduli [Usage]

5.4 Quantum Mechanics (QME)

Quantum Mechanics provides the fundamental framework for understanding the behavior of matter and energy at the scale of atoms and subatomic particles. It replaces classical determinism with probabilistic wave functions, operators in Hilbert space, and the principles of superposition and entanglement.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.4.1 Origins of Quantum Theory and Wave-Particle Duality (Pg.. 228)	1	1
⭐ 5.4.2 The Schrödinger Equation and Wave Functions (Pg.. 228)	1	1
⭐ 5.4.3 Formalism: Hilbert Space and Dirac Notation (Pg.. 229)	1	1
⭐ 5.4.4 One-Dimensional Potentials (Wells, Barriers, Tunneling) (Pg.. 229)	1	1
⭐ 5.4.5 The Quantum Harmonic Oscillator (Pg.. 229)	1	1
⭐ 5.4.6 Angular Momentum and Spin Physics (Pg.. 230)	1	1
⭐ 5.4.7 The Hydrogen Atom and Central Potentials (Pg.. 230)	1	1
⭐ 5.4.8 Time-Independent and Time-Dependent Perturbation Theory (Pg.. 230)	1	1
⭐ 5.4.9 Identical Particles and Exchange Symmetry (Pg.. 231)	1	1
⭐ 5.4.10 Scattering Theory and Cross Sections (Pg.. 231)	1	1

Table 5.4: List of KUs in the Quantum Mechanics area.

5.4.1 QME/Origins of Quantum Theory and Wave-Particle Duality (Core Tier1: 1 hr, Core Tier2: 1 hr)

Historical foundations and the transition from classical to quantum physics.
Topics:
Core

- Blackbody radiation and Planck's quantization of energy
- The photoelectric effect and Einstein's photon hypothesis
- Compton scattering and the particle-like behavior of light
- The Bohr model and the quantization of atomic orbits
- De Broglie hypothesis of matter waves and the relation $\lambda = h/p$
- Davisson-Germer experiment and electron diffraction
- Heisenberg Uncertainty Principle: $\Delta x \Delta p \geq \hbar/2$
- Bohr's complementarity principle and the nature of measurement

Learning Outcomes:
Core:

1. Calculate the energy and momentum of a photon given its wavelength [Usage]
2. Evaluate the minimum uncertainty in position for a particle with a known momentum spread [Assessment]
3. Explain the outcomes of the double-slit experiment for both light and electrons [Familiarity]

5.4.2 QME/The Schrödinger Equation and Wave Functions (Core Tier1: 1 hr, Core Tier2: 1 hr)

The fundamental equation of motion and the statistical interpretation of the wave function.
Topics:
Core

- The time-dependent Schrödinger equation: $i\hbar\frac{\partial \Psi}{\partial t} = \hat{H}\Psi$
- The time-independent Schrödinger equation and energy eigenstates
- Born's statistical interpretation of the wave function and probability density

- The normalization condition $\int |\Psi|^2 dx = 1$
- Calculation of expectation values for position and momentum
- Probability current J and the continuity equation
- Ehrenfest's theorem relating quantum averages to classical laws

Learning Outcomes:

Core:

1. Normalize a given mathematical wave function for a particle in space [Usage]
2. Compute the expectation value of kinetic energy for a Gaussian wave packet [Usage]
3. Verify that the total probability is conserved in time using the Schrödinger equation [Assessment]

5.4.3 QME/Formalism: Hilbert Space and Dirac Notation (Core Tier1: 1 hr, Core Tier2: 1 hr)

The rigorous mathematical structure of quantum mechanics using linear algebra.

Topics:

Core

- State vectors as elements of a complex Hilbert space
- Dirac notation: Brackets, kets, and inner products
- Hermitian operators and their role as physical observables
- Eigenvalue equations and the orthonormality of eigenstates
- The completeness relation and projection operators
- Commutators and the generalized uncertainty principle
- Matrix representation of operators in a discrete basis
- Unitary transformations and change of basis in Hilbert space

Learning Outcomes:

Core:

1. Apply Dirac notation to calculate the inner product and probability amplitudes [Usage]
2. Verify if a given operator is Hermitian by examining its matrix elements [Assessment]
3. Solve the eigenvalue problem for a Two-by-Two Hamiltonian matrix [Usage]

5.4.4 QME/One-Dimensional Potentials (Wells, Barriers, Tunneling) (Core Tier1: 1 hr, Core Tier2: 1 hr)

Solvable models in OneD that demonstrate key quantum effects like bound states and tunneling.

Topics:

Core

- The infinite square well: Energy levels and wave functions
- The finite square well and the existence of bound states
- Scattering at a potential step: Reflection and transmission coefficients
- Quantum tunneling through a rectangular potential barrier
- Bound states and scattering in a Dirac delta function potential
- The free particle solution and the spreading of wave packets

Learning Outcomes:

Core:

1. Calculate the ground state energy for an electron in an infinite square well [Usage]
2. Evaluate the transmission probability for a particle tunneling through a thin barrier [Assessment]
3. Sketch the qualitative behavior of wave functions for various potential profiles [Usage]

5.4.5 QME/The Quantum Harmonic Oscillator (Core Tier1: 1 hr, Core Tier2: 1 hr)

The most important model in physics, solved via algebraic and analytic methods.

Topics:

Core

- The Hamiltonian of the harmonic oscillator and its classical limit
- Algebraic method: Raising and lowering (creation and annihilation) operators

- Quantized energy levels: $E_n = (n + 1/2)\hbar\omega$
- Physical significance of the zero-point energy
- Analytic method: Power series solution and Hermite polynomials

Elective

- Introduction to coherent states and their time evolution

Learning Outcomes:

Core:

1. Apply ladder operators to find the first excited state from the ground state [Usage]
2. Analyze the probability of finding an oscillator particle in the classically forbidden region [Assessment]
3. Calculate the zero-point energy for a harmonic oscillator with a given spring constant [Usage]

5.4.6 QME/Angular Momentum and Spin Physics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Rotational symmetry and the intrinsic angular momentum of particles.

Topics:

Core

- Orbital angular momentum operators and commutation relations
- Eigenfunctions of angular momentum: Spherical harmonics Y_l^m
- Spin One-Half systems and the Stern-Gerlach experiment
- Algebra of the Pauli spin matrices
- Addition of angular momentum and Clebsch-Gordan coefficients
- Spin precession in a uniform magnetic field

Learning Outcomes:

Core:

1. Calculate the expectation value of S_z for a given spinor state [Usage]
2. Evaluate the possible values of total angular momentum for two interacting spins [Assessment]
3. Apply Pauli matrix algebra to compute the commutation relations for spin operators [Usage]

5.4.7 QME/The Hydrogen Atom and Central Potentials (Core Tier1: 1 hr, Core Tier2: 1 hr)

The quantum mechanical solution for the simplest atom and motion in spherical symmetry.

Topics:

Core

- Reduction of the 3D Schrödinger equation to the radial equation
- Centrifugal term and the effective potential
- Energy spectrum of the hydrogen atom and the Rydberg constant
- Principal, azimuthal, and magnetic quantum numbers
- Structure of the radial wave functions and Laguerre polynomials
- Atomic shell structure and the shapes of s, p, d orbitals

Learning Outcomes:

Core:

1. Determine the most probable radius for an electron in the ground state of hydrogen [Usage]
2. Analyze the degeneracy of energy levels in the hydrogen atom neglecting spin [Assessment]
3. Compute the Rydberg constant from the ground state energy of hydrogen [Usage]

5.4.8 QME/Time-Independent and Time-Dependent Perturbation Theory (Core Tier1: 1 hr, Core Tier2: 1 hr)

Approximation methods for systems that cannot be solved exactly.

Topics:

Core

- First-order energy and state corrections for non-degenerate systems
- Second-order energy corrections and their significance

- Degenerate perturbation theory and the lifting of degeneracies
- Fine structure of hydrogen: Relativistic and spin-orbit corrections
- The Zeeman effect: Atoms in external magnetic fields
- Time-dependent perturbation theory and transition amplitudes
- Fermi's Golden Rule for transition rates

Elective

- The variational principle for estimating ground state energies
- The WKB approximation for slowly varying potentials

Learning Outcomes:

Core:

1. Calculate the first-order energy shift for an oscillator with a small anharmonic perturbation [Usage]
2. Evaluate the transition rate between two states under a harmonic perturbation using Fermi's Golden Rule [Assessment]
3. Apply the variational method to estimate the ground state energy of a delta potential well [Usage]

5.4.9 QME/Identical Particles and Exchange Symmetry (Core Tier1: 1 hr, Core Tier2: 1 hr)

The physics of systems containing multiple indistinguishable particles.

Topics:

Core

- The exchange operator and symmetry/antisymmetry of states
- Distinction between Bosons and Fermions
- The Pauli exclusion principle and Slater determinants
- The Helium atom and the role of exchange energy
- Singlet and triplet spin states for Two-electron systems

Learning Outcomes:

Core:

1. State the Pauli exclusion principle and its impact on the periodic table [Familiarity]
2. Construct a Slater determinant for a system of three non-interacting fermions [Usage]
3. Determine the sign of the exchange energy for two electrons in a triplet state [Assessment]

5.4.10 QME/Scattering Theory and Cross Sections (Core Tier1: 1 hr, Core Tier2: 1 hr)

Formalism for describing collisions and interactions between particles.

Topics:

Core

- Definition of the differential and total cross sections
- The scattering amplitude $f(\theta)$ and its relation to cross section
- The first Born approximation for weak potentials
- Partial wave analysis and phase shifts

Elective

- The Optical Theorem relating forward scattering to total cross section

Learning Outcomes:

Core:

1. Apply the Born approximation to find the scattering amplitude for a Yukawa potential [Usage]
2. Analyze the effect of a hard-sphere potential on the low-energy scattering phase shift [Assessment]
3. Compute the total cross section from the partial wave expansion [Usage]

5.5 Relativity and Cosmology (RCO)

Relativity and Cosmology explore the fundamental nature of space, time, and gravity, from the local behavior of high-speed particles to the large-scale evolution of the universe. This area covers the transition from special to general relativity and the mathematical models describing the history and composition of the cosmos.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.5.1 Special Relativity: Lorentz Transformations and 4-Vectors (Pg.. 232)	1	1
⭐ 5.5.2 Relativistic Dynamics and Energy-Momentum (Pg.. 232)	1	1
⭐ 5.5.3 General Relativity Foundations: Equivalence Principle (Pg.. 233)	1	1
⭐ 5.5.4 Spacetime Curvature and Einstein Field Equations (Pg.. 233)	1	1
⭐ 5.5.5 Schwarzschild Geometry and Black Holes (Pg.. 233)	1	1
⭐ 5.5.6 Gravitational Waves (Pg.. 233)	1	1
⭐ 5.5.7 Big Bang Theory and Friedmann Equations (Pg.. 234)	1	1
⭐ 5.5.8 Cosmic Microwave Background (CMB) (Pg.. 234)	1	1
⭐ 5.5.9 Dark Matter and Dark Energy (Pg.. 234)	1	1
⭐ 5.5.10 Early Universe Inflation and Nucleosynthesis (Pg.. 235)	1	1

Table 5.5: List of KUs in the Relativity and Cosmology area.

5.5.1 RCO/Special Relativity: Lorentz Transformations and 4-Vectors (Core Tier1: 1 hr, Core Tier2: 1 hr)

The mathematical framework of flat spacetime and the unification of space and time.
Topics:
Core

- Lorentz transformations for coordinate and velocity boosts
- The invariant interval ds^2 and the Minkowski metric
- Time dilation, length contraction, and proper time τ
- Light cones, worldlines, and causal structure of spacetime
- General definition of 4-vectors and the scalar product
- Introduction to tensors and index notation in Minkowski space

Learning Outcomes:
Core:

1. Calculate the time elapsed for a twin traveling at relativistic speeds [Usage]
2. Verify the invariance of the spacetime interval under a Lorentz boost [Assessment]
3. Compute the Lorentz factor for a particle moving at 0.99c [Usage]

5.5.2 RCO/Relativistic Dynamics and Energy-Momentum (Core Tier1: 1 hr, Core Tier2: 1 hr)

Conservation laws and particle physics within the relativistic regime.
Topics:
Core

- Definition of the energy-momentum 4-vector P^μ
- Rest mass energy and the relation $E = \gamma mc^2$
- The mass-shell relation: $E^2 - (pc)^2 = (mc^2)^2$
- Dynamics of particle collisions and creation threshold energies
- Minkowski force and the relativistic version of Newton's Second Law

Learning Outcomes:
Core:

1. Compute the momentum of a particle given its total energy and rest mass [Usage]
2. Analyze a particle decay process using 4-momentum conservation [Assessment]
3. Calculate the threshold energy for pion production in proton-proton collisions [Usage]

5.5.3 RCO/General Relativity Foundations: Equivalence Principle (Core Tier1: 1 hr, Core Tier2: 1 hr)

The conceptual shift from gravity as a force to gravity as geometry.
Topics:
Core

- The Weak Equivalence Principle and universality of free fall
- Einstein's Equivalence Principle: Local physics in freely falling frames
- Derivation of gravitational redshift from the equivalence principle
- Mach's Principle and its influence on the development of GR

Learning Outcomes:
Core:

1. Explain the equivalence of a gravitational field and an accelerating frame [Familiarity]
2. Compute the gravitational redshift of light escaping from a massive star [Usage]
3. Analyze the consequences of the equivalence principle for the bending of light by gravity [Assessment]

5.5.4 RCO/Spacetime Curvature and Einstein Field Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

The mathematical core of GR: Tensors, curvature, and the field equations.
Topics:
Core

- The metric tensor $g_{\mu\nu}$ as the fundamental description of geometry
- Christoffel symbols and the covariant derivative
- The Riemann curvature tensor and tidal forces
- Ricci tensor, Ricci scalar, and the Einstein tensor $G_{\mu\nu}$
- The Einstein Field Equations (EFE): $G_{\mu\nu} = \kappa T_{\mu\nu}$
- The energy-momentum tensor for perfect fluids

Learning Outcomes:
Core:

1. Calculate the non-zero Christoffel symbols for a given simple metric [Usage]
2. Interpret the relationship between matter-energy density and spacetime curvature [Assessment]
3. Compute the Ricci scalar for a Schwarzschild metric at a given radial coordinate [Usage]

5.5.5 RCO/Schwarzschild Geometry and Black Holes (Core Tier1: 1 hr, Core Tier2: 1 hr)

Vacuum solutions of the EFE and the physics of extreme gravity.
Topics:
Core

- The Schwarzschild metric for spherically symmetric static mass
- The Schwarzschild radius R_s and the event horizon
- Orbits of light and particles in Schwarzschild geometry
- Gravitational time dilation near a massive body
- Coordinate singularities vs. physical singularities

Learning Outcomes:
Core:

1. Calculate the Schwarzschild radius for the Sun and the Earth [Usage]
2. Analyze the deflection of light by a massive object in the weak field limit [Assessment]
3. Compute the orbital period at the innermost stable circular orbit (ISCO) of a black hole [Usage]

5.5.6 RCO/Gravitational Waves (Core Tier1: 1 hr, Core Tier2: 1 hr)

Propagating ripples in the fabric of spacetime.
Topics:

Core

- Linearized gravity and the wave equation for spacetime perturbations
- Plus (+) and Cross (×) polarization modes
- The Quadrupole Formula for gravitational radiation power
- Principles of detection via laser interferometry (LIGO/VIRGO)

Learning Outcomes:

Core:

1. Estimate the dimensionless amplitude of gravitational waves from a binary pulsar [Usage]
2. Calculate the strain sensitivity required to detect a given gravitational wave signal [Usage]
3. Analyze the frequency evolution of a binary black hole inspiral during the last few seconds [Assessment]

5.5.7 RCO/Big Bang Theory and Friedmann Equations (Core Tier1: 1 hr, Core Tier2: 1 hr)

The standard cosmological model and the expansion of the universe.

Topics:

Core

- The Cosmological Principle: Isotropy and Homogeneity
- The Friedmann-Lemaître-Robertson-Walker (FLRW) metric
- Hubble's Law and the Hubble constant H_0
- The Friedmann equations for the scale factor $a(t)$
- Critical density and the density parameters $\Omega_m, \Omega_\Lambda, \Omega_k$
- Relation between redshift z and the scale factor a

Learning Outcomes:

Core:

1. Determine the approximate age of a Hubble-flow universe given H_0 [Usage]
2. Analyze the fate of the universe based on the total density parameter [Assessment]
3. Compute the critical density of the universe for a given Hubble constant [Usage]

5.5.8 RCO/Cosmic Microwave Background (CMB) (Core Tier1: 1 hr, Core Tier2: 1 hr)

The relic radiation from the early, hot universe.

Topics:

Core

- Epoch of recombination and photon decoupling
- CMB blackbody spectrum and current temperature ($T \approx 2.73$ K)
- Temperature anisotropies and seeds of cosmic structure
- The angular power spectrum and acoustic peaks

Learning Outcomes:

Core:

1. Describe the significance of the surface of last scattering [Familiarity]
2. Estimate the number density of CMB photons today [Usage]
3. Analyze how the position of the first acoustic peak constrains the geometry of the universe [Assessment]

5.5.9 RCO/Dark Matter and Dark Energy (Core Tier1: 1 hr, Core Tier2: 1 hr)

The mysterious components that dominate the universe's energy density.

Topics:

Core

- Evidence for Dark Matter from galactic rotation curves
- The Cosmological Constant Λ as Dark Energy
- Gravitational lensing as a tool for mapping dark matter mass
- Evidence for accelerating expansion from Type Ia Supernovae

- The ΛCDM standard model of cosmology

Learning Outcomes:

Core:

1. Contrast the roles of dark matter and dark energy in cosmic evolution [Familiarity]
2. Evaluate the evidence for dark matter from gravitational lensing and rotation curves [Assessment]
3. Compute Ω_m and Ω_Λ from a given set of supernova distance moduli [Usage]

5.5.10 RCO/Early Universe Inflation and Nucleosynthesis (Core Tier1: 1 hr, Core Tier2: 1 hr)

High-energy processes in the first minutes and fractions of a second.

Topics:

Core

- Inflationary theory: Solving the Horizon and Flatness problems
- Big Bang Nucleosynthesis: Origin of light elements (H, He, Li)
- Baryogenesis and the matter-antimatter asymmetry

Elective

- Symmetry breaking and phase transitions in the early universe

Learning Outcomes:

Core:

1. Evaluate the predicted helium-4 abundance from BBN theory [Assessment]
2. Compute the size of a causally connected region at recombination and compare to the observed CMB homogeneity [Usage]
3. Explain why the flatness problem motivates an inflationary epoch [Familiarity]

5.6 Condensed Matter Physics (SSP)

Condensed Matter Physics investigates the macroscopic and microscopic properties of matter in its solid and liquid phases. It explores how collective interactions between vast numbers of atoms lead to emergent phenomena such as superconductivity, magnetism, and topological phases.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.6.1 Crystal Structures and Reciprocal Lattices (Pg.. 235)	1	1
⭐ 5.6.2 X-Ray Diffraction and Lattice Dynamics (Pg.. 236)	1	1
⭐ 5.6.3 Free Electron Model and Band Theory of Solids (Pg.. 236)	1	1
⭐ 5.6.4 Semiconductors, Insulators, and Metals (Pg.. 236)	1	1
⭐ 5.6.5 Magnetic Properties: Para-, Dia-, and Ferromagnetism (Pg.. 237)	1	1
⭐ 5.6.6 Superconductivity: BCS Theory and High-Tc (Pg.. 237)	1	1
⭐ 5.6.7 Soft Matter Physics and Polymers (Pg.. 237)	1	1
⭐ 5.6.8 Nanostructures and Low-Dimensional Systems (Pg.. 238)	1	1
⭐ 5.6.9 Spintronics and Topological Insulators (Pg.. 238)	1	1
⭐ 5.6.10 Strongly Correlated Electron Systems (Pg.. 238)	1	1

Table 5.6: List of KUs in the Condensed Matter Physics area.

5.6.1 SSP/Crystal Structures and Reciprocal Lattices (Core Tier1: 1 hr, Core Tier2: 1 hr)

The mathematical description of periodic atomic arrangements in real and momentum space.

Topics:

Core

- The 14 Bravais lattices and point group symmetries
- Primitive unit cells, Wigner-Seitz cells, and the basis of atoms
- Miller indices for crystal planes and directions

- Construction of the reciprocal lattice: $\vec{b}_i = 2\pi \frac{\vec{a}_j \times \vec{a}_k}{\vec{a}_i \cdot (\vec{a}_j \times \vec{a}_k)}$
- The First Brillouin Zone and its physical significance in k-space
- Packing fractions and coordination numbers for SC, BCC, FCC, and HCP

Learning Outcomes:

Core:

1. Determine the reciprocal lattice vectors for a Face-Centered Cubic (FCC) crystal [Usage]
2. Calculate the atomic packing factor for a Diamond Cubic structure [Assessment]
3. Identify the rotational and translational symmetries in a given 2D crystal tiling [Usage]

5.6.2 SSP/X-Ray Diffraction and Lattice Dynamics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Probing crystal structures and the quantization of collective vibrations (phonons).

Topics:

Core

- Bragg's Law of diffraction: $2d \sin\theta = n\lambda$
- Laue condition for constructive interference: $\Delta\vec{k} = \vec{G}$
- The geometric structure factor S_G and systematic absences in diffraction
- Phonon dispersion relations for monatomic and diatomic chains
- Physical distinction between acoustic and optical phonon branches
- Inelastic neutron scattering as a tool for measuring phonon spectra

Learning Outcomes:

Core:

1. Calculate the diffraction angle for the (111) plane of silicon using X-rays of wavelength 1.54 Å [Usage]
2. Analyze the appearance of a frequency gap in a diatomic lattice dispersion curve [Assessment]
3. Compute the structure factor for a BCC crystal and identify forbidden reflections [Usage]

5.6.3 SSP/Free Electron Model and Band Theory of Solids (Core Tier1: 1 hr, Core Tier2: 1 hr)

The behavior of electrons in periodic potentials and the origin of energy bands.

Topics:

Core

- The classical Drude model: conductivity, drift velocity, and Hall effect
- The Sommerfeld model: Fermi energy, Fermi surface, and density of states
- Bloch's Theorem: Wavefunctions in a periodic potential $\psi(\vec{r}) = u(\vec{r})e^{i\vec{k}\cdot\vec{r}}$
- The Kronig-Penney model and the emergence of energy gaps
- The nearly-free electron approximation and perturbation at zone boundaries
- The tight-binding approximation for localized atomic orbitals

Learning Outcomes:

Core:

1. Calculate the Fermi energy and Fermi temperature for copper [Usage]
2. Evaluate the effective mass of an electron from a given $E(k)$ dispersion relation [Assessment]
3. Compute the density of states for a 3D free electron gas [Usage]

5.6.4 SSP/Semiconductors, Insulators, and Metals (Core Tier1: 1 hr, Core Tier2: 1 hr)

Classification of solids based on electronic transport and doping.

Topics:

Core

- Intrinsic semiconductors: carrier concentration and the law of mass action
- Extrinsic semiconductors: n-type and p-type doping, donor/acceptor levels
- Physics of the p-n junction: depletion region and built-in potential
- Temperature dependence of carrier mobility and conductivity

- Optical properties: direct vs. indirect bandgap semiconductors

Elective

- Operation principles of MOSFETs and Solar Cells

Learning Outcomes:

Core:

1. Determine the carrier concentration of silicon at 300K given its bandgap [Usage]
2. Analyze how the Fermi level shifts with increasing donor concentration in a semiconductor [Assessment]
3. Calculate the built-in potential of a p-n junction with given doping concentrations [Usage]

5.6.5 SSP/Magnetic Properties: Para-, Dia-, and Ferromagnetism (Core Tier1: 1 hr, Core Tier2: 1 hr)

Origins of magnetic moments and collective ordering in solids.

Topics:

Core

- Langevin diamagnetism and Curie's Law for paramagnetism
- Weiss molecular field and the Heisenberg exchange interaction
- The Curie-Weiss Law and the paramagnetic-ferromagnetic phase transition
- Antiferromagnetism (Néel temperature) and Ferrimagnetism
- Domain wall energy and the magnetic hysteresis loop
- Quantized spin waves (magnons) and their thermal excitation

Learning Outcomes:

Core:

1. Calculate the magnetic susceptibility of a paramagnetic salt at a specific temperature [Usage]
2. Compare the susceptibility curves of ferromagnetic and antiferromagnetic materials [Assessment]
3. Estimate the Curie temperature from the molecular field constant in a ferromagnet [Usage]

5.6.6 SSP/Superconductivity: BCS Theory and High-Tc (Core Tier1: 1 hr, Core Tier2: 1 hr)

Zero resistance and perfect diamagnetism in quantum fluids.

Topics:

Core

- The Meissner-Ochsenfeld effect and magnetic flux expulsion
- Type I and Type II superconductors: critical fields and vortices
- London equations and the magnetic penetration depth λ_L
- BCS theory: Cooper pairs, phonon-mediated attraction, and the energy gap
- DC and AC Josephson effects in superconducting junctions

Elective

- Overview of High-Tc cuprates and unconventional superconductivity

Learning Outcomes:

Core:

1. Describe the dependence of the critical magnetic field on temperature [Familiarity]
2. Calculate the London penetration depth for a lead superconductor [Usage]
3. Estimate the superconducting energy gap from the critical temperature using BCS theory [Assessment]

5.6.7 SSP/Soft Matter Physics and Polymers (Core Tier1: 1 hr, Core Tier2: 1 hr)

Complexity in systems where thermal energy is comparable to interaction energy.

Topics:

Core

- Liquid crystals: Nematic, Smectic, and Cholesteric phases
- Ideal chain models: Random walks and Flory theory
- Viscoelasticity and the glass transition in amorphous solids

Elective
- Colloidal stability, DLVO theory, and self-assembly

Learning Outcomes:
Core:
1. Analyze the entropic origin of elasticity in rubber-like polymers [Assessment]
2. Compute the mean squared end-to-end distance for a freely jointed chain [Usage]
3. Explain the concept of the glass transition temperature in amorphous polymers [Familiarity]

5.6.8 SSP/Nanostructures and Low-Dimensional Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Quantum confinement effects in ZeroD, OneD, and TwoD materials.

Topics:
Core
- Density of states in quantum wells (TwoD), wires (OneD), and dots (ZeroD)
- Graphene electronic structure and properties of Carbon Nanotubes
- Single-electron tunneling and Coulomb blockade
- Integral Quantum Hall Effect and Landau levels

Learning Outcomes:
Core:
1. Calculate the first energy level shift in a 10nm GaAs quantum well [Usage]
2. Analyze the density of states for a 2D electron gas versus a 1D nanowire [Assessment]
3. Compute the Landau level spacing in a 2DEG under a magnetic field [Usage]

5.6.9 SSP/Spintronics and Topological Insulators (Core Tier1: 1 hr, Core Tier2: 1 hr)

Exploiting electron spin and topology for next-generation electronics.

Topics:
Core
- Giant Magnetoresistance (GMR) and spin-valve sensors
- Spin-orbit coupling and the Spin Hall Effect

Elective
- Topological insulators: Bulk-edge correspondence and protected states
- Majorana bound states in topological superconductors

Learning Outcomes:
Core:
1. Analyze the role of spin-dependent scattering in GMR devices [Assessment]
2. Estimate the spin Hall angle in a heavy metal from experimental data [Usage]
3. Describe the concept of topologically protected edge states [Familiarity]

5.6.10 SSP/Strongly Correlated Electron Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Physics beyond the single-particle approximation: many-body effects.

Topics:
Core
- Mott insulators and the breakdown of band theory
- The Kondo Effect and magnetic impurity scattering

Elective
- The Hubbard Model for correlated electrons on a lattice
- Heavy fermion systems and mass renormalization

Learning Outcomes:
Core:
1. Evaluate when electron-electron repulsion dominates over kinetic energy in the Hubbard model [Assessment]

2. Estimate the Kondo temperature from the exchange coupling constant [Usage]
3. Explain the Mott metal-insulator transition in terms of electron correlation [Familiarity]

5.7 Nuclear and Particle Physics (NPP)

Nuclear and Particle Physics explores the fundamental constituents of matter and the forces that govern their interactions. This field spans from the structure and stability of the atomic nucleus to the high-energy interactions of quarks and leptons described by the Standard Model.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.7.1 Nuclear Structure, Binding Energy, and Shell Model (Pg.. 239)	1	1
⭐ 5.7.2 Radioactivity and Nuclear Decays (Alpha, Beta, Gamma) (Pg.. 239)	1	1
⭐ 5.7.3 Nuclear Reactions, Fission, and Fusion (Pg.. 240)	1	1
⭐ 5.7.4 Particle Accelerators and Detectors (Pg.. 240)	1	1
⭐ 5.7.5 The Standard Model of Particle Physics (Pg.. 240)	1	1
⭐ 5.7.6 Quarks, Leptons, Gauge Bosons (Pg.. 241)	1	1
⭐ 5.7.7 Symmetries and Conservation Laws (CPT) (Pg.. 241)	1	1
⭐ 5.7.8 Fundamental Interactions and Feynman Diagrams (Pg.. 241)	1	1
⭐ 5.7.9 Higgs Mechanism and Beyond the Standard Model (Pg.. 242)	1	1
⭐ 5.7.10 Neutrino Physics and Oscillation (Pg.. 242)	1	1

Table 5.7: List of KUs in the Nuclear and Particle Physics area.

5.7.1 NPP/Nuclear Structure, Binding Energy, and Shell Model (Core Tier1: 1 hr, Core Tier2: 1 hr)

The static properties of nuclei and the models describing nucleon organization.
Topics:
Core

- Nuclear radii, charge distribution, and the $R = R_0 A^{1/3}$ scaling law
- The Liquid Drop Model and the Semi-Empirical Mass Formula (SEMF)
- Binding energy per nucleon curve and nuclear stability trends
- The Nuclear Shell Model: Evidence for magic numbers
- Strong spin-orbit coupling and the reordering of nuclear energy levels
- Nuclear magnetic dipole and electric quadrupole moments
- Collective model: Rotational and vibrational excited states in nuclei

Learning Outcomes:
Core:

1. Calculate the total binding energy of a Carbon-12 nucleus using the SEMF [Usage]
2. Predict the ground-state spin and parity of odd-A nuclei using the Shell Model [Assessment]
3. Determine the neutron separation energy for a specific isotope from mass data [Usage]

5.7.2 NPP/Radioactivity and Nuclear Decays (Alpha, Beta, Gamma) (Core Tier1: 1 hr, Core Tier2: 1 hr)

The physics of unstable nuclei and the three primary modes of radioactive decay.
Topics:
Core

- The exponential decay law, half-life, and mean lifetime concepts
- Alpha decay: Gamow's theory of quantum tunneling through the Coulomb barrier
- Beta decay (β^-, β^+) and Electron Capture: The role of the neutrino
- Fermi's theory of beta decay and the Kurie plot
- Gamma decay: Multi-pole transitions and internal conversion
- Selection rules for nuclear transitions based on angular momentum and parity
- Radioactive decay chains and secular/transient equilibrium

Learning Outcomes:
Core:
1. Calculate the activity of a radioactive sample after five half-lives [Usage]
2. Analyze why the beta particle spectrum is continuous while alpha is discrete [Assessment]
3. Evaluate whether an E1 or M1 transition is allowed between two nuclear states [Usage]

5.7.3 NPP/Nuclear Reactions, Fission, and Fusion (Core Tier1: 1 hr, Core Tier2: 1 hr)

Energetics and mechanisms of nuclear transmutations and energy release.
Topics:
Core
- Reaction Q-value and threshold energy for endothermic reactions
- Nuclear cross-sections and Breit-Wigner resonance formula
- Mechanisms of induced fission and the chain reaction principle
- Nuclear fusion: The p-p chain and CNO cycle in stellar nucleosynthesis
- Neutron moderation, four-factor formula, and reactor criticality
- Classification of reactions: Direct reactions vs. compound nucleus formation

Elective
- Inertial and magnetic confinement fusion: Lawson criterion

Learning Outcomes:
Core:
1. Calculate the minimum kinetic energy required for a (α, n) reaction [Usage]
2. Analyze the mass distribution of fission fragments in U-235 [Assessment]
3. Compute the energy released in a single Deuterium-Tritium fusion event [Usage]

5.7.4 NPP/Particle Accelerators and Detectors (Core Tier1: 1 hr, Core Tier2: 1 hr)

Experimental tools for high-energy physics and radiation measurement.
Topics:
Core
- Principles of Linear Accelerators (Linacs) and RF cavities
- Cyclotrons, Synchrotrons, and the principle of phase stability
- Ionization chambers, Proportional counters, and Geiger-Müller tubes
- Scintillation counters and Photomultiplier Tubes (PMTs)
- Silicon strip and High-Purity Germanium (HPGe) detectors
- Electromagnetic/Hadronic calorimeters and Cherenkov radiation detectors
- Collider physics: Center-of-mass energy and integrated luminosity

Learning Outcomes:
Core:
1. Calculate the cyclotron frequency for a proton in a 1.5 Tesla field [Usage]
2. Select the appropriate detector type for high-resolution gamma-ray spectroscopy [Assessment]
3. Compute the integrated luminosity needed to produce a given number of events [Usage]

5.7.5 NPP/The Standard Model of Particle Physics (Core Tier1: 1 hr, Core Tier2: 1 hr)

The unified framework describing electromagnetism, weak, and strong interactions.
Topics:
Core
- The four fundamental forces: range, strength, and mediator particles
- Classification of particles: Fermions (matter) and Bosons (force carriers)
- Conceptual structure of the Standard Model Lagrangian
- Local gauge invariance as the generator of interactions
- The three generations of quarks and leptons

- Electroweak unification and the Grand Unification Theory (GUT) scale
- Asymptotic freedom and the running of coupling constants

Learning Outcomes:

Core:

1. List all particles in the Standard Model and their basic properties (spin, charge) [Familiarity]
2. Contrast the properties of the Gluon and the Photon [Assessment]
3. Explain why the strong coupling constant decreases at high energies [Usage]

5.7.6 NPP/Quarks, Leptons, Gauge Bosons (Core Tier1: 1 hr, Core Tier2: 1 hr)

Specific properties of the elementary building blocks of the universe.

Topics:

Core

- Quark flavors (u, d, s, c, b, t) and Color Charge (R, G, B)
- Hadron classification: Baryons (qqq) and Mesons (q-antiquark)
- Lepton families and the conservation of Lepton Number (L_e, L_μ, L_τ)
- The W^+, W^-, and Z^0 bosons: Mediators of the weak force
- Gluons and the principle of color confinement
- Strangeness and the discovery of kaons and hyperons
- Charmonium and Bottomonium: Spectroscopy of heavy quarks

Learning Outcomes:

Core:

1. Determine the quark content for a proton, neutron, and pion [Usage]
2. Calculate the total electric charge and strangeness of a Ω^- baryon [Assessment]
3. Assign the flavor quantum numbers to a given hadron state [Usage]

5.7.7 NPP/Symmetries and Conservation Laws (CPT) (Core Tier1: 1 hr, Core Tier2: 1 hr)

The deep connection between symmetries and physical invariants.

Topics:

Core

- Noether's Theorem: Space-time symmetries and conservation of E, p, L
- Conservation of Baryon Number (B) and Lepton Number (L)
- Isospin symmetry in the strong interaction (p/n symmetry)
- Parity (P) violation in weak interactions (Wu experiment)
- Charge Conjugation (C) and the C-symmetry
- CP violation in the neutral Kaon system
- The CPT Theorem and its implications for particle-antiparticle mass parity

Learning Outcomes:

Core:

1. Check if a proposed particle reaction violates B, L, or Q conservation [Usage]
2. Evaluate the parity of a two-pion system in a specific orbital state [Assessment]
3. Predict possible CP-violating decays in the B meson system [Usage]

5.7.8 NPP/Fundamental Interactions and Feynman Diagrams (Core Tier1: 1 hr, Core Tier2: 1 hr)

Visualizing and calculating particle interactions via exchange diagrams.

Topics:

Core

- Basic rules for drawing Feynman diagrams: Vertices and propagators
- Quantum Electrodynamics (QED) vertices and electron-photon coupling
- Weak vertices: Charged current ($W^\pm$) and Neutral current (Z)
- Quantum Chromodynamics (QCD) vertices and gluon self-interaction
- Relation between Feynman diagrams and transition matrix elements

- Off-shell particles and the physical meaning of propagators
- Introduction to loop diagrams and renormalization concepts

Learning Outcomes:

Core:

1. Draw a Feynman diagram for the decay of a muon into an electron and neutrinos [Usage]
2. Identify whether a diagram represents an EM, Weak, or Strong interaction based on vertices [Assessment]
3. Estimate the matrix element for electron-electron scattering in QED [Usage]

5.7.9 NPP/Higgs Mechanism and Beyond the Standard Model (Core Tier1: 1 hr, Core Tier2: 1 hr)

Origin of mass and the limitations of current theory.

Topics:

Core

- Spontaneous symmetry breaking and Goldstone bosons
- The Higgs field potential (Mexican hat) and vacuum expectation value
- Mass generation via Yukawa coupling to the Higgs field
- The Hierarchy Problem and fine-tuning in the Standard Model

Elective

- Introduction to Supersymmetry (SUSY) and superpartners
- Grand Unified Theories (GUTs) and WIMP candidates for dark matter

Learning Outcomes:

Core:

1. Explain how the Higgs mechanism provides mass to the W and Z bosons [Familiarity]
2. Compute the vacuum expectation value of the Higgs field from the Fermi constant [Usage]
3. Evaluate the magnitude of the hierarchy problem in the Standard Model [Assessment]

5.7.10 NPP/Neutrino Physics and Oscillation (Core Tier1: 1 hr, Core Tier2: 1 hr)

The elusive nature of neutrinos and the evidence for non-zero mass.

Topics:

Core

- Sources of neutrinos: Solar, atmospheric, and reactor neutrinos
- The Reines-Cowan experiment and neutrino cross-section scales
- Neutrino flavor mixing and the PMNS matrix
- Neutrino oscillation formula: Two-flavor approximation
- The Solar Neutrino Problem and the MSW effect

Elective

- Dirac vs. Majorana neutrinos and neutrinoless double beta decay

Learning Outcomes:

Core:

1. Calculate the oscillation length for a neutrino of energy 1 GeV [Usage]
2. Evaluate the implications of neutrino oscillations for the Standard Model mass-less assumption [Assessment]
3. Compute the electron neutrino survival probability as a function of baseline [Usage]

5.8 Quantum Information and Computing (QIC)

Quantum Information and Computing leverages the principles of quantum mechanics-superposition, entanglement, and interference-to process and transmit information in ways that surpass classical limits. This area covers the theoretical foundations, algorithmic development, and the physical architectures of quantum hardware.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.8.1 Qubits, Superposition, and Entanglement (Pg.. 243)	1	1
⭐ 5.8.2 Quantum Logic Gates and Circuits (Pg.. 243)	1	1
⭐ 5.8.3 Quantum Algorithms (Shor, Grover) (Pg.. 243)	1	1
⭐ 5.8.4 Quantum Teleportation and Cryptography (Pg.. 244)	1	1
⭐ 5.8.5 Decoherence and Quantum Error Correction (Pg.. 244)	1	1
⭐ 5.8.6 Physical Realizations of Quantum Computers (Pg.. 244)	1	1
⭐ 5.8.7 Quantum Sensing and Metrology (Pg.. 245)	1	1
⭐ 5.8.8 Quantum Networks and Communication (Pg.. 245)	1	1

Table 5.8: List of KUs in the Quantum Information and Computing area.

5.8.1 QIC/Qubits, Superposition, and Entanglement (Core Tier1: 1 hr, Core Tier2: 1 hr)

The fundamental units of quantum information and their non-classical correlations.
Topics:
Core

- The Bloch Sphere: Visualization of pure state qubits
- Linear superposition and computational basis states $|0\rangle, |1\rangle$
- The four Bell states and maximally entangled pairs
- Density matrix formalism for pure and mixed states
- Schmidt decomposition and the concept of state purification
- Tracing out subsystems and reduced density operators

Learning Outcomes:
Core:

1. Represent a given quantum state as coordinates on the Bloch Sphere [Usage]
2. Use the partial trace to determine if a two-qubit state is entangled or separable [Assessment]
3. Calculate the Von Neumann entropy to quantify the degree of entanglement in a system [Usage]

5.8.2 QIC/Quantum Logic Gates and Circuits (Core Tier1: 1 hr, Core Tier2: 1 hr)

Unitary transformations and the construction of universal quantum circuits.
Topics:
Core

- Single-qubit gates: Pauli X, Y, Z, Hadamard (H), and Phase (S, T) gates
- Two-qubit gates: CNOT, CZ, and SWAP operations
- Definition of universal gate sets for quantum computation
- Three-qubit gates: Toffoli (CCNOT) and Fredkin (CSWAP) gates
- The principle of reversible computing in quantum circuits
- Decomposing arbitrary unitary matrices into elementary gates

Learning Outcomes:
Core:

1. Construct a quantum circuit using H and CNOT gates to generate a Bell state [Usage]
2. Evaluate the resulting state vector after applying a sequence of Pauli gates [Assessment]
3. Synthesize a Toffoli gate using only 2-qubit CNOT and 1-qubit gates [Usage]

5.8.3 QIC/Quantum Algorithms (Shor, Grover) (Core Tier1: 1 hr, Core Tier2: 1 hr)

Harnessing quantum speedup for factoring, searching, and simulation.
Topics:
Core

- The Quantum Fourier Transform (QFT) and its circuit implementation
- The Deutsch-Jozsa algorithm and the concept of oracle-based speedup

- Shor's Algorithm: Period finding and the RSA cryptosystem threat
- Grover's Algorithm: Amplitude amplification and the diffusion operator
- The Quantum Phase Estimation (QPE) protocol

Elective

- Variational Quantum Eigensolver (VQE) and QAOA for NISQ devices

Learning Outcomes:

Core:

1. Apply the QFT to a 3-qubit register and determine the output state [Usage]
2. Analyze the $O(\sqrt{N})$ speedup of Grover's algorithm compared to classical linear search [Assessment]
3. Trace the steps of the Phase Estimation algorithm to find the eigenvalue of a T-gate [Usage]

5.8.4 QIC/Quantum Teleportation and Cryptography (Core Tier1: 1 hr, Core Tier2: 1 hr)

Protocols for secure information transfer and state reconstruction.

Topics:

Core

- The Quantum Teleportation protocol: Measurement and classical correction
- The No-Cloning Theorem and its implications for quantum security
- The BB84 Quantum Key Distribution (QKD) protocol
- Ekert91 protocol: Entanglement-based QKD and Bell's inequality
- Superdense coding: Sending two classical bits via one qubit
- Remote state preparation and entanglement swapping

Learning Outcomes:

Core:

1. Diagram the sequence of gates and measurements required to teleport a qubit [Usage]
2. Identify eavesdropping in a BB84 protocol by analyzing error rates in the sifted key [Assessment]
3. Execute the bit-to-gate mapping for a superdense coding transmission [Usage]

5.8.5 QIC/Decoherence and Quantum Error Correction (Core Tier1: 1 hr, Core Tier2: 1 hr)

Protecting quantum information from noise and environmental interactions.

Topics:

Core

- Types of quantum noise: Bit-flip, phase-flip, and depolarizing channels
- Relaxation (T_1) and dephasing (T_2) time scales
- The 3-qubit bit-flip code and majority voting
- The Shor 9-qubit code: Correcting arbitrary single-qubit errors
- The Stabilizer formalism for quantum error correction codes

Elective

- Introduction to surface codes and topological protection

Learning Outcomes:

Core:

1. Evaluate the impact of dephasing on the off-diagonal elements of a density matrix [Assessment]
2. Calculate the error syndrome for a 3-qubit code using stabilizer measurements [Usage]
3. Analyze the threshold theorem for fault-tolerant quantum computation [Assessment]

5.8.6 QIC/Physical Realizations of Quantum Computers (Core Tier1: 1 hr, Core Tier2: 1 hr)

The engineering of physical systems to satisfy DiVincenzo's criteria.

Topics:

Core

- DiVincenzo's criteria for a scalable quantum computer

- Superconducting qubits: Cooper pair boxes and Transmon qubits
- Trapped ion quantum computing: Paul traps and Cirac-Zoller gates
- Linear Optical Quantum Computing (LOQC) and squeezed states
- Neutral atom arrays and Rydberg blockade interactions

Elective

- Semiconductor quantum dots and spin-based qubits

Learning Outcomes:

Core:

1. Compare the coherence times and gate speeds of superconducting vs. ion trap qubits [Familiarity]
2. Analyze the Hamiltonian of a Transmon qubit and its anharmonicity [Assessment]
3. Design a 1D ion chain configuration for a specific gate operation [Usage]

5.8.7 QIC/Quantum Sensing and Metrology (Core Tier1: 1 hr, Core Tier2: 1 hr)

Using quantum states to achieve precision measurements beyond the Standard Quantum Limit.

Topics:

Core

- The Standard Quantum Limit and the Heisenberg Limit
- Squeezed light and its application in gravitational wave detection (LIGO)
- Nitrogen-Vacancy (NV) centers in diamond for magnetic field sensing
- Ramsey interferometry and the physics of atomic clocks
- Quantum Fisher Information and phase estimation precision

Learning Outcomes:

Core:

1. Calculate the measurement precision scaling for N entangled particles compared to N independent particles [Usage]
2. Evaluate the sensitivity of an NV center sensor to a nano-Tesla magnetic field [Assessment]
3. Model a Ramsey pulse sequence to measure a frequency offset [Usage]

5.8.8 QIC/Quantum Networks and Communication (Core Tier1: 1 hr, Core Tier2: 1 hr)

Scaling quantum systems via entanglement distribution and quantum repeaters.

Topics:

Core

- Quantum repeaters: Entanglement purification and Bell state measurements
- Entanglement swapping for long-distance quantum communication
- Quantum memory: Storing quantum states in light-matter interfaces
- Free-space and satellite-based quantum communication (Micius)

Elective

- Architectures for a global quantum internet

Learning Outcomes:

Core:

1. Design a quantum repeater chain to connect two nodes separated by 1000 km [Usage]
2. Evaluate the fidelity of a quantum state after retrieval from a storage medium [Assessment]
3. Calculate the probability of successful entanglement distribution over a lossy fiber link [Usage]

5.9 Optics and Photonics (OPH)

Optics and Photonics study the behavior and properties of light and its interaction with matter. This field covers the classical laws of reflection and refraction, the wave nature of light manifested in interference and diffraction, and the modern advancements in lasers, fiber optics, and nanophotonics.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.9.1 Geometrical Optics and Optical Instruments (Pg.. 246)	1	1
⭐ 5.9.2 Wave Optics: Interference and Diffraction (Pg.. 246)	1	1
⭐ 5.9.3 Polarization and Crystal Optics (Pg.. 247)	1	1
⭐ 5.9.4 Lasers: Principles and Applications (Pg.. 247)	1	1
⭐ 5.9.5 Nonlinear Optics and Frequency Conversion (Pg.. 247)	1	1
⭐ 5.9.6 Fiber Optics and Photonic Crystals (Pg.. 248)	1	1
⭐ 5.9.7 Ultrafast Optics and Spectroscopy (Pg.. 248)	1	1
⭐ 5.9.8 Nano-optics and Plasmonics (Pg.. 248)	1	1

Table 5.9: List of KUs in the Optics and Photonics area.

5.9.1 OPH/Geometrical Optics and Optical Instruments (Core Tier1: 1 hr, Core Tier2: 1 hr)

The ray approximation of light and the design of imaging systems.
Topics:
Core

- Fermat's Principle and the derivation of reflection/refraction laws
- The Lens-Maker's formula and spherical mirror equations
- Thick lenses, cardinal points, and system matrices
- Abbe theory of image formation and resolution limits
- Compound microscopes, Keplerian/Galilean telescopes, and cameras
- Paraxial ray tracing using the ABCD matrix formalism
- The five Seidel aberrations: Spherical, Coma, Astigmatism, Curvature, Distortion

Learning Outcomes:
Core:

1. Design a telescope system with a specific angular magnification using two thin lenses [Usage]
2. Evaluate the position and nature of an image formed by a combination of a mirror and a lens [Assessment]
3. Calculate the equivalent ABCD matrix for a complex optical train of three components [Usage]

5.9.2 OPH/Wave Optics: Interference and Diffraction (Core Tier1: 1 hr, Core Tier2: 1 hr)

Wave-like phenomena of light and the resolution of optical systems.
Topics:
Core

- Principle of superposition and phasor representation of waves
- Young's double-slit experiment and interference fringe intensity
- The Michelson Interferometer: Measurement of small displacements
- Fraunhofer diffraction by single, double, and N slits
- Diffraction by circular apertures and the Airy disk
- The Rayleigh criterion for the limit of resolution
- Fresnel diffraction and Fresnel zone plates
- Multi-beam interference and the Fabry-Perot interferometer

Learning Outcomes:
Core:

1. Calculate the fringe spacing on a screen for a double-slit setup with 532 nm light [Usage]
2. Determine the minimum resolvable separation of two stars using a 10-inch telescope [Assessment]
3. Compute the finesse and free spectral range of a Fabry-Perot cavity [Usage]

5.9.3 OPH/Polarization and Crystal Optics (Core Tier1: 1 hr, Core Tier2: 1 hr)

The vector nature of light and its interaction with anisotropic media.
Topics:
Core
- Linear, circular, and elliptical polarization states
- Malus's Law for polarized light intensity
- Polarization by reflection and Brewster's Angle
- Birefringence in uniaxial crystals: Ordinary and extraordinary rays
- Quarter-wave and half-wave plates: Phase retardation
- Jones vectors and matrices for polarization state analysis
- Optical activity and Faraday rotation

Learning Outcomes:
Core:
1. Determine the final intensity of unpolarized light passing through three polarizers at 45-degree offsets [Usage]
2. Analyze the output polarization state of linearly polarized light after a quarter-wave plate [Assessment]
3. Use Jones calculus to find the resulting state after a polarizer and a retarder [Usage]

5.9.4 OPH/Lasers: Principles and Applications (Core Tier1: 1 hr, Core Tier2: 1 hr)

Coherent light generation through stimulated emission.
Topics:
Core
- Einstein A and B coefficients for absorption and emission
- Conditions for lasing: Population inversion and optical pumping
- Optical resonators: Longitudinal and transverse (TEM) modes
- Energy level schemes: 3-level vs. 4-level lasers
- Gaussian beams: Beam waist, Rayleigh range, and divergence
- Pulsed operation: Q-switching and Mode-locking techniques
- Characteristics of He-Ne, Nd:YAG, and Diode lasers

Learning Outcomes:
Core:
1. Calculate the threshold gain required for lasing in a cavity with known mirror losses [Usage]
2. Explain why a 2-level system cannot achieve steady-state population inversion [Assessment]
3. Determine the focused spot size of a Gaussian laser beam using a given lens [Usage]

5.9.5 OPH/Nonlinear Optics and Frequency Conversion (Core Tier1: 1 hr, Core Tier2: 1 hr)

Light-matter interactions where polarization depends nonlinearly on the electric field.
Topics:
Core
- Nonlinear susceptibility tensors ($\chi^{(2)}, \chi^{(3)}$)
- Second Harmonic Generation (SHG) and frequency doubling
- Phase matching: Momentum conservation in nonlinear processes
- Electro-optic effects: Pockels effect and Kerr effect
- Optical Parametric Oscillation (OPO) and down-conversion

Elective
- Self-focusing, self-phase modulation, and optical solitons

Learning Outcomes:
Core:
1. Calculate the SHG conversion efficiency for a specific nonlinear crystal length [Usage]
2. Evaluate the required crystal orientation to achieve type-I phase matching [Assessment]

3. Model the phase shift induced by the Kerr effect in an optical fiber [Usage]

5.9.6 OPH/Fiber Optics and Photonic Crystals (Core Tier1: 1 hr, Core Tier2: 1 hr)

Guiding light in waveguides and periodic dielectric structures.
Topics:
Core
- Total internal reflection and step-index fiber structures
- Numerical Aperture (NA) and the acceptance cone of a fiber
- Single-mode vs. multi-mode fibers and the V-number
- Signal degradation: Attenuation and chromatic/modal dispersion
- Photonic crystals and the emergence of photonic bandgaps
- Microstructured and photonic crystal fibers (PCFs)

Learning Outcomes:
Core:
1. Calculate the numerical aperture for a fiber with core index 1.48 and cladding 1.46 [Usage]
2. Determine the maximum wavelength for single-mode operation in a given fiber [Assessment]
3. Evaluate the pulse broadening over a 50 km link due to chromatic dispersion [Usage]

5.9.7 OPH/Ultrafast Optics and Spectroscopy (Core Tier1: 1 hr, Core Tier2: 1 hr)

Generation and measurement of femtosecond and attosecond light pulses.
Topics:
Core
- Generation of femtosecond pulses via Ti:Sapphire lasers
- The time-bandwidth product and transform-limited pulses
- Group Velocity Dispersion (GVD) and pulse chirping
- Pulse characterization using intensity autocorrelation
- Time-resolved pump-probe spectroscopy techniques

Elective
- High-harmonic generation and attosecond science

Learning Outcomes:
Core:
1. Calculate the broadened pulse width after passing through a glass block with known GVD [Usage]
2. Interpret an experimental autocorrelation trace to estimate pulse duration [Assessment]
3. Model the decay of an excited state population using pump-probe data [Usage]

5.9.8 OPH/Nano-optics and Plasmonics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Light interaction with sub-wavelength structures and metallic nanostructures.
Topics:
Core
- Breaking the diffraction limit: Near-field vs. Far-field optics
- Surface Plasmon Polaritons (SPPs) at metal-dielectric interfaces
- Localized Surface Plasmon Resonance (LSPR) in metal nanoparticles
- Scanning Near-field Optical Microscopy (SNOM)
- Metamaterials and negative refractive index structures

Elective
- Optical tweezers: Force and torque on nanoparticles

Learning Outcomes:
Core:
1. Describe the decay of evanescent waves and their role in near-field optics [Familiarity]
2. Analyze the shift in LSPR frequency due to changes in the local refractive index [Assessment]
3. Calculate the wavelength of an SPP at a gold-air interface for a given excitation light [Usage]

5.10 Computational and Data-Driven Physics (CDP)

This Knowledge Area focuses on the intersection of numerical analysis, high-performance computing, and modern data science. It equips physicists with the tools to simulate systems that lack analytical solutions and to extract fundamental laws from massive datasets using machine learning and statistical inference.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.10.1 Numerical Methods for Differential Equations in Physics (Pg.. 249)	1	1
⭐ 5.10.2 Monte Carlo Methods and Simulations (Pg.. 249)	1	1
⭐ 5.10.3 High-Performance Computing (HPC) for Large-Scale Simulations (Pg.. 250)	1	1
⭐ 5.10.4 Machine Learning for Physical Discovery (Pg.. 250)	1	1
⭐ 5.10.5 Data Analysis and Uncertainty Quantification (Pg.. 250)	1	1
⭐ 5.10.6 Modeling Complex Adaptive Systems (Pg.. 251)	1	1
⭐ 5.10.7 Neural Networks for Quantum Systems (Pg.. 251)	1	1
⭐ 5.10.8 Physics-Informed Neural Networks (PINNs) (Pg.. 251)	1	1

Table 5.10: List of KUs in the Computational and Data-Driven Physics area.

5.10.1 CDP/Numerical Methods for Differential Equations in Physics (Core Tier1: 1 hr, Core Tier2: 1 hr)

Discrete approximations for solving the fundamental equations of motion and field theories.
Topics:
Core

- Runge-Kutta methods (RK2, RK4) for initial value problems
- Finite Difference Time Domain (FDTD) for Maxwell's equations
- Boundary value problems: Shooting methods and relaxation techniques
- Symplectic integrators: Verlet and leapfrog algorithms for Hamiltonian systems
- Stability analysis and the Courant-Friedrichs-Lewy (CFL) condition
- Finite Element Method (FEM) for complex geometries in elasticity and fluid flow

Learning Outcomes:
Core:

1. Implement a velocity Verlet algorithm to simulate a 1D harmonic oscillator [Usage]
2. Evaluate the scaling of discretization errors for different numerical integration orders [Assessment]
3. Solve the 2D Laplace equation for a given set of boundary potentials using Jacobi iteration [Usage]

5.10.2 CDP/Monte Carlo Methods and Simulations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Stochastic sampling techniques for multi-dimensional integration and statistical physics.
Topics:
Core

- Pseudo-random and quasi-random number generation algorithms
- The Metropolis-Hastings algorithm for Markov Chain Monte Carlo (MCMC)
- Monte Carlo simulation of the Ising model and phase transitions
- Importance sampling and variance reduction techniques
- Kinetic Monte Carlo (KMC) for time-dependent stochastic processes
- Statistical error estimation and autocorrelation time in MCMC chains

Learning Outcomes:
Core:

1. Estimate the value of π using a hit-or-miss Monte Carlo integration method [Usage]

2. Analyze the effect of critical slowing down near the Curie temperature in an Ising simulation [Assessment]
3. Develop an importance sampling scheme to integrate a peaked multidimensional function [Usage]

5.10.3 CDP/High-Performance Computing (HPC) for Large-Scale Simulations (Core Tier1: 1 hr, Core Tier2: 1 hr)

Architectures and programming models for massive parallelization of physics code.
Topics:
Core

- Shared-memory programming using OpenMP directives
- Distributed-memory systems and the Message Passing Interface (MPI)
- GPU acceleration and GPGPU programming models (CUDA, OpenCL)
- Scalability analysis: Amdahl's Law and Gustafson's Law
- Domain decomposition methods for spatial grid parallelization
- Optimization for cache locality and memory bandwidth limits

Learning Outcomes:
Core:

1. Parallelize a large-scale vector addition using OpenMP thread synchronization [Usage]
2. Measure the speedup and parallel efficiency of a code across multiple compute nodes [Assessment]
3. Implement non-blocking MPI communication for a ghost-cell exchange in a fluid simulation [Usage]

5.10.4 CDP/Machine Learning for Physical Discovery (Core Tier1: 1 hr, Core Tier2: 1 hr)

Applying statistical learning to identify patterns and surrogate models in data.
Topics:
Core

- Supervised learning: Linear regression, Lasso, and Ridge for physics models
- Unsupervised learning: Clustering and PCA for detecting phase transitions
- Physical feature engineering and dimensional reduction
- Gaussian Process Regression for uncertainty-aware surrogate modeling
- Symbolic regression for discovering analytical physical laws from data
- Kernel methods for approximating quantum many-body states

Learning Outcomes:
Core:

1. Train a regression model to predict the potential energy surface of a molecule [Usage]
2. Evaluate model overfitting using cross-validation on a spectroscopic dataset [Assessment]
3. Apply Principal Component Analysis (PCA) to visualize high-dimensional spin configurations [Usage]

5.10.5 CDP/Data Analysis and Uncertainty Quantification (Core Tier1: 1 hr, Core Tier2: 1 hr)

Statistical frameworks for validating models and quantifying experimental errors.
Topics:
Core

- Bayesian inference: Priors, likelihoods, and posterior distributions
- Maximum Likelihood Estimation (MLE) for parameter fitting
- Linear and non-linear propagation of experimental uncertainties
- Resampling techniques: Bootstrapping and Jackknife methods
- Model selection criteria: AIC, BIC, and Evidence ratios
- Global sensitivity analysis and Sobol indices

Learning Outcomes:
Core:
1. Perform a Bayesian parameter fit for a radioactive decay dataset [Usage]
2. Calculate confidence intervals for a physical constant derived from noisy measurements [Assessment]
3. Compute the Fisher Information matrix to determine the precision limits of a parameter estimate [Usage]

5.10.6 CDP/Modeling Complex Adaptive Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Dynamics of systems with emergent behavior and non-linear interactions.
Topics:
Core
- Agent-based modeling (ABM) for social and biological systems
- Network science: Scale-free and small-world network dynamics
- Self-organized criticality (SOC) and power-law distributions
- Cellular Automata: Game of Life and fluid flow models (LBM)
- Collective synchronization and the Kuramoto model

Elective
- Evolutionary game theory and ecological stability

Learning Outcomes:
Core:
1. Simulate 2D fluid flow around an obstacle using a Lattice Boltzmann method [Usage]
2. Analyze the robustness of a scale-free network against targeted attacks [Assessment]
3. Model a sandpile to demonstrate the principles of self-organized criticality [Usage]

5.10.7 CDP/Neural Networks for Quantum Systems (Core Tier1: 1 hr, Core Tier2: 1 hr)

Representing and solving many-body quantum problems using artificial neural networks.
Topics:
Core
- Restricted Boltzmann Machines (RBM) as Neural Quantum States (NQS)
- Variational Monte Carlo using neural network trial wavefunctions
- Neural network-based quantum state tomography
- CNNs for identifying topological phases in spin lattices
- Autoencoders for compressing many-body wavefunction data

Elective
- Generative models (GANs, VAEs) for sampling quantum configurations

Learning Outcomes:
Core:
1. Train an RBM to approximate the ground state of a transverse-field Ising chain [Usage]
2. Compare the accuracy of Neural Quantum States against traditional Hartree-Fock methods [Assessment]
3. Reconstruct a density matrix from experimental measurement data using a neural network [Usage]

5.10.8 CDP/Physics-Informed Neural Networks (PINNs) (Core Tier1: 1 hr, Core Tier2: 1 hr)

Integrating physical laws directly into the loss functions of deep learning architectures.
Topics:
Core
- Incorporating PDEs as residuals in neural network loss functions
- Leveraging automatic differentiation for exact derivative constraints

- Data-efficient learning: Solving inverse problems with sparse data
- Application of PINNs to the Navier-Stokes equations
- Bayesian PINNs for uncertainty quantification in PDE solutions

Elective

- Introduction to Neural Operators (DeepONet) for parametric PDEs

Learning Outcomes:

Core:

1. Solve the 1D Burgers' equation using a PINN architecture [Usage]
2. Compare the computational cost and accuracy of a PINN versus a traditional Finite Element solver [Assessment]
3. Use a PINN to estimate the viscosity coefficient of a fluid from flow visualization data [Usage]

5.11 Biophysics and Medical Physics (BMP)

Biophysics and Medical Physics apply the theories and methods of physics to understand biological systems and improve human health. From the mechanical forces within a single cell to the complex radiation physics used in cancer therapy, this field bridges the gap between the physical and life sciences.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.11.1 Physics of Biological Macromolecules (Pg.. 252)	1	1
⭐ 5.11.2 Biomechanics and Cellular Forces (Pg.. 252)	1	1
⭐ 5.11.3 Neural Physics and Information Processing in the Brain (Pg.. 253)	1	1
⭐ 5.11.4 Imaging Physics (MRI, CT, PET, Ultrasound) (Pg.. 253)	1	1
⭐ 5.11.5 Radiation Therapy and Dosimetry (Pg.. 253)	1	1
⭐ 5.11.6 Statistical Mechanics of Biological Networks (Pg.. 254)	1	1
⭐ 5.11.7 Microfluidics and Lab-on-a-Chip (Pg.. 254)	1	1

Table 5.11: List of KUs in the Biophysics and Medical Physics area.

5.11.1 BMP/Physics of Biological Macromolecules (Core Tier1: 1 hr, Core Tier2: 1 hr)

Structural and thermodynamic properties of DNA, proteins, and lipids.

Topics:

Core

- Protein folding: Energetics, entropy, and the Levinthal paradox
- Elasticity of DNA: The Worm-Like Chain (WLC) model
- Mechanics of lipid bilayers and phase transitions in membranes
- Electrostatics in cells: Poisson-Boltzmann and Debye-Hückel theory
- Molecular motors: Kinesin and Myosin as Brownian ratchets
- Single-molecule techniques: Optical and magnetic tweezers

Learning Outcomes:

Core:

1. Calculate the persistence length of a DNA molecule from a force-extension curve [Usage]
2. Evaluate the thermal stability of a protein using its Gibbs free energy landscape [Assessment]
3. Model the efficiency of a molecular motor using a stochastic Langevin approach [Usage]

5.11.2 BMP/Biomechanics and Cellular Forces (Core Tier1: 1 hr, Core Tier2: 1 hr)

Physical forces governing cell shape, motility, and adhesion.

Topics:

Core

- Elasticity of the cytoskeleton: Actin filaments and microtubules

- Viscoelastic models of cells: Maxwell and Kelvin-Voigt representations
- Principles of Traction Force Microscopy (TFM)
- Mechanisms of cell motility: Lamellipodia protrusion and retraction
- Mechanotransduction: How cells convert mechanical stimuli into chemical signals
- Osmotic pressure and the regulation of cell volume

Learning Outcomes:

Core:

1. Calculate the Young's modulus of a cell using Atomic Force Microscopy (AFM) data [Usage]
2. Analyze the creep compliance of a cellular membrane under constant stress [Assessment]
3. Model the time-dependent swelling of a cell in a hypotonic solution [Usage]

5.11.3 BMP/Neural Physics and Information Processing in the Brain (Core Tier1: 1 hr, Core Tier2: 1 hr)

Electrical and statistical properties of neurons and neural networks.

Topics:

Core

- The Hodgkin-Huxley model of action potential propagation
- Equilibrium potentials: The Nernst and Goldman-Hodgkin-Katz equations
- Synaptic transmission and Leaky Integrate-and-Fire models
- Cable theory for signal attenuation in dendrites and axons
- Neural coding: Rate coding versus temporal coding strategies
- Hopfield networks and the physics of associative memory

Learning Outcomes:

Core:

1. Calculate the Nernst potential for Potassium ions across a neuronal membrane [Usage]
2. Analyze the role of Voltage-gated channels in the initiation of an action potential [Assessment]
3. Apply the cable equation to determine the length constant of a passive dendrite [Usage]

5.11.4 BMP/Imaging Physics (MRI, CT, PET, Ultrasound) (Core Tier1: 1 hr, Core Tier2: 1 hr)

Physical principles of non-invasive visualization of the human body.

Topics:

Core

- X-ray attenuation, Beer-Lambert law, and Computed Tomography (CT)
- Nuclear Magnetic Resonance (NMR) and the Bloch equations for MRI
- Contrast mechanisms: T1 and T2 relaxation times in MRI
- Acoustic impedance and reflection in Medical Ultrasound
- Physics of Positron Emission Tomography (PET) and coincidence detection
- Image reconstruction: Filtered Back Projection and Fourier methods

Learning Outcomes:

Core:

1. Calculate the CT number (Hounsfield units) for bone and soft tissue [Usage]
2. Contrast the appearance of a tumor in T1-weighted versus T2-weighted MRI images [Assessment]
3. Determine the axial resolution of an ultrasound probe based on pulse duration [Usage]

5.11.5 BMP/Radiation Therapy and Dosimetry (Core Tier1: 1 hr, Core Tier2: 1 hr)

Interaction of ionizing radiation with tissue for therapeutic purposes.

Topics:

Core

- Photoelectric effect, Compton scattering, and Pair production in tissue
- Definitions of Absorbed Dose (Gray), Equivalent Dose (Sievert), and KERMA

- Stopping power and the Bragg Peak in Proton Therapy
- Radiobiology: The Linear-Quadratic model for cell survival
- Principles of Intensity-Modulated Radiation Therapy (IMRT)
- Dosimetry protocols and ionization chamber calibration

Learning Outcomes:
Core:

1. Calculate the absorbed dose at a specific depth using a Percent Depth Dose (PDD) curve [Usage]
2. Compare the dose distribution of a 6MV X-ray beam against a 200 MeV proton beam [Assessment]
3. Apply the Linear-Quadratic model to estimate the effect of dose fractionation [Usage]

5.11.6 BMP/Statistical Mechanics of Biological Networks (Core Tier1: 1 hr, Core Tier2: 1 hr)

Collective behavior in gene regulation, metabolism, and protein interactions.
Topics:
Core

- Dynamics of gene regulatory networks: The Hill equation and Cooperativity
- Constraint-based modeling and Metabolic Flux Analysis (MFA)
- Gillespie algorithm for stochastic gene expression
- Identification of network motifs: Feed-forward and feedback loops
- Information theory in biological signaling: Mutual information and capacity
- Perfect adaptation and robustness in chemotaxis signaling

Learning Outcomes:
Core:

1. Model a genetic toggle switch using a pair of coupled differential equations [Usage]
2. Analyze how negative feedback loops suppress stochastic noise in gene expression [Assessment]
3. Calculate the information capacity of a simple biochemical sensor [Usage]

5.11.7 BMP/Microfluidics and Lab-on-a-Chip (Core Tier1: 1 hr, Core Tier2: 1 hr)

Physics of fluids at the micrometer scale for biological analysis.
Topics:
Core

- Low Reynolds number flow and the reversibility of Stokes flow
- Surface tension, Young-Laplace equation, and capillary action
- Electro-osmotic flow and Zeta potential in microchannels
- Droplet-based microfluidics and high-throughput screening
- Mixing at the microscale: Peclet number and diffusive limits
- Principles of Organ-on-a-Chip: Mimicking physiological microenvironments

Learning Outcomes:
Core:

1. Calculate the capillary rise height of blood in a 50-micrometer glass tube [Usage]
2. Evaluate the time required for a protein to diffuse across a microfluidic channel [Assessment]
3. Design a microfluidic mixer geometry to optimize mixing efficiency at $Re < 1$ [Usage]

5.12 Astrophysics and Space Science (ASS)

Astrophysics and Space Science apply the laws of physics to the study of the universe. This Knowledge Area covers the birth, life, and death of stars, the formation of planetary systems, the large-scale structure of galaxies, and the high-energy phenomena that define our cosmic environment.

Knowledge Area (KA)	Core Tier1	Core Tier2
⭐ 5.12.1 Stellar Structure and Evolution (Pg.. 255)	1	1
⭐ 5.12.2 Planetary Science and Exoplanets (Pg.. 255)	1	1
⭐ 5.12.3 Galactic Dynamics and Structure (Pg.. 255)	1	1
⭐ 5.12.4 High-Energy Astrophysics (X-rays, Gamma-rays) (Pg.. 256)	1	1
⭐ 5.12.5 Observational Techniques and Telescopes (Pg.. 256)	1	1
⭐ 5.12.6 Solar Physics and Space Weather (Pg.. 256)	1	1
⭐ 5.12.7 Multi-messenger Astronomy (Gravitational Waves + Neutrinos) (Pg.. 257)	1	1

Table 5.12: List of KUs in the Astrophysics and Space Science area.

5.12.1 ASS/Stellar Structure and Evolution (Core Tier1: 1 hr, Core Tier2: 1 hr)

The physical processes governing the life cycles of stars.
Topics:
Core

- Hydrostatic equilibrium and the Virial Theorem for stars
- Stellar nucleosynthesis: pp-chain, CNO cycle, and triple-alpha process
- The Hertzsprung-Russell (H-R) diagram and stellar classification
- Energy transport: Radiative transfer and convection zones
- Electron/Neutron degeneracy and the Chandrasekhar/Tolman-Oppenheimer-Volkoff limits
- Post-main sequence evolution: Red giants, white dwarfs, and supernovae

Learning Outcomes:
Core:

1. Estimate the main-sequence lifetime of a star based on its mass and luminosity [Usage]
2. Analyze why high-mass stars have convective cores while low-mass stars have radiative cores [Assessment]
3. Determine the maximum mass of a non-rotating white dwarf [Usage]

5.12.2 ASS/Planetary Science and Exoplanets (Core Tier1: 1 hr, Core Tier2: 1 hr)

Formation and characteristics of planets within and beyond our Solar System.
Topics:
Core

- The Nebular Hypothesis: Accretion and planetesimal formation
- Exoplanet detection: Transit photometry and light curve analysis
- Radial velocity method: Doppler shifts and the reflex motion of stars
- Atmospheric composition, greenhouse effects, and the Habitable Zone
- Tidal interactions: Orbital migration and synchronous rotation
- High-contrast direct imaging and nulling interferometry

Learning Outcomes:
Core:

1. Calculate the radius of an exoplanet given the transit depth of its parent star [Usage]
2. Evaluate whether a planet at 0.5 AU from a K-type star lies in the habitable zone [Assessment]
3. Estimate the minimum mass ($M \sin i$) of a planet from a radial velocity amplitude [Usage]

5.12.3 ASS/Galactic Dynamics and Structure (Core Tier1: 1 hr, Core Tier2: 1 hr)

The organization, kinematics, and evolution of galaxies.
Topics:
Core

- The Hubble tuning fork: Elliptical, Spiral, and Irregular galaxies
- Galactic rotation curves and evidence for Dark Matter
- Active Galactic Nuclei (AGN) and supermassive black holes
- Kinematics of stars and the Jeans equations for galactic equilibrium

- Spiral structure theory: Density wave theory vs. stochastic formation
- Galaxy interactions, mergers, and hierarchical structure formation

Learning Outcomes:

Core:

1. Calculate the required dark matter mass to explain a flat rotation curve at 20 kpc [Usage]
2. Compare the stellar populations and gas content of elliptical versus spiral galaxies [Assessment]
3. Apply the Virial Theorem to estimate the total mass of a galaxy cluster [Usage]

5.12.4 ASS/High-Energy Astrophysics (X-rays, Gamma-rays) (Core Tier1: 1 hr, Core Tier2: 1 hr)

Extreme physical processes and cosmic accelerators.

Topics:

Core

- Accretion disks around black holes and neutron stars: Eddington luminosity
- Non-thermal emission: Synchrotron and curvature radiation
- Inverse Compton scattering and the Sunyaev-Zel'dovich effect
- Gamma-Ray Bursts: Hypernovae and neutron star mergers
- Physics of pulsars: Magnetic dipole radiation and spin-down
- Fermi acceleration mechanism in supernova remnants

Learning Outcomes:

Core:

1. Calculate the Eddington luminosity for a 10 solar mass black hole [Usage]
2. Analyze the evolution of a pulsar's period due to energy loss from magnetic dipole radiation [Assessment]
3. Determine the maximum energy of a photon scattered by ultra-relativistic electrons [Usage]

5.12.5 ASS/Observational Techniques and Telescopes (Core Tier1: 1 hr, Core Tier2: 1 hr)

Methods for detecting and measuring electromagnetic radiation from space.

Topics:

Core

- Refracting and reflecting telescopes: Aperture and resolving power
- Principles of CCD detectors, quantum efficiency, and noise sources
- Spectroscopic techniques: Dispersion, resolution, and line profiles
- Radio astronomy: Aperture synthesis and VLBI techniques
- Adaptive optics: Wavefront sensing and deformable mirrors
- Instrumentation for X-ray and Gamma-ray space observatories

Learning Outcomes:

Core:

1. Calculate the signal-to-noise ratio for a star given a specific exposure time and sky background [Usage]
2. Evaluate the theoretical diffraction limit of a 30-meter ground-based telescope [Assessment]
3. Design an interferometric array configuration to achieve a specific angular resolution at 1.4 GHz [Usage]

5.12.6 ASS/Solar Physics and Space Weather (Core Tier1: 1 hr, Core Tier2: 1 hr)

The sun's activity and its impact on the heliosphere and Earth.

Topics:

Core

- Solar structure: Core, radiative zone, convection zone, and photosphere
- The 11-year solar cycle and the Babcock-Leighton dynamo model
- The Solar Wind: Parker's theory of isothermal expansion
- Solar flares and Coronal Mass Ejections (CMEs): Magnetic reconnection

- Space weather: Geomagnetic storms and ionospheric disturbances
- Helioseismology: Using acoustic waves to probe the solar interior

Learning Outcomes:

Core:

1. Calculate the solar constant (irradiance) at the distance of Mars [Usage]
2. Analyze how magnetic reconnection triggers the release of energy in solar flares [Assessment]
3. Model the velocity of the solar wind as a function of radial distance from the Sun [Usage]

5.12.7 ASS/Multi-messenger Astronomy (Gravitational Waves + Neutrinos) (Core Tier1: 1 hr, Core Tier2: 1 hr)

Integrating electromagnetic, gravitational, and particle signals from cosmic events.

Topics:

Core

- Sources of Gravitational Waves: Binary black hole and neutron star inspirals
- LIGO/Virgo interferometry: Michelson setup with Fabry-Perot cavities
- High-energy cosmic neutrinos and detection in IceCube
- Standard Sirens: Using GW signals to measure the Hubble constant
- The physical connection between ultra-high-energy cosmic rays and neutrinos

Elective

- Space-based GW detection: LISA and pulsar timing arrays

Learning Outcomes:

Core:

1. Calculate the chirp mass of a binary system from its gravitational wave frequency evolution [Usage]
2. Evaluate the expected neutrino flux from a nearby core-collapse supernova [Assessment]
3. Estimate the luminosity distance to a binary neutron star merger using GW strain data [Usage]

Chapter 6

Curriculum

6.1 Course coding

Courses are classified into 4 levels:

- **Introductory** Code 1.. example: CS100
- **Intermediate** Code 2.. example: MA200
- **Advanced** Code 3.. and example: IS370
- **Capstone project** Code 4.. example: CS403

Courses are coded under the scheme shown in Figure 6.1.

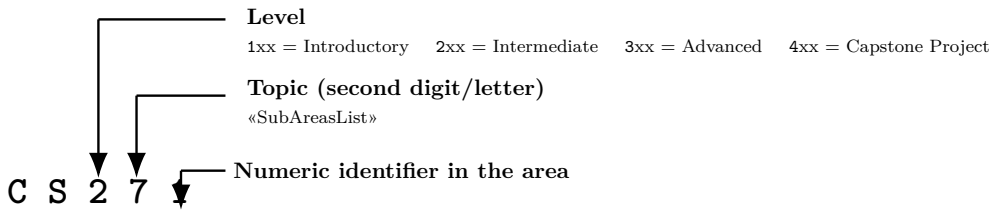


Figure 6.1: Course coding scheme.

The course type is determined by the initial letters. The possible codes for these types are:

1. **CS** Área de Ciencia de la Computación (*Computer Science* – CS);
2. **AI** Área de Inteligencia Artificial (*Artificial Intelligence* – AI);
3. **MA** Matemáticas (MA);
4. **STA** Estadística (STA);
5. **FI** Área de Física;
6. **QC** Quantum Computing
7. **FG** Área de Formación General;
8. **ID** Área de Idiomas.

At the same time, courses are classified into:

- General studies
- Specific
- Specialization

6.2 Estructura Curricular

La relación de cursos se muestra a continuación:

TH = Theory Hours, PR = Practice Hours, CR = Credits, T = Type of Course.

M = Mandatory, E = Elective.

Primer Semestre

Código	Curso	Área	HT	HP	CR	T	Requisitos
CS111	Introduction to Programming (Pág 295)	Specific	3	4	5	O	
MA100	Basic Mathematics (Pág 300)	General Estudies	3	4	5	O	
FG101	Language (Pág 303)	General Estudies	1	2	2	O	
FG105	Gender and Society (Pág 306)	General Estudies	1	2	2	O	
FG109	Environment and Sustainable Development (Pág 309)	General Estudies	2	2	3	O	
FG112	Methods of managing university personnel (Pág 312)	General Estudies	1	2	2	O	
FG140	Fundamentals of Scientific Research (Pág 315)	General Estudies	2	2	3	O	
					22		

Segundo Semestre

Código	Curso	Área	HT	HP	CR	T	Requisitos
CS100	Introduction to Computer Science (Pág 318)	Specific	2	2	3	O	
CS112	Objects-oriented programming I (Pág 328)	Specific	2	4	4	O	CS111 (1^{er} Sem)
AI161	Applied AI (Pág 333)	Specific	2	4	4	O	CS111 (1^{er} Sem)
MA111	Calculus I (Pág 338)	General Estudies	3	4	5	O	MA100 (1^{er} Sem)
FI101	General Physics (Pág 340)	General Estudies	3	4	5	O	MA111 (2^{do} Sem)
					21		

Tercer Semestre							
Código	**Curso**	**Área**	**HT**	**HP**	**CR**	**T**	**Requisitos**
CS113	Objects-oriented programming II (Pág 343)	Specific	1	4	3	O	CS112 (2do Sem)
CS1D1	Discrete Structures I (Pág 349)	Specific	2	4	4	O	MA100 (1er Sem)
CS2B1	Platform Based Development (Pág 352)	Specific	2	4	4	O	CS112 (2do Sem)
MA112	Calculus II (Pág 356)	General Estudies	3	2	4	O	MA111 (2do Sem)
MA121	Linear Algebra I (Pág 358)	Specific	2	4	4	O	MA100 (1er Sem)
ID101	Technical and professional English I (Pág 361)	General Estudies	1	4	3	O	
					22		

Cuarto Semestre							
Código	**Curso**	**Área**	**HT**	**HP**	**CR**	**T**	**Requisitos**
CS1D2	Discrete Structures II (Pág 365)	Specific	2	2	3	O	CS1D1 (3er Sem)
CS210	Algorithms and Data Structures (Pág 367)	Specific	2	4	4	O	CS113 (3er Sem)
CS271	Data Management (Pág 372)	Specific	1	4	3	O	CS112 (2do Sem), CS1D1 (3er Sem)
MA122	Linear Algebra II (Pág 377)	Specific	2	4	4	O	MA121 (3er Sem)
MA210	Calculus III (Pág 380)	General Estudies	2	4	4	O	MA112 (3er Sem)
ID102	Technical and professional English II (Pág 383)	General Estudies	1	6	4	O	ID101 (3er Sem)
					22		

Quinto Semestre

Código	Curso	Área	HT	HP	CR	T	Requisitos
CS251	Computer graphics (Pág 387)	Specialization	1	4	3	O	CS210 (4^{to} Sem)
MA211	Calculus IV (Pág 390)	Specific	1	4	3	O	MA210 (4^{to} Sem), MA121 (3^{er} Sem)
MA221	Numerical Analysis I[1] (Pág 393)	Specific	1	4	3	O	MA112 (3^{er} Sem), MA122 (4^{to} Sem), AI161 (2^{do} Sem)
MA222	Real Analysis I (Pág 396)	Specific	3	4	5	O	MA112 (3^{er} Sem), ID101 (3^{er} Sem)
MA251	Introduction to Statistics (Pág 399)	General Estudies	1	4	3	O	AI161 (2^{do} Sem)
MA281	Numbers theory (Pág 401)	General Estudies	1	2	2	O	MA100 (1^{er} Sem)
ID201	Technical and professional English III (Pág 405)	General Estudies	1	4	3	O	ID102 (4^{to} Sem)
					22		

Sexto Semestre

Código	Curso	Área	HT	HP	CR	T	Requisitos
MA212	Ordinary Differential Equations (Pág 409)	Specific	4	4	6	O	CS112 (2^{do} Sem), MA121 (3^{er} Sem), MA222 (5^{to} Sem)
MA325	Numerical Analysis II (Pág 412)	Specific	2	4	4	O	MA221 (5^{to} Sem)
MA351	Statistics and Probabilities (Pág 415)	Specific	2	4	4	O	MA251 (5^{to} Sem), CS112 (2^{do} Sem)
MA371	Functional analysis I (Pág 420)	Specific	3	4	5	O	MA222 (5^{to} Sem), ID201 (5^{to} Sem), MA122 (4^{to} Sem)
ID202	Technical and professional English IV (Pág 423)	General Estudies	1	4	3	O	ID201 (5^{to} Sem)
					22		

[1]Se recomienda que el alumno haya llevado: CS210 Algorithms and Data Structures(4^{th} Sem).

Séptimo Semestre							
Código	**Curso**	**Área**	**HT**	**HP**	**CR**	**T**	**Requisitos**
CS261	Artificial Intelligence (Pág 427)	Specialization	1	4	3	O	MA251 (5^{to} Sem)
CS3P1	Parallel and Distributed Computing (Pág 434)	Specialization	1	4	3	O	CS210 (4^{to} Sem)
MA213	Partial Differential Equations (Pág 443)	Specific	2	4	4	O	MA212 (6^{to} Sem), MA211 (5^{to} Sem)
MA326	Numerical Analysis III (Pág 446)	Specific	2	4	4	O	MA325 (6^{to} Sem)
MA361	Mathematical Modeling (Pág 449)	General Estudies	2	4	4	O	MA325 (6^{to} Sem)
ID203	Technical and professional English V (Pág 452)	General Estudies	1	6	4	O	ID202 (6^{to} Sem)
					22		

	Octavo Semestre							
	Código	**Curso**	**Área**	**HT**	**HP**	**CR**	**T**	**Requisitos**
	AI263	Introduction to Machine Learning (Pág 455)	Specific	2	4	4	O	CS261 (7^{mo} Sem)
	MA401	Capstone Project I (Pág 458)	Specialization	1	4	3	O	MA361 (7^{mo} Sem), MA371 (6^{to} Sem), 140cr
Elective	CS2S1	Operating systems (Pág 461)	Specific	3	4	5	E	CS210 (4^{to} Sem)
Elective	CS3I1	Computer Security (Pág 466)	Specific	3	4	5	E	CS210 (4^{to} Sem)
	QC251	Quantum Mechanics (Pág 472)	Specialization	4	2	5	E	FI101 (2^{do} Sem), MA371 (6^{to} Sem)
Elective	CS253	Exploratory data analysis (Pág 476)	Specialization	3	4	5	E	MA351 (6^{to} Sem)
Elective	CS370	Big Data and Analytics (Pág 480)	Specialization	3	4	5	E	CS271 (4^{to} Sem)
Elective	STA252	Inferential Statistics (Pág 484)	Specialization	3	4	5	E	MA351 (6^{to} Sem)
Elective	MA327	Numerical Analysis IV (Pág 487)	Specific	4	2	5	E	MA326 (7^{mo} Sem), MA213 (7^{mo} Sem)
Elective	MA352	Stochastic processes (Pág 490)	Specific	3	4	5	E	MA351 (6^{to} Sem), MA210 (4^{to} Sem), CS261 (7^{mo} Sem)
Elective	MA372	Functional analysis II (Pág 493)	Specific	3	4	5	E	MA371 (6^{to} Sem)
						7		

	Noveno Semestre							
	Código	**Curso**	**Área**	**HT**	**HP**	**CR**	**T**	**Requisitos**
	AI264	Deep Learning (Pág 496)	Specific	3	4	5	O	AI263 (8vo Sem)
	MA402	Capstone Project II (Pág 499)	Specialization	1	8	5	O	MA401 (8vo Sem), ID203 (7mo Sem)
Elective	CS231	Networking and Communication (Pág 503)	Specific	2	4	4	E	CS2S1 (8vo Sem)
	MA381	Algebraic Structures (Pág 508)	General Estudies	2	4	4	E	MA281 (5to Sem)
Elective	AI262	Natural Language Processing (Pág 512)	Specific	2	4	4	E	AI263 (8vo Sem)
	CS273	Time series analysis (Pág 515)	Specialization	2	4	4	E	MA351 (6to Sem)
Elective	MA341	Optimization I (Pág 519)	Specialization	2	4	4	E	MA221 (5to Sem)
	MA355	Statistical Learning Theory (Pág 522)	General Estudies	2	4	4	E	MA352 (8vo Sem), MA372 (8vo Sem)
						10		

	Décimo Semestre							
	Código	**Curso**	**Área**	**HT**	**HP**	**CR**	**T**	**Requisitos**
	CS400	Pre-professional internships (Pág 525)	Specialization	1	4	3	O	110Cr
	AI268	Computational Vision (Pág 527)	Specialization	3	4	5	O	AI264 (9no Sem)
	AI367	Topics in Artificial Intelligence (Pág 533)	Specialization	3	4	5	O	AI264 (9no Sem)
	MA403	Capstone Project III (Pág 539)	Specialization	1	8	5	O	MA402 (9no Sem)
Elective	QC381	Introduction to Quantum Computing (Pág 543)	Specialization	2	4	4	E	QC251 (8vo Sem), MA381 (9no Sem)
Elective	CS254	Advanced Data Visualization (Pág 547)	Specialization	2	4	4	E	CS253 (8vo Sem)
						18		

Total de créditos de la carrera: 188 .

6.3 Topics distributed by course

6.3.1 Computational Mathematics topics/course

		First Sem							Second Sem					
	Knowledge Units of Computational Mathematics **Courses →**	CS111	MA100	FG101	FG105	FG109	FG112	FG140	CS100	CS112	AI161	MA111	FI101	Total
⭐	NSA/Computational Arithmetic and Error Analysis (1 hour Core-Tier1, 1 hour Core-Tier2)											25		25
⭐	NSA/Roots Finding and Basic Numerical Optimization (1 hour Core-Tier1, 1 hour Core-Tier2)											25		25
	Total	0	0	0	0	0	0	0	0	0	0	50	0	

Table 6.1: Computational Mathematics topics by course from 1st to 2nd Semester

		Third Sem						Fourth Sem						
	Knowledge Units of Computational Mathematics **Courses →**	CS113	CS1D1	CS2B1	MA112	MA121	ID101	CS1D2	CS210	CS271	MA122	MA210	ID102	Total
★	NSA/Computational Arithmetic and Error Analysis (1 hour Core-Tier1, 1 hour Core-Tier2)					25					16			41
★	NSA/Numerical Linear Algebra (1 hour Core-Tier1, 1 hour Core-Tier2)					25					16			41
★	NSA/Roots Finding and Basic Numerical Optimization (1 hour Core-Tier1, 1 hour Core-Tier2)					25						20		45
★	NSA/Interpolation and Function Approximation (1 hour Core-Tier1, 1 hour Core-Tier2)				25	25					16	20		86
★	NSA/Numerical Differentiation and Integration (1 hour Core-Tier1, 1 hour Core-Tier2)				25							20		45
★	NSA/Numerical Methods for Ordinary Differential Equations (ODEs) (1 hour Core-Tier1, 1 hour Core-Tier2)										16	20		36
	Total	0	0	0	50	100	0	0	0	0	64	80	0	

Table 6.2: Computational Mathematics topics by course from 3rd to 4th Semester

		Fifth Sem							Sixth Sem					.
	Knowledge Units of Computational Mathematics **Courses →**	CS251	MA211	MA221	MA222	MA251	MA281	ID201	MA212	MA325	MA351	MA371	ID202	Total
⭐	NSA/Computational Arithmetic and Error Analysis (1 hour Core-Tier1, 1 hour Core-Tier2)			20	20									40
⭐	NSA/Numerical Linear Algebra (1 hour Core-Tier1, 1 hour Core-Tier2)			20						20				40
⭐	NSA/Roots Finding and Basic Numerical Optimization (1 hour Core-Tier1, 1 hour Core-Tier2)		20	20	20									60
⭐	NSA/Interpolation and Function Approximation (1 hour Core-Tier1, 1 hour Core-Tier2)		20	20	20					20				80
⭐	NSA/Numerical Differentiation and Integration (1 hour Core-Tier1, 1 hour Core-Tier2)		20		20					20				60
⭐	NSA/Numerical Methods for Ordinary Differential Equations (ODEs) (1 hour Core-Tier1, 1 hour Core-Tier2)		20							20				40
	Total	0	80	80	80	0	0	0	0	80	0	0	0	

Table 6.3: Computational Mathematics topics by course from 5th to 6th Semester

		Seventh Sem						.
	Knowledge Units of Computational Mathematics **Courses →**	CS261	CS3P1	MA213	MA326	MA361	ID203	Total
⭐	MMS/Mathematical Model Formulation (1 hour Core-Tier1, 1 hour Core-Tier2)					16		16
⭐	MMS/Discrete Systems Simulation (1 hour Core-Tier1, 1 hour Core-Tier2)					16		16
⭐	MMS/Continuous Systems Simulation (1 hour Core-Tier1, 1 hour Core-Tier2)					16		16
⭐	MMS/Sensitivity Analysis and Robustness (1 hour Core-Tier1, 1 hour Core-Tier2)					16		16
⭐	MMS/Model Validation and Verification (1 hour Core-Tier1, 1 hour Core-Tier2)					16		16
	Total	0	0	0	0	80	0	

Table 6.4: Computational Mathematics topics by course 7th Semester

Eighth Sem

Knowledge Units of Computational Mathematics — Courses →	AI263	MA401	CS251	CS311	QC251	CS253	CS370	STA252	MA327	MA352	MA372	Total
MMS/Discrete Systems Simulation (1 hour Core-Tier1, 1 hour Core-Tier2)										20		20
MMS/Continuous Systems Simulation (1 hour Core-Tier1, 1 hour Core-Tier2)										20		20
PSC/Advanced Probability and Stochastic Processes (1 hour Core-Tier1, 1 hour Core-Tier2)										20		20
PSC/Computational Statistical Inference (1 hour Core-Tier1, 1 hour Core-Tier2)										20		20
Total	0	0	0	0	0	0	0	0	0	80	0	

Table 6.5: Computational Mathematics topics by course 8th Semester

		Ninth Sem								Tenth Sem						
	Knowledge Units of Computational MathematicsCourses →	AI264	MA402	CS231	MA381	AI262	CS273	MA341	MA355	CS400	AI268	AI367	MA403	QC381	CS254	Total
⭐	OML/Convex and Non-Convex Optimization (1 hour Core-Tier1, 1 hour Core-Tier2)							20								20
⭐	OML/Constrained Optimization (1 hour Core-Tier1, 1 hour Core-Tier2)							20								20
⭐	OML/Optimization for Machine Learning (1 hour Core-Tier1, 1 hour Core-Tier2)							20								20
⭐	OML/Metaheuristics and Bio-Inspired Optimization (1 hour Core-Tier1, 1 hour Core-Tier2)							20								20
⭐	DSM/Foundations of Statistical Learning (1 hour Core-Tier1, 1 hour Core-Tier2)								20							20
⭐	DSM/Supervised Learning Methods (1 hour Core-Tier1, 1 hour Core-Tier2)								20							20
⭐	DSM/Unsupervised Learning Methods (1 hour Core-Tier1, 1 hour Core-Tier2)								20							20
⭐	DSM/Model Evaluation and Selection (1 hour Core-Tier1, 1 hour Core-Tier2)								20							20
	Total	0	0	0	0	0	0	80	80	0	0	0	0	0	0	

Table 6.6: Computational Mathematics topics by course from 9th to 10th Semester

6.3.2 Computer Science topics/course

		First Sem							Second Sem					
	Knowledge Units of Computer Science **Courses →**	CS111	MA100	FG101	FG105	FG109	FG112	FG140	CS100	CS112	AI161	MA111	FI101	Total
★	AI/Machine Learning Fundamentals (1 hour Core-Tier1, 2 hours Core-Tier2)								1					1
★	AI/Neural Networks and Ethics (1 hour Core-Tier1, 1 hour Core-Tier2)								1					1
★	AL/Foundational Data Structures (5 hours Core-Tier1, 3 hours Core-Tier2)								3					3
★	AL/Foundational Algorithms (4 hours Core-Tier1, 2 hours Core-Tier2)								3					3
★	AL/Advanced Algorithms (3 hours Core-Tier1, 2 hours Core-Tier2)								2					2
★	AL/Algorithmic Strategies (6 hours Core-Tier1)	6												6
★	AL/Complexity Analysis Framework (2 hours Core-Tier1, 1 hour Core-Tier2)								2					2
★	AL/Asymptotic Notation and Complexity Classes (1 hour Core-Tier1, 1 hour Core-Tier2)								1					1
★	AL/Complexity Analysis: Recursion, Amortization, and Tight Bounds (2 hours Core-Tier1, 1 hour Core-Tier2)								1					1
★	AR/Machine-Level Data Representation (1 hour Core-Tier1)								4					4
★	DM/Core Database System Concepts (2 hours Core-Tier1, 1 hour Core-Tier2)								4					4
★	FPL/Object-Oriented Programming Fundamentals (2 hours Core-Tier1, 1 hour Core-Tier2)	24							4	14				42
★	FPL/Object-Oriented Programming: Encapsulation, Subtyping, and Reflection (2 hours Core-Tier1, 1 hour Core-Tier2)									8				8
★	FPL/Functional Programming (4 hours Core-Tier1, 3 hours Core-Tier2)									10				10
★	FPL/Shell Scripting (2 hours Core-Tier1)									4				4
★	FPL/Type Systems Fundamentals (2 hours Core-Tier1, 2 hours Core-Tier2)	6								8				14
★	FPL/Systems Execution and Memory Model (2 hours Core-Tier1)									12				12
★	FPL/Language Translation and Execution (2 hours Core-Tier1, 3 hours Core-Tier2)									4				4
★	FPL/Program Abstraction and Representation (3 hours Core-Tier2)								2					2
	FPL/Advanced Programming Constructs									6				6
	FPL/Language Pragmatics									2				2
★	GIT/Fundamental Concepts of Graphics and Interactive Techniques (4 hours Core-Tier1, 3 hours Core-Tier2)								2					2
★	GIT/Applied Rendering and Techniques (15 hours Core-Tier2)								2					2
★	NC/Fundamentals of Networks and Communications (3 hours Core-Tier1)								4					4

Table 6.7: Computer Science topics by course from 1st to 2nd Semester (Part 1 of 2)

	Knowledge Units of Computer Science	Courses →	CS111	MA100	FG101	FG105	FG109	FG112	FG140	CS100	CS112	AI161	MA111	FI101	Total
			First Sem							Second Sem					
★	NC/Networked Applications (4 hours Core-Tier1)									4					4
★	OS/Role and Purpose of Operating Systems (2 hours Core-Tier1)									2					2
★	OS/Principles of Operating System (2 hours Core-Tier1)									2					2
★	OS/Memory Management (1 hour Core-Tier2)									4					4
★	SE/Version Control and CI/CD (1 hour Core-Tier1, 2 hours Core-Tier2)									2					2
★	SE/Requirements Engineering (2 hours Core-Tier2)									2					2
★	SEC/Cryptography Fundamentals (1 hour Core-Tier1, 2 hours Core-Tier2)									2					2
	Total		36	0	0	0	0	0	0	54	68	0	0	0	0

Table 6.8: Computer Science topics by course from 1st to 2nd Semester (Part 2 of 2)

		Third Sem						Fourth Sem						.
	Knowledge Units of Computer Science **Courses →**	CS113	CS1D1	CS2B1	MA112	MA121	ID101	CS1D2	CS210	CS271	MA122	MA210	ID102	Total
★	AL/Foundational Data Structures (5 hours Core-Tier1, 3 hours Core-Tier2)								32					32
★	DM/Core Database System Concepts (2 hours Core-Tier1, 1 hour Core-Tier2)									8				8
★	DM/Data Modeling (2 hours Core-Tier1, 3 hours Core-Tier2)									8				8
★	DM/Relational Databases (1 hour Core-Tier1, 3 hours Core-Tier2)									16				16
★	DM/Query Construction (2 hours Core-Tier1, 4 hours Core-Tier2)									16				16
★	DM/DBMS Internals (4 hours Core-Tier2)									8				8
★	DM/Data Security and Privacy (1 hour Core-Tier1, 2 hours Core-Tier2)									8				8
★	FPL/Object-Oriented Programming: Encapsulation, Subtyping, and Reflection (2 hours Core-Tier1, 1 hour Core-Tier2)	8												8
★	FPL/Parallel and Distributed Computing (2 hours Core-Tier1, 2 hours Core-Tier2)	12												12
★	FPL/Type Systems: Polymorphism and Complementary Typing (1 hour Core-Tier1, 1 hour Core-Tier2)	12												12
★	FPL/Systems Execution and Memory Model (2 hours Core-Tier1)	8												8
★	FPL/Memory Management (1 hour Core-Tier1)	8												8
	FPL/Advanced Programming Constructs	20												20
★	MSF/Discrete Mathematics (15 hours Core-Tier1, 6 hours Core-Tier2)		48					48						96
★	SE/Version Control and CI/CD (1 hour Core-Tier1, 2 hours Core-Tier2)	4												4
★	SPD/Common Aspects: Platforms, APIs and Constraints (3 hours Core-Tier1, 2 hours Core-Tier2)			5										5
★	SPD/Common Aspects: Platform Languages and Web Programming Patterns (2 hours Core-Tier1, 1 hour Core-Tier2)			3										3
★	SPD/Frontend, Frameworks and Web Standards (3 hours Core-Tier2)			12										12
★	SPD/Mobile Platforms: Development Fundamentals (2 hours Core-Tier2)			6										6
★	SPD/Mobile Platforms: Architecture and Constraints (2 hours Core-Tier2)			3										3
★	SPD/Mobile Platforms: Capabilities and Asynchronous Computing (1 hour Core-Tier2)			3										3
★	SPD/Interactive Data and Analytics Platforms (2 hours Core-Tier2)			6										6
	SPD/SPD-SEP/Interactive			10										10
	Total	72	48	48	0	0	0	48	32	64	0	0	0	

Table 6.9: Computer Science topics by course from 3rd to 4th Semester

Knowledge Units of Computer Science	Courses →	CS251	MA211	MA221	MA222	MA251	MA281	ID201	MA212	MA325	MA351	MA371	ID202	Total
		Fifth Sem							Sixth Sem					
⭐	GIT/Fundamental Concepts of Graphics and Interactive Techniques (4 hours Core-Tier1, 3 hours Core-Tier2)	16												16
⭐	GIT/Visualization (6 hours Core-Tier2)													16
⭐⭐	GIT/Applied Rendering and Techniques (15 hours Core-Tier2)													16
⭐⭐	GIT/Geometric Modeling (6 hours Core-Tier2)													16
⭐⭐	GIT/Shading and Advanced Rendering (6 hours Core-Tier2)													12
	Total	16	0	0	0	0	0	0	0	0	0	0	0	76

Table 6.10: Computer Science topics by course from 5th to 6th Semester

	Knowledge Units of Computer Science / Courses →	Seventh Sem CS261	CS3P1	MA213	MA326	MA361	ID203	Total
⭐	AI/Introduction to AI: Concepts and Intelligent Behavior (2 hours Core-Tier1, 1 hour Core-Tier2)	4						4
⭐	AI/Problem Characteristics and Agent Nature (1 hour Core-Tier1, 1 hour Core-Tier2)	4						4
⭐	AI/State and Uninformed Search (1 hour Core-Tier1, 2 hours Core-Tier2)	5						5
⭐	AI/Heuristic Search and CSP (1 hour Core-Tier1, 2 hours Core-Tier2)	6						6
⭐	AI/Adversarial Search and Stochastic Optimization (1 hour Core-Tier1, 3 hours Core-Tier2)	5						5
⭐	AI/Knowledge Representation and Probabilistic Reasoning Foundations (1 hour Core-Tier1, 1 hour Core-Tier2)	6						6
⭐	AI/Independence, Markov Models and Decision Making (1 hour Core-Tier1, 1 hour Core-Tier2)	6						6
⭐	AI/Machine Learning Fundamentals (1 hour Core-Tier1, 2 hours Core-Tier2)	7						7
⭐	AI/Machine Learning Data, Representations and Evaluation (1 hour Core-Tier1, 2 hours Core-Tier2)	7						7
⭐	AI/Neural Networks and Ethics (1 hour Core-Tier1, 1 hour Core-Tier2)	6						6
⭐	AI/Machine Learning Formulation and Evaluation (1 hour Core-Tier1, 2 hours Core-Tier2)	3						3
⭐	AI/Deep Learning, RL and Ethics (1 hour Core-Tier1, 2 hours Core-Tier2)	3						3
⭐	AI/AI Applications and Societal Impact (2 hours Core-Tier1, 2 hours Core-Tier2)	4						4
⭐	AI/Additional AI Applications and Societal Impact (2 hours Core-Tier1, 2 hours Core-Tier2)	4						4
⭐	PDC/Programs: Declarative Parallelism (1 hour Core-Tier1, 1 hour Core-Tier2)		4					4
⭐	PDC/Programs: Starting Activities (1 hour Core-Tier1, 1 hour Core-Tier2)		3					3
⭐	PDC/Programs: Execution Properties (1 hour Core-Tier1, 1 hour Core-Tier2)		3					3
⭐	PDC/Programs: Distribution (1 hour Core-Tier1, 1 hour Core-Tier2)		2					2
⭐	PDC/Programs: Implementation Mappings (1 hour Core-Tier1, 1 hour Core-Tier2)		2					2
	PDC/GPU Programming		4					4
⭐	PDC/Communication (2 hours Core-Tier1, 4 hours Core-Tier2)		8					8
⭐	PDC/Communication: Properties and Extensions (1 hour Core-Tier1, 1 hour Core-Tier2)		3					3
⭐	PDC/Memory and Consistency (1 hour Core-Tier1, 1 hour Core-Tier2)		3					3
⭐	PDC/Coordination (1 hour Core-Tier1, 3 hours Core-Tier2)		7					7
⭐	PDC/Coordination: Synchronization and Atomicity (1 hour Core-Tier1, 2 hours Core-Tier2)		3					3
⭐	PDC/Coordination: Advanced Properties (1 hour Core-Tier1, 2 hours Core-Tier2)		4					4

Table 6.11: Computer Science topics by course 7th Semester (Part 1 of 2)

Seventh Sem								
	Courses →	CS261	CS3P1	MA213	MA326	MA361	ID203	Total
	Knowledge Units of Computer Science							
⭐	PDC/Evaluation (1 hour Core-Tier1, 3 hours Core-Tier2)		7					7
⭐⭐	PDC/Evaluation: Formal Methods (1 hour Core-Tier1, 1 hour Core-Tier2)		3					3
⭐	PDC/Evaluation: Testing and Measurement (1 hour Core-Tier1, 1 hour Core-Tier2)		2					2
⭐⭐	PDC/Algorithms (1 hour Core-Tier1, 3 hours Core-Tier2)		4					4
⭐⭐	PDC/Algorithms: Application Domains Survey (1 hour Core-Tier1, 3 hours Core-Tier2)		4					4
⭐⭐	PDC/Algorithms: Algorithmic Domains (1 hour Core-Tier1, 3 hours Core-Tier2)		4					4
	Total	70	70	0	0	0	0	

Table 6.12: Computer Science topics by course 7^{th} Semester (Part 2 of 2)

Eighth Sem .

	Knowledge Units of Computer Science — Courses →	AI263	MA401	CS2S1	CS3I1	QC251	CS253	CS370	STA252	MA327	MA352	MA372	Total
★	DM/Core Database System Concepts (2 hours Core-Tier1, 1 hour Core-Tier2)							12					12
★	DM/NoSQL Systems (2 hours Core-Tier2)							12					12
★	DM/Data Analytics (3 hours Core-Tier2)							12					12
	DM/Distributed Databases/Cloud Computing							12					12
★	FPL/Shell Scripting (2 hours Core-Tier1)			8									8
★	OS/Role and Purpose of Operating Systems (2 hours Core-Tier1)			8									8
★	OS/Protection and Safety (2 hours Core-Tier1, 1 hour Core-Tier2)			12									12
★	OS/Scheduling (2 hours Core-Tier2)			16									16
★	OS/Memory Management (1 hour Core-Tier2)			16									16
★	OS/File Systems API and Implementation (2 hours Core-Tier2)			12									12
★	SEC/Security Mindset and Principles (1 hour Core-Tier1, 3 hours Core-Tier2)				6								6
★	SEC/Security Ethics, Law and Society (1 hour Core-Tier1, 2 hours Core-Tier2)				6								6
★	SEC/Injection and Input Validation (1 hour Core-Tier1, 2 hours Core-Tier2)				8								8
★	SEC/Malware Analysis and Advanced Security (1 hour Core-Tier1, 3 hours Core-Tier2)				6								6
★	SEC/Cryptography Fundamentals (1 hour Core-Tier1, 2 hours Core-Tier2)				8								8
★	SEC/Threat Analysis and Security Engineering (1 hour Core-Tier1, 4 hours Core-Tier2)				8								8
★	SEC/Digital Forensics Techniques (4 hours Core-Tier2)				6								6
★	SEC/Security Governance, Management and Policy (2 hours Core-Tier2)				6								6
	Total	0	0	72	54	0	0	48	0	0	0	0	

Table 6.13: Computer Science topics by course 8th Semester

		Ninth Sem								Tenth Sem						
	Knowledge Units of Computer Science **Courses →**	AI264	MA402	CS231	MA381	AI262	CS273	MA341	MA355	CS400	AI268	AI367	MA403	QC381	CS254	Total
★	AI/Introduction to AI: Concepts and Intelligent Behavior (2 hours Core-Tier1, 1 hour Core-Tier2)										5					5
★	AI/Problem Characteristics and Agent Nature (1 hour Core-Tier1, 1 hour Core-Tier2)										5					5
★	AI/Machine Learning Fundamentals (1 hour Core-Tier1, 2 hours Core-Tier2)										5	12				17
★	AI/Machine Learning Data, Representations and Evaluation (1 hour Core-Tier1, 2 hours Core-Tier2)										5	10				15
★	AI/Neural Networks and Ethics (1 hour Core-Tier1, 1 hour Core-Tier2)										4	10				14
★	AI/Machine Learning Formulation and Evaluation (1 hour Core-Tier1, 2 hours Core-Tier2)										3	3				6
★	AI/Deep Learning, RL and Ethics (1 hour Core-Tier1, 2 hours Core-Tier2)										3	3				6
★	AI/Neural Network Training and Optimization (1 hour Core-Tier1, 1 hour Core-Tier2)	10														10
★	AI/Regularization Techniques for Deep Networks (1 hour Core-Tier1, 1 hour Core-Tier2)	5														5
★	AI/Convolutional and Residual Network Architectures (1 hour Core-Tier1, 1 hour Core-Tier2)	9														9
★	AI/Transformer Architectures and Generative Models (1 hour Core-Tier1, 1 hour Core-Tier2)	16														16
★	AI/AI Applications and Societal Impact (2 hours Core-Tier1, 2 hours Core-Tier2)											6				6
★	AI/Additional AI Applications and Societal Impact (2 hours Core-Tier1, 2 hours Core-Tier2)											6				6
	AI/Natural Language Processing											20				20
	AI/Perception and Computer Vision										24					24
★	AL/Algorithmic Strategies (6 hours Core-Tier1)													12		12
★	AL/Formal Languages and Automata (4 hours Core-Tier1, 10 hours Core-Tier2)													10		10
★	AR/Quantum Architectures (2 hours Core-Tier2)													10		10
★	NC/Fundamentals of Networks and Communications (3 hours Core-Tier1)			6												6
★	NC/Networked Applications (4 hours Core-Tier1)			6												6
★	NC/Reliability Support (6 hours Core-Tier2)			6												6
★	NC/Routing and Forwarding (4 hours Core-Tier2)			6												6
★	NC/Single Hop Communication (3 hours Core-Tier2)			6												6
★	NC/Network Security (3 hours Core-Tier2)			6												6

		Ninth Sem								Tenth Sem						.
Knowledge Units of Computer Science	**Courses →**	AI264	MA402	CS231	MA381	AI262	CS273	MA341	MA355	CS400	AI268	AI367	MA403	QC381	CS254	Total
Total		40	0	48	0	0	0	0	0	0	54	70	0	32	0	

Table 6.15: Computer Science topics by course from 9th to 10th Semester (Part 2 of 2)

6.3.3 Mathematics topics/course

		First Sem							Second Sem					.
Knowledge Units of Mathematics	**Courses →**	CS111	MA100	FG101	FG105	FG109	FG112	FG140	CS100	CS112	AI161	MA111	FI101	Total
Total		0	0	0	0	0	0	0	0	0	0	0	0	

Table 6.16: Mathematics topics by course from 1st to 2nd Semester

		Third Sem							Fourth Sem					.
Knowledge Units of Mathematics	**Courses →**	CS113	CS1D1	CS2B1	MA112	MA121	ID101	CS1D2	CS210	CS271	MA122	MA210	ID102	Total
Total		0	0	0	0	0	0	0	0	0	0	0	0	

Table 6.17: Mathematics topics by course from 3rd to 4th Semester

		Fifth Sem							Sixth Sem					
	Knowledge Units of Mathematics **Courses →**	CS251	MA211	MA221	MA222	MA251	MA281	ID201	MA212	MA325	MA351	MA371	ID202	Total
⭐	NSA/Numerical Methods for Differential Equations (1 hour Core-Tier1, 1 hour Core-Tier2)								36					36
⭐	MMS/First-Order Ordinary Differential Equations (1 hour Core-Tier1, 1 hour Core-Tier2)								20					20
⭐	MMS/Higher-Order Differential Equations and Systems (1 hour Core-Tier1, 1 hour Core-Tier2)								40					40
	Total	0	0	0	0	0	0	0	96	0	0	0	0	

Table 6.18: Mathematics topics by course from 5th to 6th Semester

		Seventh Sem						
	Knowledge Units of Mathematics **Courses →**	CS261	CS3P1	MA213	MA326	MA361	ID203	Total
	Total	0	0	0	0	0	0	

Table 6.19: Mathematics topics by course 7th Semester

		Eighth Sem										
	Knowledge Units of Mathematics **Courses →**	AI263	MA401	CS2S1	CS3I1	QC251	CS253	CS370	STA252	MA327	MA352	MA372	Total
	Total	0	0	0	0	0	0	0	0	0	0	0	

Table 6.20: Mathematics topics by course 8th Semester

		Ninth Sem								Tenth Sem						.
Knowledge Units of Mathematics	**Courses →**	AI264	MA402	CS231	MA381	AI262	CS273	MA341	MA355	CS400	AI268	AI367	MA403	QC381	CS254	Total
Total		0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Table 6.21: Mathematics topics by course from 9th to 10th Semester

6.3.4 Physics topics/course

		First Sem							Second Sem					.
Knowledge Units of Physics	**Courses →**	CS111	MA100	FG101	FG105	FG109	FG112	FG140	CS100	CS112	AI161	MA111	FI101	Total
Total		0	0	0	0	0	0	0	0	0	0	0	0	

Table 6.22: Physics topics by course from 1st to 2nd Semester

		Third Sem						Fourth Sem						.
Knowledge Units of Physics	**Courses →**	CS113	CS1D1	CS2B1	MA112	MA121	ID101	CS1D2	CS210	CS271	MA122	MA210	ID102	Total
Total		0	0	0	0	0	0	0	0	0	0	0	0	

Table 6.23: Physics topics by course from 3rd to 4th Semester

Fifth Sem											Sixth Sem										
Knowledge Units of Physics	**Courses →**	CS251	MA211	MA221	MA222	MA251	MA281	ID201	MA212	MA325	MA351	MA371	ID202	Total							
Total		0	0	0	0	0	0	0	0	0	0	0	0	0							

Table 6.24: Physics topics by course from 5th to 6th Semester

Seventh Sem								
Knowledge Units of Physics	**Courses →**	CS261	CS3P1	MA213	MA326	MA361	ID203	Total
Total		0	0	0	0	0	0	0

Table 6.25: Physics topics by course 7th Semester

Eighth Sem											
Knowledge Units of Physics	**Courses →**	AI263	MA401	CS2S1	CS311	QC251	CS253	CS370	STA252	MA327	MA352	MA372	Total
Total		0	0	0	0	0	0	0	0	0	0	0	0

Table 6.26: Physics topics by course 8th Semester

		Ninth Sem								Tenth Sem						
Knowledge Units of Physics	**Courses →**	AI264	MA402	CS231	MA381	AI262	CS273	MA341	MA355	CS400	AI268	AI367	MA403	QC381	CS254	Total
Total		0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Table 6.27: Physics topics by course from 9th to 10th Semester

6.4 Expected outcomes distributed by course

The following tables provide an overview of the outcomes expected to be achieved in each course of the present curriculum. The complete list of expected outcomes can be found in Section: 1.5.1.

		First Sem							Second Sem				
	Skill/Course	CS111	MA100	FG101	FG105	FG109	FG112	FG140	CS100	CS112	AI161	MA111	FI101
ABET-1)	Identify, formulate, and solve applied scientific problems.	1	1						1		2	1	2
ABET-2)	Formulate or design systems, processes, or programs.	2							1	2			
ABET-3)	Experimentation, data analysis, and scientific judgment.							1		2			2
ABET-4)	Effective communication with diverse audiences.			2									
ABET-5)	Ethical responsibilities and impact of scientific solutions.			2	1	1			1		1		
ABET-6)	Teamwork, planning, and risk management.						1						

Table 6.28: Outcomes by course 1st to 2nd Semester

Skill/Course	Third Sem							Fourth Sem						
	CS113	CS1D1	CS2B1	MA112	MA121	ID101	CS1D2	CS210	CS271	MA122	MA210	ID102		
ABET-1) Identify, formulate, and solve applied scientific problems.		2		2	2		2	2		2	2			
ABET-2) Formulate or design systems, processes, or programs.	2	1	2				1	2	2					
ABET-3) Experimentation, data analysis, and scientific judgment.	2								2					
ABET-4) Effective communication with diverse audiences.			2			2						2		
ABET-5) Ethical responsibilities and impact of scientific solutions.														
ABET-6) Teamwork, planning, and risk management.						1						1		

Table 6.29: Outcomes by course 3rd to 4th Semester

Skill/Course	Fifth Sem							Sixth Sem						
	CS251	MA211	MA221	MA222	MA251	MA281	ID201	MA212	MA325	MA351	MA371	ID202		
ABET-1) Identify, formulate, and solve applied scientific problems.		2	2	2		2		2	2	1	2			
ABET-2) Formulate or design systems, processes, or programs.									2					
ABET-3) Experimentation, data analysis, and scientific judgment.							2					2		
ABET-4) Effective communication with diverse audiences.														
ABET-5) Ethical responsibilities and impact of scientific solutions.							1					1		
ABET-6) Teamwork, planning, and risk management.														

Table 6.30: Outcomes by course 5th to 6th Semester

		Seventh Sem					
	Skill/Course	CS261	CS3P1	MA213	MA326	MA361	ID203
ABET-1)	Identify, formulate, and solve applied scientific problems.	2			2	2	
ABET-2)	Formulate or design systems, processes, or programs.	2	2				
ABET-3)	Experimentation, data analysis, and scientific judgment.		2		2	2	
ABET-4)	Effective communication with diverse audiences.						2
ABET-5)	Ethical responsibilities and impact of scientific solutions.						
ABET-6)	Teamwork, planning, and risk management.						1

Table 6.31: Outcomes by course 7th Semester

		Eighth Sem										
	Skill/Course	AI263	MA401	CS2S1	CS3I1	QC251	CS253	CS370	STA252	MA327	MA352	MA372
ABET-1)	Identify, formulate, and solve applied scientific problems.	2	3			2			1	2	2	2
ABET-2)	Formulate or design systems, processes, or programs.	2	3	2	2		2	2				
ABET-3)	Experimentation, data analysis, and scientific judgment.				2	2	2	2		2	2	2
ABET-4)	Effective communication with diverse audiences.		3									
ABET-5)	Ethical responsibilities and impact of scientific solutions.		3									
ABET-6)	Teamwork, planning, and risk management.		3	2								

Table 6.32: Outcomes by course 8th Semester

Skill/Course	Ninth Sem										Tenth Sem									
	AI264	MA402	CS231	MA381	AI262	CS273	MA341	MA355	CS400	AI268	AI367	MA403	QC381	CS254						
ABET-1) Identify, formulate, and solve applied scientific problems.		3		2	2		2	2			2	3	2							
ABET-2) Formulate or design systems, processes, or programs.	2	3	2		2	2						3		2						
ABET-3) Experimentation, data analysis, and scientific judgment.	3			2		2	2	2					2	2						
ABET-4) Effective communication with diverse audiences.		3										3								
ABET-5) Ethical responsibilities and impact of scientific solutions.		3										3								
ABET-6) Teamwork, planning, and risk management.		3	2									3								

Table 6.33: Outcomes by course 9th to 10th Semester

6.5 Courses by Outcome

6.5.1 Outcome: ABET-1) Identify, formulate, and solve applied scientific problems.

- CS111 Introduction to Programming (1st Sem-Pg. 295)
- MA100 Basic Mathematics (1st Sem-Pg. 300)
- CS100 Introduction to Computer Science (2nd Sem-Pg. 318)
- AI161 Applied AI (2nd Sem-Pg. 333)
- MA111 Calculus I (2nd Sem-Pg. 338)
- FI101 General Physics (2nd Sem-Pg. 340)
- CS1D1 Discrete Structures I (3rd Sem-Pg. 349)
- MA112 Calculus II (3rd Sem-Pg. 356)
- MA121 Linear Algebra I (3rd Sem-Pg. 358)
- CS1D2 Discrete Structures II (4th Sem-Pg. 365)
- CS210 Algorithms and Data Structures (4th Sem-Pg. 367)
- MA122 Linear Algebra II (4th Sem-Pg. 377)
- MA210 Calculus III (4th Sem-Pg. 380)
- MA211 Calculus IV (5th Sem-Pg. 390)
- MA221 Numerical Analysis I (5th Sem-Pg. 393)
- MA222 Real Analysis I (5th Sem-Pg. 396)
- MA281 Numbers theory (5th Sem-Pg. 401)
- MA212 Ordinary Differential Equations (6th Sem-Pg. 409)
- MA325 Numerical Analysis II (6th Sem-Pg. 412)
- MA351 Statistics and Probabilities (6th Sem-Pg. 415)
- MA371 Functional analysis I (6th Sem-Pg. 420)
- CS261 Artificial Intelligence (7th Sem-Pg. 427)
- MA213 Partial Differential Equations (7th Sem-Pg. 443)
- MA326 Numerical Analysis III (7th Sem-Pg. 446)
- MA361 Mathematical Modeling (7th Sem-Pg. 449)
- AI263 Introduction to Machine Learning (8th Sem-Pg. 455)
- MA401 Capstone Project I (8th Sem-Pg. 458)
- QC251 Quantum Mechanics (8th Sem-Pg. 472)
- STA252 Inferential Statistics (8th Sem-Pg. 484)
- MA327 Numerical Analysis IV (8th Sem-Pg. 487)
- MA352 Stochastic processes (8th Sem-Pg. 490)
- MA372 Functional analysis II (8th Sem-Pg. 493)
- MA402 Capstone Project II (9th Sem-Pg. 499)
- MA381 Algebraic Structures (9th Sem-Pg. 508)
- AI262 Natural Language Processing (9th Sem-Pg. 512)
- MA341 Optimization I (9th Sem-Pg. 519)
- MA355 Statistical Learning Theory (9th Sem-Pg. 522)
- AI367 Topics in Artificial Intelligence (10th Sem-Pg. 533)
- MA403 Capstone Project III (10th Sem-Pg. 539)
- QC381 Introduction to Quantum Computing (10th Sem-Pg. 543)

6.5.2 Outcome: ABET-2) Formulate or design systems, processes, or programs.

- CS111 Introduction to Programming (1st Sem-Pg. 295)
- CS100 Introduction to Computer Science (2nd Sem-Pg. 318)
- CS112 Objects-oriented programming I (2nd Sem-Pg. 328)

- CS113 Objects-oriented programming II (3rd Sem-Pg. 343)
- CS1D1 Discrete Structures I (3rd Sem-Pg. 349)
- CS2B1 Platform Based Development (3rd Sem-Pg. 352)
- CS1D2 Discrete Structures II (4th Sem-Pg. 365)
- CS210 Algorithms and Data Structures (4th Sem-Pg. 367)
- CS271 Data Management (4th Sem-Pg. 372)
- MA325 Numerical Analysis II (6th Sem-Pg. 412)
- CS261 Artificial Intelligence (7th Sem-Pg. 427)
- CS3P1 Parallel and Distributed Computing (7th Sem-Pg. 434)
- AI263 Introduction to Machine Learning (8th Sem-Pg. 455)
- MA401 Capstone Project I (8th Sem-Pg. 458)
- CS2S1 Operating systems (8th Sem-Pg. 461)
- CS3I1 Computer Security (8th Sem-Pg. 466)
- CS253 Exploratory data analysis (8th Sem-Pg. 476)
- CS370 Big Data and Analytics (8th Sem-Pg. 480)
- AI264 Deep Learning (9th Sem-Pg. 496)
- MA402 Capstone Project II (9th Sem-Pg. 499)
- CS231 Networking and Communication (9th Sem-Pg. 503)
- AI262 Natural Language Processing (9th Sem-Pg. 512)
- CS273 Time series analysis (9th Sem-Pg. 515)
- MA403 Capstone Project III (10th Sem-Pg. 539)
- CS254 Advanced Data Visualization (10th Sem-Pg. 547)

6.5.3 Outcome: ABET-3) Experimentation, data analysis, and scientific judgment.

- FG140 Fundamentals of Scientific Research (1st Sem-Pg. 315)
- CS112 Objects-oriented programming I (2nd Sem-Pg. 328)
- FI101 General Physics (2nd Sem-Pg. 340)
- CS113 Objects-oriented programming II (3rd Sem-Pg. 343)
- CS271 Data Management (4th Sem-Pg. 372)
- CS3P1 Parallel and Distributed Computing (7th Sem-Pg. 434)
- MA213 Partial Differential Equations (7th Sem-Pg. 443)
- MA326 Numerical Analysis III (7th Sem-Pg. 446)
- MA361 Mathematical Modeling (7th Sem-Pg. 449)
- CS3I1 Computer Security (8th Sem-Pg. 466)
- QC251 Quantum Mechanics (8th Sem-Pg. 472)
- CS253 Exploratory data analysis (8th Sem-Pg. 476)
- CS370 Big Data and Analytics (8th Sem-Pg. 480)
- MA327 Numerical Analysis IV (8th Sem-Pg. 487)
- MA352 Stochastic processes (8th Sem-Pg. 490)
- MA372 Functional analysis II (8th Sem-Pg. 493)
- AI264 Deep Learning (9th Sem-Pg. 496)
- MA381 Algebraic Structures (9th Sem-Pg. 508)
- CS273 Time series analysis (9th Sem-Pg. 515)
- MA341 Optimization I (9th Sem-Pg. 519)
- MA355 Statistical Learning Theory (9th Sem-Pg. 522)
- QC381 Introduction to Quantum Computing (10th Sem-Pg. 543)
- CS254 Advanced Data Visualization (10th Sem-Pg. 547)

6.5.4 Outcome: ABET-4) Effective communication with diverse audiences.

- FG101 Language (1st Sem-Pg. 303)
- CS2B1 Platform Based Development (3rd Sem-Pg. 352)
- ID101 Technical and professional English I (3rd Sem-Pg. 361)
- ID102 Technical and professional English II (4th Sem-Pg. 383)
- ID201 Technical and professional English III (5th Sem-Pg. 405)
- ID202 Technical and professional English IV (6th Sem-Pg. 423)
- ID203 Technical and professional English V (7th Sem-Pg. 452)
- MA401 Capstone Project I (8th Sem-Pg. 458)
- MA402 Capstone Project II (9th Sem-Pg. 499)
- MA403 Capstone Project III (10th Sem-Pg. 539)

6.5.5 Outcome: ABET-5) Ethical responsibilities and impact of scientific solutions.

- FG101 Language (1st Sem-Pg. 303)
- FG105 Gender and Society (1st Sem-Pg. 306)
- FG109 Environment and Sustainable Development (1st Sem-Pg. 309)
- CS100 Introduction to Computer Science (2nd Sem-Pg. 318)
- AI161 Applied AI (2nd Sem-Pg. 333)
- MA401 Capstone Project I (8th Sem-Pg. 458)
- MA402 Capstone Project II (9th Sem-Pg. 499)
- MA403 Capstone Project III (10th Sem-Pg. 539)

6.5.6 Outcome: ABET-6) Teamwork, planning, and risk management.

- FG112 Methods of managing university personnel (1st Sem-Pg. 312)
- ID101 Technical and professional English I (3rd Sem-Pg. 361)
- ID102 Technical and professional English II (4th Sem-Pg. 383)
- ID201 Technical and professional English III (5th Sem-Pg. 405)
- ID202 Technical and professional English IV (6th Sem-Pg. 423)
- ID203 Technical and professional English V (7th Sem-Pg. 452)
- MA401 Capstone Project I (8th Sem-Pg. 458)
- CS2S1 Operating systems (8th Sem-Pg. 461)
- MA402 Capstone Project II (9th Sem-Pg. 499)
- CS231 Networking and Communication (9th Sem-Pg. 503)
- MA403 Capstone Project III (10th Sem-Pg. 539)

6.6 Distribution of courses in the program

This proposal can be analyzed by the number of credits dedicated to each area, by course levels (Introductory, Intermediate, Advanced, and Projects).

	Mandatory			Elective			Total
	General Estudies	Specific	Specialization	General Estudies	Specific	Specialization	
1st Sem	34	10					44
2nd Sem	20	22					42
3rd Sem	14	30					44
4th Sem	16	28					44
5th Sem	16	22	6				44
6th Sem	6	38					44
7th Sem	16	16	12				44
8th Sem		8	6		20	10	44
9th Sem		10	10		16	8	44
10th Sem			36			16	52
Total	122	184	70	0	36	34	**446**

Table 6.34: Distribution of credits by area by semester

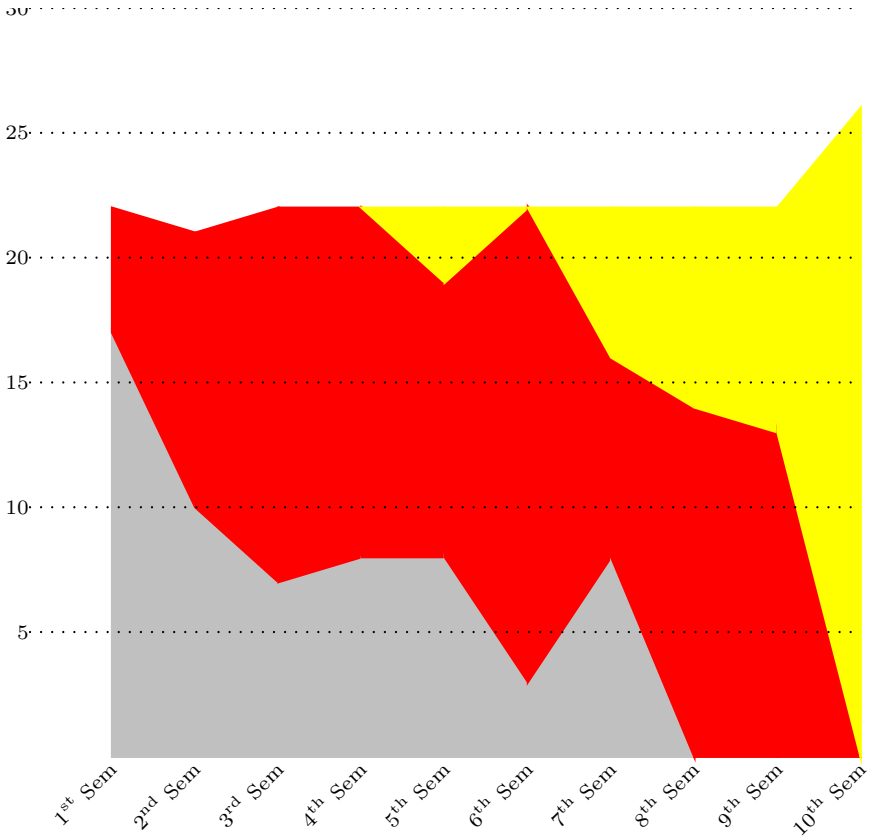


Figure 6.2: Credits by area by semester

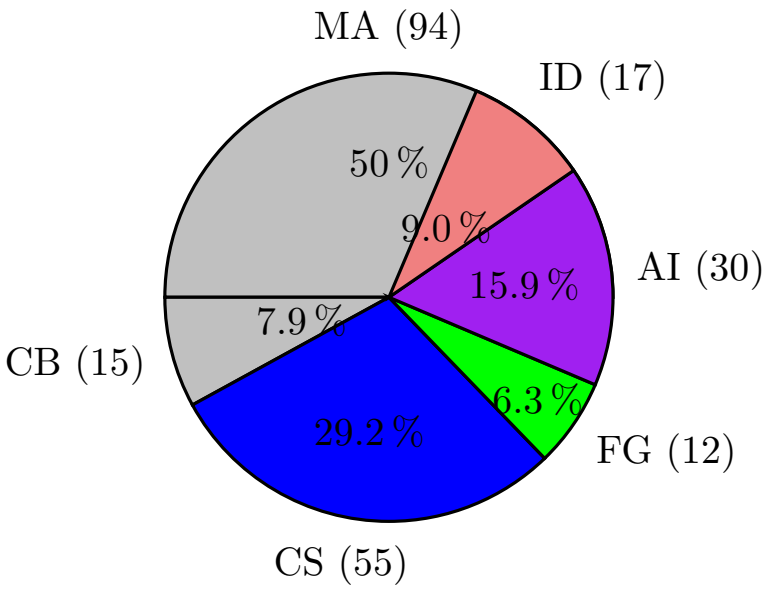


Figure 6.3: Distribution of courses by area considering credits.

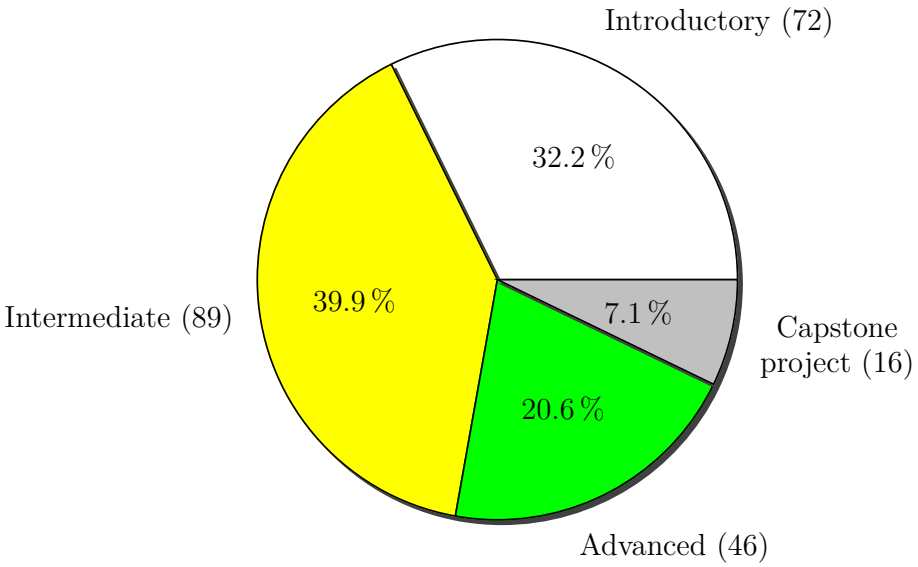


Figure 6.4: Distribution of credits by course levels.

6.7 Compatibility of the program with international standards

In this section we present the distribution of courses by areas of concentration in contrast with the international proposals for Computación Científica programs from the *Computing Curricula* of IEEE-CS/ ACM.

It should be noted that in some cases subjects may appear in more than one axis since they contain content from more than one area. For example, the operating systems course contains application units that can be classified as Information Technology but at the same time contains fundamentals of how an Operating System is structured, which belongs to the Computación Científica axis. In these cases, the credits have been divided among the corresponding axes.

1. Hardware and Architecture
   - Ninguno
2. Computer Science (115.50 Créditos)
   - CS111 Introduction to Programming (1st Sem-Pg. 295)
   - CS100 Introduction to Computer Science (2nd Sem-Pg. 318)
   - CS112 Objects-oriented programming I (2nd Sem-Pg. 328)
   - AI161 Applied AI (2nd Sem-Pg. 333)
   - CS113 Objects-oriented programming II (3rd Sem-Pg. 343)
   - CS210 Algorithms and Data Structures (4th Sem-Pg. 367)
   - CS271 Data Management (4th Sem-Pg. 372)
   - CS251 Computer graphics (5th Sem-Pg. 387)
   - CS261 Artificial Intelligence (7th Sem-Pg. 427)
   - CS3P1 Parallel and Distributed Computing (7th Sem-Pg. 434)
   - AI263 Introduction to Machine Learning (8th Sem-Pg. 455)
   - CS2S1 Operating systems (8th Sem-Pg. 461)
   - CS370 Big Data and Analytics (8th Sem-Pg. 480)
   - AI264 Deep Learning (9th Sem-Pg. 496)
   - CS231 Networking and Communication (9th Sem-Pg. 503)
   - AI262 Natural Language Processing (9th Sem-Pg. 512)
   - AI268 Computational Vision (10th Sem-Pg. 527)
   - AI367 Topics in Artificial Intelligence (10th Sem-Pg. 533)
   - QC381 Introduction to Quantum Computing (10th Sem-Pg. 543)

3. Software Engineering (7.00 Créditos)
   - CS112 Objects-oriented programming I (2nd Sem-Pg. 328)
   - CS113 Objects-oriented programming II (3rd Sem-Pg. 343)
4. Information Systems
   - Ninguno
5. Information Technology (20.50 Créditos)
   - CS2B1 Platform Based Development (3rd Sem-Pg. 352)
   - CS271 Data Management (4th Sem-Pg. 372)
   - CS2S1 Operating systems (8th Sem-Pg. 461)
   - CS3I1 Computer Security (8th Sem-Pg. 466)
   - CS231 Networking and Communication (9th Sem-Pg. 503)
6. Mathematics for Computing (14.00 Créditos)
   - CS1D1 Discrete Structures I (3rd Sem-Pg. 349)
   - CS1D2 Discrete Structures II (4th Sem-Pg. 365)
7. Basic Sciences (256.00 Créditos)
   - MA100 Basic Mathematics (1st Sem-Pg. 300)
   - FG101 Language (1st Sem-Pg. 303)
   - FG105 Gender and Society (1st Sem-Pg. 306)
   - FG109 Environment and Sustainable Development (1st Sem-Pg. 309)
   - FG112 Methods of managing university personnel (1st Sem-Pg. 312)
   - FG140 Fundamentals of Scientific Research (1st Sem-Pg. 315)
   - MA111 Calculus I (2nd Sem-Pg. 338)
   - FI101 General Physics (2nd Sem-Pg. 340)
   - MA112 Calculus II (3rd Sem-Pg. 356)
   - MA121 Linear Algebra I (3rd Sem-Pg. 358)
   - MA122 Linear Algebra II (4th Sem-Pg. 377)
   - MA210 Calculus III (4th Sem-Pg. 380)
   - MA211 Calculus IV (5th Sem-Pg. 390)
   - MA221 Numerical Analysis I (5th Sem-Pg. 393)
   - MA222 Real Analysis I (5th Sem-Pg. 396)
   - MA251 Introduction to Statistics (5th Sem-Pg. 399)
   - MA281 Numbers theory (5th Sem-Pg. 401)
   - MA212 Ordinary Differential Equations (6th Sem-Pg. 409)
   - MA325 Numerical Analysis II (6th Sem-Pg. 412)
   - MA351 Statistics and Probabilities (6th Sem-Pg. 415)
   - MA371 Functional analysis I (6th Sem-Pg. 420)
   - MA213 Partial Differential Equations (7th Sem-Pg. 443)
   - MA326 Numerical Analysis III (7th Sem-Pg. 446)
   - MA361 Mathematical Modeling (7th Sem-Pg. 449)
   - MA401 Capstone Project I (8th Sem-Pg. 458)
   - QC251 Quantum Mechanics (8th Sem-Pg. 472)
   - CS253 Exploratory data analysis (8th Sem-Pg. 476)
   - STA252 Inferential Statistics (8th Sem-Pg. 484)
   - MA327 Numerical Analysis IV (8th Sem-Pg. 487)
   - MA352 Stochastic processes (8th Sem-Pg. 490)
   - MA372 Functional analysis II (8th Sem-Pg. 493)
   - MA402 Capstone Project II (9th Sem-Pg. 499)
   - MA381 Algebraic Structures (9th Sem-Pg. 508)
   - CS273 Time series analysis (9th Sem-Pg. 515)

- MA341 Optimization I (9th Sem-Pg. 519)
- MA355 Statistical Learning Theory (9th Sem-Pg. 522)
- MA403 Capstone Project III (10th Sem-Pg. 539)
- CS254 Advanced Data Visualization (10th Sem-Pg. 547)

8. Business
   - Ninguno
9. General Knowledge (40.00 Créditos)
   - ID101 Technical and professional English I (3rd Sem-Pg. 361)
   - ID102 Technical and professional English II (4th Sem-Pg. 383)
   - ID201 Technical and professional English III (5th Sem-Pg. 405)
   - ID202 Technical and professional English IV (6th Sem-Pg. 423)
   - ID203 Technical and professional English V (7th Sem-Pg. 452)
   - CS400 Pre-professional internships (10th Sem-Pg. 525)

Referencias Bibliográficas

Chapter 7

Detailed content by course

First Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS111 Introduction to Programming (Mandatory)

1. General information

- **Semester**: 1st Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Laboratory**: 4 hours;
- **Prerrequisites**: None

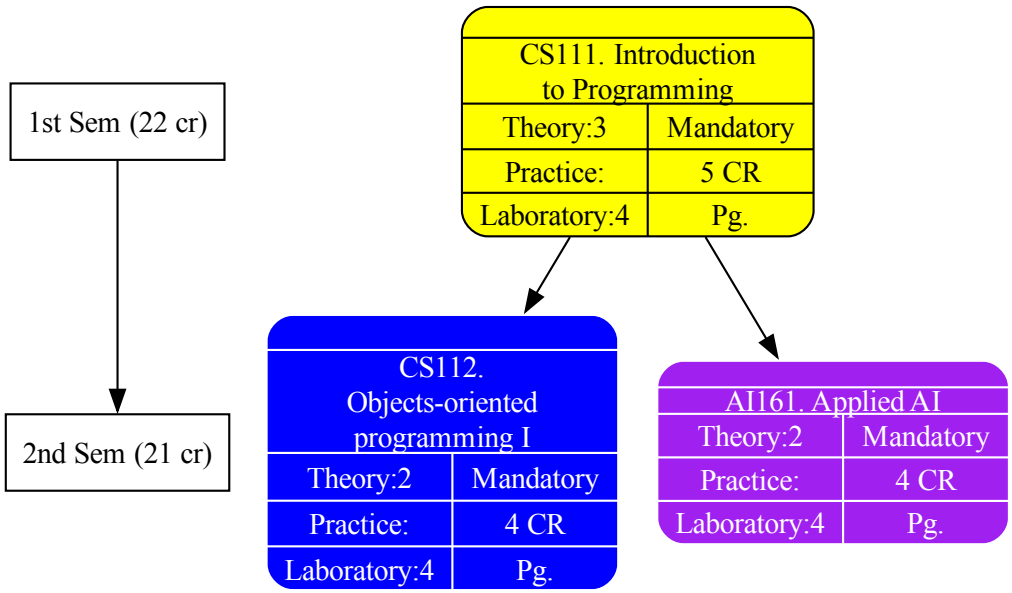


Figure 7.1: Connection Map. CS111 Introduction to Programming

2. Course Rationale

This is the first course in the sequence of introductory courses to Computer Science. This course

introduces participants to the fundamental concepts of programming using Python. Topics include data types, control structures, functions, lists, recursion, and the mechanics of execution, testing, and debugging. Programming is one of the pillars of Computer Science; any professional in the area will need to program to materialize their models and proposals.

3. Goals of the Course

- Introduce the fundamental concepts of programming.
- Develop the ability of abstraction using a programming language.
- Develop problem-solving skills through algorithmic thinking.
- Understand basic data structures and their applications.
- Master the Python programming language for problem solving.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Familiarity)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Introduction to Programming and History, 4 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Brief history of computing and programming languages. **II:** Overview of Python programming language. **III:** Programming environments and tools. **IV:** Basic program structure and execution. **V:** Writing, running, and debugging simple programs.

Learning Outcomes **I:** Identify important trends in the history of computing. [Familiarity] **II:** Discuss the historical context of programming language paradigms. [Familiarity] **III:** Compare daily life before and after personal computers and the Internet. [Assessment] **IV:** Write, compile, and run a simple Python program. [Usage]

Bibliography: [Zelle, 2010, Guttag, 2013, Brookshear and Brylow, 2019]

5.2 Type Systems Fundamentals, 6 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** A type as a set of values together with a set of operations: (a) Primitive types (e.g., numbers, Booleans) (b) Compound types built from other types (e.g., records/structs, unions, arrays, lists, functions, references using set operations) **II:** Type safety and errors caused by using values inconsistently given their intended types **III:** Goals and limitations of static and dynamic typing: detecting and eliminating errors as early as possible. **IV:** Primitive data types in Python (int, float, str, bool). **V:** Type conversion and casting.

Learning Outcomes **I:** Describe, for both a primitive and a compound type, the values that have that type [Familiarity] **II:** Describe examples of program errors detected by a type system [Familiarity] **III:** Use types and type-error messages to write and debug programs [Usage] **IV:** Explain Python's dynamic typing system. [Familiarity]

Bibliography: [Zelle, 2010, Guttag, 2013]

5.3 Object-Oriented Programming Fundamentals, 14 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Imperative programming as a subset of object-oriented programming. **II:** Definition of classes: fields, methods, and constructors. **III:** Dynamic vs static properties. **IV:** Variables, expressions, and assignments. **V:** Input and output operations. **VI:** Conditional statements (if, elif, else). **VII:** Iteration structures (for loops, while loops). **VIII:** Nested loops and loop control statements. **IX:** Exception handling basics.

Learning Outcomes **I:** Compose a class through design, implementation, and testing to meet behavioral requirements [Usage] **II:** Predict and validate control flow in a program using dynamic dispatch [Assessment] **III:** Use object-oriented encapsulation mechanisms such as interfaces and private members [Usage] **IV:** Implement programs using conditional and iterative structures. [Usage] **V:** Debug programs with syntax and logical errors. [Usage] **VI:** Design and implement simple algorithms using loops. [Usage]

Bibliography: [Zelle, 2010, Guttag, 2013, Brookshear and Brylow, 2019]

5.4 Functions and Parameter Passing, 10 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Function definition and invocation. **II:** Parameter passing mechanisms (by value, by reference). **III:** Return values and multiple returns. **IV:** Variable scope and lifetime (local, global, nonlocal). **V:** Recursion fundamentals. **VI:** Lambda expressions and higher-order functions. **VII:** Functions as first-class objects. **VIII:** Documentation and testing of functions.

Learning Outcomes **I:** Define and implement functions with proper parameter passing. [Usage] **II:** Explain the difference between pass-by-value and pass-by-reference. [Familiarity] **III:** Implement recursive solutions to simple problems. [Usage] **IV:** Use lambda expressions for functional programming constructs. [Usage] **V:** Design functions with appropriate scope and documentation. [Assessment]

Bibliography: [Zelle, 2010, Guttag, 2013, Hunt and Thomas, 1999]

5.5 Basic Data Structures, 10 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Lists and list operations (indexing, slicing, mutability). **II:** Tuples and immutability. **III:** Dictionaries and key-value mapping. **IV:** Sets and set operations. **V:** Strings as sequences. **VI:** List comprehensions and generator expressions. **VII:** Basic searching and sorting algorithms. **VIII:** Introduction to algorithmic complexity for basic operations.

Learning Outcomes **I:** Implement programs using lists, dictionaries, and sets. [Usage] **II:** Choose appropriate data structures for given problems. [Assessment] **III:** Explain the trade-offs between different data structures. [Familiarity] **IV:** Implement basic search and sort operations on lists. [Usage]

Bibliography: [Zelle, 2010, Guttag, 2013]

5.6 Object-Oriented Programming Fundamentals, 10 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Subclasses, inheritance (including multiple inheritance), and method overriding. **II:** Dynamic dispatch: definition of method-call. **III:** Object-oriented idioms for encapsulation: (a) Privacy, data hiding, and visibility of class members. (b) Interfaces revealing only method signatures. (c) Abstract base classes, traits and mixins. **IV:** Classes and objects in Python. **V:** Attributes and methods. **VI:** Inheritance and polymorphism basics. **VII:** Encapsulation and information hiding.

Learning Outcomes **I:** Build a simple class hierarchy utilizing subclassing that allows code to be reused for distinct subclasses [Usage] **II:** Explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping (the idea of a subtype being usable in a context that expects the supertype) [Familiarity] **III:** Compare and contrast the benefits and costs/impact of using inheritance (subclasses) and composition (specifically, how to base composition on higher order functions) [Assessment] **IV:** Design and implement simple classes in Python. [Usage] **V:** Explain the benefits of OOP for code organization. [Familiarity]

Bibliography: [Zelle, 2010, Guttag, 2013]

5.7 Algorithmic Strategies, 6 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Paradigms: (a) Brute-Force (e.g., linear search, selection sort, traveling salesperson, knapsack) (b) Decrease-and-Conquer: i. By a Constant (e.g., insertion sort, topological sort) ii. By a Constant Factor (e.g., binary search) iii. By a Variable Size (e.g., Euclid's) (c) Divide-and-Conquer (e.g., binary search, quicksort, mergesort, Strassen's) (d) Greedy (e.g., Dijkstra's, Kruskal's, Knapsack) (e) Transform-and-Conquer: i. Instance simplification (e.g., find duplicates via list presort) ii. Representation change (e.g., heapsort) iii. Problem reduction (e.g., least-common-multiple, linear programming) iv. Dynamic programming (e.g., Floyd's, Marshall, Bellman-Ford) (f) Space vs time tradeoffs (e.g., hashing) **II:** Iteration vs recursion (e.g., factorial, tree search) **III:** Problem decomposition strategies. **IV:** Algorithm design techniques. **V:** Debugging and testing strategies. **VI:** Tracing program execution.

Learning Outcomes **I:** For each of the paradigms in this unit: (a) Explain its definitional characteristics (b) Explain an example that demonstrates the paradigm including how this example satisfies the paradigm's characteristics. [Familiarity] **II:** Give examples of iterative and recursive algorithms that solve the same problem, explain the benefits and disadvantages of each approach [Usage] **III:** Evaluate whether a greedy approach leads to an optimal solution [Assessment] **IV:** Apply systematic debugging techniques to locate and fix errors. [Usage] **V:** Analyze algorithm complexity for simple problems. [Familiarity]

Bibliography: [Guttag, 2013, Cormen et al., 2009]

5.8 Modern Development Methods and Tool Usage, 4 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Using modern programming environments (IDEs, notebooks). **II:** Code search and reuse best practices. **III:** Using standard libraries and APIs. **IV:** Debugging and testing techniques. **V:** Version control basics. **VI:** Code documentation and style guidelines. **VII:** Virtual environments and package management.

Learning Outcomes **I:** Use modern development tools effectively. [Usage] **II:** Apply code reuse principles through libraries and APIs. [Usage] **III:** Debug and test programs systematically. [Usage] **IV:** Follow coding standards and documentation practices. [Familiarity] **V:** Manage Python packages and environments. [Familiarity]

Bibliography: [Zelle, 2010, Hunt and Thomas, 1999]

6. Bibliography

[Brookshear and Brylow, 2019] Brookshear, J. G. and Brylow, D. (2019). *Computer Science: An Overview*. Pearson, global edition edition.

[Cormen et al., 2009] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press, 3rd edition.

[Guttag, 2013] Guttag, J. V. (2013). *Introduction to Computation and Programming Using Python*. MIT Press, 1st edition.

[Hunt and Thomas, 1999] Hunt, A. and Thomas, D. (1999). *The Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley, 1st edition.

[Zelle, 2010] Zelle, J. M. (2010). *Python Programming: An Introduction to Computer Science*. Franklin, Beedle & Associates Inc, 2nd edition.

[Brookshear and Brylow, 2019] Brookshear, J. G. and Brylow, D. (2019). *Computer Science: An Overview*. Pearson, global edition edition.

[Cormen et al., 2009] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press, 3rd edition.

[Guttag, 2013] Guttag, J. V. (2013). *Introduction to Computation and Programming Using Python*. MIT Press, 1st edition.

[Hunt and Thomas, 1999] Hunt, A. and Thomas, D. (1999). *The Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley, 1st edition.

[Zelle, 2010] Zelle, J. M. (2010). *Python Programming: An Introduction to Computer Science*. Franklin, Beedle & Associates Inc, 2nd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA100 Basic Mathematics (Mandatory)

1. General information

- **Semester**: 1st Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**: None

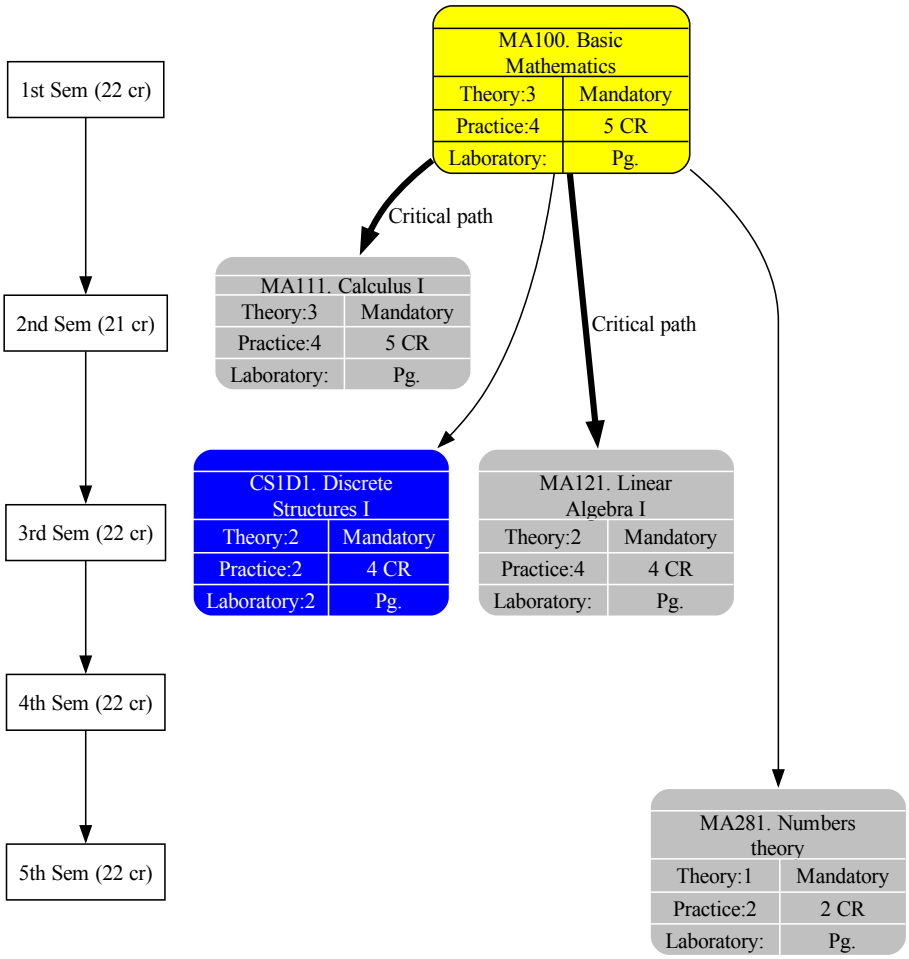


Figure 7.2: Connection Map. MA100 Basic Mathematics

2. Course Rationale

The course includes knowledge of Notions of Propositional Logic and set theory, Real number systems, Integers, the Principle of Mathematical Induction, and polynomials of one variable. Matrices, determinants, and their applications in problems involving living beings. Introduction to statistics.

3. Goals of the Course

- Develop logical reasoning and critical thinking skills through the use of formal languages.
- Understand and apply the properties of number systems (real and complex) in problem solving.

- Manipulate polynomial and matrix structures to solve systems of equations and interpret results.
- Introduce basic concepts of statistics and probability for data analysis.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Familiarity)

5. Content

5.1 Logic and Sets, 16 hr(s)

Competences: ABET-1

Topics: **I:** Propositional logic: Connectives, truth tables, and validity. **II:** Logical-mathematical quantifiers. **III:** Set theory: Operations, relations, and functions. **IV:** Families of sets and Cartesian product.

Learning Outcomes **I:** Formalize natural language statements into propositional language [Familiarity]. **II:** Operate with sets using algebraic laws and Venn diagrams [Usage]. **III:** Prove basic properties of sets and logical relations [Assessment].

Bibliography: [Lipschutz, 1991, Rojo, 1995]

5.2 Number Systems, 16 hr(s)

Competences: ABET-1

Topics: **I:** The real number system: Field and order axioms. **II:** Mathematical induction and Newton's binomial theorem. **III:** Complex numbers: Binomial, polar, and exponential forms. **IV:** Roots of complex numbers and De Moivre's theorem.

Learning Outcomes **I:** Identify the algebraic structure of real and complex numbers [Familiarity]. **II:** Apply the principle of mathematical induction in proofs of summations [Usage]. **III:** Solve equations in the complex number field using various representations [Assessment].

Bibliography: [Gentile, 1985, Álvaro Pinzón, 1973]

5.3 Polynomials and Matrices, 16 hr(s)

Competences: ABET-1

Topics: **I:** Polynomials: Division algorithm and roots of polynomials. **II:** Matrices: Types, operations, and determinants. **III:** Inverse matrix and rank of a matrix. **IV:** Systems of linear equations: Gauss-Jordan method.

Learning Outcomes **I:** Recognize the structure of polynomial rings and matrix algebra [Familiarity]. **II:** Use elementary row operations to calculate the inverse of a matrix [Usage]. **III:** Model and solve systems of linear equations applied to contextual problems [Assessment].

Bibliography: [Gentile, 1984, Rojo, 1995]

5.4 Statistics and Probability, 16 hr(s)

Competences: ABET-1

Topics: **I:** Descriptive statistics: Measures of central tendency. **II:** Dispersion measures and graphical representation. **III:** Probability concepts: Sample space and events. **IV:** Conditional probability and independence.

Learning Outcomes **I:** Describe data sets using basic statistical indicators [Familiarity]. **II:** Calculate probabilities of simple and compound events in random experiments [Usage]. **III:** Interpret statistical results for decision making in scientific environments [Assessment].

Bibliography: [Peterson, 2005]

6. Bibliography

[Gentile, 1984] Gentile, E. (1984). *Notas de álgebra*. Editorial Universitaria de Buenos Aires.

[Gentile, 1985] Gentile, E. (1985). *Aritmética Elemental*. Colección de Monografías Científicas Ser.

[Lipschutz, 1991] Lipschutz, S. (1991). *Teoría de conjuntos y temas afines*. McGraw-Hill.

[Peterson, 2005] Peterson, J. (2005). *Matemática básica*. Prentice-Hall.

[Rojo, 1995] Rojo, A. (1995). *Álgebra I*. El Ateneo.

[Álvaro Pinzón, 1973] Álvaro Pinzón (1973). *Conjuntos y estructuras*. Blaisdell Publishing Company.

[Gentile, 1984] Gentile, E. (1984). *Notas de álgebra*. Editorial Universitaria de Buenos Aires.

[Gentile, 1985] Gentile, E. (1985). *Aritmética Elemental*. Colección de Monografías Científicas Ser.

[Lipschutz, 1991] Lipschutz, S. (1991). *Teoría de conjuntos y temas afines*. McGraw-Hill.

[Peterson, 2005] Peterson, J. (2005). *Matemática básica*. Prentice-Hall.

[Rojo, 1995] Rojo, A. (1995). *Álgebra I*. El Ateneo.

[Álvaro Pinzón, 1973] Álvaro Pinzón (1973). *Conjuntos y estructuras*. Blaisdell Publishing Company.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

FG101 Language (Mandatory)

1. General information

- **Semester**: 1st Sem. **Credits**: 2
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 2 hours;
- **Prerrequisites**: None

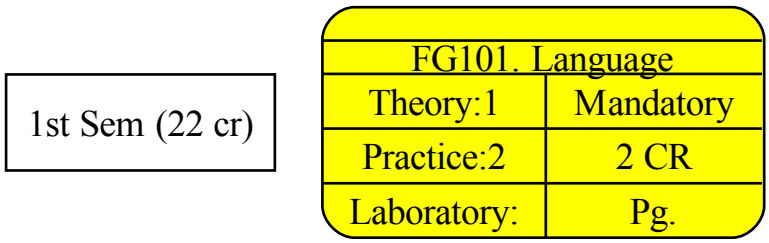


Figure 7.3: Connection Map. FG101 Language

2. Course Rationale

The Language course is fundamental for the development of the university student's communicative skills. It focuses on mastering the rules and principles that regulate the use of standard Spanish, covering phonetic, syntactic, lexical, and pragmatic levels. Likewise, it seeks to strengthen the capacity for reading comprehension, synthesis, and academic writing under international standards, ensuring that the graduate can communicate their ideas with coherence, ethics, and academic rigor.

3. Goals of the Course

- Know and apply Spanish writing standards in the drafting and editing of diverse texts.
- Understand the fundamental concepts of human communication and adapt the message to the academic reality.
- Apply basic research and writing knowledge to prepare structured academic reports.
- Execute formal speeches with responsibility, ethics, and mastery of oral expression techniques.

4. General Competences (*Outcomes*)

ABET-4) An ability to communicate effectively with a range of audiences. (Usage)

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts. (Usage)

5. Content

5.1 Spanish Language and Regulations, 16 hr(s)

Competences: ABET-4

Topics: **I:** Language, speech, and norm. **II:** Functions of language and the linguistic reality of Peru. **III:** Written accentuation: General and special rules. **IV:** Reading: Comprehension strategies and main ideas. **V:** Orthography of numerical expressions.

Learning Outcomes I: Recognize linguistic terms and language functions in various contexts [Familiarity]. **II:** Correctly use orthography and accentuation rules in text production [Usage]. **III:** Analyze the multilingual reality of Peru and characterize academic language [Assessment].

Bibliography: [Española, 2010, Española, 2014]

5.2 Academic Writing, 16 hr(s)

Competences: ABET-4

Topics: I: The academic report: Planning and structure. **II:** Direct and indirect citations, and paraphrasing according to APA standards. **III:** The sentence as a syntactic unit: Subject and predicate. **IV:** Punctuation marks: Period, comma, and semicolon. **V:** Summary of scientific articles.

Learning Outcomes I: Identify the syntactic structure of simple and compound sentences [Familiarity]. **II:** Employ citations and bibliographic references following APA regulations [Usage]. **III:** Prepare coherent summaries of scientific articles and plan academic reports [Assessment].

Bibliography: [Martínez, 2002, Nariola et al., 2007]

5.3 Text Drafting and Grammar, 16 hr(s)

Competences: ABET-4

Topics: I: Cases of redundancy, colloquialisms, and solecisms. **II:** Subordinate compound sentences: Noun, adjective, and adverbial. **III:** APA and Vancouver standards in scientific writing. **IV:** Textual coherence and cohesion: Dequeísmo and queísmo. **V:** Review and editing of the academic report.

Learning Outcomes I: Detect common writing errors such as lexical imprecision and language vices [Familiarity]. **II:** Organize paragraphs with proper grammatical cohesion and coherence [Usage]. **III:** Correct and co-evaluate academic texts to ensure standard language use [Assessment].

Bibliography: [Marín, 2008, Martínez, 2002]

5.4 Oral Expression and Public Speaking, 16 hr(s)

Competences: ABET-4

Topics: I: The art of public speaking: Oratory and discourse. **II:** Characteristics and types of oratory. **III:** Preparation of visual aids (PPT) for presentations. **IV:** Ethical responsibilities of the speaker. **V:** Final presentation of the academic report.

Learning Outcomes I: Internalize fundamental discourse techniques and active listening [Familiarity]. **II:** Design effective visual presentations that complement oral communication [Usage]. **III:** Present academic reports with fluency, propriety, and ethics before an audience [Assessment].

Bibliography: [Verderber et al., 2010]

6. Bibliography

[Española, 2010] Española, R. A. (2010). *Ortografía de la lengua española*. Espasa Libros.

[Española, 2014] Española, R. A. (2014). *Diccionario de la lengua española*. Espasa Libros, 23rd edition.

[Martínez, 2002] Martínez, M. C. (2002). *Lectura y escritura de textos. Perspectivas teóricas y talleres*. Universidad del Valle.

[Marín, 2008] Marín, M. (2008). *Lingüística y enseñanza de la lengua*. Aique Grupo Editor, 2nd edition.

[Nariola et al., 2007] Nariola, N., Naters, L., and Romero, L. (2007). *Iniciarse en la Investigación Universitaria: Exámenes, trabajos y reseñas*. Pontificia Universidad Católica del Perú, 1st edition.

[Verderber et al., 2010] Verderber, R. F., Verderber, K. S., and Sellnow, D. D. (2010). *Comunicación oral efectiva*. Cengage Learning Editores S.A., 14th edition.

[Española, 2010] Española, R. A. (2010). *Ortografía de la lengua española*. Espasa Libros.

[Española, 2014] Española, R. A. (2014). *Diccionario de la lengua española*. Espasa Libros, 23rd edition.

[Martínez, 2002] Martínez, M. C. (2002). *Lectura y escritura de textos. Perspectivas teóricas y talleres*. Universidad del Valle.

[Marín, 2008] Marín, M. (2008). *Lingüística y enseñanza de la lengua*. Aique Grupo Editor, 2nd edition.

[Nariola et al., 2007] Nariola, N., Naters, L., and Romero, L. (2007). *Iniciarse en la Investigación Universitaria: Exámenes, trabajos y reseñas*. Pontificia Universidad Católica del Perú, 1st edition.

[Verderber et al., 2010] Verderber, R. F., Verderber, K. S., and Sellnow, D. D. (2010). *Comunicación oral efectiva*. Cengage Learning Editores S.A., 14th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

FG105 Gender and Society (Mandatory)

1. General information

- **Semester**: 1st Sem. **Credits**: 2
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 2 hours;
- **Prerrequisites**: None

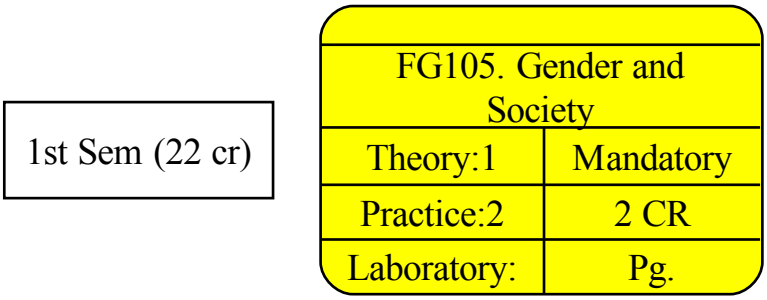


Figure 7.4: Connection Map. FG105 Gender and Society

2. Course Rationale

The Gender and Society course seeks to familiarize students with the category of gender as an analytical tool to understand social organization. The course critically addresses the implications of inequality and discrimination in the production of knowledge and in the socio-historical structure. Through this analysis, a reflective vision of the environment is promoted, fostering respect for diversity and the pursuit of equity in the exercise of citizenship and professional practice.

3. Goals of the Course

- Understand the concept of gender and its importance for the analysis of social structures and critical thinking.
- Identify historical processes of domination and the formation of gender stereotypes in Western and Peruvian society.
- Analyze the relationship between gender, division of labor, and access to public and private power spaces.
- Evaluate equality policies and the national and international regulatory framework aimed at reducing gender gaps.

4. General Competences (*Outcomes*)

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts. (Familiarity)

5. Content

5.1 Fundamental Concepts of Gender, 8 hr(s)

Competences:

Topics: I: Sex and gender: Conceptual and biological distinctions. **II:** Gender as an analytical category in the social sciences. **III:** Gender socialization and the construction of identities. **IV:** Stereotypes, prejudices, and gender roles today.

Learning Outcomes I: Distinguish the biological bases of sex from the social and cultural construction of gender [Familiarity]. **II:** Explain how gender stereotypes influence the perception of the environment and communication [Usage]. **III:** Critically analyze the production of knowledge from an inclusive gender perspective [Assessment].

Bibliography: [Butler, 2007, Lamas, 1996]

5.2 Gender, History, and Social Structure, 8 hr(s)

Competences:

Topics: I: Historical evolution of gender relations. **II:** The sexual division of labor: Public and private spaces. **III:** Gender and coloniality in Latin America. **IV:** Patriarchy and systems of domination in Peruvian history.

Learning Outcomes I: Identify historical milestones that have shaped gender gaps in Latin American society [Familiarity]. **II:** Describe the impact of the division of labor on the economic and social development of women [Usage]. **III:** Evaluate how historical power structures affect current sustainability and social equity [Assessment].

Bibliography: [Federici, 2010, Quijano, 1988]

5.3 Gender, Power, and Citizenship, 8 hr(s)

Competences:

Topics: I: Gender and the exercise of citizenship in the 21st century. **II:** Political participation and access to decision-making positions. **III:** Gender-based violence: Definitions, types, and statistics. **IV:** Masculinities and changes in power relations.

Learning Outcomes I: Recognize the various forms of gender-based violence and existing prevention mechanisms [Familiarity]. **II:** Apply ethical principles in the analysis of power relations between genders [Usage]. **III:** Argue for the importance of diversity and inclusion in the development of civil society [Assessment].

Bibliography: [Fuller, 2004, Pateman, 2016]

5.4 Public Policies and Regulatory Framework, 8 hr(s)

Competences:

Topics: I: International legal framework: CEDAW and the SDGs. **II:** National Gender Equality Policy in Peru. **III:** Gender perspective in public management and social projects. **IV:** Current challenges for substantive equality.

Learning Outcomes I: Identify the main international treaties ratified by Peru regarding gender [Familiarity]. **II:** Integrate the gender perspective in the planning and management of multidisciplinary projects [Usage]. **III:** Evaluate the impact of public equality policies on national health, safety, and economy [Assessment].

Bibliography: [Durand, 2014, MIMP, 2019]

6. Bibliography

[Butler, 2007] Butler, J. (2007). *El género en disputa: El feminismo y la subversión de la identidad*. Paidós.

[Durand, 2014] Durand, A. (2014). *Género y Ciudadanía: Retos para la inclusión en el Perú*. UNMSM.

[Federici, 2010] Federici, S. (2010). *Calibán y la bruja: Mujeres, cuerpo y acumulación originaria*. Traficantes de Sueños.

[Fuller, 2004] Fuller, N. (2004). *Jerarquías de género y ciudadanía*. Pontificia Universidad Católica del Perú.

[Lamas, 1996] Lamas, M. (1996). *El género: La construcción cultural de la diferencia sexual*. PUEG.

[MIMP,] MIMP. Política nacional de igualdad de género. Technical report, Ministerio de la Mujer y Poblaciones Vulnerables.

[Pateman, 2016] Pateman, C. (2016). *El contrato sexual*. Anthropos.

[Quijano, 1988] Quijano, A. (1988). *Modernidad, identidad y utopía en América Latina*. Sociedad y Política.

[Butler, 2007] Butler, J. (2007). *El género en disputa: El feminismo y la subversión de la identidad*. Paidós.

[Durand, 2014] Durand, A. (2014). *Género y Ciudadanía: Retos para la inclusión en el Perú*. UNMSM.

[Federici, 2010] Federici, S. (2010). *Calibán y la bruja: Mujeres, cuerpo y acumulación originaria*. Traficantes de Sueños.

[Fuller, 2004] Fuller, N. (2004). *Jerarquías de género y ciudadanía*. Pontificia Universidad Católica del Perú.

[Lamas, 1996] Lamas, M. (1996). *El género: La construcción cultural de la diferencia sexual*. PUEG.

[MIMP, 2019] MIMP (2019). Política nacional de igualdad de género. Technical report, Ministerio de la Mujer y Poblaciones Vulnerables, Lima, Perú.

[Pateman, 2016] Pateman, C. (2016). *El contrato sexual*. Anthropos.

[Quijano, 1988] Quijano, A. (1988). *Modernidad, identidad y utopía en América Latina*. Sociedad y Política.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

FG109 Environment and Sustainable Development (Mandatory)

1. General information

- **Semester**: 1st Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 2 hours;
- **Prerrequisites**: None

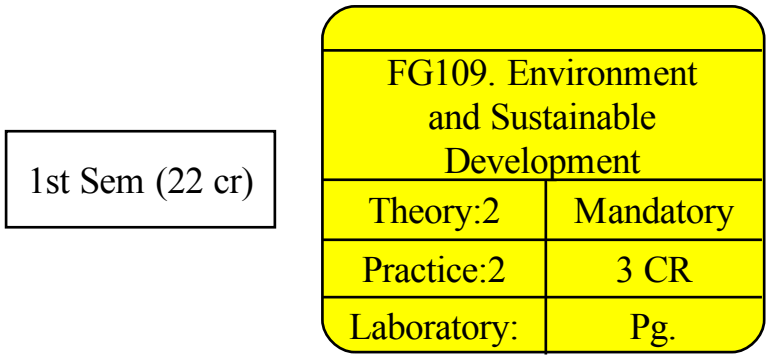


Figure 7.5: Connection Map. FG109 Environment and Sustainable Development

2. Course Rationale

The Environment and Sustainable Development course is theoretical and practical in nature, aiming to raise student awareness regarding the importance of ecosystem conservation. The course covers the principles of ecology, the impact of human activities on the biosphere, and global strategies to achieve development that balances economic growth, social equity, and environmental protection. This knowledge is crucial for computer science professionals to design technological solutions with a focus on environmental responsibility and sustainability.

3. Goals of the Course

- Understand the fundamental concepts of ecology and the functioning of biogeochemical cycles.
- Analyze the causes and consequences of major global environmental problems, such as climate change.
- Evaluate the national and international regulatory framework oriented towards environmental management and sustainability.
- Apply sustainability indicators in the evaluation of scientific and technological projects.

4. General Competences (*Outcomes*)

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts. (Familiarity)

5. Content

5.1 Ecology and Biodiversity, 16 hr(s)

Competences: ABET-5

Topics: **I:** Ecology: Basic concepts, ecosystems, and biomes. **II:** Energy flow and biogeochemical cycles in nature. **III:** Biodiversity: Importance, levels, and current threats. **IV:** Ecosystem services and valuation of natural capital.

Learning Outcomes **I:** Identify the components of an ecosystem and the interdependence relationships among living beings [Familiarity]. **II:** Explain the impact of biodiversity loss on the stability of natural systems [Usage]. **III:** Evaluate the health of a local ecosystem using biological indicators and physicochemical parameters [Assessment].

Bibliography: [Miller and Spoolman, 2012, Smith and Smith, 2001]

5.2 Environmental Impact and Climate Change, 16 hr(s)

Competences: ABET-5

Topics: **I:** Environmental pollution: Air, water, soil, and waste. **II:** The greenhouse effect and global warming. **III:** Impact of industrial and technological activities. **IV:** Ecological footprint and carbon footprint: Measurement and reduction.

Learning Outcomes **I:** Recognize the main sources of pollution and their effects on public health and the climate [Familiarity]. **II:** Calculate personal and institutional carbon footprints to propose mitigation measures [Usage]. **III:** Critically analyze the environmental impact of data centers and technological infrastructure [Assessment].

Bibliography: [Miller and Spoolman, 2012, Starr and Taggart, 2004]

5.3 Environmental Management and Regulations, 16 hr(s)

Competences: ABET-5

Topics: **I:** Environmental legal framework in Peru and the Ministry of Environment. **II:** Environmental Impact Assessment (EIA) and certifications. **III:** Natural resource management: Water, energy, and protected areas. **IV:** Environmental ethics and corporate social responsibility.

Learning Outcomes **I:** Describe the functions of environmental regulatory bodies at the national level [Familiarity]. **II:** Apply ethical and legal principles in the design of environmental management plans for projects [Usage]. **III:** Argue for the importance of transparency and citizen participation in environmental issues [Assessment].

Bibliography: [Strange and Bayley, 2012, Primack, 1993]

5.4 Strategies for Sustainable Development, 16 hr(s)

Competences: ABET-5

Learning Outcomes **I:** Recognize the 17 Sustainable Development Goals and their relationship with scientific computing [Familiarity]. **II:** Integrate sustainability criteria and circular economy into the software life cycle [Usage]. **III:** Propose innovative technological solutions to solve contemporary socio-environmental challenges [Assessment].

Bibliography: [Strange and Bayley, 2012, Miller and Spoolman, 2012]

6. Bibliography

[Miller and Spoolman, 2012] Miller, G. T. and Spoolman, S. (2012). *Living in the Environment.* Cengage Learning.

[Primack,] Primack, R. *Essentials of Conservation Biology.* Sinauer Associates Inc.

[Smith and Smith, 2001] Smith, R. L. and Smith, T. (2001). *Ecología.* Pearson Educación.

[Starr and Taggart,] Starr, C. and Taggart, R. *Biología: La unidad y diversidad de la vida.* Thomson Editors.

[Strange and Bayley,] Strange, T. and Bayley, A. *Desarrollo sostenible: Integrar la economía, la sociedad y el medio ambiente.* OCDE / UNAM.

[Miller and Spoolman, 2012] Miller, G. T. and Spoolman, S. (2012). *Living in the Environment.* Cengage Learning, Canada, 17th edition.

[Primack, 1993] Primack, R. (1993). *Essentials of Conservation Biology.* Sinauer Associates Inc., Sunderland, Massachusetts.

[Smith and Smith, 2001] Smith, R. L. and Smith, T. (2001). *Ecología.* Pearson Educación, Madrid, España, 4th edition.

[Starr and Taggart, 2004] Starr, C. and Taggart, R. (2004). *Biología: La unidad y diversidad de la vida.* Thomson Editors, México D.F.

[Strange and Bayley, 2012] Strange, T. and Bayley, A. (2012). *Desarrollo sostenible: Integrar la economía, la sociedad y el medio ambiente.* OCDE / UNAM, México.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

FG112 Methods of managing university personnel (Mandatory)

1. General information

- **Semester**: 1st Sem. **Credits**: 2
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 2 hours;
- **Prerrequisites**: None

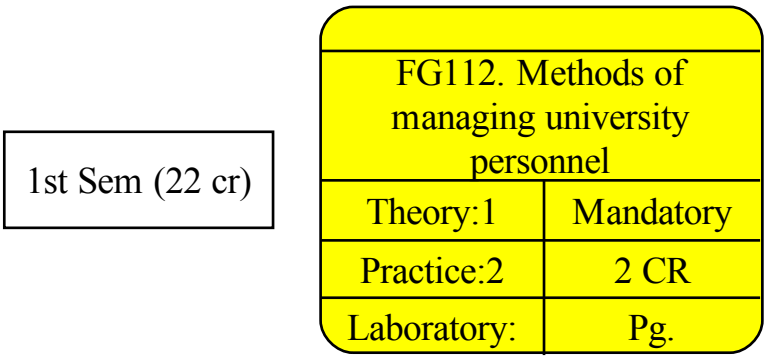


Figure 7.6: Connection Map. FG112 Methods of managing university personnel

2. Course Rationale

The course University Personal Management Methods is theoretical-practical in nature and aims to empower students through the development of intrapersonal competencies and intellectual work strategies. The course integrates self-knowledge and strategic planning processes with advanced learning techniques and digital tools. Through this comprehensive approach, students will be able to manage their behavior in the face of change, optimize their academic performance, and conduct research with ethical rigor, fostering a strategic vision to face the challenges of the contemporary university and professional environment.

3. Goals of the Course

- Develop a personal strategic plan integrating autonomous learning techniques and SWOT analysis.
- Apply cognitive and metacognitive strategies to optimize the processing, retention, and communication of scientific information.
- Enhance leadership, ethics, and collaborative work skills in face-to-face and virtual environments.
- Produce monographic research papers and effective presentations following academic integrity standards and use of ICTs.

4. General Competences (*Outcomes*)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Familiarity)

5. Content

5.1 Strategic Management and Learning, 12 hr(s)

Competences:

Topics: **I:** Basic concepts of management and university learning. **II:** Self-knowledge and personal diagnosis (SWOT). **III:** Personal vision, mission, and values. **IV:** Cognitive processes, attention, and learning strategies. **V:** Metacognition: Learning to learn and self-regulation.

Learning Outcomes **I:** Identify the principles of management and cognitive factors involved in academic success [Familiarity]. **II:** Develop a strategic planning matrix oriented towards individual development [Usage]. **III:** Evaluate the alignment between personal values and continuous learning processes [Assessment].

Bibliography: [Alcaide, 2021, Bernardo and Méndez, 2020, Fischman, 2018]

5.2 Intellectual Work Techniques and ICTs, 12 hr(s)

Competences:

Topics: **I:** Comprehensive reading, underlining, and note-taking. **II:** Visual organizers: Concept and mind maps. **III:** Time management and personal productivity. **IV:** Personal Learning Environment (PLE) and Web 2.0 tools. **V:** Collaborative work in virtual environments.

Learning Outcomes **I:** Recognize the usefulness of visual organizers and ICT tools in studying [Familiarity]. **II:** Use synthesis techniques and time management to optimize performance [Usage]. **III:** Design an efficient personal learning environment integrating digital and productivity tools [Assessment].

Bibliography: [Pallarés, 2017, Novak, 2019, Lazzati, 2019]

5.3 Leadership, Ethics, and Research, 12 hr(s)

Competences:

Topics: **I:** Leadership: Styles, theories, and conflict management. **II:** The monograph: Structure, search, and source selection. **III:** Ethical use of information and plagiarism prevention (Citation standards). **IV:** Social responsibility and professional ethics in project management.

Learning Outcomes **I:** Understand the methodological structure of research and the principles of effective leadership [Familiarity]. **II:** Select reliable information sources for solving management problems [Usage]. **III:** Write academic documents respecting intellectual property and ethical principles [Assessment].

Bibliography: [Cortina, 2021, Montenegro and Núñez, 2022, Sampieri and Mendoza, 2018]

5.4 Communication and Decision Making, 12 hr(s)

Competences:

Topics: **I:** The decision-making process: Models and risks. **II:** Inclusive academic communication: Synchronous and asynchronous. **III:** Oral presentations and preparation of visual aids. **IV:** Emotional intelligence and proactive adaptation to change.

Learning Outcomes **I:** Identify the critical stages for decision-making and effective communication [Familiarity]. **II:** Apply assertive communication strategies to lead and present project results [Usage]. **III:** Resolve conflict situations and make decisions based on critical thinking and emotional intelligence [Assessment].

Bibliography: [Bernardo and Méndez, 2020, Lazzati, 2019, Zimbardo and Gerrig, 2017]

6. Bibliography

[Alcaide,] Alcaide, F. *Aprendiendo de los mejores: Tu desarrollo personal es tu destino.* Alienta Editorial.

[Bernardo and Méndez,] Bernardo, J. and Méndez, J. *Estrategias de aprendizaje para universitarios: Un enfoque práctico y digital.* Pirámide.

[Cortina,] Cortina, A. *Ética para líderes: La responsabilidad de gobernar.* Paidós.

[Fischman,] Fischman, D. *El camino del líder: Historias ancestrales y vivencias personales.* Planeta.

[Lazzati,] Lazzati, S. *Decisiones gerenciales: Anatomía y proceso.* Granica.

[Montenegro and Núñez,] Montenegro, M. and Núñez, R. *Guía metodológica para la investigación y elaboración de monografías.* Fondo Editorial PUCP.

[Novak,] Novak, J. *Aprender a aprender: 30 años después.* Morata.

[Pallarés,] Pallarés, E. *Estudiar con éxito: Técnicas y hábitos para aprender mejor.* Paraninfo.

[Sampieri and Mendoza,] Sampieri, R. H. and Mendoza, C. *Metodología de la investigación: Las rutas cuantitativa, cualitativa y mixta.* McGraw-Hill.

[Zimbardo and Gerrig,] Zimbardo, P. and Gerrig, R. *Psicología y vida (Edición 19).* Pearson.

[Alcaide, 2021] Alcaide, F. (2021). *Aprendiendo de los mejores: Tu desarrollo personal es tu destino.* Alienta Editorial, Barcelona.

[Bernardo and Méndez, 2020] Bernardo, J. and Méndez, J. (2020). *Estrategias de aprendizaje para universitarios: Un enfoque práctico y digital.* Pirámide, Madrid.

[Cortina, 2021] Cortina, A. (2021). *Ética para líderes: La responsabilidad de gobernar.* Paidós, Barcelona.

[Fischman, 2018] Fischman, D. (2018). *El camino del líder: Historias ancestrales y vivencias personales.* Planeta, Lima.

[Lazzati, 2019] Lazzati, S. (2019). *Decisiones gerenciales: Anatomía y proceso.* Granica, Buenos Aires.

[Montenegro and Núñez, 2022] Montenegro, M. and Núñez, R. (2022). *Guía metodológica para la investigación y elaboración de monografías.* Fondo Editorial PUCP, Lima.

[Novak, 2019] Novak, J. (2019). *Aprender a aprender: 30 años después.* Morata, Madrid.

[Pallarés, 2017] Pallarés, E. (2017). *Estudiar con éxito: Técnicas y hábitos para aprender mejor.* Paraninfo, Madrid.

[Sampieri and Mendoza, 2018] Sampieri, R. H. and Mendoza, C. (2018). *Metodología de la investigación: Las rutas cuantitativa, cualitativa y mixta.* McGraw-Hill, México D.F.

[Zimbardo and Gerrig, 2017] Zimbardo, P. and Gerrig, R. (2017). *Psicología y vida (Edición 19).* Pearson, México D.F.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

FG140 Fundamentals of Scientific Research (Mandatory)

1. General information

- **Semester**: 1st Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 2 hours;
- **Prerrequisites**: None

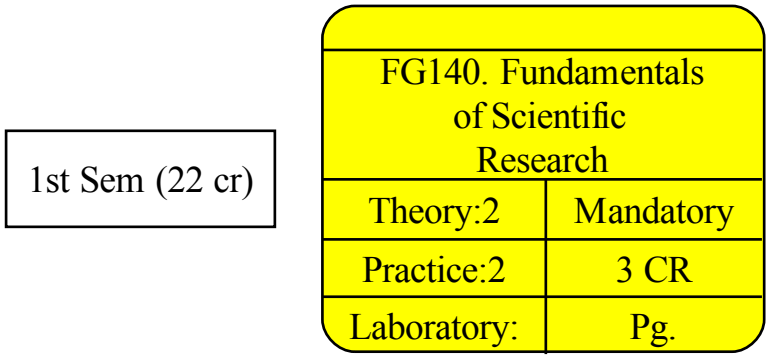


Figure 7.7: Connection Map. FG140 Fundamentals of Scientific Research

2. Course Rationale

The Fundamentals of Scientific Research course provides the student with the basic knowledge and tools for the development of high-quality scientific research. The course addresses the problematization of reality, theoretical foundation, methodology, and ethics in research. It seeks to develop in the student the capacity for critical observation, academic rigor, and intellectual honesty necessary to produce new knowledge that contributes to the advancement of science and technology.

3. Goals of the Course

- Understand the concept of science, the scientific method, and the stages of the research process.
- Identify and pose relevant research problems based on the observation of reality.
- Apply techniques for searching and managing scientific information in specialized databases.
- Know the standards for scientific writing and the ethical principles governing the publication of results.

4. General Competences (*Outcomes*)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Familiarity)

5. Content

5.1 Science and Scientific Method, 12 hr(s)

Competences: ABET-3

Topics: **I:** Scientific knowledge: Characteristics and levels. **II:** Science: Classification and objectives. **III:** The scientific method: Stages and application. **IV:** Ethics in research and best practices.

Learning Outcomes **I:** Differentiate scientific knowledge from empirical or everyday knowledge [Familiarity]. **II:** Sequentially apply the stages of the scientific method in practical cases [Usage]. **III:** Evaluate the importance of ethics and scientific integrity in project development [Assessment].

Bibliography: [Bunge, 2014, Sampieri et al., 2014]

5.2 Scientific Information Management, 12 hr(s)

Competences: ABET-3

Topics: **I:** Information search: Strategies and databases. **II:** Bibliographic management tools (Mendeley, Zotero). **III:** Critical reading of scientific articles. **IV:** Citation and reference standards (APA, Vancouver, IEEE).

Learning Outcomes **I:** Identify reliable information sources and high-impact academic databases [Familiarity]. **II:** Use bibliographic managers to organize and cite sources in an academic document [Usage]. **III:** Write a coherent and well-founded literature review on a specific topic [Assessment].

Bibliography: [Rodríguez, 2013, Upegui, 2011]

5.3 Research Project Planning, 12 hr(s)

Competences: ABET-3

Topics: **I:** Identification and delimitation of the research problem. **II:** Formulation of the research question and objectives. **III:** Justification and feasibility of the research. **IV:** Hypothesis formulation and definition of variables.

Learning Outcomes **I:** Recognize the fundamental components of a scientific research problem [Familiarity]. **II:** Formulate mathematical or logical hypotheses that can be experimentally tested [Usage]. **III:** Design the preliminary methodological framework for a rigorous scientific study [Assessment].

Bibliography: [Sampieri et al., 2014, Kothari, 2009]

5.4 Scientific Writing and Communication, 12 hr(s)

Competences: ABET-3

Topics: **I:** Structure of the scientific article (IMRaD). **II:** Technical writing: Clarity, precision, and brevity. **III:** Tables, figures, and data representation. **IV:** The publication process and peer review.

Learning Outcomes **I:** Understand the standard structure of scientific reports and communications [Familiarity]. **II:** Prepare effective reports using technical language and appropriate visual representations [Usage]. **III:** Present and orally defend research results in an inclusive manner [Assessment].

Bibliography: [Olson, 2014, Russey et al., 2006]

6. Bibliography

[Bunge, 2014] Bunge, M. (2014). *La ciencia, su método y su filosofía*. DeBolsillo.

[Kothari, 2009] Kothari, C. (2009). *Research Methodology: Methods and Techniques*. New Age International Publishers, 2nd edition.

[Olson, 2014] Olson, L. (2014). *Guide to Academic and Scientific Publication*. eacademia.

[Rodríguez, 2013] Rodríguez, L. J. M. (2013). *Cómo buscar y usar información científica: Guía para estudiantes universitarios*. Universidad de Cantabria.

[Russey et al., 2006] Russey, W., Ebel, H., and Bliefert, C. (2006). *How to Write a Successful Science Thesis*. Wiley-VCH.

[Sampieri et al., 2014] Sampieri, R. H., Collado, C. F., and Baptista, M. P. (2014). *Metodología de la investigación*. McGraw-Hill, 6th edition.

[Upegui, 2011] Upegui, A. A. S. (2011). *Manual de redacción académica e investigativa*. Católica del Norte Fundación Universitaria.

[Bunge, 2014] Bunge, M. (2014). *La ciencia, su método y su filosofía*. DeBolsillo.

[Kothari, 2009] Kothari, C. (2009). *Research Methodology: Methods and Techniques*. New Age International Publishers, 2nd edition.

[Olson, 2014] Olson, L. (2014). *Guide to Academic and Scientific Publication*. eacademia.

[Rodríguez, 2013] Rodríguez, L. J. M. (2013). *Cómo buscar y usar información científica: Guía para estudiantes universitarios*. Universidad de Cantabria.

[Russey et al., 2006] Russey, W., Ebel, H., and Bliefert, C. (2006). *How to Write a Successful Science Thesis*. Wiley-VCH.

[Sampieri et al., 2014] Sampieri, R. H., Collado, C. F., and Baptista, M. P. (2014). *Metodología de la investigación*. McGraw-Hill, 6th edition.

[Upegui, 2011] Upegui, A. A. S. (2011). *Manual de redacción académica e investigativa*. Católica del Norte Fundación Universitaria.

Second Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS100 Introduction to Computer Science (Mandatory)

1. General information

- **Semester**: 2nd Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 2 hours;
- **Prerrequisites**: None

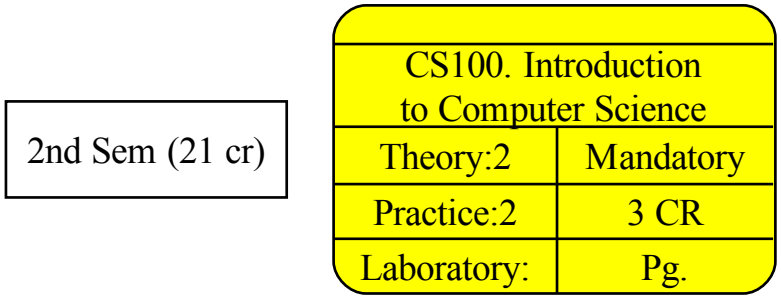


Figure 7.8: Connection Map. CS100 Introduction to Computer Science

2. Course Rationale

This course serves as the foundation for understanding the fundamental concepts of computational thinking applicable across various professions.

The course provides, starting from ground zero, a panoramic view of: introductory computational thinking, data storage, computer architecture, operating systems, networks and the Internet, algorithms, sorting methods, software engineering, databases, data structures, software engineering, computer graphics, artificial intelligence among others.

Designed as an introductory course to Computer Science, the concepts are presented in a playful manner and using an Active Learning methodology. Throughout the course, active audience participation is encouraged, akin to a theatrical performance.

The related knowledge areas covered are directly aligned with the Computing Curricula ACM/IEEE-CS.

The course **does not require** any prior knowledge in computer handling topics and can be taken by student from any field.

3. Goals of the Course

- Introduce the fundamental concepts of Computational Thinking and Computer Science to students from any professional background.
- Develop their ability to abstract.
- Understand how Computational Thinking is applied in each of their professions.
- Apply advanced concepts in a simplified manner in any career.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Familiarity)

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts. (Familiarity)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Familiarity)

5. Content

5.1 Computational Thinking (Part I), 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Explanation of the evaluation system. **II:** General course instructions. **III:** Definition of Computing. **IV:** Computing as a Human-Computer binomial. **V:** Distortions in the definition of computing. **VI:** Computing as the automation of abstraction.

VII: Computing and Engineering: similarities and differences. **VIII:** Algorithmic problem-solving. **IX:** Dynamics: Understanding the execution of an algorithm at human speed.

Learning Outcomes **I:** Apply the fundamental concepts of computing in real-life situations. [Usage] **II:** Identify distortions of Computing in real-life situations. [Usage]

III: Clearly identify at least 3 contexts of using the word“Engineer” in English. [Assessment] **IV:** Identify the limitations of humans in solving computational problems. [Usage]

Bibliography: [Brookshear and Brylow, 2019]

5.2 Computational Thinking. Part II, 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Decomposition (Breaking a problem in smaller pieces) **II:** Abstraction (Focus on important things) **III:** Pattern recongnition (Identify similar sequences based on previous problems) **IV:** Designing algorithms

Learning Outcomes **I:** Apply Computational Thinking to real-world problems. [Usage]

Bibliography: [Brookshear and Brylow, 2019]

5.3 General concepts, 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Binary vs. decimal numbering. **II:** Character representation: the ASCII table. **III:** Binary search. **IV:** Computational complexity of an algorithm.

Learning Outcomes **I:** Apply various numbering systems to real-world problems. [Usage] **II:** Understand the internal representation of characters in the ASCII and UTF-8 tables. [Familiarity] **III:** Apply Divide and Conquer algorithmic strategy. [Usage] **IV:** Determine basic analysis of algorithmic complexity. [Usage]

Bibliography: [Brookshear and Brylow, 2019]

5.4 Machine-Level Data Representation, 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Overview and history of computer architecture **II:** Bits, bytes, and words **III:** Unsigned, signed and two's complement representations **IV:** Numeric data representation and number bases: (a) Fixed-point (b) Floating-point **V:** Representation of non-numeric data **VI:** Representation of records, arrays and UTF data types

Learning Outcomes **I:** Discuss why everything in computers are data, including instructions [Debate] **II:** Explain how fixed-length number representations can affect accuracy and precision [Explain] **III:** Describe how negative integers are stored in sign-magnitude and two's-complement representations [Describe] **IV:** Discuss how different formats can represent numerical data [Debate] **V:** Explain the bit-level representation of non-numeric data, such as characters, strings, records, and arrays [Explain] **VI:** Translate numerical data from one format to another [Translate] **VII:** Describe how a single adder (without overflow detection) can handle both signed (two's complement) and unsigned (binary) input without "knowing" which format a given input is using [Describe]

Bibliography: [Brookshear and Brylow, 2019]

5.5 Program Abstraction and Representation, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Programs that take (other) programs as input such as interpreters, compilers, type-checkers, documentation generators **II:** Components of a language: (a) Definitions of alphabets, delimiters, sentences, syntax, and semantics (b) Syntax vs semantics **III:** Program as a set of non-ambiguous meaningful sentences **IV:** Basic programming abstractions: constants, variables, declarations (including nested declarations), command, expression, assignment, selection, definite and indefinite iteration, iterators, function, procedure, modules, exception handling **V:** Types of variables: static, local, nonlocal, global; need and issues with nonlocal and global variables. **VI:** Scope rules: static vs dynamic; visibility of variables; side-effects. **VII:** Environment vs store and their properties **VIII:** Data and control abstraction **IX:** Mechanisms for information exchange between program units such as procedures, functions, and modules: nonlocal variables, global variables, parameter-passing, import-export between modules **X:** Low level instruction representation such as virtual machine instructions, assembly language, and binary representation

Learning Outcomes **I:** Illustrate the scope of variables and visibility using simple programs [Apply] **II:** Illustrate different types of parameter-passing using simple pseudo programming language [Apply] **III:** Explain side-effect using global and nonlocal variables and how to fix such programs [Explain] **IV:** Explain how programs that process other programs treat the other programs as their input data [Explain] **V:** Describe a grammar and an abstract syntax tree for a small language [Describe]

Bibliography: [Brookshear and Brylow, 2019]

5.6 Cryptography Fundamentals, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Basic concepts on Cryptography **II:** Public-keys, Private-keys

Learning Outcomes **I:** Explain the role of cryptography in supporting security and privacy [Explain] **II:** Discuss the risks of inventing one's own cryptographic methods [Debate] **III:** Discuss the importance of prime numbers in cryptography and explain their use in cryptographic algorithms [Debate] **IV:** Implement and cryptanalyze classical ciphers [Implement]

Bibliography: [Brookshear and Brylow, 2019]

5.7 Memory Management, 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Review of physical memory, address translation and memory management hardware See also: Memory Hierarchy, Discrete Mathematics **II:** Impact of memory hierarchy including cache concept, cache lookup, and per-CPU caching on operating system mechanisms and policy See also: Memory Hierarchy, Latency, Cache and Memory Hierarchy, Virtualization and Isolation **III:** Logical and physical addressing, address space virtualization. See also: Memory Hierarchy, Discrete Mathematics **IV:** Concepts of paging, page replacement, thrashing and allocation of pages and frames **V:** Allocation/deallocation/storage techniques (algorithms and data structure) performance and flexibility (a) Arenas, slab allocators, free lists, size classes, heterogeneously sized pages (huge pages) **VI:** Memory caching and cache coherence and the effect of flushing the cache to avoid speculative execution vulnerabilities. See also: Functional Organization, Memory Hierarchy, Latency, Cache and Memory Hierarchy, Virtualization and Isolation **VII:** Virtual memory: leveraging virtual memory hardware for OS services and efficiency

Learning Outcomes **I:** Explain memory hierarchy and cost-performance tradeoffs [Explain] **II:** Summarize the principles of virtual memory as applied to caching and paging [Summarize] **III:** Evaluate the tradeoffs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed [Evaluate] **IV:** Describe the reason for and use of cache memory (performance and proximity, how caches complicate isolation and virtual machine abstraction) [Describe]

Bibliography: [Brookshear and Brylow, 2019]

5.8 Role and Purpose of Operating Systems, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Operating systems mediate between general purpose hardware and application-specific software. **II:** Universal operating system functions (e.g., process, user and device interfaces, persistence of data) **III:** Influences of security, networking, multimedia, parallel and distributed computing

Learning Outcomes **I:** Understand the objectives and functions of modern operating systems [Explain] **II:** Evaluate the design issues in different usage scenarios (e.g., real time OS, mobile, server) [Evaluate] **III:** Understand how evolution and stability are desirable and mutually antagonistic in operating systems function [Explain]

Bibliography: [Brookshear and Brylow, 2019]

5.9 Principles of Operating System, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Operating system software design and approaches (e.g., monolithic, layered, modular, microkernel, unikernel) **II:** Abstractions, processes, and resources **III:** The evolution of the link between hardware architecture and the operating system functions **IV:** Leveraging interrupts from hardware level: service routines and implementations. See also: Assembly Level Machine Organization (a) Timer interrupts for implementing time slices (b) I/O interrupts for putting blocking threads to sleep without polling

Learning Outcomes **I:** Understand how the application of software design approaches to operating systems design/implementation (e.g., layered, modular, etc.) affects the robustness and maintainability of an operating system [Explain] **II:** Categorize system calls by purpose [Categorize] **III:** Apply OS techniques for isolation, protection, and throughput across OS functions (e.g., starvation similarities in process scheduling, disk request scheduling, semaphores, etc.) and beyond [Apply]

Bibliography: [Brookshear and Brylow, 2019]

5.10 Fundamentals of Networks and Communications, 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Importance of networking in contemporary computing, and associated challenges. See also: Digital Equity and Accessibility, Social Impact and AI, Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation **II:** Organization of the internet (e.g., users, Internet Service Providers, autonomous systems, content providers, content delivery networks) **III:** Layers and their roles (application, transport, network, datalink, and physical) **IV:** Network elements (e.g., routers, switches, hubs, access points, and hosts) **V:** Basic queueing concepts (e.g., relationship with latency, congestion, service levels, etc.)

Learning Outcomes **I:** Articulate the organization of the internet [Articulate] **II:** List and define the appropriate network terminology [List] **III:** Describe the layered structure of a typical networked architecture [Describe]

Bibliography: [Brookshear and Brylow, 2019]

5.11 Networked Applications, 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Naming and address schemes (e.g., DNS, and Uniform Resource Identifiers) **II:** Distributed application paradigms (e.g., client/server, peer-to-peer, cloud, edge, and fog). See also: Communication, Coordination **III:** Diversity of networked application demands (e.g., latency, bandwidth, and loss tolerance). See also: Communication, Sustainable Design and Pervasive Computing, Environmental Footprint of Computing Systems, Systemic Effects and Social Context, Digital Equity and Accessibility, Social Impact and AI **IV:** Coverage of application-layer protocols (e.g., HTTP) **V:** Interactions with TCP, UDP, and Socket APIs. See also: Programs: Distribution, Programs: Execution Properties, GPU Programming, Programs: Implementation Mappings, Programs: Declarative Parallelism, Programs: Starting Activities

Learning Outcomes **I:** Define the principles of naming, addressing, resource location [Define] **II:** Analyze the needs of specific networked application demands [Analyze]

Bibliography: [Brookshear and Brylow, 2019]

5.12 Complexity Analysis Framework, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Complexity Analysis Framework: (a) Best, average, and worst-case performance of an algorithm (b) Empirical and relative (Order of Growth) measurements (c) Input size and primitive operations (d) Time and space efficiency **II:** Empirical measurements of performance **III:** Time and space tradeoffs in algorithms

Learning Outcomes **I:** Prepare a presentation that explains to first year students the basic concepts of algorithmic complexity including best, average, and worst-case algorithm behavior, Big-O, Omega, and Theta notations, complexity classes, time and space tradeoffs, empirical measurement, and impact on practical problems [Explain] **II:** For each algorithm in the Foundational Data Structures, Foundational Algorithms, Advanced Algorithms unit, explain its runtime complexity class and why it belongs to this class [Explain] **III:** Informally evaluate the foundational complexity class of simple algorithms [Evaluate] **IV:** Develop empirical studies to determine and validate hypotheses about the runtime complexity of various algorithms by running algorithms on input of various sizes and comparing actual performance to the theoretical analysis [Create] **V:** Explain examples that illustrate time-space tradeoffs of algorithms [Explain] **VI:** Explain how tree balance affects the efficiency of binary search tree operations [Explain]

Bibliography: [Brookshear and Brylow, 2019]

5.13 Asymptotic Notation and Complexity Classes, 1 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: I: Asymptotic complexity analysis (average and worst-case bounds): (a) Big-O, Big-Omega, and Big-Theta formal notations (b) Foundational Complexity Classes and Representative Examples/Problems: i. $O(1)$ Constant (e.g., array access) ii. $O(\log_2 n)$ Logarithmic (e.g., binary search) iii. $O(n)$ Linear (e.g., linear search) iv. $O(n \log_2 n)$ Log Linear (e.g., mergesort) v. $O(n^2)$ Quadratic (e.g., selection sort) vi. $O(n^c)$ Polynomial (e.g., $O(n^3)$ Gaussian elimination) vii. $O(2^n)$ Exponential (e.g., Knapsack, Satisfiability (SAT), Traveling Sales-Person (TSP), all subsets) viii. $O(n!)$ Factorial (e.g., Hamiltonian circuit, all permutations)

Learning Outcomes I: Using examples, explain each of the foundational complexity classes in this unit [Explain] **II:** For each foundational complexity class in this unit, explain an algorithm that demonstrates the associated runtime complexity [Explain] **III:** Explain to a non-technical audience the significance of tractable versus intractable algorithms using an intuitive explanation of Big-O complexity [Explain]

Bibliography: [Brookshear and Brylow, 2019]

5.14 Complexity Analysis: Recursion, Amortization, and Tight Bounds, 1 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: I: Little-o, Little-Omega, and Little Theta notations

Learning Outcomes I: Given a problem to program for which there may be several algorithmic approaches, evaluate them and determine which are feasible, and select one that is optimal in implementation and run-time behavior [Evaluate]

Bibliography: [Brookshear and Brylow, 2019]

5.15 Foundational Data Structures, 3 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: I: Abstract Data Type (ADT) and operations on an ADT: (a) Dictionary operations (insert, delete, find) **II:** Arrays: (a) Numeric vs non-numeric, character strings (b) Single (vector) vs multidimensional (matrix) **III:** Records/Structs/Tuples and Objects **IV:** Linked lists (for historical reasons): (a) Single vs Double and Linear vs Circular **V:** Stacks **VI:** Queues and deques: (a) Heap-based priority queue **VII:** Hash tables/maps: (a) Collision resolution and complexity (e.g., probing, chaining, rehash) **VIII:** Trees: (a) Binary, n-ary, and search trees (b) Balanced (e.g., AVL, Red-Black, Heap)

Learning Outcomes I: For each ADT/Data-Structure in this unit: (a) Explain its definition, properties, representation(s), and associated ADT operations. (b) Explain step-by-step how the ADT operations associated with the data structure transform it. [Explain] **II:** Explain how collision avoidance and collision resolution is handled in hash tables [Explain] **III:** Explain the heap property and the use of heaps as an implementation of a priority queue [Explain]

Bibliography: [Brookshear and Brylow, 2019]

5.16 Foundational Algorithms, 3 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: I: Graphs (e.g., [un]directed, [a]cyclic, [un]connected, and [un]weighted): (a) Graph representation: adjacency list vs matrix **II:** Search algorithms: (a) $O(n)$ complexity (e.g., linear/sequential array/list search) (b) $O(\log_2 n)$ complexity (e.g., binary search) (c) $O(\log_b n)$ complexity (e.g., uninformed depth/breadth-first tree search) **III:** Sorting algorithms (e.g., stable, unstable): (a) $O(n^2)$ complexity (e.g., insertion, selection) (b) $O(n \log n)$ complexity (e.g.,

quicksort, merge, timsort) **IV:** Graph algorithms: (a) Shortest path (e.g., Dijkstra's, Floyd's) (b) Minimal spanning tree (e.g., Prim's, Kruskal's) **V:** Sorting algorithms: (a) $O(n \log n)$ complexity heapsort (b) Pseudo $O(n)$ complexity (e.g., bucket, counting, radix) **VI:** Graph algorithms: (a) Transitive closure (e.g., Warshall's) (b) Topological sort

Learning Outcomes I: For each algorithm in this unit explain step-by-step how the algorithm operates [Explain] **II:** For each algorithmic approach (e.g., sorting) in this unit apply a prototypical example of the approach (e.g., merge sort) [Apply] **III:** Given requirements for a problem, develop multiple solutions using various data structures and algorithms. Subsequently, evaluate the suitability, strengths, and weaknesses selecting an approach that best satisfies the requirements [Create] **IV:** Explain factors beyond computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data [Explain]

Bibliography: [Brookshear and Brylow, 2019]

5.17 Advanced Algorithms, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: I: Cryptography algorithms (e.g., SHA-256) **II:** Parallel algorithms

Learning Outcomes I: An appreciation of quantum computation and its application to certain problems [Explain]

Bibliography: [Brookshear and Brylow, 2019]

5.18 Core Database System Concepts, 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: I: Purpose and advantages of database systems **II:** Components of database systems **III:** Design of core DBMS functions (e.g., query mechanisms, transaction management, buffer management, access methods) **IV:** Transaction management **V:** Normalization **VI:** Approaches for managing large volumes of data (e.g., NoSQL database systems, use of MapReduce) **VII:** Distributed databases/cloud-based systems **VIII:** Use of a declarative query language

Learning Outcomes I: Identify at least four advantages that using a database system provides [Analyze] **II:** Enumerate the components of a (relational) database system [Enumerate] **III:** Defend the value of data independence [Defend] **IV:** Compose a simple select-project-join query in SQL [Compose] **V:** Describe the advantages for eliminating duplicate repeated data [Describe] **VI:** Outline how MapReduce uses parallelism to process data efficiently [Outline]

Bibliography: [Brookshear and Brylow, 2019]

5.19 Object-Oriented Programming Fundamentals, 4 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: I: Imperative programming as a subset of object-oriented programming. **II:** Object-oriented design: (a) Decomposition into objects carrying state and having behavior. (b) Class-hierarchy design for modeling. **III:** Definition of classes: fields, methods, and constructors. **IV:** Subclasses, inheritance (including multiple inheritance), and method overriding. **V:** Dynamic dispatch: definition of method-call. **VI:** Object-oriented idioms for encapsulation: (a) Privacy, data hiding, and visibility of class members. (b) Interfaces revealing only method signatures. (c) Abstract base classes, traits and mixins.

Learning Outcomes I: Enumerate the differences between imperative and object-oriented programming paradigms [Enumerate] **II:** Compose a class through design, implementation, and testing to meet behavioral requirements [Compose] **III:** Use object-oriented encapsulation mechanisms such as interfaces and private members [Use]

Bibliography: [Brookshear and Brylow, 2019]

5.20 Version Control and CI/CD, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Software configuration management and version control: Software Development Practices (a) Configuration in version control, reproducible builds/configuration. (b) Version control branching strategies. Development branches vs release branches. Trunk-based development. (c) Merging/rebasing strategies, when relevant. **II:** Release management. **III:** Software process automation: (a) Build systems - the value of fast, hermetic, reproducible builds, compare/contrast approaches to building a project. (b) Continuous Integration (CI) - the use of automation and automated tests to do preliminary validation that the current head/trunk revision builds and passes (basic) tests. (c) Continuous Deployment (CD) - the use of automation to automatically release every change that passes the automated tests to the production environment, ensuring frequent and reliable deliveries. (d) Dependency management - updating external/upstream dependencies, package management, SemVer. **IV:** Tool integration concepts and mechanisms. Software Development Practices **V:** Use of modern IDE facilities - debugging, refactoring, searching/indexing, ML-powered code assistants, etc. Software Development Practices

Learning Outcomes **I:** Describe the difference between centralized and distributed software configuration management [Describe] **II:** Describe how version control can be used to help manage software release management [Describe] **III:** Identify configuration items and use a source code control tool in a small team-based project [Analyze] **IV:** Understand the use of CI/CD systems as a ground-truth for the state of the team's shared code (build and test success) [Explain] **V:** Demonstrate the capability to use software tools in support of the development of a software product of medium size [Demonstrate]

Bibliography: [Brookshear and Brylow, 2019]

5.21 Requirements Engineering, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Describe functional requirements using, for example, use cases or user stories. (a) Using at least one method of documenting and structuring functional requirements. (b) Understanding how the method supports design and implementation. (c) Strengths and weaknesses of using a specific approach. **II:** Requirements elicitation. (a) Sources of requirements, for example, users, administrators, or support personnel. (b) Methods of requirement gathering, for example, surveys, interviews, or behavioral analysis. **III:** Non-functional requirements, for example, security, usability, or performance, also called as Quality Attributes. Sustainable Design and Pervasive Computing, Environmental Footprint of Computing Systems, Systemic Effects and Social Context **IV:** Risk identification and management, including ethical considerations surrounding the proposed product. Ethics Codes and Legal Accountability, Professional Responsibility and Public Policy, Professional Values, Nature and Continuing Development, Ethical Dissent and Cultural Context, Workplace Ethics **V:** Communicating and/or formalizing requirement specifications. **VI:** Prototyping a tool for both eliciting and validating/confirming requirements. **VII:** Product evolution: when requirements change, how to understand what effect that has and what changes need to be made. **VIII:** Effort estimation: (a) Learning techniques for better estimating the effort required to complete a task; (b) Practicing estimation and comparing it to how long tasks take; (c) Effort estimation is quite difficult, so students are likely to be way off in many cases, but seeing the process play out with their own work is valuable.

Learning Outcomes **I:** Compare different methods of eliciting requirements along multiple axes [Compare] **II:** Identify differences between two methods of describing functional requirements (e.g., customer interviews, user studies) and the situations where each would be preferred [Analyze] **III:** Identify which behaviors are required, allowed, or barred from a given set of requirements

and a list of candidate behaviors [Analyze] **IV:** Collect a set of requirements for a simple software system [Analyze] **V:** Identify areas of a software system that need to be changed, given a description of the system and a set of new requirements to be implemented [Analyze] **VI:** Identify the functional and non-functional requirements in a set of requirements [Analyze] **VII:** Estimate the time to complete a set of tasks, then compare estimates to the actual time taken [Estimate] **VIII:** Determine an implementation sequence for a set of tasks, adhering to dependencies between them, with a goal to retire risk as early as possible [Determine] **IX:** Write a requirement specification for a simple software system [Write]

Bibliography: [Brookshear and Brylow, 2019]

5.22 Machine Learning Fundamentals, 1 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Definition and examples of a broad variety of machine learning tasks: (a) Supervised learning: i. Classification ii. Regression (b) Reinforcement learning (c) Unsupervised learning: i. Clustering **II:** Fundamental ideas: (a) No free lunch theorem: no one learner can solve all problems; representational design decisions have consequences. (b) Sources of error and undecidability in machine learning **III:** A simple statistical-based supervised learning such as linear regression or decision trees: (a) Focus on how they work without going into mathematical or optimization details; enough to understand and use existing implementations correctly

Learning Outcomes **I:** Describe the differences among the three main styles of learning (supervised, reinforcement, and unsupervised) and determine which is appropriate to a particular problem domain [Explain] **II:** Differentiate the terms of AI, machine learning, and deep learning [Evaluate] **III:** Explain how machine learning works as an optimization/search process [Explain]

Bibliography: [Brookshear and Brylow, 2019]

5.23 Neural Networks and Ethics, 1 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Basic neural networks: (a) Fundamentals of understanding how neural networks work and their training process, without details of the calculations (b) Basic introduction to generative neural networks (e.g., large language models) **II:** Ethics for Machine Learning: (a) Focus on real data, real scenarios, and case studies (b) Dataset/algorithmic/evaluation bias and unintended consequences **III:** Deep learning: (a) Deep feed-forward networks (intuition only, no mathematics) (b) Convolutional neural networks (intuition only, no mathematics) (c) Visualization of learned feature representations from deep nets (d) Other architectures (generative NN, recurrent NN, transformers, etc.) **IV:** Ethics for Machine Learning: (a) Continued focus on real data, real scenarios, and case studies (b) Privacy (c) Fairness (d) Intellectual property (e) Explainability

Learning Outcomes **I:** Describe the neural network training process and resulting learned representations [Explain] **II:** Visualize the training progress of a neural network through learning curves in a well-established toolkit (e.g., TensorBoard) and visualize the learned features of the network [Apply] **III:** Given a real-world application of machine learning, describe ethical issues regarding the choices of data, preprocessing steps, algorithm selection, and visualization/presentation of results [Apply]

Bibliography: [Brookshear and Brylow, 2019]

5.24 Fundamental Concepts of Graphics and Interactive Techniques, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Overview of computer graphics pipeline and interactive systems **II:** Basic rendering concepts: rasterization, ray tracing, shading models **III:** Color theory and color spaces **IV:** Coordinate systems and transformations **V:** Basic interaction techniques: input devices, event handling **VI:** Advanced rendering concepts: global illumination, physically based rendering **VII:** Graphics hardware and acceleration techniques **VIII:** Human perception and visual cognition

Learning Outcomes **I:** Explain the basic stages of the computer graphics pipeline [Explain] **II:** Describe basic rendering techniques such as rasterization and ray tracing [Describe] **III:** Apply color theory to design visually effective graphics [Apply] **IV:** Implement coordinate transformations for 2D and 3D graphics [Implement] **V:** Design basic interactive applications using common input devices and event handling [Design] **VI:** Compare advanced rendering techniques such as global illumination and physically based rendering [Compare] **VII:** Analyze the role of graphics hardware in accelerating rendering tasks [Analyze] **VIII:** Explain how human perception influences graphics and interaction design [Explain]

Bibliography: [Brookshear and Brylow, 2019]

5.25 Applied Rendering and Techniques, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Ray tracing: acceleration structures, sampling, shading **II:** Texture mapping and filtering

Learning Outcomes **I:** Implement rasterization algorithms for primitive rendering [Implement] **II:** Implement basic ray tracing with acceleration structures [Implement]

Bibliography: [Brookshear and Brylow, 2019]

5.26 Closure Class: How Does a Search Engine Like Google Work?, 2 hr(s)

Competences: ABET-1,ABET-2,ABET-5

Topics: **I:** Problem analysis **II:** The index does not grow linearly with the size of the indexed information. **III:** Response time does not depend on the size of the "database." **IV:** Response time does not depend on the number of occurrences found. **V:** Combining various data structures to reach a solution. **VI:** Analyzing the scalability of the solution.

Learning Outcomes **I:** Understand the principles under which a search engine is created [Usage] **II:** Correctly apply data structures to solve the problem [Usage] **III:** Apply concepts related to algorithmic complexity in a search engine [Usage].

Bibliography: [Brookshear and Brylow, 2019]

6. Bibliography

[Brookshear and Brylow, 2019] Brookshear, J. G. and Brylow, D. (2019). *Computer Science: An Overview*. Pearson, global edition edition.

[Brookshear and Brylow, 2019] Brookshear, J. G. and Brylow, D. (2019). *Computer Science: An Overview*. Pearson, global edition edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS112 Objects-oriented programming I (Mandatory)

1. General information

- **Semester**: 2nd Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS111 Introduction to Programming (1st Sem-Pg. 295)

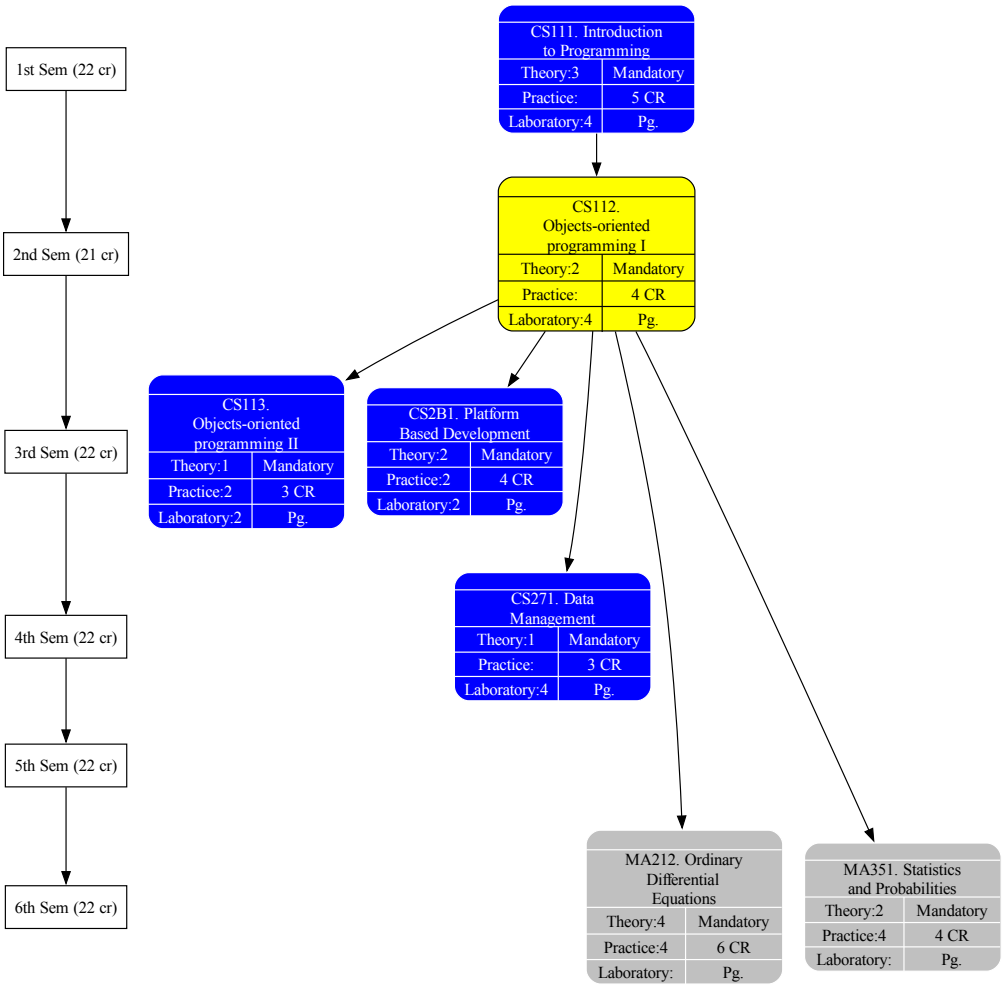


Figure 7.9: Connection Map. CS112 Objects-oriented programming I

2. Course Rationale

This is the second course in the introductory Computer Science programming sequence. It incorporates a programming language transition as a pedagogical strategy and focuses on core Object-Oriented Programming (OOP) concepts in C++17/C++20, expanding on the foundations from the first course while introducing low-level programming concepts. The course provides a solid foundation for advanced topics covered in the subsequent course of the curriculum.

3. Goals of the Course

- Introduce the student to the foundations of the object orientation paradigm in C++
- Develop skills in C++17/C++20 programming including templates and STL
- Understand memory management and low-level programming concepts
- Develop problem-solving skills using OOP principles
- Master intermediate C++ programming techniques

4. General Competences (*Outcomes*)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Language Pragmatics, 2 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Problem domains and programming paradigm **II:** Criteria for good programming language design: (a) Principles of language design such as orthogonality (b) Defining control and iteration constructs (c) Modularization of large software **III:** Brief review of programming paradigms **IV:** Comparison between functional and imperative programming **V:** History of programming languages (emphasis on C and C++)

Learning Outcomes **I:** Discuss the historical context of programming language paradigms [Familiarity]. **II:** Evaluate language design trade-offs for specific applications [Assessment].

Bibliography: [Stroustrup, 2013, Josuttis, 2019]

5.2 Language Translation and Execution, 4 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Execution models for JIT (Just-In-Time), compiler, interpreter **II:** Use of intermediate code, e.g., bytecode **III:** Execution as native code or within a virtual machine **IV:** Virtual machines and intermediate languages **V:** Compilation vs. interpretation models

Learning Outcomes **I:** Explain and understand the differences between compiled, JIT, and interpreted language implementations, including the benefits and limitations of each [Familiarity] **II:** Differentiate syntax and parsing from semantics and evaluation [Usage] **III:** Illustrate ambiguity in parsing using nested if-then-else/arithmetic expression and show resolution using precedence order [Assessment]

Bibliography: [Stroustrup, 2013, Deitel and Deitel, 2017]

5.3 Type Systems Fundamentals, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** A type as a set of values together with a set of operations: (a) Primitive types (e.g., numbers, Booleans) (b) Compound types built from other types (e.g., records/structs, unions, arrays, lists, functions, references using set operations) **II:** Generic types (parametric polymorphism): (a) Definition and advantages of polymorphism: parametric, subtyping, overloading, and coercion (b) Comparison of monomorphic and polymorphic types (c) Comparison with ad-hoc polymorphism (overloading) and subtype polymorphism (d) Generic parameters and typing (e) Use of generic libraries such as collections (f) Comparison with ad hoc polymorphism (overloading)

and subtype polymorphism (g) Prescriptive vs descriptive polymorphism (h) Implementation models of polymorphic types (i) Subtyping **III:** Declaration models (linking, visibility, scope, lifetime) **IV:** Overview of type checking in C++ **V:** Type conversion and casting in C++

Learning Outcomes **I:** Define and use program pieces (such as functions, classes, methods) that use generic types, including for collections [Usage] **II:** Discuss the differences among generics, subtyping, and overloading [Familiarity] **III:** Explain multiple benefits and limitations of static typing in writing, maintaining, and debugging software [Assessment]

Bibliography: [Stroustrup, 2013, Josuttis, 2019]

5.4 Object-Oriented Programming Fundamentals, 14 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Definition of classes: fields, methods, and constructors. **II:** Subclasses, inheritance (including multiple inheritance), and method overriding. **III:** Dynamic dispatch: definition of method-call. **IV:** Subtyping: (a) Subtype polymorphism; implicit upcasts in typed languages. (b) Notion of behavioral replacement: subtypes acting like supertype. (c) Relationship between subtyping and inheritance. **V:** Classes and objects in C++ **VI:** Constructors and destructors **VII:** Access specifiers and encapsulation **VIII:** Inheritance hierarchies **IX:** Polymorphism and virtual functions

Learning Outcomes **I:** Compose a class through design, implementation, and testing to meet behavioral requirements [Usage] **II:** Build a simple class hierarchy utilizing subclassing that allows code to be reused for distinct subclasses [Usage] **III:** Predict and validate control flow in a program using dynamic dispatch [Assessment] **IV:** Design C++ classes with proper encapsulation [Usage]. **V:** Implement inheritance hierarchies for code reuse [Usage].

Bibliography: [Stroustrup, 2013, Josuttis, 2019]

5.5 Systems Execution and Memory Model, 12 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Direct, indirect, and indexed access to memory location **II:** Run-time representation of data abstractions such as variables, arrays, vectors, records, pointer-based data elements such as linked-lists and trees, and objects **III:** Low level allocation and accessing of high-level data structures such as basic data types, n-dimensional array, vector, record, and objects **IV:** Return from procedure as automatic deallocation mechanism for local data elements in the stack **V:** Manual memory management: allocating, de-allocating, and reusing heap memory **VI:** Automated memory management: garbage collection as an automated technique using the notion of reachability **VII:** Memory layout (stack, heap, static) **VIII:** Pointer arithmetic and operations **IX:** Dynamic memory allocation (new/delete) **X:** Memory leaks and dangling pointers **XI:** Smart pointers (unique_ptr, shared_ptr)

Learning Outcomes **I:** Explain why memory leaks and dangling pointer problems occur, and what can be done by a programmer to avoid/fix them [Usage] **II:** Explain how programming language implementations typically organize memory into global data, text, heap, and stack sections and how features such as recursion and memory management map to this memory model [Assessment] **III:** Explain how a core language construct, such as data abstractions and control abstractions, is executed [Assessment] **IV:** Implement RAII for resource management [Usage]. **V:** Debug memory-related errors in C++ programs [Usage].

Bibliography: [Stroustrup, 2013, Deitel and Deitel, 2017]

5.6 Functional Programming, 10 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Lambda expressions and evaluation: (a) Variable binding and scope rules. (b) Parameter-passing. (c) Nested lambda expressions and reduction order. **II:** Using higher-order functions (taking, returning, and storing functions). **III:** Function templates and class templates **IV:** Template specialization **V:** Lambda expressions in C++11/14 **VI:** Standard Template Library concepts

Learning Outcomes **I:** Develop useful functions that take and return other functions [Usage] **II:** Correctly interpret variables and lexical scope in a program using function closures [Assessment] **III:** Use functional encapsulation mechanisms such as closures and modular interfaces [Usage] **IV:** Implement generic functions using templates [Usage]. **V:** Use lambda expressions for functional programming [Usage].

Bibliography: [Stroustrup, 2013, Josuttis, 2019]

5.7 Object-Oriented Programming: Encapsulation, Subtyping, and Reflection, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Collection classes, iterators, and other common library components. **II:** Sequence containers (vector, list, deque) **III:** Associative containers (set, map, unordered_set) **IV:** Container adapters (stack, queue, priority_queue) **V:** Iterators and algorithms **VI:** Functors and predicates

Learning Outcomes **I:** Use collection classes and iterators effectively to solve a problem [Usage] **II:** Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language [Usage] **III:** Compare and contrast mechanisms to define and protect data elements within procedural, functional, and object-oriented approaches [Assessment] **IV:** Select appropriate STL containers for specific needs [Usage].

Bibliography: [Stroustrup, 2013, Josuttis, 2019]

5.8 Advanced Programming Constructs, 6 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Object-oriented abstractions: multiple inheritance, mixins, traits, multimethods **II:** Metaprogramming: macros, generative programming, model-based development **III:** Operator overloading **IV:** Move semantics and rvalue references **V:** Exception handling in C++ **VI:** Namespaces and modular programming

Learning Outcomes **I:** Use various advanced programming constructs and idioms correctly [Usage] **II:** Discuss how various advanced programming constructs aim to improve program structure, software quality, and programmer productivity [Assessment] **III:** Discuss how various advanced programming constructs interact with the definition and implementation of other language features [Familiarity] **IV:** Implement move semantics for performance optimization [Usage].

Bibliography: [Stroustrup, 2013, Josuttis, 2019]

5.9 Shell Scripting, 4 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** System commands: (a) Interface with operating systems **II:** Piping **III:** File abstraction and operators **IV:** Programs and processes **V:** Automating compilation and testing of C++ programs through scripts (Makefiles, build/test scripts)

Learning Outcomes I: Create and execute automated scripts to manage various system tasks [Usage]

Bibliography: [Robbins and Beebe, 2005]

6. Bibliography

[Deitel and Deitel, 2017] Deitel, H. and Deitel, P. (2017). *C++17: The Complete Guide*. Pearson, Boston, MA, 10th edition.

[Josuttis, 2019] Josuttis, N. M. (2019). *C++17: The Complete Guide*. Pearson Education, Indianapolis, IN, 1st edition.

[Robbins and Beebe, 2005] Robbins, A. and Beebe, N. H. (2005). *Classic Shell Scripting*. O'Reilly Media, Sebastopol, CA, 1st edition.

[Stroustrup, 2013] Stroustrup, B. (2013). *The C++ Programming Language*. Addison-Wesley Professional, Upper Saddle River, NJ, 4th edition.

[Deitel and Deitel, 2017] Deitel, H. and Deitel, P. (2017). *C++17: The Complete Guide*. Pearson, Boston, MA, 10th edition.

[Josuttis, 2019] Josuttis, N. M. (2019). *C++17: The Complete Guide*. Pearson Education, Indianapolis, IN, 1st edition.

[Robbins and Beebe, 2005] Robbins, A. and Beebe, N. H. (2005). *Classic Shell Scripting*. O'Reilly Media, Sebastopol, CA, 1st edition.

[Stroustrup, 2013] Stroustrup, B. (2013). *The C++ Programming Language*. Addison-Wesley Professional, Upper Saddle River, NJ, 4th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

AI161 Applied AI (Mandatory)

1. General information

- **Semester**: 2nd Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS111 Introduction to Programming (1st Sem-Pg. 295)

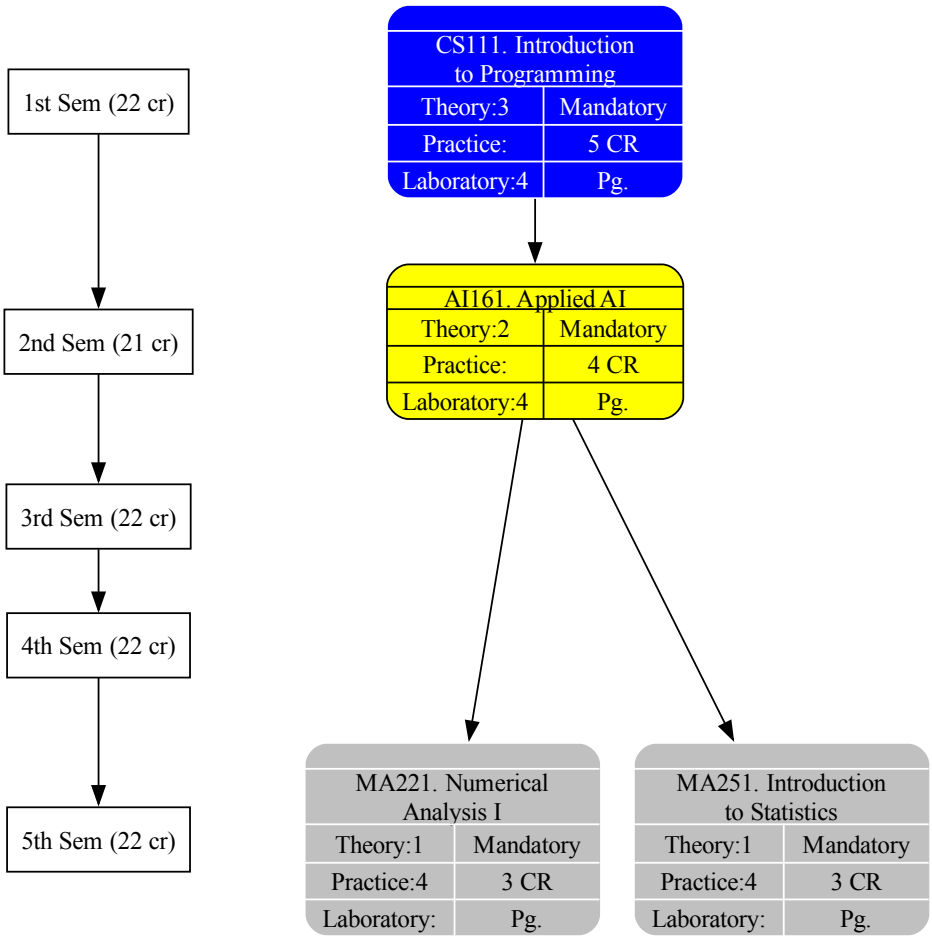


Figure 7.10: Connection Map. AI161 Applied AI

2. Course Rationale

This course provides a practical introduction to Artificial Intelligence (AI) for students from all scientific and engineering disciplines. Focused on developing AI literacy and practical skills, it covers fundamental concepts, modern AI tools (including Western and Chinese platforms), and responsible usage. Students will learn to effectively interact with diverse AI systems, write quality prompts, and apply AI solutions to problems across various domains while understanding ethical implications and cultural contexts of AI deployment.

3. Goals of the Course

- Develop comprehensive AI literacy by understanding fundamental concepts, capabilities, and limitations of modern AI systems across different platforms and cultures.
- Master effective prompt engineering techniques and interaction patterns with various AI tools (Western: ChatGPT, Claude, Gemini; Chinese: DeepSeek, Kimi, ERNIE).
- Apply diverse AI tools to solve practical problems in scientific, engineering, and academic contexts while maintaining critical evaluation of outputs across platforms.
- Understand ethical considerations, biases, cultural contexts, and responsible usage of AI technologies in global professional and academic settings.
- Develop workflows that integrate multiple AI tools to enhance productivity and problem-solving capabilities while understanding regional strengths and specializations.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts. (Familiarity)

5. Content

5.1 AI Fundamentals and Global Landscape, 16 hr(s)

Competences: ABET-1,ABET-5

Topics: **I:** What is AI? Definitions, history, and current global landscape. **II:** Types of AI systems: chatbots, image generators, research assistants across regions. **III:** Western AI ecosystems: OpenAI, Anthropic, Google, Microsoft. **IV:** Chinese AI ecosystems: DeepSeek, Kimi, ERNIE, Zhipu AI, Baidu. **V:** AI capabilities and limitations: comparative analysis across platforms. **VI:** Digital literacy in the AI era: critical thinking about diverse AI outputs.

Learning Outcomes **I:** Apply fundamental AI knowledge to identify system types and their capabilities [Usage]. **II:** Analyze global intelligent systems and their distinctive characteristics [Assessment]. **III:** Critically evaluate AI outputs across different cultural contexts [Assessment].

Bibliography: [Ng, 2019, Mollick and Mollick, 2024, UNESCO, 2023, Institute, 2024]

5.2 Effective AI Interaction and Cross-Platform Prompt Engineering, 20 hr(s)

Competences: ABET-1,ABET-5

Topics: **I:** Principles of effective prompt writing: clarity, context, constraints across platforms. **II:** Prompt patterns: persona, template, chain-of-thought, few-shot for different AI systems. **III:** Western tools: specific techniques for ChatGPT, Claude, Gemini, Copilot. **IV:** Chinese tools: specific features and best practices for DeepSeek, Kimi, ERNIE. **V:** Cross-platform strategies: leveraging different AI strengths for complex tasks. **VI:** Iterative refinement: how to improve prompts based on outputs from diverse systems. **VII:** Practical sessions: comparative prompt writing workshops across platforms.

Learning Outcomes **I:** Design effective prompt solutions for diverse AI systems [Assessment]. **II:** Develop cross-platform workflows that integrate different AI tools [Usage]. **III:** Implement iterative refinement strategies to optimize AI outputs [Usage].

Bibliography: [White et al., 2023, Prompt Engineering Institute, 2024, Microsoft, 2024, DeepSeek, 2024]

5.3 AI Applications Across Disciplines and Platforms, 16 hr(s)

Competences: ABET-1,ABET-5

Topics: **I:** AI for research: literature review, data analysis using multiple AI assistants. **II:** AI for writing: academic papers, reports with cross-platform verification. **III:** AI for problem-solving: scientific calculations, engineering design with specialized tools. **IV:** AI for creativity: brainstorming, concept development across cultural contexts. **V:** Platform-specific strengths: when to use Western vs. Chinese AI tools. **VI:** Discipline-specific workshops: tailored applications using diverse AI ecosystems. **VII:** Case studies: real-world applications in scientific research across regions.

Learning Outcomes **I:** Apply AI fundamentals to solve discipline-specific problems [Usage]. **II:** Design solutions that integrate multiple intelligent systems for complex tasks [Assessment]. **III:** Develop practical applications using Western and Chinese AI tools [Usage]. **IV:** Implement workflows that leverage specific strengths of each platform [Assessment].

Bibliography: [Mollick, 2023, Google, 2024, Cloud, 2024, OpenAI, 2024]

5.4 Global AI Ethics and Responsible Usage, 12 hr(s)

Competences: ABET-1,ABET-5

Topics: **I:** Understanding AI biases: Western and Eastern cultural perspectives. **II:** Ethical frameworks: comparing EU AI Act, Chinese regulations, and global standards. **III:** Academic integrity: proper citation and AI usage in coursework across platforms. **IV:** Privacy and data security: regional differences in AI data handling. **V:** Cultural sensitivity: navigating AI outputs in global contexts. **VI:** Environmental impact: sustainability considerations of different AI systems. **VII:** Developing personal guidelines for ethical AI usage in international settings. **VIII:** Case studies: ethical dilemmas in AI deployment across regions.

Learning Outcomes **I:** Demonstrate professional responsibility in using AI technologies [Assessment]. **II:** Apply professional ethical principles in global AI contexts [Usage]. **III:** Develop ethical guidelines to mitigate biases in AI systems [Assessment]. **IV:** Evaluate ethical implications of AI deployment across different cultures [Usage].

Bibliography: [UNESCO, 2023, Union, 2024, Bender et al., 2021, of Cyberspace Studies, 2024, Tencent AI Lab, 2023]

6. Bibliography

[Bender et al., 2021] Bender, E. M., Gebru, T., et al. (2021). On the dangers of stochastic parrots: Can language models be too big? *FAccT '21: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency.* Foundational paper on AI ethics with ongoing relevance.

[Cloud, 2024] Cloud, A. (2024). AI Ethics and best practices in chinese context. Chinese perspective on AI ethics and implementation guidelines.

[DeepSeek, 2024] DeepSeek (2024). Deepseek model documentation and best practices. Official documentation for DeepSeek AI models and usage guidelines.

[Google, 2024] Google (2024). Introduction to responsible ai. Google's updated framework for responsible AI development and use.

[Institute, 2024] Institute, T. R. (2024). Ai development in china: Current status and future trends. Technical report, Tencent Research Institute. Comprehensive analysis of Chinese AI ecosystem development.

[Microsoft, 2024] Microsoft (2024). Prompt crafting for ai systems. Official Microsoft prompt engineering guidance updated for 2024.

[Mollick, 2023] Mollick, E. (2023). Chatgpt and how ai disrupts industries. *Harvard Business Review*. Analysis of AI's practical impact across global sectors.

[Mollick and Mollick, 2024] Mollick, E. and Mollick, L. (2024). *Co-Intelligence: Living and Working with AI*. Penguin Random House. Practical guide to human-AI collaboration.

[Ng, 2019] Ng, A. (2019). AI for everyone. Technical report, DeepLearning.AI. Continuously updated online course, available on Coursera.

[of Cyberspace Studies, 2024] of Cyberspace Studies, C. A. (2024). Ai governance and ethics in china. Technical report, Cyberspace Administration of China. Official Chinese perspective on AI governance and ethical standards.

[OpenAI, 2024] OpenAI (2024). Best practices for prompt engineering. Updated official prompt engineering guidelines from OpenAI.

[Prompt Engineering Institute, 2024] Prompt Engineering Institute (2024). Prompt engineering guide. `https://www.promptingguide.ai/`. Recurso integral de ingeniería de prompts actualizado continuamente. Accedido: 15 de enero de 2024.

[Tencent AI Lab, 2023] Tencent AI Lab (2023). Responsible ai practices in chinese tech industry. Technical report, Tencent AI Lab. Industry perspective on AI ethics from leading Chinese tech company.

[UNESCO, 2023] UNESCO (2023). AI and education: Guidance for policy-makers. Technical report, UNESCO. Latest guidance on AI literacy and ethical implementation in education.

[Union, 2024] Union, E. (2024). AI Act: Regulatory framework. Technical report, European Commission. Comprehensive AI regulation framework effective 2024.

[White et al., 2023] White, J. et al. (2023). A prompt pattern catalog to enhance prompt engineering with chatgpt. Peer-reviewed research on prompt engineering patterns.

[Bender et al., 2021] Bender, E. M., Gebru, T., et al. (2021). On the dangers of stochastic parrots: Can language models be too big? *FAccT '21: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. Foundational paper on AI ethics with ongoing relevance.

[Cloud, 2024] Cloud, A. (2024). AI Ethics and best practices in chinese context. Chinese perspective on AI ethics and implementation guidelines.

[DeepSeek, 2024] DeepSeek (2024). Deepseek model documentation and best practices. Official documentation for DeepSeek AI models and usage guidelines.

[Google, 2024] Google (2024). Introduction to responsible ai. Google's updated framework for responsible AI development and use.

[Institute, 2024] Institute, T. R. (2024). Ai development in china: Current status and future trends. Technical report, Tencent Research Institute. Comprehensive analysis of Chinese AI ecosystem development.

[Microsoft, 2024] Microsoft (2024). Prompt crafting for ai systems. Official Microsoft prompt engineering guidance updated for 2024.

[Mollick, 2023] Mollick, E. (2023). Chatgpt and how ai disrupts industries. *Harvard Business Review*. Analysis of AI's practical impact across global sectors.

[Mollick and Mollick, 2024] Mollick, E. and Mollick, L. (2024). *Co-Intelligence: Living and Working with AI*. Penguin Random House. Practical guide to human-AI collaboration.

[Ng, 2019] Ng, A. (2019). AI for everyone. Technical report, DeepLearning.AI. Continuously updated online course, available on Coursera.

[of Cyberspace Studies, 2024] of Cyberspace Studies, C. A. (2024). Ai governance and ethics in china. Technical report, Cyberspace Administration of China. Official Chinese perspective on AI governance and ethical standards.

[OpenAI, 2024] OpenAI (2024). Best practices for prompt engineering. Updated official prompt engineering guidelines from OpenAI.

[Prompt Engineering Institute, 2024] Prompt Engineering Institute (2024). Prompt engineering guide. `https://www.promptingguide.ai/`. Recurso integral de ingeniería de prompts actualizado continuamente. Accedido: 15 de enero de 2024.

[Tencent AI Lab, 2023] Tencent AI Lab (2023). Responsible ai practices in chinese tech industry. Technical report, Tencent AI Lab. Industry perspective on AI ethics from leading Chinese tech company.

[UNESCO, 2023] UNESCO (2023). AI and education: Guidance for policy-makers. Technical report, UNESCO. Latest guidance on AI literacy and ethical implementation in education.

[Union, 2024] Union, E. (2024). AI Act: Regulatory framework. Technical report, European Commission. Comprehensive AI regulation framework effective 2024.

[White et al., 2023] White, J. et al. (2023). A prompt pattern catalog to enhance prompt engineering with chatgpt. Peer-reviewed research on prompt engineering patterns.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA111 Calculus I (Mandatory)

1. General information

- **Semester**: 2nd Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA100 Basic Mathematics (1st Sem-Pg. 300)

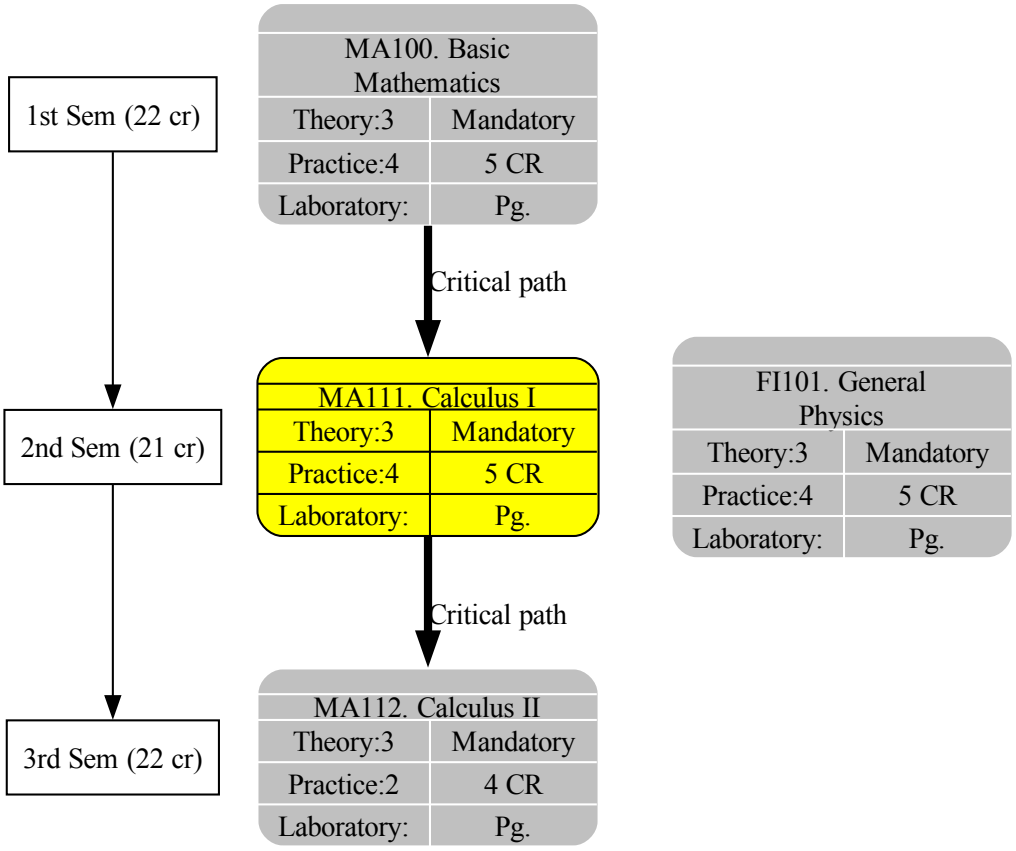


Figure 7.11: Connection Map. MA111 Calculus I

2. Course Rationale

The course includes knowledge of binary relations and real functions of a real variable. Limits, Continuity, and Derivatives of real functions, applications (maxima and minima), and Differentials applied to the solution of problems involving living beings.

3. Goals of the Course

- Understand the fundamental concepts of limits and continuity.
- Master differentiation techniques for algebraic and transcendental functions.
- Apply derivatives to solve optimization problems and analyze rates of change.
- Develop a rigorous approach to the formal definition of a derivative.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Familiarity)

5. Content

5.1 Computational Arithmetic and Error Analysis, 25 hr(s)

Competences: ABET-1

Topics: **I:** Floating-point arithmetic and the IEEE standard **II:** Sources of error: truncation, rounding, and discretization **III:** Error propagation and condition number **IV:** Stability of numerical algorithms **V:** Absolute, relative, and forward/backward error

Learning Outcomes **I:** Define the different types of errors in numerical computation (rounding, truncation, discretization) [Familiarity] **II:** Explain the principles of floating-point representation and arithmetic according to the IEEE standard [Familiarity] **III:** Calculate the absolute and relative error of an approximation [Usage] **IV:** Analyze the propagation of errors in a sequence of arithmetic operations [Usage] **V:** Distinguish between the condition of a problem and the stability of an algorithm [Assessment] **VI:** Compute the condition number of a simple problem (e.g., matrix-vector multiplication) [Usage]

Bibliography: [Stewart, 2016, Larson and Edwards, 2018]

5.2 Roots Finding and Basic Numerical Optimization, 25 hr(s)

Competences: ABET-1

Topics: **I:** Bisection method and fixed-point iteration **II:** Newton's method and secant method for root finding **III:** Convergence analysis and order of convergence **IV:** Gradient descent for univariate optimization **V:** Rate of convergence and stopping criteria

Learning Outcomes **I:** Implement root-finding algorithms (bisection, Newton, secant) and analyze their convergence [Usage] **II:** Classify the order of convergence of iterative methods [Assessment] **III:** Apply gradient descent to solve simple univariate optimization problems [Usage] **IV:** Design appropriate stopping criteria for iterative methods [Usage]

Bibliography: [Stewart, 2016, Thomas et al., 2014]

6. Bibliography

[Larson and Edwards, 2018] Larson, R. and Edwards, B. H. (2018). *Cálculo*. Cengage Learning, 10th edition.

[Stewart, 2016] Stewart, J. (2016). *Cálculo de una variable: Trascendentes tempranas*. Cengage Learning, 8th edition.

[Thomas et al., 2014] Thomas, G. B., Weir, M. D., and Hass, J. (2014). *Cálculo de una variable*. Pearson, 13th edition.

[Larson and Edwards, 2018] Larson, R. and Edwards, B. H. (2018). *Cálculo*. Cengage Learning, 10th edition.

[Stewart, 2016] Stewart, J. (2016). *Cálculo de una variable: Trascendentes tempranas*. Cengage Learning, 8th edition.

[Thomas et al., 2014] Thomas, G. B., Weir, M. D., and Hass, J. (2014). *Cálculo de una variable*. Pearson, 13th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

FI101 General Physics (Mandatory)

1. General information

- **Semester**: 2nd Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA111 Calculus I (2nd Sem-Pg. 338)

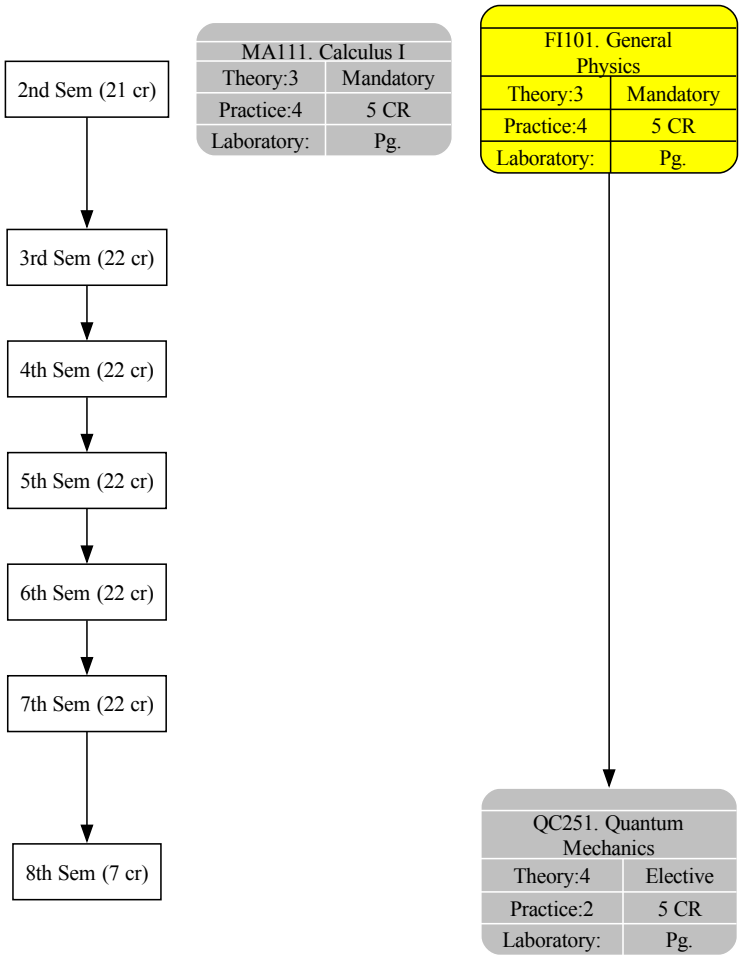


Figure 7.12: Connection Map. FI101 General Physics

2. Course Rationale

The course includes knowledge of the fundamental principles and laws of Physics, Newtonian Classical Mechanics, Kinematics, Statics, Dynamics, Energy and Work, conservative and non-conservative forces, Energy and Momentum. It emphasizes acoustic phenomena, electromagnetic waves, and optics with applications in biological sciences.

3. Goals of the Course

- Understand and apply Newton's laws in the analysis of motion and equilibrium of bodies.

- Apply the concepts of work, energy, and conservation in solving complex mechanical problems.
- Analyze the laws of thermodynamics, fluid behavior, and heat transfer phenomena.
- Study the principles of optics and wave phenomena to understand the nature of light and sound.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Particle and Rigid Body Mechanics, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Physical quantities, systems of units, and vectors. **II:** Kinematics: Motion in one and two dimensions. **III:** Dynamics: Newton's laws and their applications. **IV:** Statics and equilibrium of the rigid body.

Learning Outcomes **I:** Describe the motion of bodies using kinematic and vector magnitudes [Familiarity]. **II:** Apply Newton's laws to determine forces and accelerations in physical systems [Usage]. **III:** Solve static equilibrium problems through the analysis of forces and torque [Assessment].

Bibliography: [Sears et al., 2013, Tipler and Mosca, 2010]

5.2 Work, Energy, and Momentum, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Work and kinetic energy. **II:** Potential energy and conservation of mechanical energy. **III:** Linear momentum and impulse. **IV:** Elastic and inelastic collisions.

Learning Outcomes **I:** Identify forms of energy and their transformation in conservative physical processes [Familiarity]. **II:** Use the work-energy theorem to analyze dynamic systems [Usage]. **III:** Apply the conservation of momentum in solving collision problems [Assessment].

Bibliography: [Sears et al., 2013, Resnick et al., 2005]

5.3 Fluid Mechanics and Thermodynamics, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Fluid statics and dynamics (Bernoulli and Archimedes). **II:** Temperature, heat, and laws of thermodynamics. **III:** Kinetic theory of gases. **IV:** Thermodynamic processes and heat engines.

Learning Outcomes **I:** Understand the fundamental properties of fluids and heat transfer mechanisms [Familiarity]. **II:** Calculate state variables in thermal processes using the laws of thermodynamics [Usage]. **III:** Model the behavior of fluids and gases through the implementation of fundamental equations [Assessment].

Bibliography: [Sears et al., 2013, Serway and Jewett, 2018]

5.4 Waves and Optics, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Wave motion and acoustic phenomena. **II:** Nature and propagation of light. **III:** Geometric optics: Reflection and refraction. **IV:** Electromagnetic waves and interference.

Learning Outcomes **I:** Recognize the characteristics of mechanical and electromagnetic waves in the environment [Familiarity]. **II:** Analyze image formation in mirrors and lenses using ray tracing [Usage]. **III:** Evaluate interference and diffraction phenomena based on the wave model of light [Assessment].

Bibliography: [Alonso and Finn, 1971, Sears et al., 2009]

6. Bibliography

[Alonso and Finn, 1971] Alonso, M. and Finn, E. (1971). *Física I.* Fondo Educativo Interamericano.

[Resnick et al., 2005] Resnick, R., Halliday, D., and Krane, K. (2005). *Física Vol. 1.* CECSA, 5th edition.

[Sears et al., 2009] Sears, F., Zemansky, M., Young, H., and Freedman, R. (2009). *Física Universitaria, Vol. 2.* Pearson, 12th edition.

[Sears et al., 2013] Sears, F., Zemansky, M., Young, H., and Freedman, R. (2013). *Física Universitaria, Vol. 1.* Pearson, 13th edition.

[Serway and Jewett, 2018] Serway, R. and Jewett, J. (2018). *Física para Ciencias e Ingeniería Vol. 1.* Cengage Learning, 10th edition.

[Tipler and Mosca, 2010] Tipler, P. and Mosca, G. (2010). *Física para la Ciencia y la Tecnología. Vol. 1.* Reverté, 6th edition.

[Alonso and Finn, 1971] Alonso, M. and Finn, E. (1971). *Física I.* Fondo Educativo Interamericano.

[Resnick et al., 2005] Resnick, R., Halliday, D., and Krane, K. (2005). *Física Vol. 1.* CECSA, 5th edition.

[Sears et al., 2009] Sears, F., Zemansky, M., Young, H., and Freedman, R. (2009). *Física Universitaria, Vol. 2.* Pearson, 12th edition.

[Sears et al., 2013] Sears, F., Zemansky, M., Young, H., and Freedman, R. (2013). *Física Universitaria, Vol. 1.* Pearson, 13th edition.

[Serway and Jewett, 2018] Serway, R. and Jewett, J. (2018). *Física para Ciencias e Ingeniería Vol. 1.* Cengage Learning, 10th edition.

[Tipler and Mosca, 2010] Tipler, P. and Mosca, G. (2010). *Física para la Ciencia y la Tecnología. Vol. 1.* Reverté, 6th edition.

Third Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS113 Objects-oriented programming II (Mandatory)

1. General information

- **Semester**: 3rd Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - CS112 Objects-oriented programming I (2nd Sem-Pg. 328)

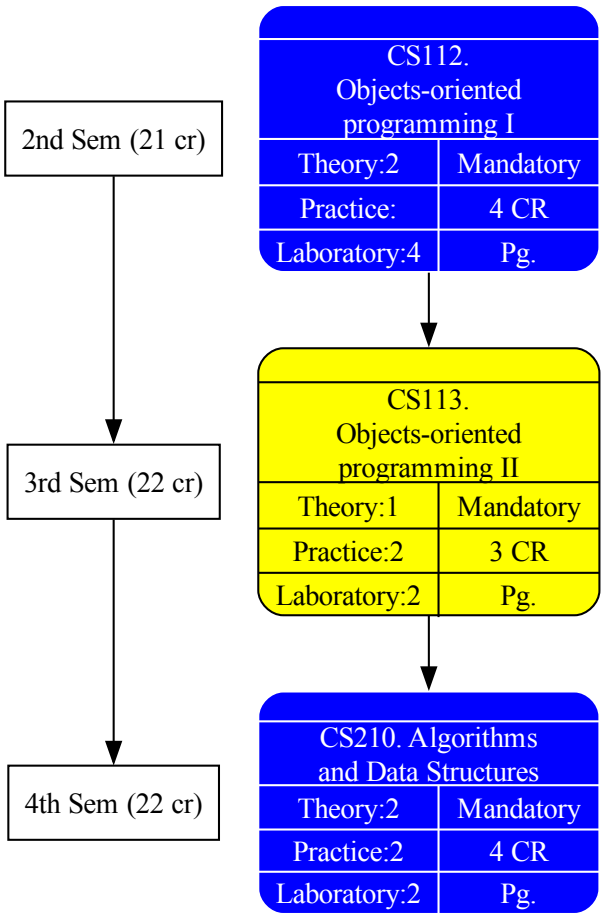


Figure 7.13: Connection Map. CS113 Objects-oriented programming II

2. Course Rationale

This is the third course in the sequence of introductory courses in computer science. This course explores advanced C++17/C++20 features focusing on high-performance systems development. Key topics include template metaprogramming, move semantics, perfect forwarding, RAII optimization,

multiple/virtual inheritance patterns, concurrency with std::thread and async/await, lock-free programming, modern C++ paradigms such as CRTP and expression templates, and interfacing C++ with other languages. The course prepares students for game engines, high-performance computing, and embedded development where C++ dominates.

3. Goals of the Course

- Master advanced C++17/C++20 features including templates and metaprogramming
- Develop skills in concurrent and systems programming
- Understand performance optimization techniques
- Prepare for high-performance computing applications
- Master template metaprogramming and compile-time computation

4. General Competences (*Outcomes*)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Object-Oriented Programming: Encapsulation, Subtyping, and Reflection, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Collection classes, iterators, and other common library components. **II:** Associative containers (std::set, std::map, unordered variants) **III:** Container adapters (stack, queue, priority_queue) **IV:** Advanced STL algorithms **V:** Functors and predicates **VI:** Iterator categories and traits

Learning Outcomes **I:** Use collection classes and iterators effectively to solve a problem [Usage] **II:** Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language [Assessment] **III:** Compare and contrast the benefits and costs/impact of using inheritance (subclasses) and composition (specifically, how to base composition on higher order functions) [Usage] **IV:** Implement custom allocators for STL containers [Usage].

Bibliography: [Stroustrup, 2013, Josuttis, 2019, Vandevoorde et al., 2017]

5.2 Type Systems: Polymorphism and Complementary Typing, 12 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Generic types (parametric polymorphism): (a) Definition and advantages of polymorphism: parametric, subtyping, overloading, and coercion (b) Comparison of monomorphic and polymorphic types (c) Comparison with ad-hoc polymorphism (overloading) and subtype polymorphism (d) Generic parameters and typing (e) Use of generic libraries such as collections (f) Comparison with ad hoc polymorphism (overloading) and subtype polymorphism (g) Prescriptive vs descriptive polymorphism (h) Implementation models of polymorphic types (i) Subtyping **II:** SFINAE (Substitution Failure Is Not An Error): historical foundation of constrained generic programming in C++, the basis of pre-C++20 libraries. **III:** Concepts (C++20): declarative syntax to constrain template parameters, the modern successor to SFINAE with better error diagnostics. **IV:** if constexpr (C++17): compile-time branching, replacing much of the tag dispatch previously handled with SFINAE. **V:** Perfect forwarding and universal references **VI:** Variadic templates **VII:** Compile-time computation with constexpr and consteval

Learning Outcomes **I:** Define and use program pieces (such as functions, classes, methods) that use generic types, including for collections [Assessment] **II:** Discuss the differences among generics, subtyping, and overloading [Assessment] **III:** Apply Concepts to constrain template parameters and improve error diagnostics [Usage]. **IV:** Distinguish when to use Concepts, if constexpr, or legacy SFINAE when maintaining existing C++ code [Assessment]. **V:** Implement perfect forwarding for efficient argument passing [Usage].

Bibliography: [Stroustrup, 2013, Vandevoorde et al., 2017, Stroustrup, 2022]

5.3 Memory Management, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Low level allocation and accessing of high-level data structures such as basic data types, n-dimensional array, vector, record, and objects **II:** Return from procedure as automatic deallocation mechanism for local data elements in the stack **III:** Manual memory management: allocating, de-allocating, and reusing heap memory **IV:** Automated memory management: garbage collection as an automated technique using the notion of reachability **V:** Lvalues and rvalues in C++11/14/17/20 **VI:** Move constructors and move assignment operators **VII:** Rvalue references and reference collapsing **VIII:** Resource Acquisition Is Initialization (RAII) pattern **IX:** Smart pointer customization **X:** Memory leak detection and prevention

Learning Outcomes **I:** Explain why memory leaks and dangling pointer problems occur, and what can be done by a programmer to avoid/fix them [Usage] **II:** Explain how programming language implementations typically organize memory into global data, text, heap, and stack sections and how features such as recursion and memory management map to this memory model [Assessment] **III:** Explain how a core language construct, such as data abstractions and control abstractions, is executed [Assessment] **IV:** Implement move semantics for resource optimization [Usage]. **V:** Design RAII classes for exception-safe resource management [Usage].

Bibliography: [Meyers, 2014, Stroustrup, 2013]

5.4 Parallel and Distributed Computing, 12 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Safety and liveness: (a) Race conditions (b) Dependencies/preconditions (c) Fault models (d) Termination **II:** Programming models: (a) Actor models (b) Procedural and reactive models (c) Synchronous/asynchronous programming models (d) Data parallelism **III:** Communication and coordination: (a) Mutexes (b) Message-passing (c) Shared memory (d) Cobegin-coend (e) Monitors (f) Channels (g) Threads (h) Guards **IV:** std::thread and thread management **V:** std::jthread (C++20): automatic joining (RAII) and cooperative cancellation via stop_token **VI:** Mutexes, locks, and condition variables **VII:** C++20 synchronization primitives: std::latch, std::barrier, std::counting_semaphore **VIII:** Atomic operations and lock-free programming **IX:** std::async and futures **X:** std::execution and the senders-receivers model (C++26, experimental): a preview of where standard C++ concurrency is heading, presented at a conceptual level only, not evaluated in lab given its limited availability (partial in GCC) **XI:** Thread-safe design patterns **XII:** Data races and deadlock prevention

Learning Outcomes **I:** Implement correct concurrent programs using multiple programming models, such as shared memory, actors, futures, synchronization constructs, and data-parallelism primitives [Usage] **II:** Use synchronization constructions such as monitor/synchronized methods in a simple program [Usage] **III:** Use a message-passing model to analyze a communication protocol [Assessment] **IV:** Implement lock-free data structures using atomics [Usage]. **V:** Differentiate std::thread from std::jthread and justify when to use each [Usage]. **VI:** Analyze thread-safety of concurrent algorithms [Assessment].

Bibliography: [Williams, 2019, Stroustrup, 2013]

5.5 Advanced Programming Constructs, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Object-oriented abstractions: multiple inheritance, mixins, traits, multimethods **II:** Metaprogramming: macros, generative programming, model-based development **III:** Multiple inheritance and virtual inheritance **IV:** CRTP (Curiously Recurring Template Pattern) **V:** deducing this (C++23): simplifies CRTP by removing the need for static_cast<Derived&>(*this) for static polymorphism **VI:** Expression templates **VII:** Type traits and compile-time introspection **VIII:** Policy-based design **IX:** Interface design for libraries

Learning Outcomes **I:** Use various advanced programming constructs and idioms correctly [Usage] **II:** Discuss how various advanced programming constructs aim to improve program structure, software quality, and programmer productivity [Assessment] **III:** Discuss how various advanced programming constructs interact with the definition and implementation of other language features [Assessment] **IV:** Implement CRTP for static polymorphism [Usage]. **V:** Design extensible library interfaces using policy-based design [Usage].

Bibliography: [Stroustrup, 2013, Gregor, 2017]

5.6 Systems Execution and Memory Model, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Run-time representation of data abstractions such as variables, arrays, vectors, records, pointer-based data elements such as linked-lists and trees, and objects **II:** Data structures for translation, execution, translation, and code mobility such as stack, heap, aliasing (sharing using pointers), indexed sequence and string **III:** Implementation of the dynamic dispatch introduced in OOP I: virtual function table (vtable), vtable pointer (vptr), and its memory/performance cost versus static dispatch **IV:** Interfacing C++ with other languages (FFI) **V:** System calls and OS interaction **VI:** Memory-mapped I/O **VII:** Performance profiling and benchmarking **VIII:** Optimization techniques for HPC

Learning Outcomes **I:** Explain how a core language construct, such as data abstractions and control abstractions, is executed [Usage] **II:** Explain how programming language implementations typically organize memory into global data, text, heap, and stack sections and how features such as recursion and memory management map to this memory model [Assessment] **III:** Explain why memory leaks and dangling pointer problems occur, and what can be done by a programmer to avoid/fix them [Usage] **IV:** Profile and optimize C++ programs for performance [Usage]. **V:** Explain how the compiler implements runtime polymorphism via vtables [Assessment]. **VI:** Interface C++ code with C libraries [Usage].

Bibliography: [Stroustrup, 2013, Williams, 2019]

5.7 Advanced Programming Constructs, 6 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Object-oriented abstractions: multiple inheritance, mixins, traits, multimethods **II:** Creational, structural, and behavioral patterns **III:** Thread-safe singleton patterns **IV:** Modern C++ design idioms **V:** Dependency injection and inversion of control

Learning Outcomes **I:** Use various advanced programming constructs and idioms correctly [Usage] **II:** Discuss how various advanced programming constructs aim to improve program structure, software quality, and programmer productivity [Assessment] **III:** Discuss how various advanced programming constructs interact with the definition and implementation of other language features [Familiarity] **IV:** Implement thread-safe design patterns [Usage]. **V:** Evaluate architectural trade-offs in C++ systems [Assessment].

Bibliography: [Gamma et al., 1994, Stroustrup, 2013]

5.8 Advanced Programming Constructs, 6 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Control abstractions: exception handling, continuations, monads. **II:** Compiler optimizations and flags **III:** Benchmarking methodologies **IV:** Cache-friendly data structures **V:** Branch prediction and alignment **VI:** Reducing compilation time **VII:** Continuous integration for performance testing

Learning Outcomes **I:** Use various advanced programming constructs and idioms correctly [Usage] **II:** Discuss how various advanced programming constructs aim to improve program structure, software quality, and programmer productivity [Assessment] **III:** Discuss how various advanced programming constructs interact with the definition and implementation of other language features [Familiarity] **IV:** Apply optimization techniques to improve program performance [Usage]. **V:** Interpret benchmark results to guide optimization decisions [Usage].

Bibliography: [Meyers, 2014, Stroustrup, 2013]

5.9 Version Control and CI/CD, 4 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Software process automation: (a) Build systems - the value of fast, hermetic, reproducible builds, compare/contrast approaches to building a project. (b) Continuous Integration (CI) - the use of automation and automated tests to do preliminary validation that the current head/trunk revision builds and passes (basic) tests. (c) Continuous Deployment (CD) - the use of automation to automatically release every change that passes the automated tests to the production environment, ensuring frequent and reliable deliveries. (d) Dependency management - updating external/upstream dependencies, package management, SemVer. **II:** Build systems for C++: CMake, the de facto industry standard **III:** Modern alternatives: xmake, Meson, and Bazel – comparing declarative approaches versus Lua/Python-based ones **IV:** Organizing multi-file projects: translation units, headers, static and dynamic libraries **V:** Dependency management in C++ (vcpkg, Conan)

Learning Outcomes **I:** Understand the use of CI/CD systems as a ground-truth for the state of the team's shared code (build and test success) [Familiarity] **II:** Configure a multi-file C++ project using a declarative build system [Usage]. **III:** Compare CMake with at least one modern alternative (xmake, Meson, or Bazel) in terms of build speed and reproducibility [Assessment].

Bibliography: [Stroustrup, 2013, Stroustrup, 2022]

6. Bibliography

[Gamma et al., 1994] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, Reading, MA, 1st edition.

[Gregor, 2017] Gregor, D. (2017). *Advanced C++: Master the Technique of Template Metaprogramming*. Addison-Wesley Professional, Upper Saddle River, NJ, 1st edition.

[Josuttis, 2019] Josuttis, N. M. (2019). *C++17: The Complete Guide*. Pearson Education, Indianapolis, IN, 1st edition.

[Meyers, 2014] Meyers, S. (2014). *Effective Modern C++: 42 Specific Ways to Improve Your Use of C++11 and C++14*. O'Reilly Media, Sebastopol, CA, 1st edition.

[Stroustrup, 2013] Stroustrup, B. (2013). *The C++ Programming Language*. Addison-Wesley Professional, Upper Saddle River, NJ, 4th edition.

[Stroustrup, 2022] Stroustrup, B. (2022). *A Tour of C++*. Addison-Wesley Professional, Boston, MA, 3rd edition.

[Vandevoorde et al., 2017] Vandevoorde, D., Josuttis, N. M., and Gregor, D. (2017). *C++ Templates: The Complete Guide*. Addison-Wesley Professional, Upper Saddle River, NJ, 2nd edition.

[Williams, 2019] Williams, A. (2019). *C++ Concurrency in Action*. Manning Publications, Shelter Island, NY, 2nd edition.

[Gamma et al., 1994] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, Reading, MA, 1st edition.

[Gregor, 2017] Gregor, D. (2017). *Advanced C++: Master the Technique of Template Metaprogramming*. Addison-Wesley Professional, Upper Saddle River, NJ, 1st edition.

[Josuttis, 2019] Josuttis, N. M. (2019). *C++17: The Complete Guide*. Pearson Education, Indianapolis, IN, 1st edition.

[Meyers, 2014] Meyers, S. (2014). *Effective Modern C++: 42 Specific Ways to Improve Your Use of C++11 and C++14*. O'Reilly Media, Sebastopol, CA, 1st edition.

[Stroustrup, 2013] Stroustrup, B. (2013). *The C++ Programming Language*. Addison-Wesley Professional, Upper Saddle River, NJ, 4th edition.

[Stroustrup, 2022] Stroustrup, B. (2022). *A Tour of C++*. Addison-Wesley Professional, Boston, MA, 3rd edition.

[Vandevoorde et al., 2017] Vandevoorde, D., Josuttis, N. M., and Gregor, D. (2017). *C++ Templates: The Complete Guide*. Addison-Wesley Professional, Upper Saddle River, NJ, 2nd edition.

[Williams, 2019] Williams, A. (2019). *C++ Concurrency in Action*. Manning Publications, Shelter Island, NY, 2nd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS1D1 Discrete Structures I (Mandatory)

1. General information

- **Semester**: 3rd Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - MA100 Basic Mathematics (1st Sem-Pg. 300)

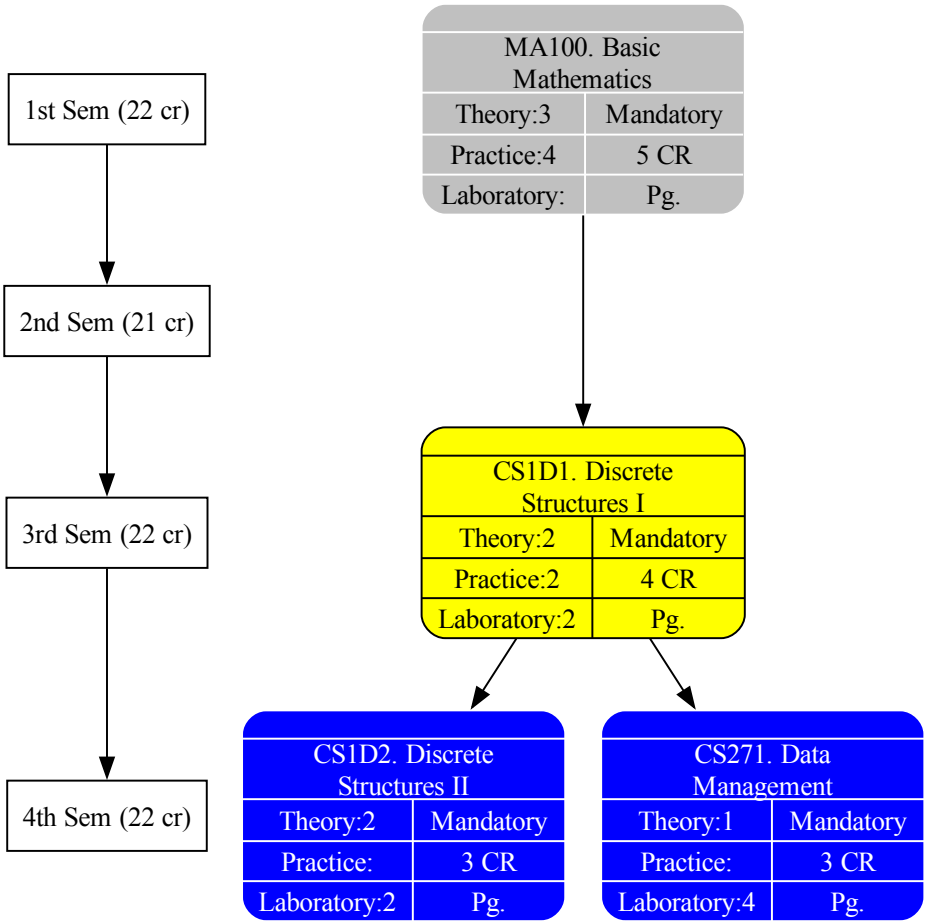


Figure 7.14: Connection Map. CS1D1 Discrete Structures I

2. Course Rationale

Discrete structures are the backbone of computer science. This first course focuses on the fundamental logic and formal reasoning necessary to evaluate the correctness of computational processes. Students will master the formal language of mathematics to avoid ambiguity and will learn various proof techniques essential for the rigorous analysis of algorithms and software specifications.

3. Goals of the Course

- Understand the principles of mathematical logic and propositional calculus.

- Apply predicate logic to model complex constraints in computing.
- Master multiple proof techniques, including direct, contradiction, and induction.
- Develop the ability to formally argue the correctness of mathematical statements.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Familiarity)

5. Content

5.1 Logic and Foundations, 24 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Logic: propositional and predicate logic, quantifiers, proofs **II:** Sets: operations, cardinality, power sets, Cartesian products **III:** Relations: properties, equivalence relations, partial orders **IV:** Functions: injective, surjective, bijective, composition, inverse **V:** Advanced logic: proof systems, completeness, decidability **VI:** Advanced set theory: infinite sets, Cantor's diagonal argument

Learning Outcomes **I:** Explain propositional and predicate logic [Explain] **II:** Construct simple logical proofs [Apply] **III:** Define basic set operations and relations [Explain] **IV:** Define properties of relations [Explain] **V:** Define types of functions [Explain]

Bibliography: [Rosen, 2019, Lehman et al., 2018]

5.2 Proof Techniques and Advanced Structures, 24 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Logic: propositional and predicate logic, quantifiers, proofs **II:** Advanced relations: closures, lattices **III:** Advanced functions: recursive functions, computability

Learning Outcomes **I:** Construct simple logical proofs [Apply] **II:** Apply set operations to solve problems [Apply] **III:** Determine if a relation is an equivalence relation [Apply] **IV:** Compose functions and find inverses [Apply]

Bibliography: [Rosen, 2019, Velleman, 2019]

6. Bibliography

[Lehman et al., 2018] Lehman, E., Leighton, F. T., and Meyer, A. R. (2018). *Mathematics for Computer Science*. MIT OpenCourseWare.

[Rosen, 2019] Rosen, K. H. (2019). *Discrete Mathematics and Its Applications*. McGraw-Hill Education, 8th edition.

[Velleman, 2019] Velleman, D. J. (2019). *How to Prove It: A Structured Approach*. Cambridge University Press, 3rd edition.

[Lehman et al., 2018] Lehman, E., Leighton, F. T., and Meyer, A. R. (2018). *Mathematics for Computer Science*. MIT OpenCourseWare.

[Rosen, 2019] Rosen, K. H. (2019). *Discrete Mathematics and Its Applications*. McGraw-Hill Education, 8th edition.

[Velleman, 2019] Velleman, D. J. (2019). *How to Prove It: A Structured Approach*. Cambridge University Press, 3rd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS2B1 Platform Based Development (Mandatory)

1. General information

- **Semester**: 3rd Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - CS112 Objects-oriented programming I (2nd Sem-Pg. 328)

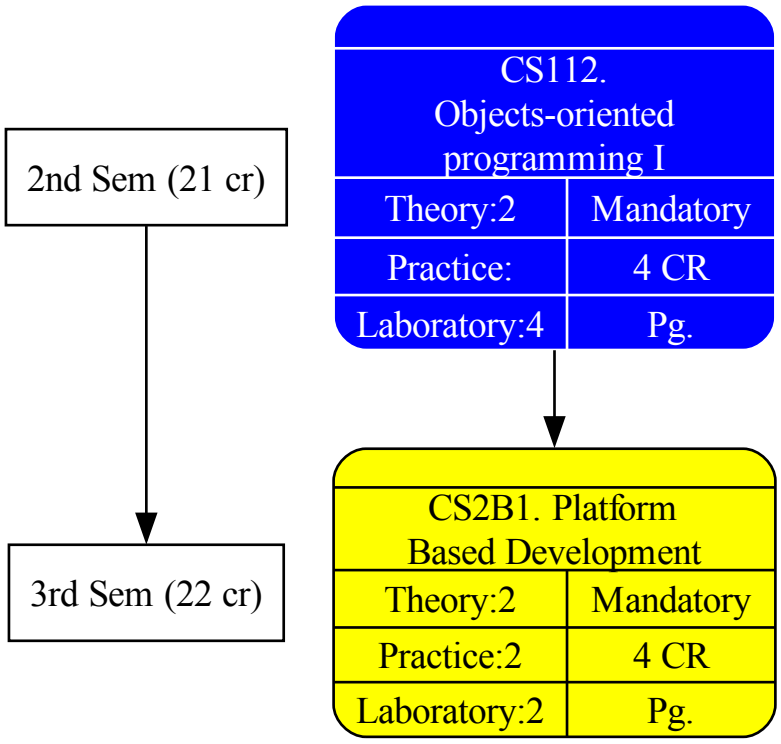


Figure 7.15: Connection Map. CS2B1 Platform Based Development

2. Course Rationale

Platform-based development focuses on the design and implementation of applications within specific computing environments. This course utilizes the Software Process and Design (SPD) framework to explore the unique constraints and capabilities of Web, Mobile, and Interactive platforms. Students will learn to apply shared architectural concerns and platform-specific APIs to build robust, scalable, and user-centric software solutions.

3. Goals of the Course

- Understand the shared aspects and unique constraints of modern platforms.
- Develop professional applications for Web and Mobile environments.
- Apply interactive computing principles to enhance user experience.
- Design software architectures that leverage platform-specific services.

4. General Competences (*Outcomes*)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

ABET-4) An ability to communicate effectively with a range of audiences. (Usage)

5. Content

5.1 Common Aspects: Platforms, APIs and Constraints, 5 hr(s)

Competences: ABET-2,ABET-4

Topics: I: Overview of development platforms (i.e., web, mobile, game, robotics, embedded, and interactive). (a) Input/sensors/control devices/haptic devices (b) Resource constraints i. Computational ii. Data storage iii. Memory iv. Communication (c) Requirements - security, uptime availability, fault tolerance Reliability Concepts, Reliability Engineering, Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering (d) Output/actuators/haptic devices **II:** Programming via platform-specific Application Programming Interface (API) vs traditional application construction **III:** Programming under platform constraints and requirements (e.g., available development tools, development, security considerations) AI and Hardware Security, Access Control and Applications, Security Mindset and Principles

Learning Outcomes I: List the constraints of mobile programming [List]

Bibliography: [Fielding, 2000, Gasca, 2021]

5.2 Common Aspects: Platform Languages and Web Programming Patterns, 3 hr(s)

Competences: ABET-2,ABET-4

Topics: I: Overview of platform Languages (e.g., Python, Swift, Lua, Kotlin) **II:** Techniques for learning and mastering a platform-specific programming language

Learning Outcomes I: List the characteristics of scripting languages [List] **II:** Describe the three-tier model of web programming [Describe] **III:** Describe how the state is maintained in web programming [Describe]

Bibliography: [Fielding, 2000, Gasca, 2021]

5.3 Frontend, Frameworks and Web Standards, 12 hr(s)

Competences: ABET-2,ABET-4

Topics: I: Web programming languages (e.g., HTML5, JavaScript, Typescript, CSS) **II:** Web platforms, frameworks, or meta-frameworks (a) Cloud services (b) API, Web Components **III:** Software as a Service (SaaS) **IV:** Web standards such as document object model, accessibility Accessibility and Inclusive Design **V:** Security and Privacy Considerations Computer Crimes and Attacks, Security Laws, Policies and Responsibility

Learning Outcomes I: Compare and contrast web programming with general-purpose programming [Compare] **II:** Describe the differences between Software-as-a-Service (SaaS) and traditional software products [Describe] **III:** Discuss how web standards impact software development [Debate] **IV:** Evaluate an existing web application against current web standards [Evaluate]

Bibliography: [Flanagan, 2020, Duckett, 2014, Fielding, 2000]

5.4 Mobile Platforms: Development Fundamentals, 6 hr(s)

Competences: ABET-2,ABET-4

Topics: **I:** Development with (a) Mobile programming languages (b) Mobile programming environments **II:** Mobile platform constraints (a) User interface design Understanding the User: Individual goals and interactions with others (b) Security **III:** Access (a) Accessing data through APIs Query Construction (b) Designing API endpoints for mobile apps: pitfalls and design considerations (c) Network and web interfaces Data Modeling, Fundamentals of Networks and Communications

Learning Outcomes **I:** Compare mobile programming with general-purpose programming [Compare] **II:** Evaluate the pros and cons of native and cross-platform mobile application development [Evaluate]

Bibliography: [Deitel et al., 2017, Hellman, 2013]

5.5 Mobile Platforms: Architecture and Constraints, 3 hr(s)

Competences: ABET-2,ABET-4

Topics: **I:** Development (a) Native versus cross-platform development (b) Software design/architecture patterns for mobile applications API Design and Modeling, Principles and Software Architecture, Software Design Quality and Evaluation **II:** Mobile platform constraints (a) Responsive user interface design Accessibility and Inclusive Design (b) Heterogeneity and mobility of devices (c) Differences in user experiences (e.g., between mobile and web-based applications) (d) Power and performance tradeoff **III:** Specification and testing Software Development Practices, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices

Learning Outcomes **I:** Develop a location-aware mobile application with data API integration [Create] **II:** Build a sensor-driven mobile application capable of logging data on a remote server [Create] **III:** Create a communication app incorporating telephony and instant messaging [Create]

Bibliography: [Deitel et al., 2017, Hellman, 2013]

5.6 Mobile Platforms: Capabilities and Asynchronous Computing, 3 hr(s)

Competences: ABET-2,ABET-4

Topics: **I:** Mobile computing affordances (a) Location-aware applications (b) Sensor-driven computing (e.g., gyroscope, accelerometer, health data from a watch) (c) Telephony and instant messaging (d) Augmented reality Immersion **II:** Asynchronous computing Algorithms (a) Difference from traditional synchronous programming (b) Handling success via callbacks (c) Handling errors asynchronously (d) Testing asynchronous code and typical problems in testing

Learning Outcomes **I:** Develop a location-aware mobile application with data API integration [Create] **II:** Build a sensor-driven mobile application capable of logging data on a remote server [Create] **III:** Create a communication app incorporating telephony and instant messaging [Create]

Bibliography: [Deitel et al., 2017, Hellman, 2013]

5.7 SPD-SEP/Interactive, 10 hr(s)

Competences: ABET-2,ABET-4

Topics: **I:** Ethical guidelines when using AI models to assist in journalism and content creation **II:** Accountability for AI-generated outputs **III:** Behavior among prompt programmers and AI developers **IV:** Trust with the public when using AI models

Learning Outcomes I: Indicate a framework for accountability in AI model deployment, including clear documentation and attribution [Evaluate] **II:** Discuss ethical codes of conduct and professional standards relevant to prompt programming and AI development [Debate] **III:** Create communication plans and materials to educate the public about AI capabilities, limitations, and ethical safeguards [Create]

Bibliography: [Duckett, 2014, Deitel et al., 2017]

5.8 Interactive Data and Analytics Platforms, 6 hr(s)

Competences: ABET-2,ABET-4

Topics: I: Computational Notebook Platforms (e.g., Jupyter notebooks; Google Colab; R; SPSS; Observable) **II:** Cloud SQL and Data Analysis Platforms (e.g., BigQuery) Query Construction (a) Apache Spark (b) Data Visualizations Visualization **III:** Interactive Presentations Backed by Data **IV:** Design Tools Requiring Low-Latency Feedback Loops (a) Rendering tools (b) Graphic design tools

Learning Outcomes I: Analyze large datasets interactively [Analyze] **II:** Create a backing track for a musical performance, such as live coding [Create] **III:** Create compelling computational notebooks that construct a narrative for a given journalistic goal/story [Create] **IV:** Implement interactive code that uses a dataset and generates exploratory graphics [Implement] **V:** Contextualize the attributes of different data analysis styles, such as interactive vs engineered pipeline [Contextualize] **VI:** Write a program using a notebook computing platform (e.g., searching, sorting, or graph manipulation) [Write]

Bibliography: [Flanagan, 2020]

6. Bibliography

[Deitel et al., 2017] Deitel, P., Deitel, H., and Deitel, A. (2017). *Android How to Program*. Pearson, 3rd edition.

[Duckett, 2014] Duckett, J. (2014). *JavaScript and JQuery: Interactive Front-End Web Development*. Wiley.

[Fielding, 2000] Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine.

[Flanagan, 2020] Flanagan, D. (2020). *JavaScript: The Definitive Guide*. O'Reilly Media, 7th edition.

[Gasca, 2021] Gasca, J. (2021). *Web Security for Developers*. No Starch Press.

[Hellman, 2013] Hellman, E. (2013). *Android Programming: Pushing the Limits*. Wiley.

[Deitel et al., 2017] Deitel, P., Deitel, H., and Deitel, A. (2017). *Android How to Program*. Pearson, 3rd edition.

[Duckett, 2014] Duckett, J. (2014). *JavaScript and JQuery: Interactive Front-End Web Development*. Wiley.

[Fielding, 2000] Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine.

[Flanagan, 2020] Flanagan, D. (2020). *JavaScript: The Definitive Guide*. O'Reilly Media, 7th edition.

[Gasca, 2021] Gasca, J. (2021). *Web Security for Developers*. No Starch Press.

[Hellman, 2013] Hellman, E. (2013). *Android Programming: Pushing the Limits*. Wiley.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA112 Calculus II (Mandatory)

1. General information

- **Semester**: 3rd Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 2 hours;
- **Prerrequisites**:
  - MA111 Calculus I (2nd Sem-Pg. 338)

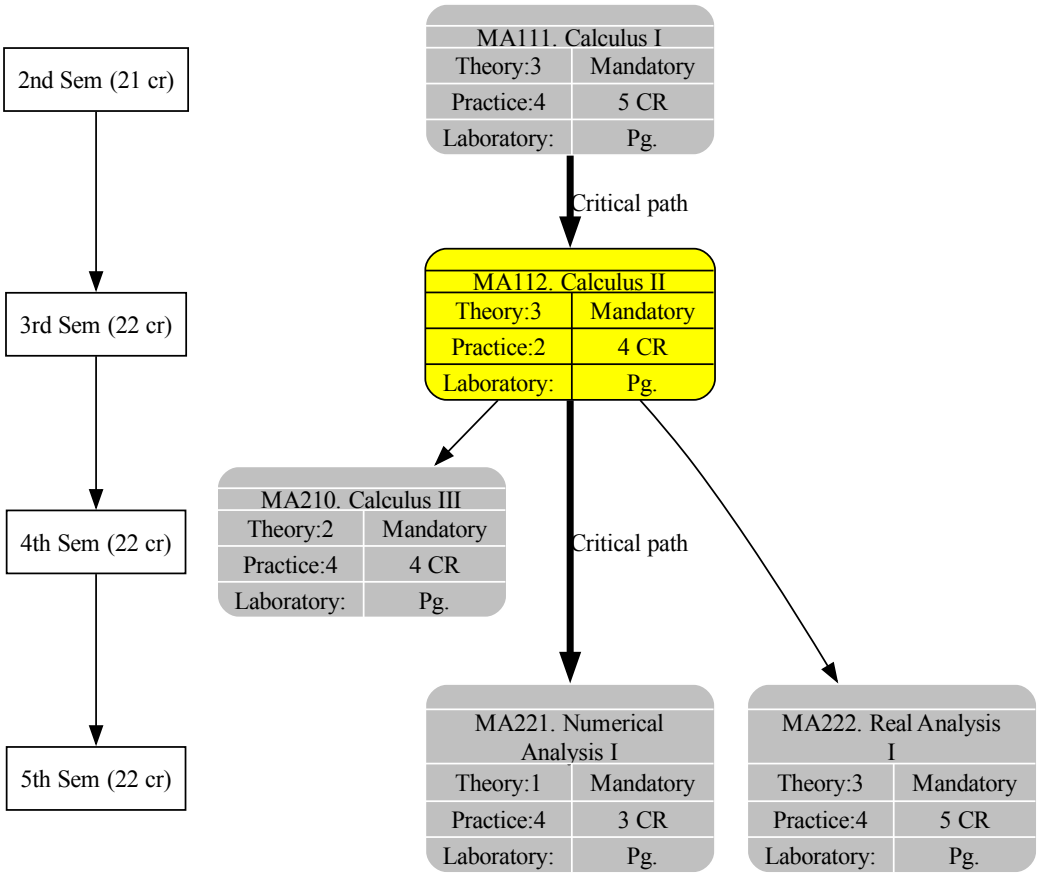


Figure 7.16: Connection Map. MA112 Calculus II

2. Course Rationale

The course includes knowledge of Integral Calculus: the Indefinite Integral (antiderivative). Integration Techniques. Definite Integral. Integration of hyperbolic and inverse functions. Applications of the Definite Integral (Areas of plane regions, Volumes, Length, Surface Area). Improper Integral. Applications in problems involving living beings.

3. Goals of the Course

- Understand the definite integral as a limit of Riemann sums.
- Master various integration techniques (substitution, parts, partial fractions).

- Apply the Fundamental Theorem of Calculus to solve complex problems.
- Use integrals to calculate areas between curves and volumes of solids of revolution.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Numerical Differentiation and Integration, 25 hr(s)

Competences: ABET-1

Topics: **I:** Finite difference approximations (forward, backward, central) **II:** Richardson extrapolation **III:** Newton-Cotes formulas (Trapezoidal rule, Simpson's rule) **IV:** Gaussian quadrature **V:** Error analysis for numerical integration

Learning Outcomes **I:** Compute derivatives of functions using finite difference schemes [Usage] **II:** Apply Richardson extrapolation to improve the accuracy of numerical approximations [Usage] **III:** Implement numerical integration rules (Trapezoidal, Simpson, Gaussian) [Usage] **IV:** Compare the accuracy and efficiency of different numerical integration methods [Assessment] **V:** Analyze the error for numerical differentiation and integration formulas [Assessment]

Bibliography: [Stewart, 2016, Thomas et al., 2014]

5.2 Interpolation and Function Approximation, 25 hr(s)

Competences: ABET-1

Topics: **I:** Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Stewart, 2016, Thomas et al., 2014]

6. Bibliography

[Stewart, 2016] Stewart, J. (2016). *Cálculo de una variable: Trascendentes tempranas*. Cengage Learning, 8th edition.

[Thomas et al., 2014] Thomas, G. B., Weir, M. D., and Hass, J. (2014). *Cálculo de una variable*. Pearson, 13th edition.

[Stewart, 2016] Stewart, J. (2016). *Cálculo de una variable: Trascendentes tempranas*. Cengage Learning, 8th edition.

[Thomas et al., 2014] Thomas, G. B., Weir, M. D., and Hass, J. (2014). *Cálculo de una variable*. Pearson, 13th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA121 Linear Algebra I (Mandatory)

1. General information

- **Semester**: 3rd Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA100 Basic Mathematics (1st Sem-Pg. 300)

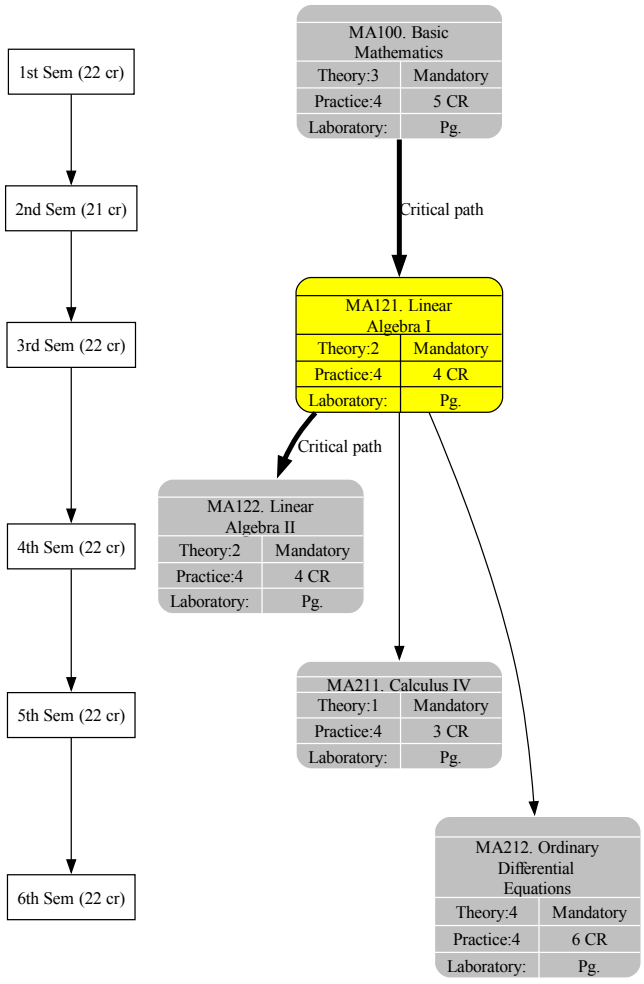


Figure 7.17: Connection Map. MA121 Linear Algebra I

2. Course Rationale

This course introduces the fundamental concepts of linear algebra, providing the mathematical foundations for the study of vector spaces, linear transformations, and systems of linear equations. It develops abstract thinking and problem-solving skills through matrices, determinants, and vectors. The course is fundamental for applications in science, engineering, and computing.

3. Goals of the Course

- Understand the concepts of vector spaces and subspaces

- Master matrix operations and determinants
- Solve systems of linear equations using different methods
- Apply linear transformations and compute eigenvalues and eigenvectors

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Numerical Linear Algebra, 25 hr(s)

Competences: ABET-1

Topics: **I:** LU decomposition and its applications **II:** QR decomposition and least squares problems **III:** Singular Value Decomposition (SVD) and its applications **IV:** Condition number and numerical stability of linear systems **V:** Iterative methods for linear systems (Jacobi, Gauss-Seidel, Conjugate Gradient)

Learning Outcomes **I:** Implement LU and QR decomposition algorithms for solving linear systems [Usage] **II:** Explain the geometric and algebraic meaning of the Singular Value Decomposition [Familiarity] **III:** Use SVD for applications such as matrix approximation and principal component analysis [Usage] **IV:** Analyze the stability of a linear system using its condition number [Assessment] **V:** Select and implement an appropriate iterative method for solving large, sparse linear systems [Assessment]

Bibliography: [Lay et al., 2016, Strang, 2016]

5.2 Computational Arithmetic and Error Analysis, 25 hr(s)

Competences: ABET-1

Topics: **I:** Floating-point arithmetic and the IEEE standard **II:** Sources of error: truncation, rounding, and discretization **III:** Error propagation and condition number **IV:** Stability of numerical algorithms **V:** Absolute, relative, and forward/backward error

Learning Outcomes **I:** Define the different types of errors in numerical computation (rounding, truncation, discretization) [Familiarity] **II:** Explain the principles of floating-point representation and arithmetic according to the IEEE standard [Familiarity] **III:** Calculate the absolute and relative error of an approximation [Usage] **IV:** Analyze the propagation of errors in a sequence of arithmetic operations [Usage] **V:** Distinguish between the condition of a problem and the stability of an algorithm [Assessment] **VI:** Compute the condition number of a simple problem (e.g., matrix-vector multiplication) [Usage]

Bibliography: [Lay et al., 2016, Strang, 2016]

5.3 Roots Finding and Basic Numerical Optimization, 25 hr(s)

Competences: ABET-1

Topics: **I:** Bisection method and fixed-point iteration **II:** Newton's method and secant method for root finding **III:** Convergence analysis and order of convergence **IV:** Gradient descent for univariate optimization **V:** Rate of convergence and stopping criteria

Learning Outcomes **I:** Implement root-finding algorithms (bisection, Newton, secant) and analyze their convergence [Usage] **II:** Classify the order of convergence of iterative methods [Assessment] **III:** Apply gradient descent to solve simple univariate optimization problems [Usage] **IV:** Design appropriate stopping criteria for iterative methods [Usage]

Bibliography: [Lay et al., 2016, Strang, 2016]

5.4 Interpolation and Function Approximation, 25 hr(s)

Competences: ABET-1

Topics: **I:** Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Lay et al., 2016, Anton and Rorres, 2014]

6. Bibliography

[Anton and Rorres, 2014] Anton, H. and Rorres, C. (2014). *Álgebra Lineal con Aplicaciones*. Limusa Wiley, 11th edition.

[Lay et al., 2016] Lay, D. C., Lay, S. R., and McDonald, J. J. (2016). *Álgebra Lineal y sus Aplicaciones*. Pearson, 5th edition.

[Strang, 2016] Strang, G. (2016). *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 5th edition.

[Anton and Rorres, 2014] Anton, H. and Rorres, C. (2014). *Álgebra Lineal con Aplicaciones*. Limusa Wiley, 11th edition.

[Lay et al., 2016] Lay, D. C., Lay, S. R., and McDonald, J. J. (2016). *Álgebra Lineal y sus Aplicaciones*. Pearson, 5th edition.

[Strang, 2016] Strang, G. (2016). *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 5th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

ID101 Technical and professional English I (Mandatory)

1. General information

- **Semester**: 3rd Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 4 hours;
- **Prerrequisites**: None

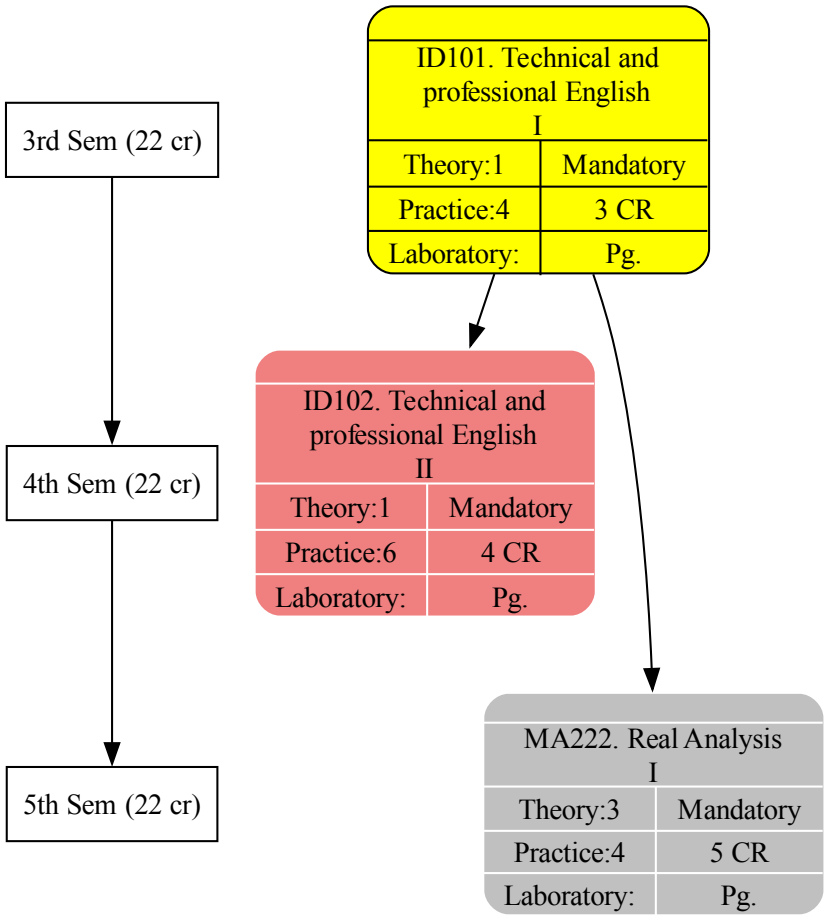


Figure 7.18: Connection Map. ID101 Technical and professional English I

2. Course Rationale

A fundamental part of the integral formation of a professional is the ability to communicate in a foreign language in addition to the native language itself. It not only broadens its cultural horizon but also allows a more humane and comprehensive view of people's lives. In the case of foreign languages, English is undoubtedly the most pratical because it is spoken around the world. There is no country where it is not spoken. In careers related to tourist services English is perhaps the most important practical tool that the student must master from the outset as part of his comprehensive education.

3. Goals of the Course

- Know the English language and its grammatical structure.
- Identify situations and employ dialogues related to them.

4. General Competences (*Outcomes*)

ABET-4) An ability to communicate effectively with a range of audiences. (Usage)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Familiarity)

5. Content

5.1 Hello everybody!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Verb To Be. **II:** Affirmative sentences , Negatives and Questions. **III:** Number Expressions. **IV:** Objects and Countries. **V:** Expressions to greet and make presentations.

Learning Outcomes **I:** At the end of the first unit, each student, understanding the grammar of the present tense is able to express a greater quantity of expressions of time and also to use sentences with the verb To Be to express situation and state. **II:** That the student is able to analyze and express ideas about dates and numbers in order.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.2 Meeting people!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Possessive adjectives. **II:** Expressions to find prices. **III:** Possession expressions **IV:** Vocabulary of Family, Food and Drinks. **V:** Formal requests. **VI:** Informal letters.

Learning Outcomes **I:** At the end of the second unit, students having identified how to express orders and make offerings in restaurants use them in various situations. Explain and apply food and drink vocabulary.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.3 The world of work, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Simple present tense. Auxiliaries. **II:** Affirmative sentences, Negatives and Questions. **III:** Common verbs and occupations. **IV:** Indications for expressing the time.

Learning Outcomes **I:** At the end of the third unit, students having recognized the characteristics of the present simple, use it to make descriptions of various types. Describe people and places and give directions. Express time.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.4 Take it easy!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Present Simple 2. **II:** Affirmative sentences , Negatives and Questions. **III:** Use of Verbs of entertainment. **IV:** Free time. **V:** The seasons of the year. **VI:** Expressions of social activities.

Learning Outcomes **I:** At the conclusion of the fourth unit, the students having identified the idea of expressing ideas of free time actions in Simple and Continuous Present. Express ideas of stations and related activities.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.5 Where do you live?, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Use There is/There are. **II:** Sentences with Prepositions. **III:** Expressions of Quantity. **IV:** Vocabulary of airplanes and places. **V:** Expressions of direction indications.

Learning Outcomes **I:** At the end of the fifth unit, students, based on the understanding of the present continuous time, will elaborate sentences using ideas of location and place. They will also assimilate the need to express objects in common use. They will acquire vocabulary to describe the parts of a house using expressions to ask for directions.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.6 Can you speak English?, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Can/cant. **II:** Past of verb to be. Use of Could **III:** Vocabulary of Countries and languages. **IV:** Expressions for using the phone **V:** Writing formal letters. **VI:** Readings.

Learning Outcomes **I:** At the end of the sixth unit, students having learned the fundamentals of using auxiliary mode, will create sentences applied to the appropriate context. They emphasize the difference between languages and nationalities. They describe feelings. Use expressions on the phone.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.7 Then and now!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Past Simple. **II:** Expressions of past tense. **III:** Vocabulary regular and irregular verbs **IV:** Expressions to describe the climate. **V:** Writing descriptive paragraphs. **VI:** Special occasions.

Learning Outcomes **I:** At the end of the seventh unit, students having learned the basics of structuring the Simple Past experience the need to be able to express this type of time in actions. They will practice in appropriate contexts. They emphasize the difference between irregular and regular verbs. They describe actions with several verbs. They use expressions to describe the climate.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

6. Bibliography

[Cambridge, 2006] Cambridge (2006). *Diccionario Inglés-Espanol Cambridge*. Editorial Oxford.

[MacGrew, 1999] MacGrew, J. (1999). *Focus on Grammar Basic*. Editorial Oxford.

[Soars and John, 2002d] Soars, L. and John (2002d). *American Headway N 2 Student Book*. Editorial Oxford.

[Cambridge, 2006] Cambridge (2006). *Diccionario Inglés-Espanol Cambridge*. Editorial Oxford.

[MacGrew, 1999] MacGrew, J. (1999). *Focus on Grammar Basic*. Editorial Oxford.

[Soars and John, 2002d] Soars, L. and John (2002d). *American Headway N 2 Student Book*. Editorial Oxford.

Fourth Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS1D2 Discrete Structures II (Mandatory)

1. General information

- **Semester**: 4th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - CS1D1 Discrete Structures I (3rd Sem-Pg. 349)

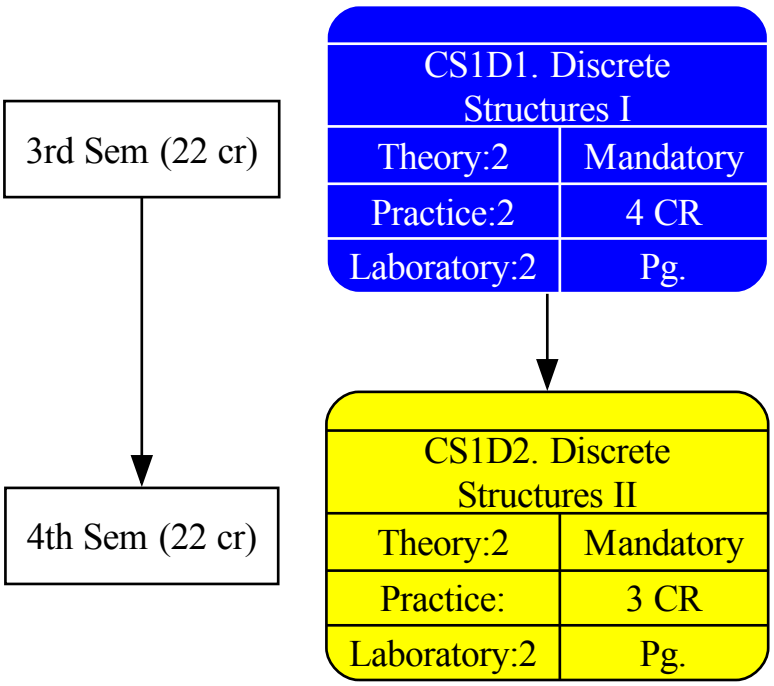


Figure 7.19: Connection Map. CS1D2 Discrete Structures II

2. Course Rationale

Continuing from Discrete Structures I, this course explores more complex mathematical structures such as sets, relations, and functions, which are the basis for data modeling in databases and programming languages. Additionally, students will study counting techniques and recurrence relations, providing the analytical tools required to estimate the computational complexity of algorithms.

3. Goals of the Course

- Perform complex operations on sets, functions, and relations.
- Model database relations and function properties formally.
- Apply counting techniques to analyze search spaces and probability.
- Solve recurrence relations to analyze the efficiency of recursive algorithms.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Familiarity)

5. Content

5.1 Set Theory, Relations and Functions, 24 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Sets: operations, cardinality, power sets, Cartesian products **II:** Relations: properties, equivalence relations, partial orders **III:** Functions: injective, surjective, bijective, composition, inverse

Learning Outcomes **I:** Define basic set operations and relations [Assessment] **II:** Apply set operations to solve problems [Usage] **III:** Define properties of relations [Assessment] **IV:** Determine if a relation is an equivalence relation [Usage] **V:** Define types of functions [Assessment] **VI:** Compose functions and find inverses [Usage]

Bibliography: [Rosen, 2019, Scheinerman, 2012]

5.2 Counting and Recurrence, 24 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Counting: permutations, combinations, binomial theorem **II:** Advanced counting: generating functions, inclusion-exclusion

III: Recurrence relations: solving simple recurrences **IV:** Advanced recurrence: generating functions, asymptotic behavior

Learning Outcomes **I:** Explain permutations and combinations [Assessment] **II:** Apply counting techniques to solve problems [Usage] **III:** Define recurrence relations [Assessment] **IV:** Solve simple recurrence relations [Usage]

V: Develop generating functions for counting problems [Usage] **VI:** Apply generating functions to solve complex recurrence relations [Assessment]

Bibliography: [Rosen, 2019, Grimaldi, 2003]

6. Bibliography

[Grimaldi, 2003] Grimaldi, R. P. (2003). *Discrete and Combinatorial Mathematics: An Applied Introduction*. Pearson, 5th edition.

[Rosen, 2019] Rosen, K. H. (2019). *Discrete Mathematics and Its Applications*. McGraw-Hill Education, 8th edition.

[Scheinerman, 2012] Scheinerman, E. R. (2012). *Mathematics: A Discrete Introduction*. Cengage Learning, 3rd edition.

[Grimaldi, 2003] Grimaldi, R. P. (2003). *Discrete and Combinatorial Mathematics: An Applied Introduction*. Pearson, 5th edition.

[Rosen, 2019] Rosen, K. H. (2019). *Discrete Mathematics and Its Applications*. McGraw-Hill Education, 8th edition.

[Scheinerman, 2012] Scheinerman, E. R. (2012). *Mathematics: A Discrete Introduction*. Cengage Learning, 3rd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS210 Algorithms and Data Structures (Mandatory)

1. General information

- **Semester**: 4th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - CS113 Objects-oriented programming II (3rd Sem-Pg. 343)

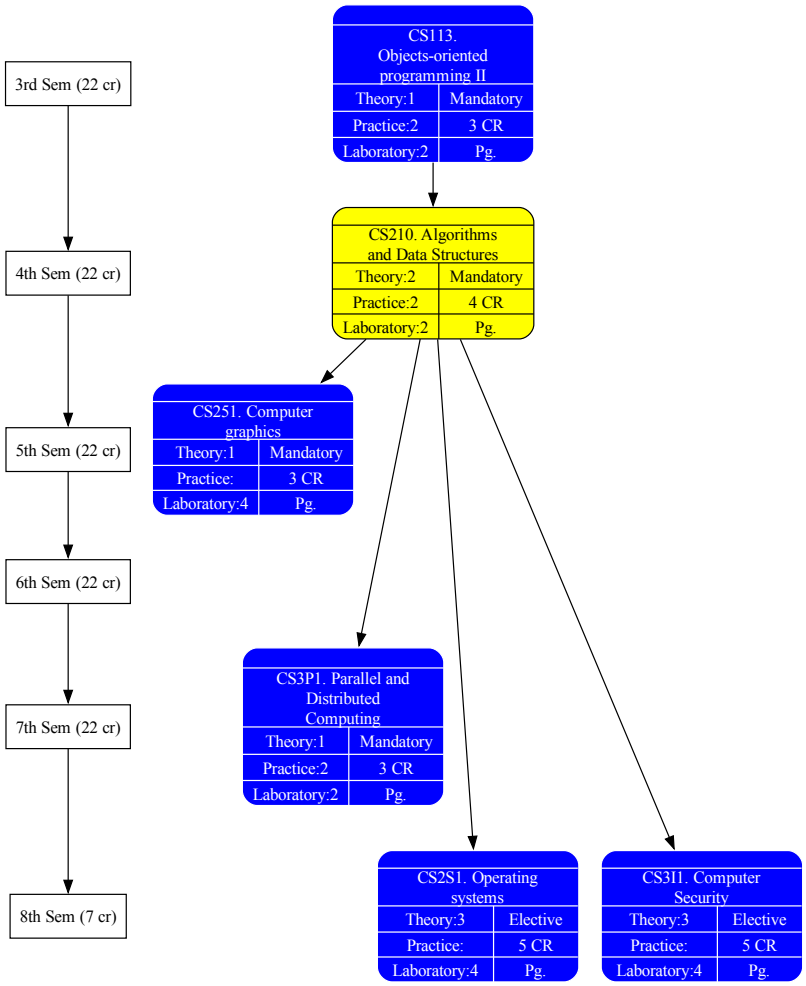


Figure 7.20: Connection Map. CS210 Algorithms and Data Structures

2. Course Rationale

This course provides a comprehensive study of fundamental data structures and algorithms with emphasis on implementation using advanced C++ features. Building on prior C++ knowledge, students will implement generic data structures using templates and traits, ensuring type flexibility and performance. The course covers linear structures (vectors, various linked lists), tree structures (binary, AVL, B-trees), heaps, hash tables, tries, and concurrent data structures. Special focus is placed on modern C++ techniques including variadic templates for generic operations and concurrent programming for thread-safe data structures.

3. Goals of the Course

- Implement generic data structures using templates and traits.
- Master advanced tree structures and balanced trees.
- Understand and implement concurrent data structures.
- Apply variadic templates for generic operations on containers.
- Analyze algorithmic complexity of all implemented structures.
- Design efficient data structures for specific problem constraints.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Advanced Foundation for Data Structures, 8 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Template metaprogramming review and advanced traits. **II:** Type traits and SFINAE for conditional compilation. **III:** Variadic templates and parameter packs. **IV:** Perfect forwarding and universal references. **V:** Lambda expressions with generic captures. **VI:** Policy-based design and CRTP (Curiously Recurring Template Pattern). **VII:** Concept-based constraints (C++20 concepts if applicable).

Learning Outcomes **I:** Design type traits for customizing data structure behavior [Usage]. **II:** Implement variadic templates for generic container operations [Usage]. **III:** Apply policy-based design to create configurable data structures [Assessment]. **IV:** Use perfect forwarding to implement efficient constructors [Usage].

Bibliography: [Stroustrup, 2013, Vandevoorde et al., 2017]

5.2 Foundational Data Structures, 8 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Abstract Data Type (ADT) and operations on an ADT: (a) Dictionary operations (insert, delete, find) **II:** Arrays: (a) Numeric vs non-numeric, character strings (b) Single (vector) vs multidimensional (matrix) **III:** Records/Structs/Tuples and Objects **IV:** Linked lists (for historical reasons): (a) Single vs Double and Linear vs Circular **V:** Stacks **VI:** Queues and deques: (a) Heap-based priority queue **VII:** Hash tables/maps: (a) Collision resolution and complexity (e.g., probing, chaining, rehash) **VIII:** Trees: (a) Binary, n-ary, and search trees (b) Balanced (e.g., AVL, Red-Black, Heap) **IX:** Sets

Learning Outcomes **I:** For each ADT/Data-Structure in this unit: (a) Explain its definition, properties, representation(s), and associated ADT operations. (b) Explain step-by-step how the ADT operations associated with the data structure transform it. [Usage] **II:** For each algorithm in this unit explain step-by-step how the algorithm operates [Usage] **III:** Implement vector with customizable allocation policy using templates [Usage]. **IV:** Design generic linked list nodes with trait-based memory management [Usage]. **V:** Compare time/space complexity of different linear structures [Assessment]. **VI:** Apply variadic templates to initialize containers with multiple elements [Usage].

Bibliography: [Cormen et al., 2009, Stroustrup, 2013]

5.3 Foundational Data Structures, 8 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Trees: (a) Binary, n-ary, and search trees (b) Balanced (e.g., AVL, Red-Black, Heap) **II:** Binary tree template implementation with iterator support. **III:** Binary search tree with trait-based comparison operators. **IV:** Tree traversal algorithms (iterative and recursive). **V:** Expression trees and their applications. **VI:** Template-based node structures with configurable payload.

Learning Outcomes **I:** Given requirements for a problem, develop multiple solutions using various data structures and algorithms. Subsequently, evaluate the suitability, strengths, and weaknesses selecting an approach that best satisfies the requirements [Assessment] **II:** Explain factors beyond computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data [Familiarity] **III:** Implement generic binary tree with configurable traversal order [Usage]. **IV:** Design expression tree evaluator using visitor pattern [Usage]. **V:** Analyze tree operation complexities for different balancing strategies [Assessment].

Bibliography: [Cormen et al., 2009, Knuth, 1997]

5.4 Balanced and Multi-way Trees, 8 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** AVL tree implementation with template-based balancing trait. **II:** Red-black tree properties and operations. **III:** B-Tree template implementation for disk-based storage. **IV:** 2-3 Tree and 2-3-4 Tree structures. **V:** Digital trie (prefix tree) implementation. **VI:** Suffix trees and their applications in string processing. **VII:** Node splitting/merging strategies for B-trees.

Learning Outcomes **I:** Implement self-balancing trees using policy-based design [Usage]. **II:** Design B-tree for database index applications [Usage]. **III:** Evaluate trade-offs between different balanced tree structures [Assessment]. **IV:** Implement trie for efficient string prefix matching [Usage].

Bibliography: [Cormen et al., 2009, Knuth, 1997, Sedgewick and Wayne, 2011]

5.5 Foundational Data Structures, 8 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Trees: (a) Binary, n-ary, and search trees (b) Balanced (e.g., AVL, Red-Black, Heap) **II:** Binary heap implementation using array representation. **III:** Min-heap and max-heap variants. **IV:** Heapify algorithms and complexity analysis. **V:** Priority queue ADT implementation. **VI:** Binomial heap and Fibonacci heap overview. **VII:** Heap applications: Dijkstra's algorithm, heap sort.

Learning Outcomes **I:** Explain the heap property and the use of heaps as an implementation of a priority queue [Familiarity] **II:** Implement generic heap with configurable comparison trait [Usage]. **III:** Design priority queue supporting multiple priority update strategies [Usage]. **IV:** Analyze heap operations complexity for different heap types [Assessment]. **V:** Apply heap in graph algorithm implementations [Usage].

Bibliography: [Cormen et al., 2009, Sedgewick and Wayne, 2011]

5.6 Foundational Data Structures, 8 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Hash tables/maps: (a) Collision resolution and complexity (e.g., probing, chaining, rehash) **II:** Hash function design and properties. **III:** Separate chaining implementation with linked lists. **IV:** Open addressing: linear probing, quadratic probing, double hashing. **V:** Perfect hashing and minimal perfect hashing. **VI:** Cuckoo hashing and hopscotch hashing. **VII:** Hash table resizing strategies and load factor management.

Learning Outcomes **I:** Explain how collision avoidance and collision resolution is handled in hash tables [Familiarity] **II:** Design generic hash table with configurable collision resolution [Usage]. **III:** Implement custom hash functions for user-defined types [Usage]. **IV:** Evaluate performance of different hashing schemes [Assessment]. **V:** Apply hash tables in compiler symbol table implementation [Usage].

Bibliography: [Cormen et al., 2009, Knuth, 1997]

5.7 Concurrent Data Structures, 8 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Thread-safe data structure design principles. **II:** Lock-based concurrent linked lists. **III:** Lock-free and wait-free algorithms. **IV:** Concurrent hash tables with fine-grained locking. **V:** Atomic operations and memory ordering. **VI:** Concurrent queues (bounded and unbounded). **VII:** Concurrent stacks and elimination techniques.

Learning Outcomes **I:** Design thread-safe data structures using mutex and lock guards [Usage]. **II:** Implement lock-free linked list using atomic operations [Usage]. **III:** Analyze performance of concurrent vs sequential data structures [Assessment]. **IV:** Apply concurrent queues in producer-consumer patterns [Usage].

Bibliography: [Herlihy and Shavit, 2012, Williams, 2019]

5.8 Specialized Structures and Applications, 8 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Persistent data structures. **II:** Disjoint-set union (union-find) with path compression. **III:** Bloom filters and their probabilistic properties. **IV:** Skip lists for probabilistic balanced search. **V:** Segment trees and Fenwick trees for range queries. **VI:** Spatial data structures: k-d trees, quadtrees. **VII:** Cache-oblivious data structures.

Learning Outcomes **I:** Implement persistent data structures with structural sharing [Usage]. **II:** Design union-find with both union by rank and path compression [Usage]. **III:** Apply specialized structures to solve domain-specific problems [Assessment]. **IV:** Evaluate space-time tradeoffs in probabilistic data structures [Assessment].

Bibliography: [Okasaki, 1999, Sedgewick and Wayne, 2011]

6. Bibliography

[Cormen et al., 2009] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press, 3rd edition.

[Herlihy and Shavit, 2012] Herlihy, M. and Shavit, N. (2012). *The Art of Multiprocessor Programming*. Morgan Kaufmann, revised 1st edition.

[Knuth, 1997] Knuth, D. E. (1997). *The Art of Computer Programming, Volume 1: Fundamental Algorithms*. Addison-Wesley, 3rd edition.

[Okasaki, 1999] Okasaki, C. (1999). *Purely Functional Data Structures*. Cambridge University Press.

[Sedgewick and Wayne, 2011] Sedgewick, R. and Wayne, K. (2011). *Algorithms*. Addison-Wesley, 4th edition.

[Stroustrup, 2013] Stroustrup, B. (2013). *The C++ Programming Language*. Addison-Wesley, 4th edition.

[Vandevoorde et al., 2017] Vandevoorde, D., Josuttis, N. M., and Gregor, D. (2017). *C++ Templates: The Complete Guide*. Addison-Wesley, 2nd edition.

[Williams, 2019] Williams, A. (2019). *C++ Concurrency in Action*. Manning Publications, 2nd edition.

[Cormen et al., 2009] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press, 3rd edition.

[Herlihy and Shavit, 2012] Herlihy, M. and Shavit, N. (2012). *The Art of Multiprocessor Programming*. Morgan Kaufmann, revised 1st edition.

[Knuth, 1997] Knuth, D. E. (1997). *The Art of Computer Programming, Volume 1: Fundamental Algorithms*. Addison-Wesley, 3rd edition.

[Okasaki, 1999] Okasaki, C. (1999). *Purely Functional Data Structures*. Cambridge University Press.

[Sedgewick and Wayne, 2011] Sedgewick, R. and Wayne, K. (2011). *Algorithms*. Addison-Wesley, 4th edition.

[Stroustrup, 2013] Stroustrup, B. (2013). *The C++ Programming Language*. Addison-Wesley, 4th edition.

[Vandevoorde et al., 2017] Vandevoorde, D., Josuttis, N. M., and Gregor, D. (2017). *C++ Templates: The Complete Guide*. Addison-Wesley, 2nd edition.

[Williams, 2019] Williams, A. (2019). *C++ Concurrency in Action*. Manning Publications, 2nd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS271 Data Management (Mandatory)

1. General information

- **Semester**: 4th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS112 Objects-oriented programming I (2nd Sem-Pg. 328)
  - CS1D1 Discrete Structures I (3rd Sem-Pg. 349)

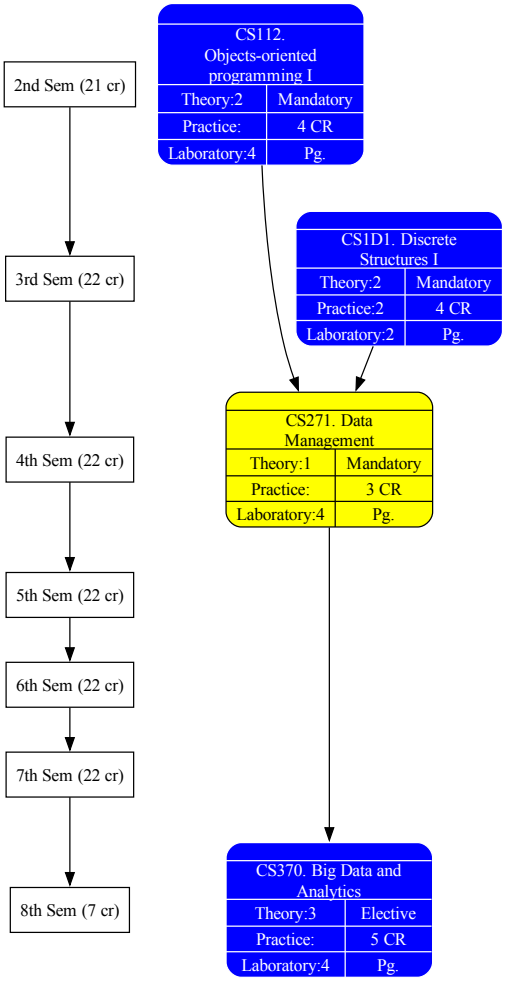


Figure 7.21: Connection Map. CS271 Data Management

2. Course Rationale

This course introduces fundamental concepts and techniques in data management, focusing on relational database systems. Students will learn data modeling using entity-relationship diagrams, relational database design principles, normalization theory, and Structured Query Language (SQL). The course covers core database system components, transaction management, and basic query processing. Emphasis is placed on practical skills for designing, implementing, and querying databases,

as well as understanding the role of data in organizations and the associated ethical considerations in data management.

3. Goals of the Course

- Design and implement normalized relational databases.
- Formulate complex queries using SQL for data retrieval and manipulation.
- Apply data modeling techniques using entity-relationship diagrams.
- Understand database system architecture and transaction management.
- Recognize ethical and security considerations in data management.
- Utilize basic database administration tools and techniques.

4. General Competences (*Outcomes*)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Core Database System Concepts, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Purpose and advantages of database systems **II:** Components of database systems **III:** Design of core DBMS functions (e.g., query mechanisms, transaction management, buffer management, access methods) **IV:** Database architecture, data independence, and data abstraction **V:** Transaction management **VI:** Normalization **VII:** Approaches for managing large volumes of data (e.g., NoSQL database systems, use of MapReduce) **VIII:** How to support CRUD-only applications **IX:** Distributed databases/cloud-based systems **X:** Structured, semi-structured, and unstructured data **XI:** Use of a declarative query language **XII:** Systems supporting structured and/or stream content

Learning Outcomes **I:** Identify at least four advantages that using a database system provides [Familiarity] **II:** Enumerate the components of a (relational) database system [Familiarity] **III:** Explain the advantages of database systems over file systems [Usage]. **IV:** Describe the three-schema architecture and data independence [Familiarity]. **V:** Identify different types of database users and their roles [Familiarity].

Bibliography: [Ramakrishnan and Gehrke, 2002, Silberschatz et al., 2019]

5.2 Data Modeling, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Conceptual models (e.g., entity-relationship, UML diagrams) **II:** Entity sets, relationship sets, and attributes **III:** Cardinality constraints: one-to-one, one-to-many, many-to-many **IV:** Weak entity sets and identifying relationships **V:** Extended E-R features: specialization, generalization, aggregation **VI:** UML class diagrams for data modeling

Learning Outcomes **I:** Describe the components of the E-R (or some other non-relational) data model [Familiarity] **II:** Model a given environment using a conceptual data model [Usage] **III:** Design E-R diagrams for real-world scenarios [Usage]. **IV:** Convert E-R diagrams to relational schemas [Usage]. **V:** Apply extended E-R features for complex modeling [Assessment].

Bibliography: [Ramakrishnan and Gehrke, 2002, Elmasri and Navathe, 2015]

5.3 Relational Databases, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Entity and referential integrity: Candidate key, superkeys **II:** Relational database design **III:** Relational algebra operations: select, project, join, union, etc. **IV:** Integrity constraints: domain, key, entity, referential **V:** Relational calculus: tuple and domain calculus **VI:** Mapping E-R diagrams to relational schemas

Learning Outcomes **I:** Describe the defining characteristics behind the relational data model [Familiarity] **II:** Comment on the difference between a foreign key and a superkey [Usage] **III:** Express queries using relational algebra [Usage]. **IV:** Design relational schemas with appropriate constraints [Usage]. **V:** Map complex E-R diagrams to normalized relations [Assessment].

Bibliography: [Ramakrishnan and Gehrke, 2002, Elmasri and Navathe, 2015]

5.4 Query Construction, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** SQL Query Formation: (a) Interactive SQL execution (b) Programmatic execution of an SQL query **II:** Data definition language: CREATE, ALTER, DROP **III:** Data types and domain constraints creation and management **IV:** View definition and materialized views **V:** Database schema design and implementation

Learning Outcomes **I:** Compose SQL queries that incorporate select, project, join, union, intersection, set difference, and set division [Usage] **II:** Define, in SQL, a relation schema, including all integrity constraints and delete/update triggers [Usage] **III:** Create and modify database schemas using SQL DDL [Usage]. **IV:** Implement integrity constraints at schema level [Usage]. **V:** Design and create appropriate indexes for performance [Assessment].

Bibliography: [Ramakrishnan and Gehrke, 2002, Beaulieu, 2009]

5.5 Query Construction, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Complex SQL queries: nested subqueries, correlated subqueries **II:** Aggregation functions and GROUP BY operations **III:** Set operations: UNION, INTERSECT, EXCEPT **IV:** JOIN operations: inner, outer, natural, cross joins **V:** Data manipulation: INSERT, UPDATE, DELETE **VI:** Transaction control: COMMIT, ROLLBACK, SAVEPOINT

Learning Outcomes **I:** Determine when a nested SQL query is correlated or not [Usage] **II:** Compose an SQL query to update a tuple in a relation [Usage] **III:** Formulate complex queries with multiple joins and subqueries [Usage]. **IV:** Apply aggregation for data analysis and reporting [Usage]. **V:** Implement data modification operations with transaction control [Assessment].

Bibliography: [Ramakrishnan and Gehrke, 2002, Beaulieu, 2009]

5.6 Relational Databases, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Introduction to Functional dependency theory **II:** Normalization Theory: (a) Decomposition of a schema; lossless-join, and dependency-preservation properties of a decomposition (b) Normal forms (BCNF) (c) Denormalization (for efficiency) **III:** Functional dependencies and inference rules **IV:** Normal forms: 1NF, 2NF, 3NF, BCNF **V:** Decomposition algorithms: lossless-join and dependency-preserving **VI:** Practical considerations in normalization

Learning Outcomes **I:** Describe how functional dependency theory generalizes the notion of key [Familiarity] **II:** Detect which normal form a given decomposition yields [Usage] **III:** Identify functional dependencies in relational schemas [Usage]. **IV:** Normalize relations to BCNF [Assessment]. **V:** Evaluate decompositions for lossless-join and dependency preservation [Assessment].

Bibliography: [Ramakrishnan and Gehrke, 2002, Elmasri and Navathe, 2015]

5.7 DBMS Internals, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Transaction Management: (a) Isolation Levels (b) ACID (c) Serializability (d) Distributed Transactions **II:** Concurrency Control: (a) 2-Phase Locking (b) Deadlocks handling strategies (c) Quorum-based consistency models **III:** ACID properties: Atomicity, Consistency, Isolation, Durability **IV:** Serializability and conflict serializability **V:** Lock-based concurrency control: two-phase locking **VI:** Deadlock detection and prevention **VII:** Isolation levels and their trade-offs

Learning Outcomes **I:** Describe the four properties for a correct transaction manager [Familiarity] **II:** Outline the principle of serializability [Familiarity] **III:** Explain ACID properties and their importance [Usage]. **IV:** Analyze transaction schedules for serializability [Assessment]. **V:** Apply locking protocols for concurrency control [Usage].

Bibliography: [Ramakrishnan and Gehrke, 2002, Bernstein et al., 1987]

5.8 Data Security and Privacy, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Differences between data security and data privacy **II:** Personally identifying information (PII) and its protection **III:** Ethical considerations in ensuring the security and privacy of data **IV:** Database security: authentication, authorization, encryption **V:** SQL injection attacks and prevention **VI:** Introduction to NoSQL databases and big data **VII:** Data governance and compliance

Learning Outcomes **I:** Describe the differences in the goals for data security and data privacy [Familiarity] **II:** Identify and mitigate risks associated with different approaches to protecting data [Usage] **III:** Implement basic database security measures [Usage]. **IV:** Prevent SQL injection vulnerabilities [Assessment]. **V:** Discuss ethical considerations in data collection and use [Assessment].

Bibliography: [Ramakrishnan and Gehrke, 2002, Elmasri and Navathe, 2015]

6. Bibliography

[Beaulieu, 2009] Beaulieu, A. (2009). *Learning SQL*. O'Reilly Media, 2nd edition.

[Bernstein et al., 1987] Bernstein, P. A., Hadzilacos, V., and Goodman, N. (1987). *Concurrency Control and Recovery in Database Systems*. Addison-Wesley.

[Elmasri and Navathe, 2015] Elmasri, R. and Navathe, S. B. (2015). *Fundamentals of Database Systems*. Pearson, 7th edition.

[Ramakrishnan and Gehrke, 2002] Ramakrishnan, R. and Gehrke, J. (2002). *Database Management Systems*. McGraw-Hill, 3rd edition.

[Silberschatz et al., 2019] Silberschatz, A., Korth, H. F., and Sudarshan, S. (2019). *Database System Concepts*. McGraw-Hill, 7th edition.

[Beaulieu, 2009] Beaulieu, A. (2009). *Learning SQL*. O'Reilly Media, 2nd edition.

[Bernstein et al., 1987] Bernstein, P. A., Hadzilacos, V., and Goodman, N. (1987). *Concurrency Control and Recovery in Database Systems*. Addison-Wesley.

[Elmasri and Navathe, 2015] Elmasri, R. and Navathe, S. B. (2015). *Fundamentals of Database Systems*. Pearson, 7th edition.

[Ramakrishnan and Gehrke, 2002] Ramakrishnan, R. and Gehrke, J. (2002). *Database Management Systems*. McGraw-Hill, 3rd edition.

[Silberschatz et al., 2019] Silberschatz, A., Korth, H. F., and Sudarshan, S. (2019). *Database System Concepts*. McGraw-Hill, 7th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA122 Linear Algebra II (Mandatory)

1. General information

- **Semester**: 4th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA121 Linear Algebra I (3rd Sem-Pg. 358)

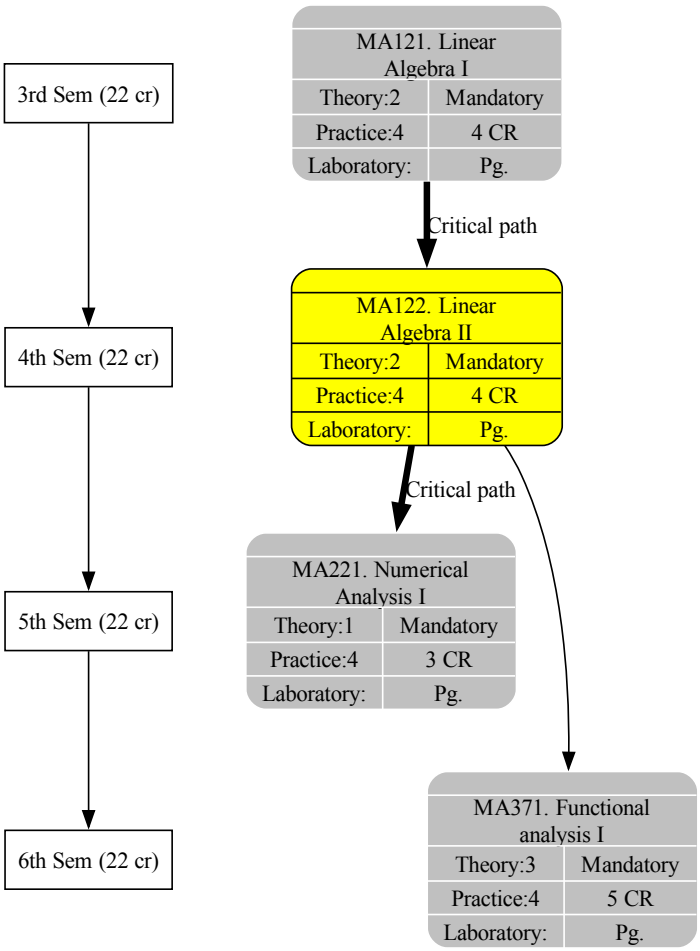


Figure 7.22: Connection Map. MA122 Linear Algebra II

2. Course Rationale

This course deepens advanced concepts of linear algebra, focusing on inner product spaces, advanced linear transformations, matrix decompositions, and computational applications. It develops skills to analyze complex algebraic structures and apply matrix methods to problems in data science, signal processing, and optimization.

3. Goals of the Course

- Master inner product spaces and their properties

- Apply advanced matrix decompositions (SVD, QR)
- Analyze linear transformations in inner product spaces
- Solve approximation and least squares problems

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Numerical Linear Algebra, 16 hr(s)

Competences: ABET-1

Topics: **I:** LU decomposition and its applications **II:** QR decomposition and least squares problems **III:** Singular Value Decomposition (SVD) and its applications **IV:** Condition number and numerical stability of linear systems **V:** Iterative methods for linear systems (Jacobi, Gauss-Seidel, Conjugate Gradient)

Learning Outcomes **I:** Implement LU and QR decomposition algorithms for solving linear systems [Usage] **II:** Explain the geometric and algebraic meaning of the Singular Value Decomposition [Familiarity] **III:** Use SVD for applications such as matrix approximation and principal component analysis [Usage] **IV:** Analyze the stability of a linear system using its condition number [Assessment] **V:** Select and implement an appropriate iterative method for solving large, sparse linear systems [Assessment]

Bibliography: [Strang, 2016, Lay et al., 2016]

5.2 Computational Arithmetic and Error Analysis, 16 hr(s)

Competences: ABET-1

Topics: **I:** Floating-point arithmetic and the IEEE standard **II:** Sources of error: truncation, rounding, and discretization **III:** Error propagation and condition number **IV:** Stability of numerical algorithms **V:** Absolute, relative, and forward/backward error

Learning Outcomes **I:** Define the different types of errors in numerical computation (rounding, truncation, discretization) [Familiarity] **II:** Explain the principles of floating-point representation and arithmetic according to the IEEE standard [Familiarity] **III:** Calculate the absolute and relative error of an approximation [Usage] **IV:** Analyze the propagation of errors in a sequence of arithmetic operations [Usage] **V:** Distinguish between the condition of a problem and the stability of an algorithm [Assessment] **VI:** Compute the condition number of a simple problem (e.g., matrix-vector multiplication) [Usage]

Bibliography: [Strang, 2016, Lay et al., 2016]

5.3 Interpolation and Function Approximation, 16 hr(s)

Competences: ABET-1

Topics: **I:** Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Strang, 2016, Lay et al., 2016]

5.4 Numerical Methods for Ordinary Differential Equations (ODEs), 16 hr(s)

Competences: ABET-1

Topics: **I:** Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Strang, 2016, Trefethen and III, 1997]

6. Bibliography

[Lay et al., 2016] Lay, D. C., Lay, S. R., and McDonald, J. J. (2016). *Álgebra Lineal y sus Aplicaciones*. Pearson, 5th edition.

[Strang, 2016] Strang, G. (2016). *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 5th edition.

[Trefethen and III, 1997] Trefethen, L. N. and III, D. B. (1997). *Numerical Linear Algebra*. SIAM.

[Lay et al., 2016] Lay, D. C., Lay, S. R., and McDonald, J. J. (2016). *Álgebra Lineal y sus Aplicaciones*. Pearson, 5th edition.

[Strang, 2016] Strang, G. (2016). *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 5th edition.

[Trefethen and III, 1997] Trefethen, L. N. and III, D. B. (1997). *Numerical Linear Algebra*. SIAM.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA210 Calculus III (Mandatory)

1. General information

- **Semester**: 4th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA112 Calculus II (3rd Sem-Pg. 356)

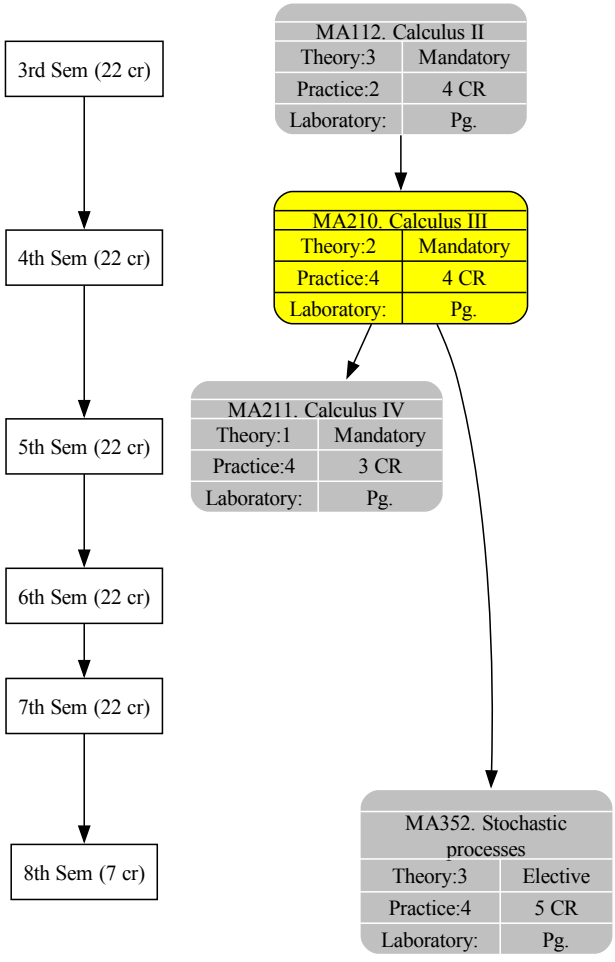


Figure 7.23: Connection Map. MA210 Calculus III

2. Course Rationale

The nature of the course is theoretical-practical; its purpose is to develop a conceptual and methodological framework in students aimed at providing a solid mathematical foundation for understanding all natural phenomena. It includes the following thematic units: Differential calculus in several variables. Integral calculus in several variables. Vector calculus.

3. Goals of the Course

- Understand functions of several variables and their partial derivatives

- Apply differential and integral calculus in three dimensions
- Solve optimization problems with multiple variables
- Master multiple integrals and their applications

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Numerical Differentiation and Integration, 20 hr(s)

Competences: ABET-1

Topics: **I:** Finite difference approximations (forward, backward, central) **II:** Richardson extrapolation **III:** Newton-Cotes formulas (Trapezoidal rule, Simpson's rule) **IV:** Gaussian quadrature **V:** Error analysis for numerical integration

Learning Outcomes **I:** Compute derivatives of functions using finite difference schemes [Usage] **II:** Apply Richardson extrapolation to improve the accuracy of numerical approximations [Usage] **III:** Implement numerical integration rules (Trapezoidal, Simpson, Gaussian) [Usage] **IV:** Compare the accuracy and efficiency of different numerical integration methods [Assessment] **V:** Analyze the error for numerical differentiation and integration formulas [Assessment]

Bibliography: [Stewart, 2016, Thomas et al., 2014]

5.2 Numerical Methods for Ordinary Differential Equations (ODEs), 20 hr(s)

Competences: ABET-1

Topics: **I:** Euler's method and its variants (forward, backward) **II:** Runge-Kutta methods **III:** Multistep methods (Adams-Bashforth, Adams-Moulton) **IV:** Stability and convergence analysis for ODE solvers **V:** Systems of ODEs and higher-order ODEs

Learning Outcomes **I:** Implement basic ODE solvers (Euler, Runge-Kutta) [Usage] **II:** Analyze the stability and convergence properties of ODE numerical methods [Assessment] **III:** Select an appropriate numerical method for a given ODE based on stability and accuracy requirements [Assessment] **IV:** Solve systems of ODEs and convert higher-order ODEs to first-order systems [Usage]

Bibliography: [Stewart, 2016, Thomas et al., 2014]

5.3 Interpolation and Function Approximation, 20 hr(s)

Competences: ABET-1

Topics: **I:** Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Stewart, 2016, Thomas et al., 2014]

5.4 Roots Finding and Basic Numerical Optimization, 20 hr(s)

Competences: ABET-1

Topics: **I:** Bisection method and fixed-point iteration **II:** Newton's method and secant method for root finding **III:** Convergence analysis and order of convergence **IV:** Gradient descent for univariate optimization **V:** Rate of convergence and stopping criteria

Learning Outcomes **I:** Implement root-finding algorithms (bisection, Newton, secant) and analyze their convergence [Usage] **II:** Classify the order of convergence of iterative methods [Assessment] **III:** Apply gradient descent to solve simple univariate optimization problems [Usage] **IV:** Design appropriate stopping criteria for iterative methods [Usage]

Bibliography: [Stewart, 2016, Thomas et al., 2014]

6. Bibliography

[Stewart, 2016] Stewart, J. (2016). *Cálculo de varias variables: Trascendentes tempranas*. Cengage Learning, 8th edition.

[Thomas et al., 2014] Thomas, G. B., Weir, M. D., and Hass, J. (2014). *Cálculo de varias variables*. Pearson, 13th edition.

[Stewart, 2016] Stewart, J. (2016). *Cálculo de varias variables: Trascendentes tempranas*. Cengage Learning, 8th edition.

[Thomas et al., 2014] Thomas, G. B., Weir, M. D., and Hass, J. (2014). *Cálculo de varias variables*. Pearson, 13th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

ID102 Technical and professional English II (Mandatory)

1. General information

- **Semester**: 4th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 6 hours;
- **Prerrequisites**:
  - ID101 Technical and professional English I (3rd Sem-Pg. 361)

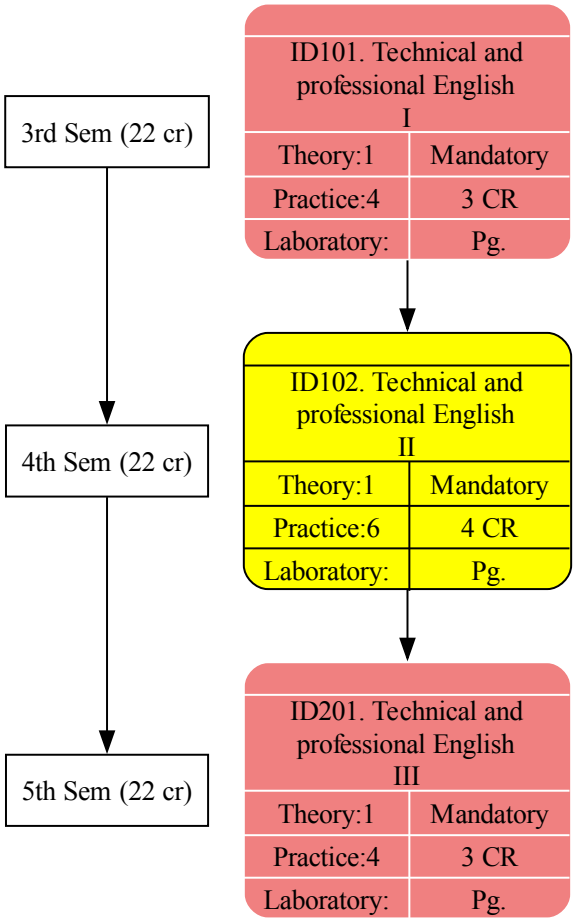


Figure 7.24: Connection Map. ID102 Technical and professional English II

2. Course Rationale

A fundamental part of the integral formation of a professional is the ability to communicate in a foreign language in addition to the native language itself. It not only broadens its cultural horizon but also allows a more humane and comprehensive view of people's lives. In the case of foreign languages, English is undoubtedly the most practical because it is spoken around the world. There is no country where it is not spoken. In careers related to tourist services english is perhaps the most important practical tool that the student must master from the outset as part of his comprehensive education.

3. Goals of the Course

- Develop the ability to fluently speak the language..
- Increase vocabulary and handle simple expressions

4. General Competences (*Outcomes*)

ABET-4) An ability to communicate effectively with a range of audiences. (Usage)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Familiarity)

5. Content

5.1 How long ago?, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: I: Simple past **II:** Negative sentences with ago. **III:** Conjunctions **IV:** Expressions of Time in past **V:** Phonetic relations and symbols **VI:** Expressions to give the date

Learning Outcomes I: At the end of the eighth unit, each student, understanding the grammar of the past tense is able to express a greater number of expressions of time and also to use prepositions to describe varied places and times. It is also capable of analyzing and expressing ideas about dates and numbers in order.

Bibliography: [Soars and John, 2002a, Cambridge, 2006, MacGrew, 1999]

5.2 Food you like!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: I: Accounting and Non-Accounting Nouns **II:** Expressions with Would like and I'd like **III:** Quantifiers **IV:** Meals around the world **V:** Formal requests **VI:** Formal letters

Learning Outcomes I: At the end of the ninth unit, students having identified how to express orders and make collections, uses in various situations. Express situations and states related to quantities. Explain and apply food and drinks vocabulary.

Bibliography: [Soars and John, 2002a, Cambridge, 2006, MacGrew, 1999]

5.3 The world of work, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: I: Adjectives **II:** Sentences with Comparative Adjectives **III:** Sentences with Superlative Adjectives **IV:** Cities and countryside **V:** Directional indications

Learning Outcomes I: At the end of the tenth unit, students who have recognized the characteristics of adjectives use these to make comparisons of various types. Describes people and places and directions. They will use conjunctions to unite type ideas.

Bibliography: [Soars and John, 2002a, Cambridge, 2006, MacGrew, 1999]

5.4 Looking good!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Present continuous **II:** Affirmative sentences, Negatives and Questions **III:** Use of Whose **IV:** Possessive pronouns **V:** Clothing and colors **VI:** Expressions to wear in clothing stores **VII:** Phonetic symbols.

Learning Outcomes **I:** At the end of the eleventh unit, students having identified the idea of expressing ideas of actions that occur at the time or that relate at any time structure sentences in Present Progressive. They express ideas of possession with regard to clothes and colors.

Bibliography: [Soars and John, 2002a, Cambridge, 2006, MacGrew, 1999]

5.5 Life is an adventure!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Use of going to **II:** Future time sentences **III:** Expressions of Quantity. **IV:** Action verbs **V:** Vocabulary of the climate **VI:** Expressions of Suggestion **VII:** Write a postcard

Learning Outcomes **I:** At the end of the twelfth unit, students, from the understanding of future time, will elaborate sentences using the necessary elements. They will also assimilate the need to express purposeful infinitives. They will acquire vocabulary to describe the climate. Expressions will be presented to make and ask for suggestions.

Bibliography: [Soars and John, 2002a, Cambridge, 2006, MacGrew, 1999]

5.6 You‘re pretty smart!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Forms of Questions **II:** Adverbs and Adjectives **III:** Vocabulary description of feelings **IV:** Expressions for train travel **V:** Writing Short Stories **VI:** Readings

Learning Outcomes **I:** At the end of the thirteenth unit, students having learned the fundamentals of structuring various questions, will perform application work in appropriate contexts. They emphasize the difference between adjectives and adverbs. They describe feelings. They use expressions to catch a train. They assume the idea is suffixes and prefixes.

Bibliography: [Soars and John, 2002a, Cambridge, 2006, MacGrew, 1999]

5.7 Have you ever?, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Perfect present **II:** Keywords with never, ever, and yet **III:** Vocabulary verbs in Past participle **IV:** Expressions for airplane travel **V:** Writing thank-you letters **VI:** Readings

Learning Outcomes **I:** At the end of the fourteenth unit, students having known the fundamentals of the structure of the Present Perfect experience the need to express this type of time in actions. They will practice in appropriate contexts. They emphasize the difference between simple past and perfect present. Describe actions with never, ever, and yet. They use expressions to use at an airport.

Bibliography: [Soars and John, 2002a, Cambridge, 2006, MacGrew, 1999]

6. Bibliography

[Cambridge, 2006] Cambridge (2006). *Diccionario Inglés-Espanol Cambridge*. Editorial Oxford.

[MacGrew, 1999] MacGrew, J. (1999). *Focus on Grammar Basic*. Editorial Oxford.

[Soars and John, 2002a] Soars, L. and John (2002a). *American Headway N 1 Student Book*. Editorial Oxford.

Fifth Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS251 Computer graphics (Mandatory)

1. General information

- **Semester**: 5th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS210 Algorithms and Data Structures (4th Sem-Pg. 367)

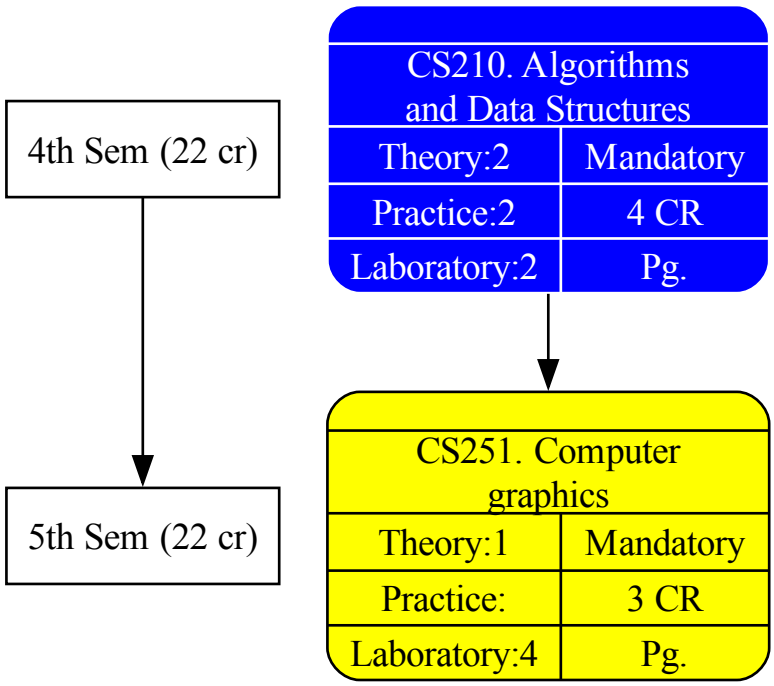


Figure 7.25: Connection Map. CS251 Computer graphics

2. Course Rationale

Computer Graphics is a fundamental area of Computer Science that focuses on the mathematical and computational techniques for generating and manipulating images. This course provides students with the foundations of 2D and 3D modeling, rendering pipelines, and the physical principles of light and color, enabling them to develop high-performance visual applications, simulations, and interactive systems.

3. Goals of the Course

- Understand the graphics pipeline and coordinate system transformations.
- Implement rendering algorithms for realistic and real-time visualization.
- Apply mathematical modeling to create and manipulate 3D objects.

4. General Competences (*Outcomes*)

5. Content

5.1 Fundamental Concepts of Graphics and Interactive Techniques, 16 hr(s)

Competences:

Topics: **I:** Overview of computer graphics pipeline and interactive systems **II:** Basic rendering concepts: rasterization, ray tracing, shading models **III:** Color theory and color spaces **IV:** Coordinate systems and transformations **V:** Basic interaction techniques: input devices, event handling **VI:** Advanced rendering concepts: global illumination, physically based rendering **VII:** Graphics hardware and acceleration techniques **VIII:** Human perception and visual cognition

Learning Outcomes **I:** Explain the basic stages of the computer graphics pipeline [Explain] **II:** Describe basic rendering techniques such as rasterization and ray tracing [Describe] **III:** Apply color theory to design visually effective graphics [Apply] **IV:** Implement coordinate transformations for 2D and 3D graphics [Implement] **V:** Design basic interactive applications using common input devices and event handling [Design] **VI:** Compare advanced rendering techniques such as global illumination and physically based rendering [Compare] **VII:** Analyze the role of graphics hardware in accelerating rendering tasks [Analyze] **VIII:** Explain how human perception influences graphics and interaction design [Explain]

Bibliography: [Marschner and Shirley, 2021, Hearn et al., 2014]

5.2 Applied Rendering and Techniques, 16 hr(s)

Competences:

Topics: **I:** Rasterization algorithms: scan conversion, antialiasing **II:** Ray tracing: acceleration structures, sampling, shading **III:** Texture mapping and filtering **IV:** Shading models: Lambertian, Phong, BRDF **V:** Advanced techniques: shadow mapping, environment mapping, deferred rendering

Learning Outcomes **I:** Implement rasterization algorithms for primitive rendering [Implement] **II:** Implement basic ray tracing with acceleration structures [Implement] **III:** Apply texture mapping and filtering techniques to enhance visual realism [Apply] **IV:** Compare different shading models and their applications [Compare] **V:** Implement advanced rendering techniques such as shadow mapping or deferred rendering [Implement]

Bibliography: [Marschner and Shirley, 2021, Hearn et al., 2014, Angel and Shreiner, 2014]

5.3 Geometric Modeling, 16 hr(s)

Competences:

Topics: **I:** Geometric representations: polygons, curves, surfaces, implicit forms **II:** Modeling techniques: splines, subdivision surfaces, constructive solid geometry **III:** Procedural modeling and generation **IV:** Geometric algorithms: intersection, distance, decomposition

Learning Outcomes **I:** Describe different geometric representations and their trade-offs [Describe] **II:** Implement modeling techniques such as spline curves or subdivision surfaces [Implement] **III:** Create procedural models using algorithms or rules [Create] **IV:** Apply geometric algorithms to solve problems like intersection testing or mesh processing [Apply]

Bibliography: [Marschner and Shirley, 2021, Angel and Shreiner, 2014]

5.4 Visualization, 16 hr(s)

Competences:

Topics: **I:** Principles of data visualization: perception, cognition, and effective encoding **II:** Types of visualizations: scientific, information, geographical **III:** Visualization techniques for multivariate, temporal, and hierarchical data **IV:** Interactive visualization and exploration **V:** Evaluation of visualization effectiveness

Learning Outcomes **I:** Apply principles of perception and cognition to design effective visualizations [Apply] **II:** Select appropriate visualization techniques for different data types and tasks [Evaluate] **III:** Implement interactive visualizations for data exploration [Implement] **IV:** Evaluate the effectiveness of a visualization using empirical methods [Evaluate]

Bibliography: [Marschner and Shirley, 2021]

5.5 Shading and Advanced Rendering, 12 hr(s)

Competences:

Topics: **I:** Advanced shading: physically based shading, subsurface scattering **II:** Global illumination: radiosity, photon mapping, path tracing **III:** Real-time rendering techniques: level of detail, culling, occlusion **IV:** GPU programming and shaders

Learning Outcomes **I:** Implement advanced shading models for realistic materials [Implement] **II:** Compare global illumination algorithms and their trade-offs [Compare] **III:** Optimize real-time rendering using techniques like level of detail and culling [Create] **IV:** Develop GPU shaders for custom rendering effects [Create]

Bibliography: [Marschner and Shirley, 2021, Angel and Shreiner, 2014]

6. Bibliography

[Angel and Shreiner, 2014] Angel, E. and Shreiner, D. (2014). *Interactive Computer Graphics: A Top-Down Approach with WebGL*. Pearson, 7th edition.

[Hearn et al., 2014] Hearn, D., Baker, M. P., and Carithers, W. (2014). *Computer Graphics with OpenGL*. Pearson, 4th edition.

[Marschner and Shirley, 2021] Marschner, S. and Shirley, P. (2021). *Fundamentals of Computer Graphics*. A K Peters/CRC Press, 5th edition.

[Angel and Shreiner, 2014] Angel, E. and Shreiner, D. (2014). *Interactive Computer Graphics: A Top-Down Approach with WebGL*. Pearson, 7th edition.

[Hearn et al., 2014] Hearn, D., Baker, M. P., and Carithers, W. (2014). *Computer Graphics with OpenGL*. Pearson, 4th edition.

[Marschner and Shirley, 2021] Marschner, S. and Shirley, P. (2021). *Fundamentals of Computer Graphics*. A K Peters/CRC Press, 5th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA211 Calculus IV (Mandatory)

1. General information

- **Semester**: 5^{th} Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA210 Calculus III (4^{th} Sem-Pg. 380)
  - MA121 Linear Algebra I (3^{rd} Sem-Pg. 358)

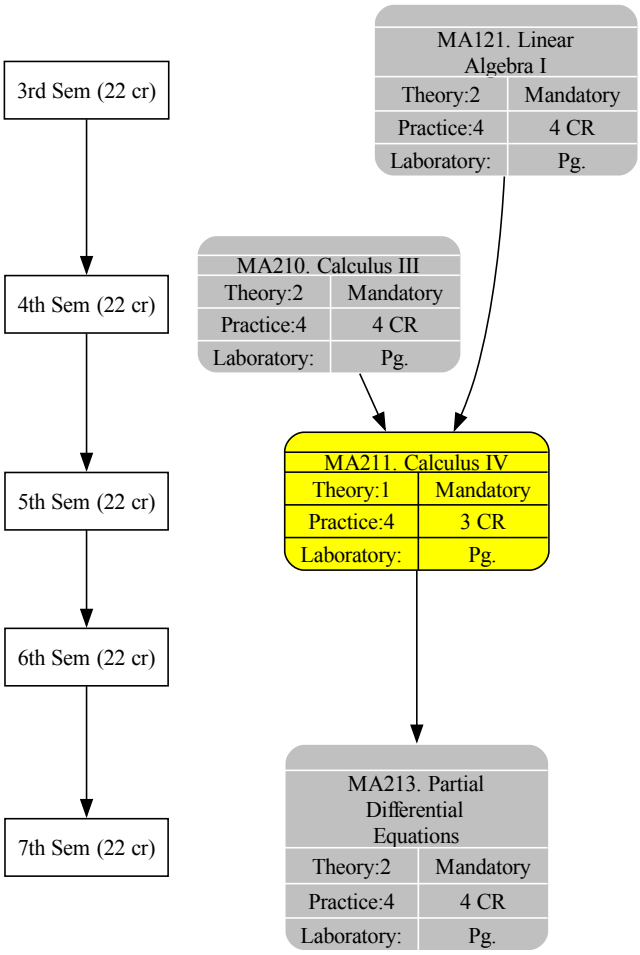


Figure 7.26: Connection Map. MA211 Calculus IV

2. Course Rationale

The nature of the course is theoretical-practical; its purpose is to develop a conceptual and methodological framework aimed at providing a solid mathematical foundation for understanding all natural phenomena. It includes the following thematic units: Sequences in R^n, C, (R^m, R). Directional derivatives and implicit differentiation. Isometries of R^n, conics, and quadric surfaces. Vector fields.

3. Goals of the Course

- Understand the concepts of vector fields and differential operators
- Apply line and surface integrals to physical problems
- Master the fundamental theorems of vector calculus
- Solve problems of flux, circulation, and vector potentials

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Numerical Differentiation and Integration, 20 hr(s)

Competences: ABET-1

Topics: **I:** Finite difference approximations (forward, backward, central) **II:** Richardson extrapolation **III:** Newton-Cotes formulas (Trapezoidal rule, Simpson's rule) **IV:** Gaussian quadrature **V:** Error analysis for numerical integration

Learning Outcomes **I:** Compute derivatives of functions using finite difference schemes [Usage] **II:** Apply Richardson extrapolation to improve the accuracy of numerical approximations [Usage] **III:** Implement numerical integration rules (Trapezoidal, Simpson, Gaussian) [Usage] **IV:** Compare the accuracy and efficiency of different numerical integration methods [Assessment] **V:** Analyze the error for numerical differentiation and integration formulas [Assessment]

Bibliography: [Marsden and Tromba, 2003, Stewart, 2016]

5.2 Numerical Methods for Ordinary Differential Equations (ODEs), 20 hr(s)

Competences: ABET-1

Topics: **I:** Euler's method and its variants (forward, backward) **II:** Runge-Kutta methods **III:** Multistep methods (Adams-Bashforth, Adams-Moulton) **IV:** Stability and convergence analysis for ODE solvers **V:** Systems of ODEs and higher-order ODEs

Learning Outcomes **I:** Implement basic ODE solvers (Euler, Runge-Kutta) [Usage] **II:** Analyze the stability and convergence properties of ODE numerical methods [Assessment] **III:** Select an appropriate numerical method for a given ODE based on stability and accuracy requirements [Assessment] **IV:** Solve systems of ODEs and convert higher-order ODEs to first-order systems [Usage]

Bibliography: [Marsden and Tromba, 2003, Stewart, 2016]

5.3 Interpolation and Function Approximation, 20 hr(s)

Competences: ABET-1

Topics: **I:** Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Marsden and Tromba, 2003, Stewart, 2016]

5.4 Roots Finding and Basic Numerical Optimization, 20 hr(s)

Competences: ABET-1

Topics: **I:** Bisection method and fixed-point iteration **II:** Newton's method and secant method for root finding **III:** Convergence analysis and order of convergence **IV:** Gradient descent for univariate optimization **V:** Rate of convergence and stopping criteria

Learning Outcomes **I:** Implement root-finding algorithms (bisection, Newton, secant) and analyze their convergence [Usage] **II:** Classify the order of convergence of iterative methods [Assessment] **III:** Apply gradient descent to solve simple univariate optimization problems [Usage] **IV:** Design appropriate stopping criteria for iterative methods [Usage]

Bibliography: [Marsden and Tromba, 2003, Stewart, 2016]

6. Bibliography

[Marsden and Tromba, 2003] Marsden, J. E. and Tromba, A. J. (2003). *Cálculo Vectorial*. Pearson, 5th edition.

[Stewart, 2016] Stewart, J. (2016). *Cálculo de varias variables: Trascendentes tempranas*. Cengage Learning, 8th edition.

[Marsden and Tromba, 2003] Marsden, J. E. and Tromba, A. J. (2003). *Vector Calculus*. Pearson, 5th edition.

[Stewart, 2016] Stewart, J. (2016). *Multivariable Calculus: Early Transcendentals*. Cengage Learning, 8th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA221 Numerical Analysis I (Mandatory)

1. General information

- **Semester**: 5th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA112 Calculus II (3rd Sem-Pg. 356)
  - MA122 Linear Algebra II (4th Sem-Pg. 377)
  - AI161 Applied AI (2nd Sem-Pg. 333)

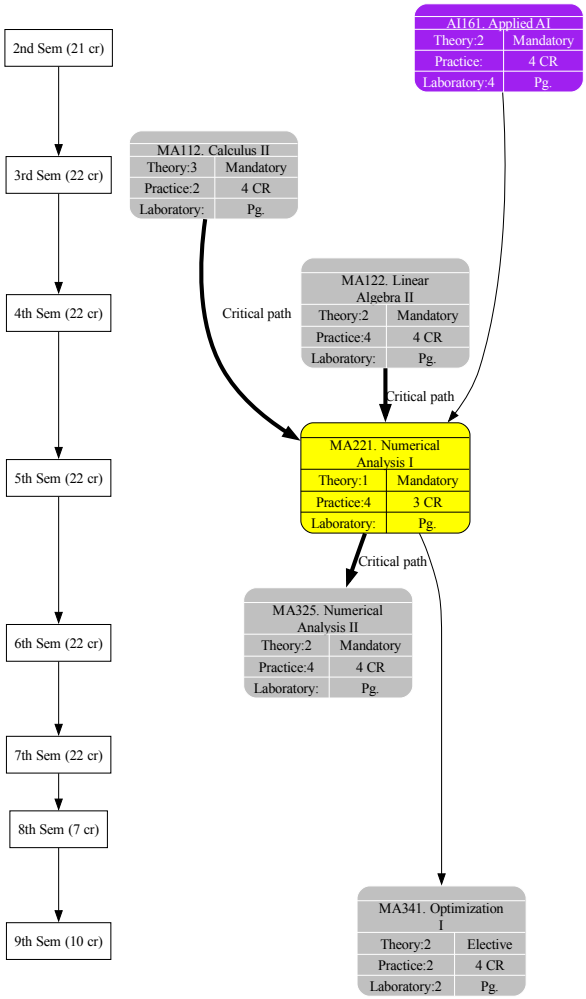


Figure 7.27: Connection Map. MA221 Numerical Analysis I

2. Course Rationale

This course introduces fundamental numerical methods for solving mathematical problems through computational algorithms. It develops skills for implementing and analyzing numerical methods in linear algebra, nonlinear equations, and interpolation, with emphasis on error analysis and numerical stability.

3. Goals of the Course

- Understand fundamentals of computational arithmetic and error analysis
- Implement numerical methods for linear systems of equations
- Solve nonlinear equations using iterative methods
- Apply function interpolation and approximation techniques

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Computational Arithmetic and Error Analysis, 20 hr(s)

Competences: ABET-1

Topics: **I:** Floating-point arithmetic and the IEEE standard **II:** Sources of error: truncation, rounding, and discretization **III:** Error propagation and condition number **IV:** Stability of numerical algorithms **V:** Absolute, relative, and forward/backward error

Learning Outcomes **I:** Define the different types of errors in numerical computation (rounding, truncation, discretization) [Familiarity] **II:** Explain the principles of floating-point representation and arithmetic according to the IEEE standard [Familiarity] **III:** Calculate the absolute and relative error of an approximation [Usage] **IV:** Analyze the propagation of errors in a sequence of arithmetic operations [Usage] **V:** Distinguish between the condition of a problem and the stability of an algorithm [Assessment] **VI:** Compute the condition number of a simple problem (e.g., matrix-vector multiplication) [Usage]

Bibliography: [Burden et al., 2016, Chapra and Canale, 2015]

5.2 Numerical Linear Algebra, 20 hr(s)

Competences: ABET-1

Topics: **I:** LU decomposition and its applications **II:** QR decomposition and least squares problems **III:** Singular Value Decomposition (SVD) and its applications **IV:** Condition number and numerical stability of linear systems **V:** Iterative methods for linear systems (Jacobi, Gauss-Seidel, Conjugate Gradient)

Learning Outcomes **I:** Implement LU and QR decomposition algorithms for solving linear systems [Usage] **II:** Explain the geometric and algebraic meaning of the Singular Value Decomposition [Familiarity] **III:** Use SVD for applications such as matrix approximation and principal component analysis [Usage] **IV:** Analyze the stability of a linear system using its condition number [Assessment] **V:** Select and implement an appropriate iterative method for solving large, sparse linear systems [Assessment]

Bibliography: [Burden et al., 2016, Chapra and Canale, 2015]

5.3 Roots Finding and Basic Numerical Optimization, 20 hr(s)

Competences: ABET-1

Topics: **I:** Bisection method and fixed-point iteration **II:** Newton's method and secant method for root finding **III:** Convergence analysis and order of convergence **IV:** Gradient descent for univariate optimization **V:** Rate of convergence and stopping criteria

Learning Outcomes I: Implement root-finding algorithms (bisection, Newton, secant) and analyze their convergence [Usage] **II:** Classify the order of convergence of iterative methods [Assessment] **III:** Apply gradient descent to solve simple univariate optimization problems [Usage] **IV:** Design appropriate stopping criteria for iterative methods [Usage]

Bibliography: [Burden et al., 2016, Chapra and Canale, 2015]

5.4 Interpolation and Function Approximation, 20 hr(s)

Competences: ABET-1

Topics: I: Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes I: Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Burden et al., 2016, Chapra and Canale, 2015]

6. Bibliography

[Burden et al., 2016] Burden, R. L., Faires, J. D., and Burden, A. M. (2016). *Análisis Numérico*. Cengage Learning, 10th edition.

[Chapra and Canale, 2015] Chapra, S. C. and Canale, R. P. (2015). *Métodos Numéricos para Ingenieros*. McGraw-Hill, 7th edition.

[Burden et al., 2016] Burden, R. L., Faires, J. D., and Burden, A. M. (2016). *Análisis Numérico*. Cengage Learning, 10th edition.

[Chapra and Canale, 2015] Chapra, S. C. and Canale, R. P. (2015). *Métodos Numéricos para Ingenieros*. McGraw-Hill, 7th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA222 Real Analysis I (Mandatory)

1. General information

- **Semester**: 5th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA112 Calculus II (3rd Sem-Pg. 356)
  - ID101 Technical and professional English I (3rd Sem-Pg. 361)

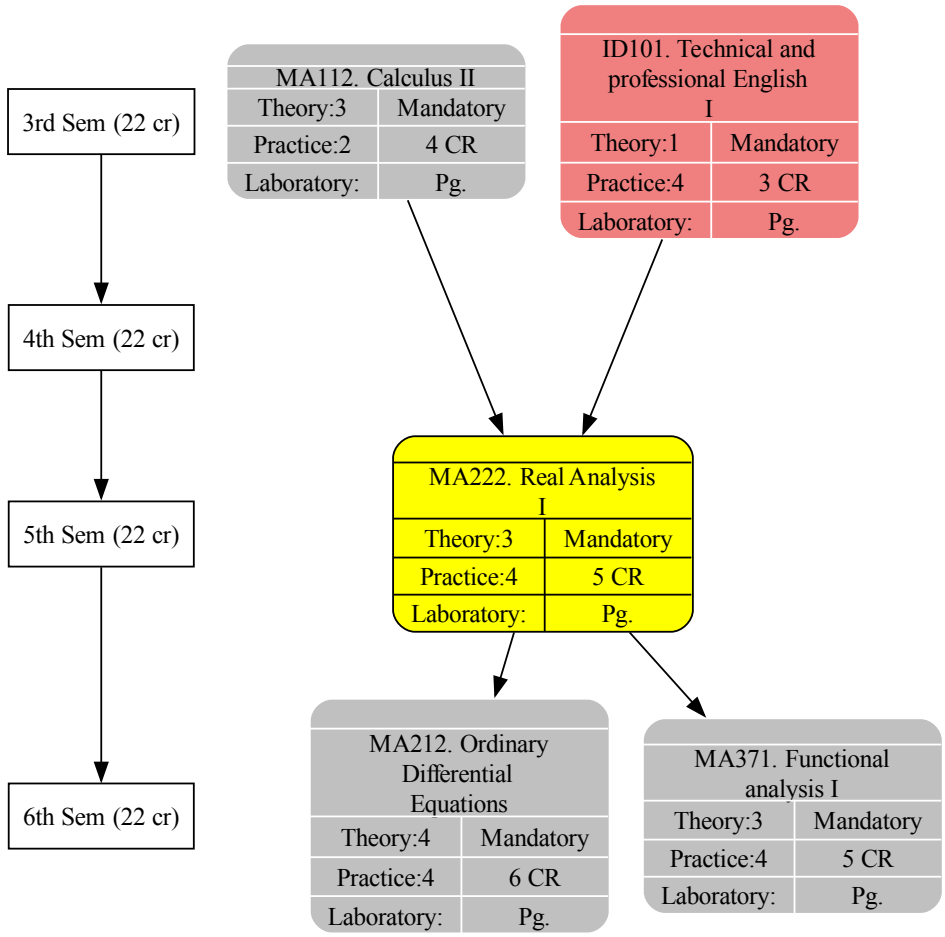


Figure 7.28: Connection Map. MA222 Real Analysis I

2. Course Rationale

The subject belongs to the area of Specific Studies; the nature of the course is theoretical-practical, and its purpose is oriented toward providing a solid mathematical foundation for understanding all natural phenomena. It includes the following thematic units: Countability. Sequences of real numbers. Topology of R. Limit and continuity of real functions of a real variable. Derivation. Riemann-Stieltjes integral.

3. Goals of the Course

- Understand the structure of real numbers and their properties
- Analyze convergence of sequences and numerical series
- Master concepts of limits and continuity of functions
- Develop skills in rigorous mathematical proof

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Computational Arithmetic and Error Analysis, 20 hr(s)

Competences: ABET-1

Topics: **I:** Floating-point arithmetic and the IEEE standard **II:** Sources of error: truncation, rounding, and discretization **III:** Error propagation and condition number **IV:** Stability of numerical algorithms **V:** Absolute, relative, and forward/backward error

Learning Outcomes **I:** Define the different types of errors in numerical computation (rounding, truncation, discretization) [Familiarity] **II:** Explain the principles of floating-point representation and arithmetic according to the IEEE standard [Familiarity] **III:** Calculate the absolute and relative error of an approximation [Usage] **IV:** Analyze the propagation of errors in a sequence of arithmetic operations [Usage] **V:** Distinguish between the condition of a problem and the stability of an algorithm [Assessment] **VI:** Compute the condition number of a simple problem (e.g., matrix-vector multiplication) [Usage]

Bibliography: [Rudin, 1976, Bartle and Sherbert, 2011]

5.2 Roots Finding and Basic Numerical Optimization, 20 hr(s)

Competences: ABET-1

Topics: **I:** Bisection method and fixed-point iteration **II:** Newton's method and secant method for root finding **III:** Convergence analysis and order of convergence **IV:** Gradient descent for univariate optimization **V:** Rate of convergence and stopping criteria

Learning Outcomes **I:** Implement root-finding algorithms (bisection, Newton, secant) and analyze their convergence [Usage] **II:** Classify the order of convergence of iterative methods [Assessment] **III:** Apply gradient descent to solve simple univariate optimization problems [Usage] **IV:** Design appropriate stopping criteria for iterative methods [Usage]

Bibliography: [Rudin, 1976, Bartle and Sherbert, 2011]

5.3 Interpolation and Function Approximation, 20 hr(s)

Competences: ABET-1

Topics: **I:** Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Rudin, 1976, Bartle and Sherbert, 2011]

5.4 Numerical Differentiation and Integration, 20 hr(s)

Competences: ABET-1

Topics: **I:** Finite difference approximations (forward, backward, central) **II:** Richardson extrapolation **III:** Newton-Cotes formulas (Trapezoidal rule, Simpson's rule) **IV:** Gaussian quadrature **V:** Error analysis for numerical integration

Learning Outcomes **I:** Compute derivatives of functions using finite difference schemes [Usage] **II:** Apply Richardson extrapolation to improve the accuracy of numerical approximations [Usage] **III:** Implement numerical integration rules (Trapezoidal, Simpson, Gaussian) [Usage] **IV:** Compare the accuracy and efficiency of different numerical integration methods [Assessment] **V:** Analyze the error for numerical differentiation and integration formulas [Assessment]

Bibliography: [Rudin, 1976, Bartle and Sherbert, 2011]

6. Bibliography

[Bartle and Sherbert, 2011] Bartle, R. G. and Sherbert, D. R. (2011). *Introduction to Real Analysis*. Wiley, 4th edition.

[Rudin, 1976] Rudin, W. (1976). *Principles of Mathematical Analysis*. McGraw-Hill, 3rd edition.

[Bartle and Sherbert, 2011] Bartle, R. G. and Sherbert, D. R. (2011). *Introduction to Real Analysis*. Wiley, 4th edition.

[Rudin, 1976] Rudin, W. (1976). *Principles of Mathematical Analysis*. McGraw-Hill, 3rd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA251 Introduction to Statistics (Mandatory)

1. General information

- **Semester**: 5th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - AI161 Applied AI (2nd Sem-Pg. 333)

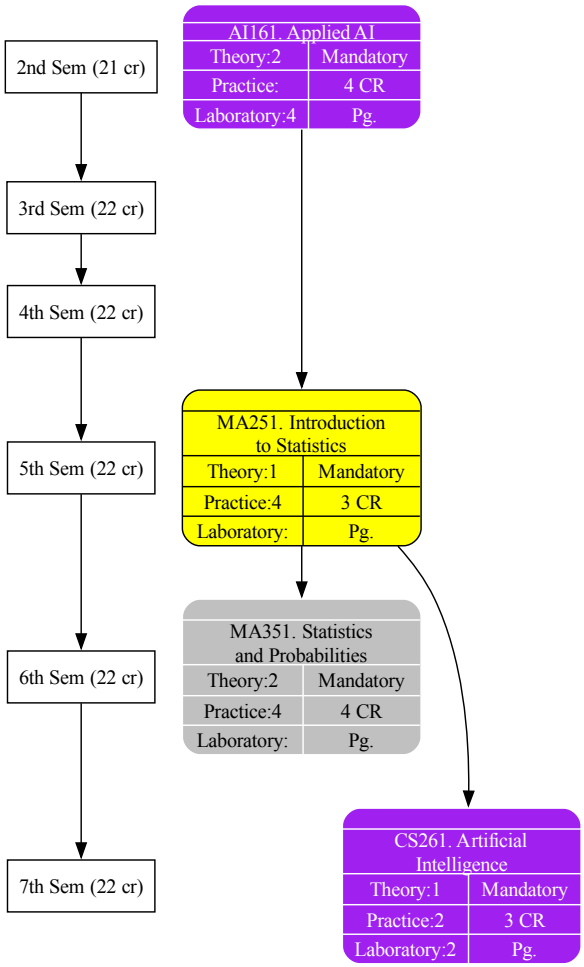


Figure 7.29: Connection Map. MA251 Introduction to Statistics

2. Course Rationale

Write justification for this course here ...

3. Goals of the Course

- Write your first goal here..
- Write your second goal here..

4. General Competences (*Outcomes*)

5. Content

5.1 Unit title, 2 hr(s)

Competences:

Topics: **I:** Topic1 **II:** Topic2

Learning Outcomes **I:** LearningOutcome1 [Familiarity]. **II:** LearningOutcome2 [Usage]. **III:** LearningOutcome3 [Assessment].

Bibliography: [Force, 2020, ACM/IEEE-CS/AAAI Joint Task Force, 2023]

6. Bibliography

[ACM/IEEE-CS/AAAI Joint Task Force, 2023] ACM/IEEE-CS/AAAI Joint Task Force (2023). Cs2023: Acm/ieee-cs/aaai computer science curricula. Technical report, ACM Press and IEEE Computer Society Press and AAAI Press.

[Force, 2020] Force, A.-C. J. T. (2020). Computing curricula 2020. Technical report, ACM Press and IEEE Computer Society Press.

[ACM/IEEE-CS/AAAI Joint Task Force, 2023] ACM/IEEE-CS/AAAI Joint Task Force (2023). Cs2023: Acm/ieee-cs/aaai computer science curricula. Technical report, ACM Press and IEEE Computer Society Press and AAAI Press.

[Force, 2020] Force, A.-C. J. T. (2020). Computing curricula 2020. Technical report, ACM Press and IEEE Computer Society Press.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA281 Numbers theory (Mandatory)

1. General information

- **Semester**: 5^{th} Sem. **Credits**: 2
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 2 hours;
- **Prerrequisites**:
  - MA100 Basic Mathematics (1^{st} Sem-Pg. 300)

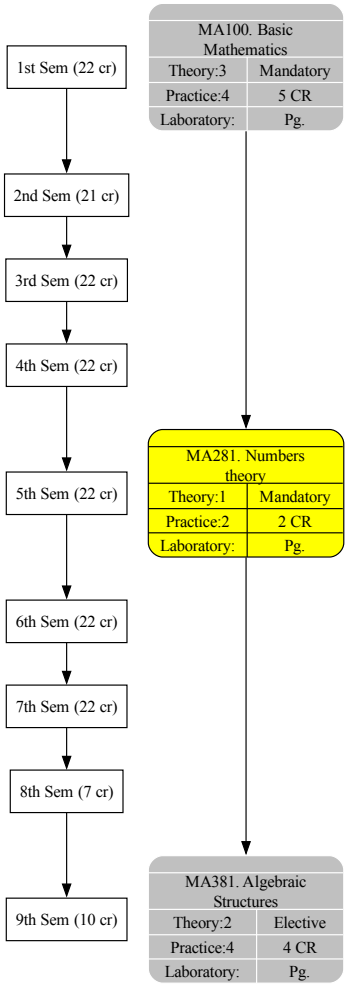


Figure 7.30: Connection Map. MA281 Numbers theory

2. Course Rationale

Number Theory is a fundamental branch of pure mathematics that studies the properties of integers, serving as the foundation for numerous applications in computational mathematics, cryptography, and information theory. This course equips students with the theoretical and algorithmic tools to analyze problems of divisibility, primality, congruences, and Diophantine equations, developing rigorous logical-abstract thinking. The computational focus allows for the implementation and validation of theoretical concepts, laying the groundwork for areas such as public-key cryptography and factorization algorithms.

3. Goals of the Course

- Understand and apply the fundamental principles of divisibility, the Euclidean algorithm, and the fundamental theorem of arithmetic.
- Analyze and solve problems involving linear congruences and systems of congruences using the Chinese Remainder Theorem.
- Study basic arithmetic functions and their applications in the distribution of prime numbers.
- Introduce the concepts of quadratic residues and the law of quadratic reciprocity, understanding their theoretical importance.
- Implement basic numerical algorithms (Euclid's algorithm, primality testing, modular exponentiation) to validate theoretical concepts.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Divisibility and Congruences, 16 hr(s)

Competences: ABET-1

Topics: I: Axioms of the integers: Well-Ordering Principle and Mathematical Induction. **II:** Divisibility: Division Algorithm, Greatest Common Divisor (Extended Euclidean Algorithm) and Bézout's Lemma. **III:** Prime and composite numbers. Fundamental Theorem of Arithmetic. Infinitude of primes. **IV:** Congruences: definition, algebraic properties, and residue classes modulo n. **V:** Computational applications: implementation of the Euclidean algorithm and basic primality tests.

Learning Outcomes I: Understand the fundamental axioms of the integers and the proof technique of induction [Familiarity]. **II:** Use the extended Euclidean algorithm to compute the greatest common divisor and the Bézout identity [Usage]. **III:** Prove basic properties of prime numbers and congruences, applying them to elementary problems [Assessment].

Bibliography: [Rosen, 2011, Apostol, 1976]

5.2 Diophantine Equations and the Chinese Remainder Theorem, 16 hr(s)

Competences: ABET-1

Topics: I: Linear Diophantine equations: conditions for solvability and general solution method. **II:** Chinese Remainder Theorem for systems of linear congruences with coprime moduli. **III:** Modular arithmetic: the ring $\mathbb{Z}_n$, units, and zero divisors. **IV:** Fermat's and Euler's Theorems. Euler's totient function ϕ and its properties. **V:** Applications in basic cryptography: Caesar cipher and foundations of RSA.

Learning Outcomes I: Identify the conditions for the existence of solutions to linear Diophantine equations [Familiarity]. **II:** Apply the Chinese Remainder Theorem to solve systems of congruences and calendar problems [Usage]. **III:** Use Fermat's and Euler's theorems to simplify modular exponentiation calculations and analyze basic algebraic structures [Assessment].

Bibliography: [Rosen, 2011, Burton, 2011]

5.3 Arithmetic Functions and Multiplicative Number Theory, 16 hr(s)

Competences: ABET-1

Topics: I: Arithmetic functions: Definition and examples (tau, sigma, phi, mu). **II:** Dirichlet convolution and the Möbius inversion formula. **III:** Perfect, amicable, and abundant numbers. Open conjectures. **IV:** Orders and primitive roots modulo a prime. **V:** Discrete logarithms: definition and applications in key exchange protocols (Diffie-Hellman).

Learning Outcomes I: Recognize the properties of the most important arithmetic functions [Familiarity]. **II:** Apply Möbius inversion to solve counting problems in number theory [Usage]. **III:** Compute orders and primitive roots, relating them to the cyclic structure of $\mathbb{Z}_p^*$ [Assessment].

Bibliography: [Apostol, 1976, Nathanson, 2000]

5.4 Quadratic Residues and Computational Applications, 16 hr(s)

Competences: ABET-1

Topics: I: Quadratic residues and non-residues. Legendre symbol and its properties. **II:** Law of Quadratic Reciprocity. Proof and applications in computing the Legendre symbol. **III:** Jacobi symbol and its use in probabilistic primality tests. **IV:** Introduction to primality testing: Miller-Rabin test. **V:** Integer factorization: Pollard's $p-1$ method and notions of the RSA algorithm.

Learning Outcomes I: Determine whether a number is a quadratic residue modulo a prime using the Legendre symbol [Familiarity]. **II:** Apply the Law of Quadratic Reciprocity to compute Legendre and Jacobi symbols efficiently [Usage]. **III:** Implement basic primality testing and factorization algorithms, analyzing their complexity and limitations [Assessment].

Bibliography: [Rosen, 2011, Hoffstein et al., 2008]

6. Bibliography

[Apostol, 1976] Apostol, T. M. (1976). *Introduction to Analytic Number Theory*. Springer-Verlag, 1st edition. Texto clásico y riguroso para el estudio de funciones aritméticas y teoría analítica de números.

[Burton, 2011] Burton, D. M. (2011). *Elementary Number Theory*. McGraw-Hill Education, 7th edition. Libro introductorio muy didáctico, con abundantes ejemplos y ejercicios.

[Hoffstein et al., 2008] Hoffstein, J., Pipher, J., and Silverman, J. H. (2008). *An Introduction to Mathematical Cryptography*. Springer, 1st edition. Excelente puente entre la teoría de números y la criptografía aplicada.

[Nathanson, 2000] Nathanson, M. B. (2000). *Elementary Methods in Number Theory*, volume 195 of *Graduate Texts in Mathematics*. Springer. Enfoque en métodos combinatorios y aditivos, con demostraciones elegantes.

[Rosen, 2011] Rosen, K. H. (2011). *Elementary Number Theory and Its Applications*. Pearson, 6th edition. Texto ampliamente utilizado, con un balance excelente entre teoría, ejemplos y aplicaciones computacionales.

[Apostol, 1976] Apostol, T. M. (1976). *Introduction to Analytic Number Theory*. Springer-Verlag, 1st edition. Texto clásico y riguroso para el estudio de funciones aritméticas y teoría analítica de números.

[Burton, 2011] Burton, D. M. (2011). *Elementary Number Theory*. McGraw-Hill Education, 7th edition. Libro introductorio muy didáctico, con abundantes ejemplos y ejercicios.

[Hoffstein et al., 2008] Hoffstein, J., Pipher, J., and Silverman, J. H. (2008). *An Introduction to Mathematical Cryptography*. Springer, 1st edition. Excelente puente entre la teoría de números y la criptografía aplicada.

[Nathanson, 2000] Nathanson, M. B. (2000). *Elementary Methods in Number Theory*, volume 195 of *Graduate Texts in Mathematics*. Springer. Enfoque en métodos combinatorios y aditivos, con demostraciones elegantes.

[Rosen, 2011] Rosen, K. H. (2011). *Elementary Number Theory and Its Applications*. Pearson, 6th edition. Texto ampliamente utilizado, con un balance excelente entre teoría, ejemplos y aplicaciones computacionales.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

ID201 Technical and professional English III (Mandatory)

1. General information

- **Semester**: 5th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - ID102 Technical and professional English II (4th Sem-Pg. 383)

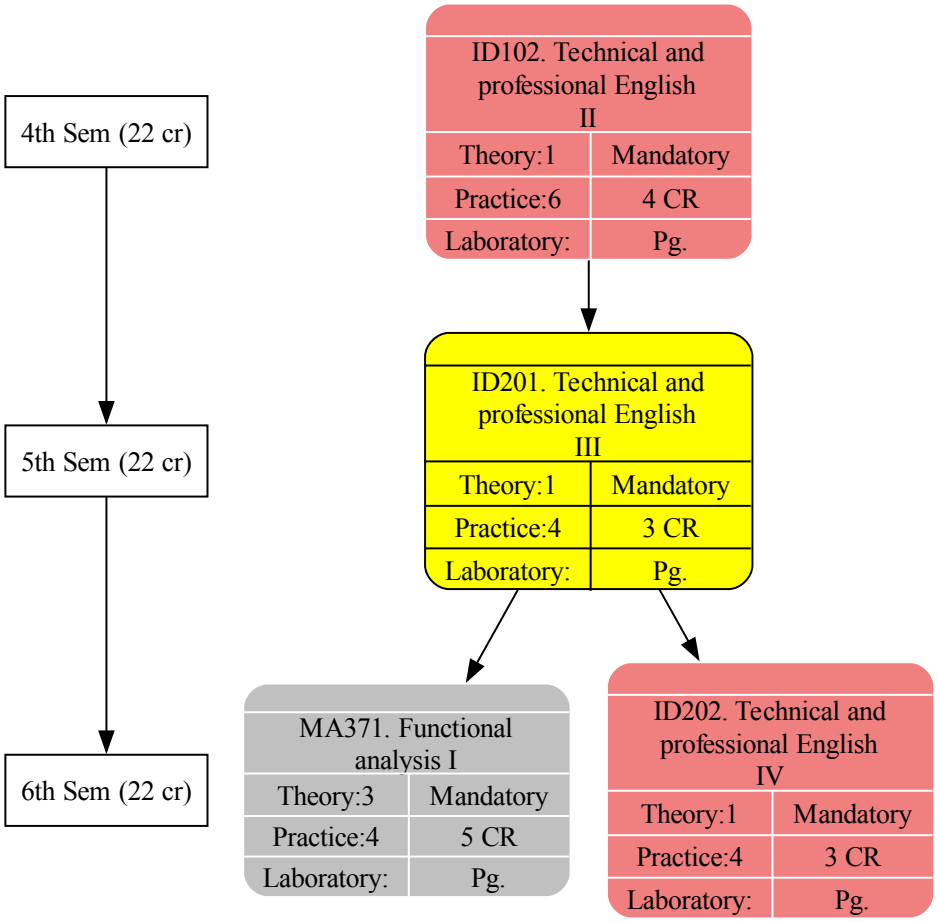


Figure 7.31: Connection Map. ID201 Technical and professional English III

2. Course Rationale

A fundamental part of the integral formation of a professional is the ability to communicate in a foreign language in addition to the native language itself. It not only broadens its cultural horizon but also allows a more humane and comprehensive view of life. In the case of foreign languages, undoubtedly English is the most practical because it is spoken around the world. There is no country where it is not spoken. In careers related to tourist services, English is perhaps the most important practical tool that the student must master from the outset as part of his / her integral education

3. Goals of the Course

- Train the student to understand and hold a conversation.
- Provide techniques of llation of ideas .

4. General Competences (*Outcomes*)

ABET-4) An ability to communicate effectively with a range of audiences. (Usage)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Familiarity)

5. Content

5.1 Getting to know you!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Present, Past, and Future Times. **II:** Interrogative sentences with Wh-. **III:** Words with more than one meaning. **IV:** Parts of the sentence **V:** Expressions for free time

Learning Outcomes **I:** At the end of the first unit, each of the students, understanding the grammar of present, past and future times, is able to express a greater number of actions in the form of sentences. He is also able to express ideas in the form of questions. Assume the idea of words with more than one meaning. Use social expressions in entertainment situations.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.2 The way we live!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Simple present tense. **II:** Present Continuous Time. **III:** Collocations. **IV:** Vocabulary of the countries of the world. **V:** Expressions of anger. **VI:** Connectors.

Learning Outcomes **I:** Ath the end of the second unit, students having identified the present form of expression recognize the difference between the forms of the same and apply it properly. They describe the countries accurately. They take expressions to show interest. Use connectors to join various ideas.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.3 It all went wrong!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Past simple tense. **II:** Continuous past tense. **III:** Irregular Verbs. **IV:** Time expressions. **V:** Connectors of time.

Learning Outcomes **I:** At the end of the third unit, students having recognized the characteristics of past times use them properly. They use prefixes and suffixes to create and recognize new words. They describe time in a broad way. They will use conjunctions to unite type ideas.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.4 Let's go shopping!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Expressions of Indefinite Quantity. **II:** Affirmative sentences, Negatives and Questions. **III:** Use of Articles. **IV:** Product prices. **V:** Filling of formats and surveys **VI:** Expressions for shopping

Learning Outcomes **I:** At the end of the fourth unit, students having identified the idea of quantity express different situations that involve it. Recognize and apply articles to nouns. They assume the idea of shopping with the help of expressions. They express money prices and ideas. They fill several formats. They express attitudes.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.5 What do you want to do?, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Verbal Patterns I. **II:** Future Intentions. **III:** Verbs of Perception. **IV:** Vocabulary of feelings. **V:** Expressions of Plans and Ambitions.

Learning Outcomes **I:** At the end of the fifth unit, students, from the understanding of the idea of verbal patterns, will elaborate sentences using the necessary elements. They will also assimilate the need to express future intentions. They will acquire vocabulary to describe feelings. Expressions will be presented to describe plans and ambitions.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.6 The best in the world!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** What's it like?. **II:** Adjectives **III:** Comparative and superlative. **IV:** Synonyms and antonyms. **V:** Indications of direction . **VI:** Readings.

Learning Outcomes **I:** At the end of the sixth unit, students having known the fundamentals of using adjectives, structure sentences with different forms of adjectives in appropriate contexts. They emphasize the difference between types of cities and towns and lifestyles. They use expressions indicating directions.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.7 Fame!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Present Perfect and Simple Past **II:** Expressions for, ever, since **III:** Adverbs **IV:** Expressions that come in pairs **V:** Short answers **VI:** Celebrities

Learning Outcomes **I:** At the end of the seventh unit, students have learned the fundamentals of structuring the present perfect time and differentiate it from the simple past. They emphasize the difference between forms of adjectives. Describe ideas of music. They use expressions to give short answers. They assume the idea of giving extra explanations of the elements of a sentence.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

6. Bibliography

[Cambridge, 2006] Cambridge (2006). *Diccionario Inglés-Espanol Cambridge*. Editorial Oxford.

[MacGrew, 1999] MacGrew, J. (1999). *Focus on Grammar Basic*. Editorial Oxford.

[Soars and John, 2002d] Soars, L. and John (2002d). *American Headway N 2 Student Book*. Editorial Oxford.

Sixth Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA212 Ordinary Differential Equations (Mandatory)

1. General information

- **Semester**: 6th Sem. **Credits**: 6
- **Hour of this course**: **Theory**: 4 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - CS112 Objects-oriented programming I (2nd Sem-Pg. 328)
  - MA121 Linear Algebra I (3rd Sem-Pg. 358)
  - MA222 Real Analysis I (5th Sem-Pg. 396)

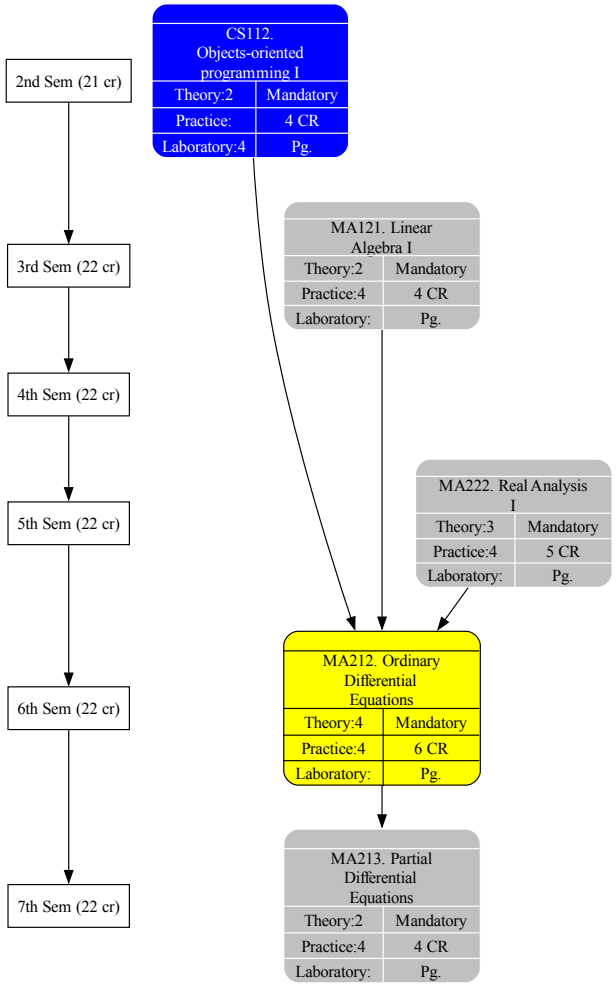


Figure 7.32: Connection Map. MA212 Ordinary Differential Equations

2. Course Rationale

The Ordinary Differential Equations course provides the mathematical tools to analyze systems that evolve with respect to a single independent variable. It focuses on the modeling of dynamic

systems through first-order equations, higher-order equations, and systems of equations, using both analytical and numerical methods. It is fundamental for the understanding of mechanics, electrical circuits, and population models.

3. Goals of the Course

- Solve first-order and higher-order ordinary differential equations using analytical methods.
- Apply the Laplace transform to solve initial value problems involving discontinuous functions.
- Analyze the stability of linear and nonlinear systems of differential equations.
- Implement numerical methods for approximating solutions in complex models.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 First-Order Ordinary Differential Equations, 20 hr(s)

Competences: ABET-1

Topics: **I:** Modeling with first-order equations: growth, mixtures, and cooling

Learning Outcomes **I:** Classify first-order ODEs and select an appropriate solution method (separation of variables, linear equation, exact equation, or integrating factor) [Familiarity] **II:** Formulate and solve first-order models for growth and decay, mixing problems, Newton's law of cooling, and Torricelli's law [Usage] **III:** Analyze autonomous first-order ODEs qualitatively using direction fields, and identify equilibrium points and their stability [Assessment]

Bibliography: [Boyce et al., 2017, Zill, 2018]

5.2 Higher-Order Differential Equations and Systems, 40 hr(s)

Competences: ABET-1

Topics: **I:** Definition and properties of the Laplace Transform; inverse transforms and translation theorems **II:** Derivatives of a transform and convolution integrals

Learning Outcomes **I:** State the Picard-Lindelöf existence and uniqueness theorem and interpret its implications for the continuation of solutions of a higher-order ODE [Familiarity] **II:** Solve linear systems of ODEs using eigenvalue methods and sketch their phase portraits [Usage] **III:** Apply the Laplace transform to solve ODEs with piecewise and impulsive forcing functions [Assessment]

Bibliography: [Boyce et al., 2017, Zill, 2018]

5.3 Numerical Methods for Differential Equations, 36 hr(s)

Competences: ABET-1

Topics: **I:** Phase portraits of 2D systems: equilibria, stability, and limit cycles **II:** Stability of critical points and nonlinear systems

Learning Outcomes **I:** Compare explicit and implicit ODE solvers and explain their relative stability for stiff problems [Familiarity] **II:** Implement a Runge-Kutta method with adaptive step control and apply it to a system of ODEs [Usage] **III:** Analyze the stability and convergence of a finite difference scheme for a parabolic PDE using the Lax equivalence theorem [Assessment]

Bibliography: [Boyce et al., 2017, Zill, 2018]

6. Bibliography

[Boyce et al., 2017] Boyce, W. E., DiPrima, R. C., and Meade, D. B. (2017). *Ecuaciones Diferenciales y Problemas con Valores en la Frontera*. Limusa Wiley, 11th edition.

[Zill, 2018] Zill, D. G. (2018). *Ecuaciones Diferenciales con Aplicaciones de Modelado*. Cengage Learning, 11th edition.

[Boyce et al., 2017] Boyce, W. E., DiPrima, R. C., and Meade, D. B. (2017). *Ecuaciones Diferenciales y Problemas con Valores en la Frontera*. Limusa Wiley, 11th edition.

[Zill, 2018] Zill, D. G. (2018). *Ecuaciones Diferenciales con Aplicaciones de Modelado*. Cengage Learning, 11th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA325 Numerical Analysis II (Mandatory)

1. General information

- **Semester**: 6th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA221 Numerical Analysis I (5th Sem-Pg. 393)

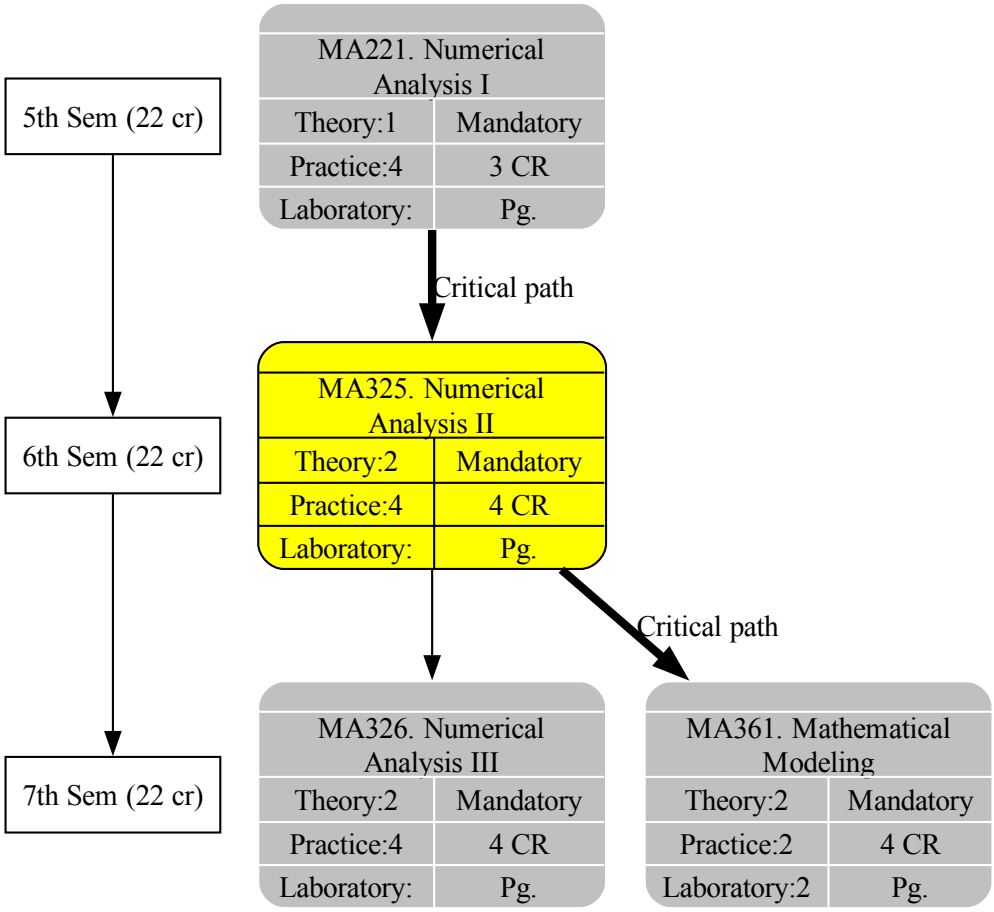


Figure 7.33: Connection Map. MA325 Numerical Analysis II

2. Course Rationale

This course advances the study of numerical methods for advanced mathematical problems, focusing on differential equations, numerical integration, and eigenvalue problems. It develops skills for implementing and analyzing sophisticated numerical algorithms with applications in engineering and computational sciences, with emphasis on stability and computational efficiency.

3. Goals of the Course

- Solve ordinary and partial differential equations numerically

- Apply advanced numerical integration and differentiation methods
- Analyze eigenvalue problems and matrix decompositions
- Evaluate stability and convergence of advanced numerical methods

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Numerical Methods for Ordinary Differential Equations (ODEs), 20 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Euler's method and its variants (forward, backward) **II:** Runge-Kutta methods **III:** Multistep methods (Adams-Bashforth, Adams-Moulton) **IV:** Stability and convergence analysis for ODE solvers **V:** Systems of ODEs and higher-order ODEs

Learning Outcomes **I:** Implement basic ODE solvers (Euler, Runge-Kutta) [Usage] **II:** Analyze the stability and convergence properties of ODE numerical methods [Assessment] **III:** Select an appropriate numerical method for a given ODE based on stability and accuracy requirements [Assessment] **IV:** Solve systems of ODEs and convert higher-order ODEs to first-order systems [Usage]

Bibliography: [Burden et al., 2016, Chapra and Canale, 2015]

5.2 Numerical Differentiation and Integration, 20 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Finite difference approximations (forward, backward, central) **II:** Richardson extrapolation **III:** Newton-Cotes formulas (Trapezoidal rule, Simpson's rule) **IV:** Gaussian quadrature **V:** Error analysis for numerical integration

Learning Outcomes **I:** Compute derivatives of functions using finite difference schemes [Usage] **II:** Apply Richardson extrapolation to improve the accuracy of numerical approximations [Usage] **III:** Implement numerical integration rules (Trapezoidal, Simpson, Gaussian) [Usage] **IV:** Compare the accuracy and efficiency of different numerical integration methods [Assessment] **V:** Analyze the error for numerical differentiation and integration formulas [Assessment]

Bibliography: [Burden et al., 2016, Chapra and Canale, 2015]

5.3 Numerical Linear Algebra, 20 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** LU decomposition and its applications **II:** QR decomposition and least squares problems **III:** Singular Value Decomposition (SVD) and its applications **IV:** Condition number and numerical stability of linear systems **V:** Iterative methods for linear systems (Jacobi, Gauss-Seidel, Conjugate Gradient)

Learning Outcomes **I:** Implement LU and QR decomposition algorithms for solving linear systems [Usage] **II:** Explain the geometric and algebraic meaning of the Singular Value Decomposition [Familiarity] **III:** Use SVD for applications such as matrix approximation and principal component analysis [Usage] **IV:** Analyze the stability of a linear system using its condition number

[Assessment] **V:** Select and implement an appropriate iterative method for solving large, sparse linear systems [Assessment]

Bibliography: [Burden et al., 2016, Chapra and Canale, 2015]

5.4 Interpolation and Function Approximation, 20 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Polynomial interpolation (Lagrange, Newton forms) **II:** Spline interpolation (cubic splines) **III:** Least squares approximation **IV:** Runge's phenomenon and Chebyshev nodes **V:** Error analysis for interpolation

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Burden et al., 2016, Chapra and Canale, 2015]

6. Bibliography

[Burden et al., 2016] Burden, R. L., Faires, J. D., and Burden, A. M. (2016). *Análisis Numérico*. Cengage Learning, 10th edition.

[Chapra and Canale, 2015] Chapra, S. C. and Canale, R. P. (2015). *Métodos Numéricos para Ingenieros*. McGraw-Hill, 7th edition.

[Burden et al., 2016] Burden, R. L., Faires, J. D., and Burden, A. M. (2016). *Análisis Numérico*. Cengage Learning, 10th edition.

[Chapra and Canale, 2015] Chapra, S. C. and Canale, R. P. (2015). *Métodos Numéricos para Ingenieros*. McGraw-Hill, 7th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA351 Statistics and Probabilities (Mandatory)

1. General information

- **Semester**: 6th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA251 Introduction to Statistics (5th Sem-Pg. 399)
  - CS112 Objects-oriented programming I (2nd Sem-Pg. 328)

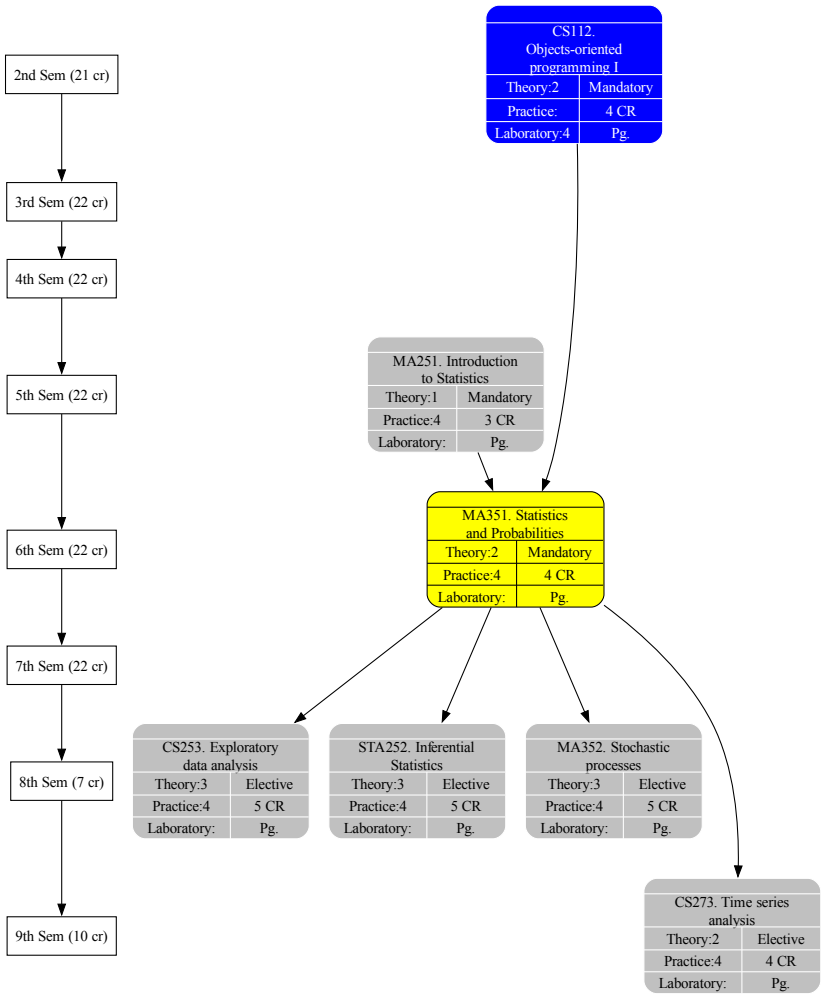


Figure 7.34: Connection Map. MA351 Statistics and Probabilities

2. Course Rationale

Probability and statistics provide the mathematical foundation for reasoning under uncertainty, analyzing data, and building predictive models – skills that are essential across computing, artificial intelligence, computational mathematics, and engineering alike. This course develops a rigorous understanding of probability theory and its use in describing random phenomena, together with the fundamentals of descriptive statistics, statistical inference (estimation, hypothesis testing, and Bayesian

reasoning), and regression modeling, so that students can summarize and analyze data, quantify uncertainty, and draw sound quantitative conclusions in their own field of application.

3. Goals of the Course

- Understand the fundamentals of probability theory and descriptive statistics.
- Apply probability distributions and statistical inference methods to analyze data and quantify uncertainty.
- Develop skills to model random phenomena and build predictive statistical models such as linear regression.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Familiarity)

5. Content

5.1 Numerical Summaries: Location, Spread, and Shape, 4 hr(s)

Competences: ABET-1

Topics: **I:** Measures of location: mean, median, mode, trimmed mean, and weighted mean **II:** Measures of spread: variance, standard deviation, IQR, MAD, and range **III:** Measures of shape: skewness and kurtosis; quantiles and percentiles **IV:** Robust summary statistics: resistance to outliers and heavy-tailed distributions

Learning Outcomes **I:** Calculate and interpret measures of location, spread, and shape for a univariate dataset [Usage] **II:** Select appropriate summary statistics based on the distributional properties of the data [Usage] **III:** Compare classical and robust summary statistics and justify the choice for a given dataset [Assessment]

Bibliography: [Devore, 2016]

5.2 Sample Spaces and Events, 4 hr(s)

Competences: ABET-1

Topics: **I:** Random experiments, outcomes, and sample spaces: finite, countable, and uncountable **II:** Set-theoretic operations on events: union, intersection, complement, and De Morgan's laws **III:** Sigma-algebras and measurable spaces as the foundation for rigorous probability **IV:** Borel sets and the Borel sigma-algebra on the real line

Learning Outcomes **I:** Define sample spaces, events, and sigma-algebras for a given random experiment [Familiarity] **II:** Apply set-theoretic operations to compute probabilities of compound events [Usage] **III:** Construct appropriate sigma-algebras for measurable spaces arising in statistical problems [Assessment]

Bibliography: [Ross, 2014, Devore, 2016]

5.3 Probability Axioms and Properties, 4 hr(s)

Competences: ABET-1

Topics: **I:** Kolmogorov's three axioms of probability and their consequences **II:** Derived properties: complement rule, addition rule, monotonicity, and continuity of probability **III:** Classical, frequentist, and subjective (Bayesian) interpretations of probability **IV:** Counting methods: permutations, combinations, and the multiplication principle

Learning Outcomes **I:** State Kolmogorov's axioms and derive fundamental probability properties [Familiarity] **II:** Calculate probabilities of events using the addition rule and complement rule [Usage] **III:** Apply counting methods to compute probabilities in finite sample spaces [Usage]

Bibliography: [Ross, 2014, Devore, 2016]

5.4 Conditional Probability and Independence, 4 hr(s)

Competences: ABET-1

Topics: **I:** Definition and interpretation of conditional probability P(A|B) and its properties **II:** Bayes' theorem and the law of total probability: derivation and applications **III:** Independence of events: pairwise and mutual independence **IV:** Conditional independence and its role in probabilistic graphical models

Learning Outcomes **I:** Explain conditional probability and independence and their relationship [Familiarity] **II:** Apply Bayes' theorem and the law of total probability to solve probability problems [Usage] **III:** Analyze whether events are independent or conditionally independent in a given probability model [Assessment]

Bibliography: [Ross, 2014, Devore, 2016]

5.5 Discrete Random Variables and Distributions, 4 hr(s)

Competences: ABET-1

Topics: **I:** Probability mass functions (PMF) and cumulative distribution functions (CDF) for discrete random variables **II:** Key discrete distributions: Bernoulli, Binomial, Geometric, Negative Binomial, Poisson, Hypergeometric **III:** Properties of discrete distributions: mean, variance, skewness, and kurtosis **IV:** Poisson approximation to the Binomial and limiting distribution arguments

Learning Outcomes **I:** Identify and describe the key discrete probability distributions and their parameters [Familiarity] **II:** Calculate probabilities, means, and variances using PMFs of standard discrete distributions [Usage] **III:** Select an appropriate discrete distribution to model a given random phenomenon and justify the choice [Assessment]

Bibliography: [Ross, 2014, Devore, 2016]

5.6 Continuous Random Variables and Distributions, 4 hr(s)

Competences: ABET-1

Topics: **I:** Probability density functions (PDF) and CDFs for continuous random variables **II:** Key continuous distributions: Uniform, Normal, Exponential, Gamma, Beta, Chi-squared, t, F **III:** The Normal distribution: properties, standard normal, and its central role in statistical inference **IV:** The exponential family of distributions: canonical form and sufficient statistics

Learning Outcomes **I:** Describe the properties and parameters of key continuous probability distributions [Familiarity] **II:** Calculate probabilities and quantiles for standard continuous distributions [Usage] **III:** Identify distributions belonging to the exponential family and derive their sufficient statistics [Assessment]

Bibliography: [Ross, 2014, Devore, 2016]

5.7 Joint, Marginal, and Conditional Distributions, 4 hr(s)

Competences: ABET-1

Topics: **I:** Joint PMFs and PDFs for two or more random variables **II:** Marginal and conditional distributions derived from joint distributions **III:** Covariance, correlation, and their properties; independence implies zero covariance (converse not always true) **IV:** The bivariate normal distribution and its conditional distributions

Learning Outcomes **I:** Derive marginal and conditional distributions from a given joint distribution [Usage] **II:** Calculate covariance and correlation between two random variables [Usage] **III:** Analyze the conditional structure of the bivariate normal distribution [Assessment]

Bibliography: [Ross, 2014, Devore, 2016]

5.8 Limit Theorems: LLN and CLT, 4 hr(s)

Competences: ABET-1

Topics: **I:** Weak law of large numbers: convergence in probability and its implications for estimation **II:** Central limit theorem: statement, conditions, and role in justifying normal approximations **III:** Applications of the CLT: normal approximation to the binomial and sampling distributions **IV:** Strong law of large numbers: almost sure convergence **V:** The delta method: asymptotic distribution of smooth functions of estimators

Learning Outcomes **I:** State the LLN and CLT and explain the conditions under which they hold [Familiarity] **II:** Apply the CLT to derive normal approximations and sampling distributions [Usage] **III:** Apply the delta method to derive asymptotic distributions of nonlinear estimators [Assessment]

Bibliography: [Ross, 2014, Devore, 2016]

5.9 Point Estimation, 4 hr(s)

Competences: ABET-1

Topics: **I:** Desirable properties: unbiasedness, consistency, efficiency, and sufficiency **II:** Method of moments estimation: derivation and asymptotic properties **III:** Maximum likelihood estimation: derivation, properties, and invariance principle **IV:** Cramer-Rao lower bound and the concept of Fisher information

Learning Outcomes **I:** Describe the key properties of estimators and explain the trade-offs between them [Familiarity] **II:** Derive method of moments and MLE estimators for standard distributions [Usage] **III:** Evaluate the efficiency of an estimator using the Cramer-Rao lower bound [Assessment]

Bibliography: [Devore, 2016]

5.10 Interval Estimation and Confidence Intervals, 4 hr(s)

Competences: ABET-1

Topics: **I:** Constructing confidence intervals using pivotal quantities and exact distributions **II:** Correct interpretation of confidence intervals: coverage probability and common misconceptions **III:** Large-sample confidence intervals based on asymptotic normality **IV:** Duality between confidence intervals and hypothesis tests

Learning Outcomes **I:** Construct confidence intervals for means, proportions, and variances using exact and large-sample methods [Usage] **II:** Interpret confidence intervals correctly and identify common misinterpretations [Usage] **III:** Explain the duality between confidence intervals and hypothesis tests and use it to draw inferences [Assessment]

Bibliography: [Devore, 2016]

5.11 Hypothesis Testing Foundations, 4 hr(s)

Competences: ABET-1

Topics: **I:** Neyman-Pearson framework: null/alternative hypotheses, rejection regions, Type I and II errors **II:** p-values: definition, correct interpretation, and common misuse; power and sample size **III:** Common parametric tests: z-test, t-test, F-test, chi-squared test **IV:** Neyman-Pearson lemma and uniformly most powerful tests

Learning Outcomes **I:** Explain the logic of hypothesis testing, Type I and II errors, and the meaning of a p-value [Familiarity] **II:** Apply standard parametric tests and interpret the results in context [Usage] **III:** Derive the most powerful test for a simple hypothesis using the Neyman-Pearson lemma [Assessment]

Bibliography: [Devore, 2016]

5.12 Bayesian Inference, 4 hr(s)

Competences: ABET-1

Topics: **I:** Bayesian framework: prior, likelihood, posterior, and the role of Bayes' theorem **II:** Prior distributions: informative, non-informative, and conjugate priors **III:** Posterior inference: point estimates, credible intervals, and posterior predictive distributions **IV:** Bayesian model comparison: Bayes factors and posterior model probabilities

Learning Outcomes **I:** Explain the Bayesian approach to inference and contrast it with the frequentist approach [Familiarity] **II:** Derive the posterior distribution for standard conjugate models [Usage] **III:** Apply Bayes factors to compare statistical models and interpret the evidence [Assessment]

Bibliography: [Ross, 2014, Devore, 2016]

5.13 Simple Linear Regression, 4 hr(s)

Competences: ABET-1

Topics: **I:** Ordinary least squares estimation: derivation of coefficients and their properties **II:** Inference for regression coefficients: t-tests, confidence intervals, and p-values **III:** Coefficient of determination R-squared and analysis of variance decomposition **IV:** Gauss-Markov theorem: BLUE property of OLS under classical assumptions

Learning Outcomes **I:** Fit a simple linear regression model and interpret the estimated coefficients [Usage] **II:** Conduct hypothesis tests and construct confidence intervals for regression parameters [Usage] **III:** Verify the Gauss-Markov assumptions and assess their consequences when violated [Assessment]

Bibliography: [Devore, 2016]

6. Bibliography

[Devore, 2016] Devore, J. L. (2016). *Probability and Statistics for Engineering and the Sciences.* Cengage Learning.

[Ross, 2014] Ross, S. M. (2014). *A First Course in Probability.* Pearson.

[Devore, 2016] Devore, J. L. (2016). *Probability and Statistics for Engineering and the Sciences.* Cengage Learning.

[Ross, 2014] Ross, S. M. (2014). *A First Course in Probability.* Pearson.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA371 Functional analysis I (Mandatory)

1. General information

- **Semester**: 6th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA222 Real Analysis I (5th Sem-Pg. 396)
  - ID201 Technical and professional English III (5th Sem-Pg. 405)
  - MA122 Linear Algebra II (4th Sem-Pg. 377)

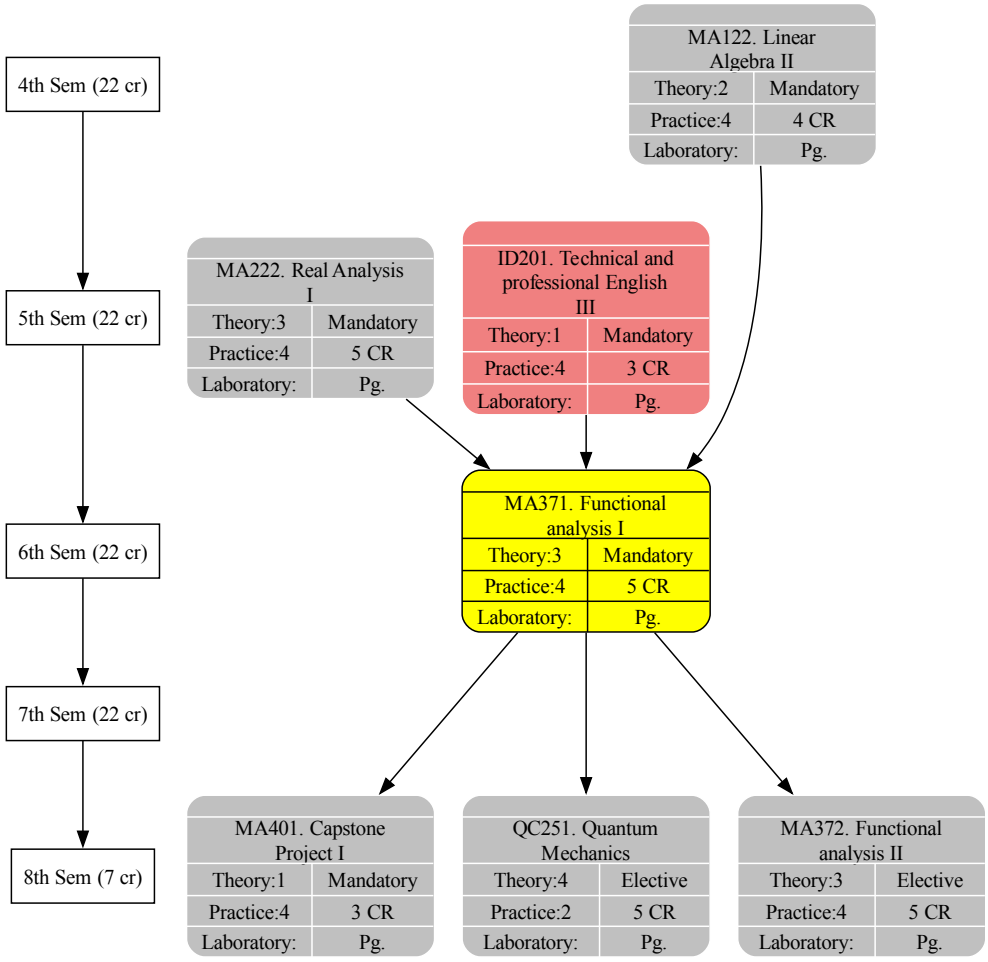


Figure 7.35: Connection Map. MA371 Functional analysis I

2. Course Rationale

Functional analysis is a central pillar of modern mathematics, providing the framework to solve problems in differential equations, quantum mechanics, and numerical analysis. This course bridges the gap between linear algebra and infinite-dimensional spaces. By introducing Measure Theory as a

rigorous foundation for integration, the course enables the study of Banach and Hilbert spaces, linear operators, and the fundamental theorems that govern functional systems.

3. Goals of the Course

- Establish a rigorous understanding of Lebesgue measure and integration theory.
- Master the topological and algebraic structures of Banach and Hilbert spaces.
- Apply the fundamental theorems of functional analysis (Hahn-Banach, Open Mapping, Closed Graph).
- Analyze the properties of linear operators and their spectral theory in infinite dimensions.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Measure and Integration, 16 hr(s)

Competences: ABET-1

Topics: **I:** σ-algebras and measurable spaces **II:** Lebesgue measure on $\mathbb{R}^n$ **III:** Measurable functions and Lebesgue integration **IV:** Convergence theorems (Monotone, Dominated, Fatou's Lemma) **V:** L^p spaces: completeness and duality

Learning Outcomes **I:** Construct the Lebesgue measure and identify measurable sets [Assessment] **II:** Evaluate integrals using convergence theorems [Usage] **III:** Prove the completeness of L^p spaces [Familiarity]

Bibliography: [Folland, 1999, Royden and Fitzpatrick, 2010]

5.2 Normed and Banach Spaces, 16 hr(s)

Competences: ABET-1

Topics: **I:** Normed vector spaces and completeness **II:** Schauder bases and infinite dimensionality **III:** Linear operators and functionals **IV:** Dual spaces and the Hahn-Banach theorem **V:** Category theorems: Baire, Open Mapping, and Closed Graph

Learning Outcomes **I:** Verify the completeness of various functional spaces [Assessment] **II:** Apply the Hahn-Banach theorem to extend linear functionals [Usage] **III:** Utilize the Open Mapping theorem to determine operator boundedness [Assessment]

Bibliography: [Kreyszig, 1989, Brezis, 2011]

5.3 Hilbert Spaces, 16 hr(s)

Competences: ABET-1

Topics: **I:** Inner product spaces and Hilbert structure **II:** Orthogonality, projection theorem, and Riesz representation **III:** Orthonormal sets and Fourier series in Hilbert spaces **IV:** Self-adjoint, unitary, and normal operators

Learning Outcomes **I:** Construct orthogonal projections in Hilbert spaces [Usage] **II:** Represent linear functionals using the Riesz theorem [Assessment] **III:** Analyze the geometry of infinite-dimensional inner product spaces [Familiarity]

Bibliography: [Kreyszig, 1989, Conway, 1990]

5.4 Operator Theory and Spectral Theory, 16 hr(s)

Competences: ABET-1

Topics: **I:** Compact operators on Banach and Hilbert spaces **II:** The Fredholm alternative **III:** Spectral theory of bounded linear operators **IV:** Spectral theorem for compact self-adjoint operators

Learning Outcomes **I:** Identify compact operators and their properties [Assessment] **II:** Determine the spectrum of linear operators [Usage] **III:** Apply the spectral theorem to decompose operators [Assessment]

Bibliography: [Brezis, 2011, Yosida, 1980]

6. Bibliography

[Brezis, 2011] Brezis, H. (2011). *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Springer Science & Business Media.

[Conway, 1990] Conway, J. B. (1990). *A Course in Functional Analysis*. Springer-Verlag, 2nd edition.

[Folland, 1999] Folland, G. B. (1999). *Real Analysis: Modern Techniques and Their Applications*. Wiley-Interscience, 2nd edition.

[Kreyszig, 1989] Kreyszig, E. (1989). *Introductory Functional Analysis with Applications*. Wiley.

[Royden and Fitzpatrick, 2010] Royden, H. and Fitzpatrick, P. (2010). *Real Analysis*. Pearson, 4th edition.

[Yosida, 1980] Yosida, K. (1980). *Functional Analysis*. Springer-Verlag, 6th edition.

[Brezis, 2011] Brezis, H. (2011). *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Springer Science & Business Media.

[Conway, 1990] Conway, J. B. (1990). *A Course in Functional Analysis*. Springer-Verlag, 2nd edition.

[Folland, 1999] Folland, G. B. (1999). *Real Analysis: Modern Techniques and Their Applications*. Wiley-Interscience, 2nd edition.

[Kreyszig, 1989] Kreyszig, E. (1989). *Introductory Functional Analysis with Applications*. Wiley.

[Royden and Fitzpatrick, 2010] Royden, H. and Fitzpatrick, P. (2010). *Real Analysis*. Pearson, 4th edition.

[Yosida, 1980] Yosida, K. (1980). *Functional Analysis*. Springer-Verlag, 6th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

ID202 Technical and professional English IV (Mandatory)

1. General information

- **Semester**: 6th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - ID201 Technical and professional English III (5th Sem-Pg. 405)

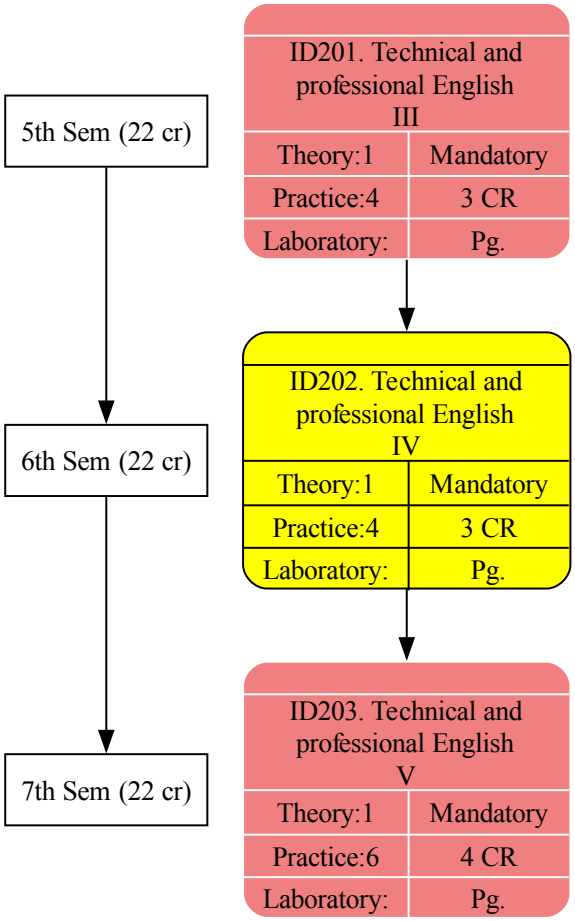


Figure 7.36: Connection Map. ID202 Technical and professional English IV

2. Course Rationale

A fundamental part of the integral formation of a professional is the ability to communicate in a foreign language in addition to the native language itself. It not only broadens its cultural horizon but also allows a more humane and comprehensive view of life. In the case of foreign languages, English is undoubtedly the most practical because it is spoken around all the world. There is no country where it is not spoken. In addition to being vital to your professional career

3. Goals of the Course

- Increase the level of conversation in different subjects, in the students. As well as the ability to write and read documentation of all kinds.
- Bring the student to a more intense expression in the language domain.

4. General Competences (*Outcomes*)

ABET-4) An ability to communicate effectively with a range of audiences. (Usage)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Familiarity)

5. Content

5.1 Do and don't!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Mode Auxiliaries should, must and have got to. **II:** Affirmative, negative and interrogative sentences with modals. **III:** Terms for formal letters. **IV:** Parts of short answers. **V:** Expressions for occupations.

Learning Outcomes **I:** At the end of the eighth unit, each of the students, understanding the grammar of should and must auxiliaries, is able to express a greater number of actions in an obligatory and suggestive way. Also be able to express ideas describing occupations. Assumes the need to write formal letters

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.2 Going places!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Present and Future Present Time with Will **II:** First conditional **III:** Collocations **IV:** Vocabulary of prepositions of place and time **V:** Expressions of connection of ideas

Learning Outcomes **I:** At the end of the ninth unit, students having identified how to express present recognize the difference between future forms and apply them properly. They describe conditions accurately. They assume expressions to show place location. They use expressions of time and connectors to unite several ideas.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.3 Scared to death!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Infinitive and gerund verb patterns **II:** What + Infinitive **III:** Something + infinitive **IV:** Expressions of feelings **V:** Exclamations of surprise

Learning Outcomes **I:** At the end of the tenth unit of students, the chapters recognize and use the patterns of times in the past properly. They use exclamation marks. And describe feelings. They will use conjunctions to unite type ideas.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.4 Things that changed the world!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Passive Voice Â Â Â Â Â Â **II:** Affirmative Prayers, Negatives and Questions Â Â Â Â Â Â **III:** Use of participles, verbs and nouns that go together Â Â Â Â Â Â **IV:** Signals. Signs and notes Â Â Â Â Â Â **V:** Summaries Â Â Â Â Â Â **VI:** Expressions to indicate prohibition

Learning Outcomes **I:** At the end of the eleventh unit ,the students having identified the idea of passive actions describe actions appropriately in diverse situations that involve it. They recognize and apply participations. They assume the idea of respecting public signs and signals. They express ideas of habits. They make summaries.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.5 Dreams and reality!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Second Conditional Â Â Â Â Â Â **II:** Auxiliar of mode "might" Â Â Â Â Â Â **III:** Phrase Verbs Â Â Â Â Â Â **IV:** Social expressions vocabulary Â Â Â Â Â Â **V:** Adverbs Â Â Â Â Â Â **VI:** Expressions to give advice

Learning Outcomes **I:** At the end of the twelfth unit, students, starting from understanding the idea of Conditionals and expressing the possibility of elaborating sentences using the necessary elements. They will also assimilate the need for verbal phrases (2 word verbs). They will acquire vocabulary to describe social expressions.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.6 Making a living!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Present Perfect Continuous Â Â Â Â Â Â **II:** Present Continuous Â Â Â Â Â Â **III:** Occupations Â Â Â Â Â Â **IV:** Word formation Â Â Â Â Â Â **V:** Adverbs Â Â Â Â Â Â **VI:** Expressions of use on the phone

Learning Outcomes **I:** At the end of the thirteenth unit, they structure sentences with actions that include present and past in appropriate contexts. They emphasize the difference between types of occupations. Use appropriate expressions for telephone conversations.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

5.7 All you need is love!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Past Perfect and Past Simple Â Â Â Â Â Â **II:** Report Expressions Â Â Â Â Â Â **III:** Expressions of words in different contexts Â Â Â Â Â Â **IV:** Short and formal farewells Â Â Â Â Â Â **V:** Love Stories

Learning Outcomes **I:** At the end of the fourteenth unit, students having learned the fundamentals of structuring past perfect time, differentiate it from the simple past. They emphasize the difference between words in different contexts. Describe farewell ideas. They use expressions to write love stories. They assume the idea of giving and doing interviews.

Bibliography: [Soars and John, 2002d, Cambridge, 2006, MacGrew, 1999]

6. Bibliography

[Cambridge, 2006] Cambridge (2006). *Diccionario Inglés-Espanol Cambridge*. Editorial Oxford.

[MacGrew, 1999] MacGrew, J. (1999). *Focus on Grammar Basic*. Editorial Oxford.

[Soars and John, 2002d] Soars, L. and John (2002d). *American Headway N 2 Student Book*. Editorial Oxford.

Seventh Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS261 Artificial Intelligence (Mandatory)

1. General information

- **Semester**: 7th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - MA251 Introduction to Statistics (5th Sem-Pg. 399)

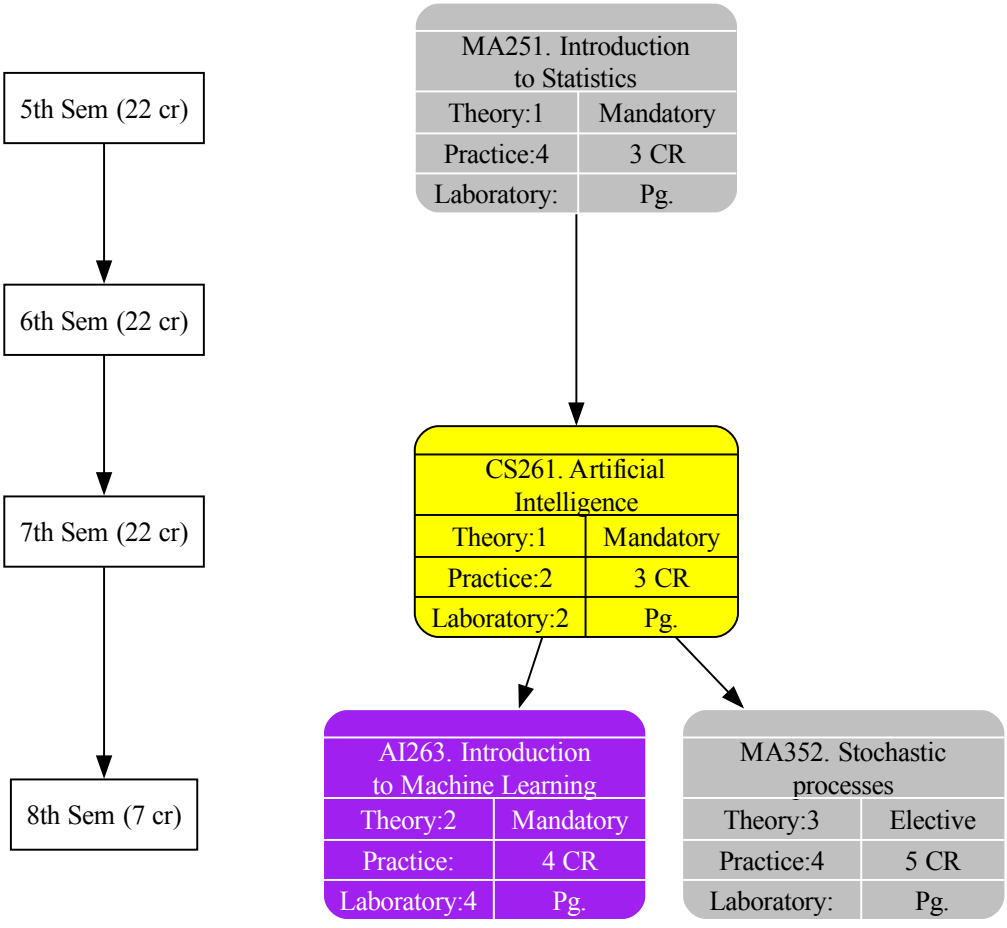


Figure 7.37: Connection Map. CS261 Artificial Intelligence

2. Course Rationale

Artificial Intelligence (AI) is a fundamental field of computer science that seeks to create systems capable of performing tasks that typically require human intelligence. This course provides an introduction to the techniques used to build such systems, including search algorithms, probabilistic

reasoning, and machine learning. Students will learn to model complex problems and implement intelligent agents, while considering the social and ethical implications of these technologies.

3. Goals of the Course

- Understand the fundamental concepts and history of Artificial Intelligence.
- Implement and evaluate search algorithms for problem-solving.
- Master the basics of machine learning, including supervised and unsupervised learning.
- Apply probabilistic reasoning to handle uncertainty in intelligent systems.
- Analyze the ethical challenges and social impact of AI.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Introduction to AI: Concepts and Intelligent Behavior, 4 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Overview of AI problems, Examples of successful recent AI applications **II:** Definitions of agents with examples (e.g., reactive, deliberative) **III:** What is intelligent behavior?: (a) The Turing test and its flaws (b) Multimodal input and output (c) Simulation of intelligent behavior (d) Rational versus non-rational reasoning **IV:** Overview of AI Applications, growth, and impact (economic, societal, ethics) **V:** Philosophical issues **VI:** History of AI

Learning Outcomes **I:** Describe the Turing test and the "Chinese Room" thought experiment [Explain] **II:** Differentiate between optimal reasoning/behavior and human-like reasoning/behavior [Evaluate] **III:** Differentiate the terms: AI, machine learning, and deep learning [Explain]

Bibliography: [Russell and Norvig, 2020, Luger, 2008]

5.2 Problem Characteristics and Agent Nature, 4 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Problem characteristics: (a) Fully versus partially observable (b) Single versus multi-agent (c) Deterministic versus stochastic (d) Static versus dynamic (e) Discrete versus continuous **II:** Nature of agents: (a) Autonomous, semi-autonomous, mixed-initiative autonomy (b) Reflexive, goal-based, and utility-based (c) Decision making under uncertainty and with incomplete information (d) The importance of perception and environmental interactions (e) Learning-based agents (f) Embodied agents: i. sensors, dynamics, effectors **III:** Practice identifying problem characteristics in example environments **IV:** Additional depth on nature of agents with examples **V:** Additional depth on AI Applications, Growth, and Impact (economic, societal, ethics, security)

Learning Outcomes **I:** Enumerate the characteristics of a specific problem [Apply]

Bibliography: [Russell and Norvig, 2020, Luger, 2008]

5.3 State and Uninformed Search, 5 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** State space representation of a problem: (a) Specifying states, goals, and operators (b) Factoring states into representations (hypothesis spaces) (c) Problem solving by graph search: i. e.g., Graphs as a space, and tree traversals as exploration of that space ii. Dynamic construction of the graph (not given upfront) **II:** Uninformed graph search for problem solving: (a) Breadth-first search (b) Depth-first search: i. With iterative deepening (c) Uniform cost search **III:** Space and time complexities of graph search algorithms **IV:** Bidirectional search **V:** Understanding the search space: (a) Constructing search trees (b) Dynamic search spaces (c) Combinatorial explosion of search space (d) Search space topology (e.g., ridges, saddle points, local minima)

Learning Outcomes **I:** Design the state space representation for a puzzle (e.g., N-queens or 3-jug problem) [Apply] **II:** Select and implement an appropriate uninformed search algorithm for a problem (e.g., tic-tac-toe), and characterize its time and space complexities [Apply]

Bibliography: [Russell and Norvig, 2020, Luger, 2008]

5.4 Heuristic Search and CSP, 6 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Heuristic graph search for problem solving: (a) Heuristic construction and admissibility (b) Hill-climbing (c) Local minima and the search landscape: i. Local vs global solutions (d) Greedy best-first search (e) A* search **II:** Beam search **III:** Implementation of A* search **IV:** Constraint satisfaction **V:** Variations on A* (IDA*, SMA*, RBFS)

Learning Outcomes **I:** Select and implement an appropriate informed search algorithm for a problem after designing a helpful heuristic function (e.g., a robot navigating a 2D gridworld) [Apply] **II:** Evaluate whether a heuristic for a given problem is admissible/can guarantee an optimal solution [Evaluate] **III:** Design and implement A*/beam search to solve a problem, and compare it against other search algorithms in terms of the solution cost, number of nodes expanded, etc [Apply] **IV:** Compare and contrast various heuristic searches vis-a-vis applicability to a given problem [Evaluate] **V:** Model a logic or Sudoku puzzle as a constraint satisfaction problem, solve it with backtrack search, and determine how much arc consistency can reduce the search space [Apply]

Bibliography: [Russell and Norvig, 2020, Luger, 2008]

5.5 Adversarial Search and Stochastic Optimization, 5 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Two-player adversarial games: (a) Minimax search (b) Alpha-beta pruning: i. Ply cutoff **II:** Local search **III:** Tabu search **IV:** Two-player adversarial games: (a) The horizon effect (b) Opening playbooks/endgame solutions (c) What it means to ”solve” a game (e.g., checkers) **V:** Implementation of minimax search, beam search **VI:** Expectimax search (MDP-solving) and chance nodes **VII:** Stochastic search: (a) Simulated annealing (b) Genetic algorithms (c) Monte-Carlo tree search

Learning Outcomes **I:** Apply minimax search in a two-player adversarial game (e.g., connect four), using heuristic evaluation at a particular depth to compute the scores to back up [Apply] **II:** Apply minimax search with alpha-beta pruning to prune search space in a two-player adversarial game (e.g., connect four) [Apply] **III:** Compare and contrast genetic algorithms with classic search techniques, explaining when it is most appropriate to use a genetic algorithm to learn a model versus other forms of optimization (e.g., gradient descent) [Evaluate] **IV:** Design and implement a genetic algorithm solution to a problem [Apply] **V:** Design and implement a simulated annealing schedule to avoid local minima in a problem [Apply]

Bibliography: [Russell and Norvig, 2020, Luger, 2008]

5.6 Machine Learning Fundamentals, 7 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Definition and examples of a broad variety of machine learning tasks: (a) Supervised learning: i. Classification ii. Regression (b) Reinforcement learning (c) Unsupervised learning: i. Clustering **II:** Fundamental ideas: (a) No free lunch theorem: no one learner can solve all problems; representational design decisions have consequences. (b) Sources of error and undecidability in machine learning **III:** A simple statistical-based supervised learning such as linear regression or decision trees: (a) Focus on how they work without going into mathematical or optimization details; enough to understand and use existing implementations correctly **IV:** The overfitting problem/controlling solution complexity (regularization, pruning - intuition only): (a) The bias (underfitting) - variance (overfitting) tradeoff

Learning Outcomes **I:** Describe the differences among the three main styles of learning (supervised, reinforcement, and unsupervised) and determine which is appropriate to a particular problem domain [Explain] **II:** Differentiate the terms of AI, machine learning, and deep learning [Evaluate] **III:** Frame an application as a classification problem, including the available input features and output to be predicted (e.g., identifying alphabetic characters from pixel grid input) [Apply] **IV:** Identify overfitting in the context of a problem and learning curves and describe solutions to overfitting [Evaluate] **V:** Explain how machine learning works as an optimization/search process [Explain]

Bibliography: [Russell and Norvig, 2020, Alpaydin, 2020]

5.7 Machine Learning Data, Representations and Evaluation, 7 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Working with Data: (a) Data preprocessing: i. Importance and pitfalls of preprocessing choices (b) Handling missing values (imputing, flag-as-missing): i. Implications of imputing vs flag-as-missing (c) Encoding categorical variables, encoding real-valued data (d) Normalization/standardization (e) Emphasis on real data, not textbook examples **II:** Representations: (a) Hypothesis spaces and complexity (b) Simple basis feature expansion, such as squaring univariate features (c) Learned feature representations **III:** Machine learning evaluation: (a) Separation of train, validation, and test sets (b) Performance metrics for classifiers (c) Estimation of test performance on held-out data (d) Tuning the parameters of a machine learning model with a validation set (e) Importance of understanding what a model is doing, where its pitfalls/shortcomings are, and the implications of its decisions

Learning Outcomes **I:** Apply two or more simple statistical learning algorithms to a classification task and measure the classifiers' accuracy [Apply] **II:** Implement a statistical learning algorithm and the corresponding optimization process to train the classifier and obtain a prediction on new data [Apply] **III:** Explain proper ML evaluation procedures, including the differences between training and testing performance, and what can go wrong with the evaluation process leading to inaccurate reporting of ML performance [Explain] **IV:** Compare two machine learning algorithms on a dataset, implementing the data preprocessing and evaluation methodology (e.g., metrics and handling of train/test splits) from scratch [Apply]

Bibliography: [Russell and Norvig, 2020, Alpaydin, 2020]

5.8 Neural Networks and Ethics, 6 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Basic neural networks: (a) Fundamentals of understanding how neural networks work and their training process, without details of the calculations (b) Basic introduction to generative neural networks (e.g., large language models) **II:** Ethics for Machine Learning: (a) Focus on real data, real scenarios, and case studies (b) Dataset/algorithmic/evaluation bias and unintended consequences

Learning Outcomes I: Describe the neural network training process and resulting learned representations [Explain]

Bibliography: [Russell and Norvig, 2020, Alpaydin, 2020]

5.9 Machine Learning Formulation and Evaluation, 3 hr(s)

Competences: ABET-2

Topics: I: Formulation of simple machine learning as an optimization problem, such as least squares linear regression or logistic regression: (a) Objective function (b) Gradient descent (c) Regularization to avoid overfitting (mathematical formulation) **II:** Ensembles of models: (a) Simple weighted majority combination **III:** Performance evaluation: (a) Other metrics for classification (e.g., error, precision, recall) (b) Performance metrics for regressors (c) Confusion matrix (d) Cross-validation: i. Parameter tuning (grid/random search, via cross-validation)

Learning Outcomes I: Compare and contrast several learning techniques (e.g., decision trees, logistic regression, naive Bayes, neural networks, and belief networks), providing examples of when each strategy is superior [Compare] **II:** Evaluate the performance of a simple learning system on a real-world dataset [Evaluate] **III:** Characterize the state of the art in learning theory, including its achievements and shortcomings [Explain] **IV:** Explain the problem of overfitting, along with techniques for detecting and managing the problem [Explain] **V:** Explain the triple tradeoff among the size of a hypothesis space, the size of the training set, and performance accuracy [Explain]

Bibliography: [Goodfellow et al., 2016]

5.10 Deep Learning, RL and Ethics, 3 hr(s)

Competences: ABET-2

Topics: I: Deep learning: (a) Deep feed-forward networks (intuition only, no mathematics) (b) Convolutional neural networks (intuition only, no mathematics) (c) Visualization of learned feature representations from deep nets (d) Other architectures (generative NN, recurrent NN, transformers, etc.) **II:** Overview of reinforcement learning methods **III:** Two or more applications of machine learning algorithms: (a) E.g., medicine and health, economics, vision, natural language, robotics, game play **IV:** Ethics for Machine Learning: (a) Continued focus on real data, real scenarios, and case studies (b) Privacy (c) Fairness (d) Intellectual property (e) Explainability

Learning Outcomes I: Visualize the training progress of a neural network through learning curves in a well-established toolkit (e.g., TensorBoard) and visualize the learned features of the network [Apply] **II:** Given a real-world application of machine learning, describe ethical issues regarding the choices of data, preprocessing steps, algorithm selection, and visualization/presentation of results [Apply]

Bibliography: [Goodfellow et al., 2016]

5.11 Knowledge Representation and Probabilistic Reasoning Foundations, 6 hr(s)

Competences: ABET-1,ABET-2

Topics: I: Types of representations: (a) Symbolic, logical: i. Creating a representation from a natural language problem statement (b) Learned subsymbolic representations (c) Graphical models (e.g., naive Bayes, Bayesian network) **II:** Review of probabilistic reasoning, Bayes theorem **III:** Bayesian reasoning: (a) Bayesian inference **IV:** Random variables and probability distributions: (a) Axioms of probability (b) Probabilistic inference (c) Bayes' Rule (derivation) (d) Bayesian inference (more complex examples)

Learning Outcomes **I:** Given a natural language problem statement, encode it as a symbolic or logical representation [Apply] **II:** Compute a probabilistic inference in a real-world problem using Bayes' theorem to determine the probability of a hypothesis given evidence [Apply] **III:** Apply Bayes' rule to determine the probability of a hypothesis given evidence [Apply] **IV:** Compute the probability of outcomes and test whether outcomes are independent [Apply]

Bibliography: [Russell and Norvig, 2020]

5.12 Independence, Markov Models and Decision Making, 6 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Independence **II:** Conditional Independence **III:** Markov chains and Markov models **IV:** Utility and decision making

Learning Outcomes **I:** Explain how we can make decisions under uncertainty, using concepts such as Bayes theorem and utility [Explain]

Bibliography: [Russell and Norvig, 2020]

5.13 AI Applications and Societal Impact, 4 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** At least one application of AI to a specific problem and field, such as medicine, health, sustainability, social media, economics, education, robotics, etc. (choose at least one for the CS Core): (a) Formulating and evaluating a specific application as an AI problem: i. How to deal with underspecified or ill-posed problems (b) Data availability/scarcity and cleanliness: i. Basic data cleaning and preprocessing ii. Data set bias (c) Algorithmic bias (d) Evaluation bias (e) Assessment of societal implications of the application **II:** Deployed deep generative models: (a) High-level overview of deep image generative models (e.g., as of 2023, DALL-E, Midjourney, Stable Diffusion, etc.), their uses, and their shortcomings/pitfalls. (b) High-level overview of large language models (e.g., as of 2023, ChatGPT, Bard, etc.), their uses, and their shortcomings/pitfalls. **III:** Overview of societal impact of AI: (a) Ethics (b) Fairness (c) Trust/explainability (d) Privacy and usage of training data (e) Human autonomy and oversight/regulations/legal requirements (f) Sustainability

Learning Outcomes **I:** Given a real-world application domain and problem, formulate an AI solution to it, identifying proper data/input, preprocessing, representations, AI techniques, and evaluation metrics/methodology [Apply]

Bibliography: [Russell and Norvig, 2020, Alpaydin, 2020]

5.14 Additional AI Applications and Societal Impact, 4 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** One or more additional applications of AI to a broad set of problems and diverse fields, such as medicine, health, sustainability, social media, economics, education, robotics, etc. (choose a different area from that chosen for the CS Core): (a) Formulating and evaluating a specific application as an AI problem: i. How to deal with underspecified or ill-posed problems (b) Data availability/scarcity and cleanliness: i. Basic data cleaning and preprocessing ii. Data set bias (c) Algorithmic bias (d) Evaluation bias (e) Assessment of societal implications of the application **II:** Additional depth on deployed deep generative models: (a) Introduction to how deep image generative models work, (e.g., as of 2023, DALL-E, Midjourney, Stable Diffusion) including discussion of attention (b) Introduction to how large language models work, (e.g., as of 2023, ChatGPT, Bard) including discussion of attention (c) Idea of foundational models, how to use them, and the benefits/issues with training them from big data **III:** Analysis and discussion of the societal impact of AI: (a) Ethics (b) Fairness (c) Trust/explainability (d) Privacy

and usage of training data (e) Human autonomy and oversight/regulations/legal requirements (f) Sustainability

Learning Outcomes I: Analyze the societal impact of one or more specific real-world AI applications, identifying issues regarding ethics, fairness, bias, trust, and explainability [Evaluate] **II:** Describe some of the failure modes of current deep generative models for language or images, and how this could affect their use in an application [Describe]

Bibliography: [Russell and Norvig, 2020, Alpaydin, 2020]

6. Bibliography

[Alpaydin, 2020] Alpaydin, E. (2020). *Introduction to Machine Learning*. MIT Press, 4th edition.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Luger, 2008] Luger, G. F. (2008). *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Pearson, 6th edition.

[Russell and Norvig, 2020] Russell, S. and Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition.

[Alpaydin, 2020] Alpaydin, E. (2020). *Introduction to Machine Learning*. MIT Press, 4th edition.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Luger, 2008] Luger, G. F. (2008). *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Pearson, 6th edition.

[Russell and Norvig, 2020] Russell, S. and Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS3P1 Parallel and Distributed Computing (Mandatory)

1. General information

- **Semester**: 7th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - CS210 Algorithms and Data Structures (4th Sem-Pg. 367)

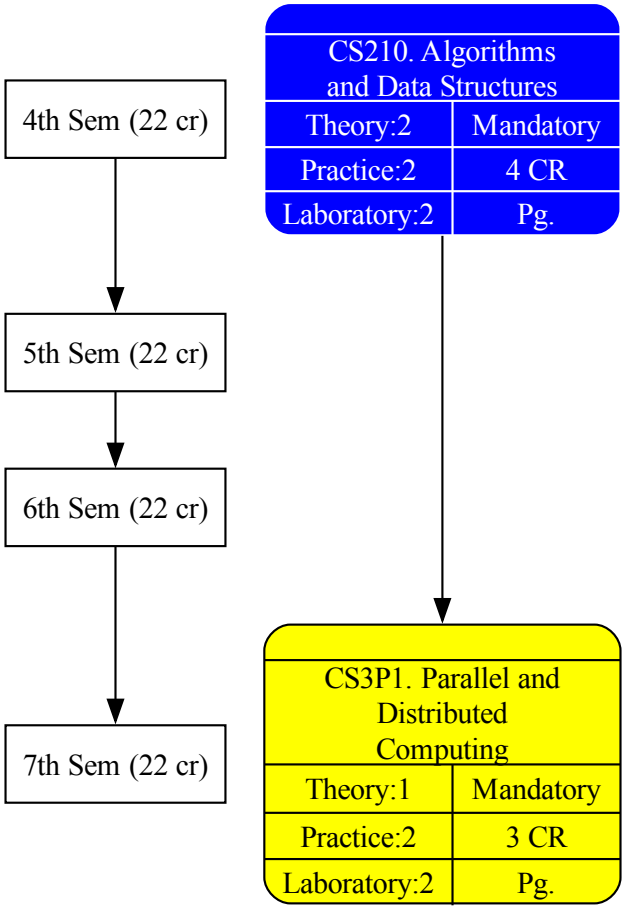


Figure 7.38: Connection Map. CS3P1 Parallel and Distributed Computing

2. Course Rationale

With the end of frequency scaling in single-core processors, parallel and distributed computing has become essential for high-performance software development. This course introduces students to the programming models, communication patterns, and coordination mechanisms required to exploit multi-core systems and distributed clusters. Students will learn to design scalable algorithms and evaluate their performance under different computational constraints.

3. Goals of the Course

- Design and implement parallel programs using shared and distributed memory.
- Master communication and synchronization protocols in distributed systems.
- Evaluate performance and scalability through formal metrics.
- Apply parallel algorithms to solve complex computational problems.

4. General Competences (*Outcomes*)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Programs: Declarative Parallelism, 4 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Parallelism (a) Declarative parallelism: Determining which actions can, or must not, be performed in parallel, at the level of instructions, functions, closures, compound actions, sessions, tasks, and services is the core idea underlying PDC algorithms; failing to do so is the primary source of errors. See also: Algorithms (b) Defining order: for example, using "happens-before" relationships or series-parallel directed acyclic graphs (DAGs) representing programs. (c) Independence: determining when order does not matter, in terms of commutativity, dependencies, preconditions. (d) Guaranteeing order between otherwise parallel actions when necessary, including locks, safe publication; and enforcing communication - sending a message happens before receiving it; and relaxing it when not necessary.

Learning Outcomes I: Graphically represent (as a Directed Acyclic Graph - DAG) how to parallelize a compound numeric expression; for example, $a = (b + c) * (d + e)$ [Design] **II:** Explain why consistency and fault tolerance concepts do not arise in purely sequential programs [Explain]

Bibliography: [Pacheco and Malensek, 2021]

5.2 Programs: Starting Activities, 3 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Starting activities (a) Options allowing actions to be performed (eventually) at locations range from hardwired to configuration scripts; also establishing communication and resource management; these are expressed differently in languages and contexts, typically relying on automated provisioning and management by platforms. See also: Resource Management (b) Procedural: Allowing multiple actions to start at a given program point; for example, starting new threads, possibly delimiting their scope or organizing them into hierarchical groups. (c) Reactive: Enabling upon event occurrence by installing an event handler, with less control over when actions start or finish, and may apply even on single processors. (d) Dependent: Enabling upon completion of others; for example, sequencing sets of parallel actions. See also: Coordination. (e) Granularity: The execution cost of action bodies should outweigh the overhead of organizing them.

Learning Outcomes I: Write a service that creates a thread (or other form of procedural activation) to return a requested web page to each new client [Write]

Bibliography: [Pacheco and Malensek, 2021]

5.3 Programs: Execution Properties, 3 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Execution Properties (a) Non-deterministic execution of actions without order. (b) Consistency: Ensuring agreement among parties on values and predicates when necessary to avoid races, maintain safety and atomicity, or reach consensus. (c) Fault tolerance: Handling failures in parts or communication, including (Byzantine) misbehavior due to unreliable parties and protocols, when necessary to maintain progress or availability. See also: System Reliability. (d) Trade-offs are a focus of evaluation. See also: Evaluation.

Learning Outcomes I: Write a function that counts events, such as network packet receptions, efficiently [Write]

Bibliography: [van Steen and Tanenbaum, 2023]

5.4 Programs: Distribution, 2 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Distribution (a) Defining locations as devices that execute actions, including hardware components, remote hosts; may also include external uncontrolled devices, hosts, and users. See also: Interfacing and Communication (b) A device may time-share or emulate multiple parallel actions using fewer processors through scheduling and virtualization. See also: Scheduling (c) Naming or identifying locations (e.g., device IDs) and actions as parts (e.g., thread IDs). (d) Activities across locations may communicate through media. See also: Communication

Learning Outcomes I: Write a filter/map/reduce program in multiple styles [Write]

Bibliography: [van Steen and Tanenbaum, 2023]

5.5 Programs: Implementation Mappings, 2 hr(s)

Competences: ABET-2,ABET-3

Topics: I: One or more of the following mappings and mechanisms across layered systems: (a) Instruction-level and data parallelism in the CPU. See also: Functional Organization. (b) SIMD and heterogeneous data parallelism. See also: Heterogeneous Architectures. (c) Scheduled concurrency on multi-core, tasks, actors. See also: Scheduling. (d) Clusters, clouds; elastic provisioning. See also: Common Aspects: Platforms, APIs and Constraints, Common Aspects: Platform Languages and Web Programming Patterns. (e) Networked distributed systems. See also: Networked Applications. (f) Emerging technologies such as quantum computing and molecular computing.

Learning Outcomes I: Explain the trade-offs between different mapping strategies in terms of performance, cost, and implementation complexity [Explain]

Bibliography: [Pacheco and Malensek, 2021, mei W. Hwu et al., 2022]

5.6 GPU Programming, 4 hr(s)

Competences: ABET-2

Topics: I: General-Purpose GPU Computing (GPGPU) (a) GPU architecture overview: streaming multiprocessors (SMs), CUDA cores, and warp execution model. (b) SIMT (Single Instruction, Multiple Threads) execution model and its differences from CPU SIMD. (c) Host-device interaction: data transfer between CPU (host) and GPU (device) memory via PCIe. (d) Use cases: scientific computing, machine learning, image processing, and large-scale simulation.
II: CUDA Programming Model (a) CUDA execution hierarchy: grids, thread blocks, and individual threads; mapping to GPU hardware. (b) Kernel definition and launch syntax; configuration parameters (grid and block dimensions). (c) Thread indexing: threadIdx, blockIdx,

blockDim, gridDim; computing global indices in 1D, 2D, and 3D. (d) Warp divergence: performance implications of branching within a warp. (e) Intra-block synchronization primitive: __syncthreads(). **III:** CUDA Memory Hierarchy (a) Global memory: large, high-latency, accessible by all threads; coalesced access patterns for performance. (b) Shared memory: low-latency on-chip memory shared within a thread block; bank conflicts. (c) Registers and local memory: per-thread private storage. (d) Constant and texture memory: read-only caches optimized for specific access patterns. (e) Unified Memory: simplified programming model with automatic host-device data migration.

Learning Outcomes I: Write a CUDA kernel that performs a data-parallel operation (e.g., vector addition, matrix multiplication) and correctly configure grid and block dimensions [Write] **II:** Analyze the impact of global memory access patterns (coalesced vs. strided) and shared memory usage on GPU kernel performance [Analyze] **III:** Design a parallel algorithm using the CUDA programming model, including thread hierarchy, memory allocation strategy, and host-device data transfers [Design]

Bibliography: [Kirk and mei W. Hwu, 2016, Corporation, 2024]

5.7 Communication, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Media (a) Varieties: channels (message passing or I/O), shared memory, heterogeneous, data stores (b) Reliance on the availability and nature of underlying hardware, connectivity, and protocols; language support, emulation. See also: Interfacing and Communication **II:** Channels (a) Explicit (usually named) party-to-party communication media (b) APIs: Sockets, architectural, language-based, and toolkit constructs, such as Message Passing Interface (MPI), and layered constructs such as Remote Procedure Call (RPC). See also: Fundamentals of Networks and Communications (c) I/O channel APIs **III:** Memory (a) Shared memory architectures in which parties directly communicate only with memory at given addresses, with extensions to heterogeneous memory supporting multiple memory stores with explicit data transfer across them; for example, GPU local and shared memory, Direct Memory Access (DMA) (b) Memory hierarchies: Multiple layers of sharing domains, scopes, and caches; locality: latency, false-sharing (c) Consistency properties: Bitwise atomicity limits, coherence, local ordering **IV:** Data Stores (a) Cooperatively maintained data structures implementing maps and related ADTs (b) Varieties: Owned, shared, sharded, replicated, immutable, versioned

Learning Outcomes I: Explain the similarities and differences among: (1) Party A sends a message on channel X with contents 1 received by party B (2) A sets shared variable X to 1, read by B (3) A sets "X=1" in a distributed shared map accessed by B [Explain] **II:** Write a program that distributes different segments of a data set to multiple workers, and collects results (for the simplest example, summing segments of an array) [Write] **III:** Write a parallel program that requests data from multiple sites and summarizes them using some form of reduction [Write] **IV:** Compare the performance of buffered versus unbuffered versions of a producer-consumer program [Compare]

Bibliography: [van Steen and Tanenbaum, 2023, Kleppmann, 2017]

5.8 Communication: Properties and Extensions, 3 hr(s)

Competences: ABET-2,ABET-3

Topics: I: One or more of the following properties and extensions (a) Topologies: Unicast, Multicast, Mailboxes, Switches; Routing via hardware and software interconnection networks (b) Media concurrency properties: Ordering, consistency, idempotency, overlapping communication with computation (c) Media performance: Latency, bandwidth (throughput) contention (congestion), responsiveness (liveness), reliability (error and drop rates), protocol-based progress (acks, timeouts, mediation) (d) Media security properties: integrity, privacy, authentication, authorization.

See also: Information Flow and Non-Interference, Injection and Input Validation, Memory Safety and Types, Malware Analysis and Advanced Security (e) Data formats: Marshaling, validation, encryption, compression (f) Channel policies: Endpoints, sessions, buffering, saturation response (waiting vs dropping), rate control (g) Multiplexing and demultiplexing many relatively slow I/O devices or parties; completion-based and scheduler-based techniques; async-await, select and polling APIs (h) Formalization and analysis of channel communication; for example, CSP (i) Applications of queuing theory to model and predict performance.

Learning Outcomes **I:** Determine whether a given communication scheme provides sufficient security properties for a given usage [Determine] **II:** Give an example of a scenario in which blocking message sends can deadlock [Create] **III:** Describe at least one design technique for avoiding liveness failures in programs using multiple locks [Describe]

Bibliography: [van Steen and Tanenbaum, 2023, Kleppmann, 2017]

5.9 Memory and Consistency, 3 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Memory models, data consistency, and fault tolerance (a) Memory models: sequential and release/acquire consistency (b) Memory management; including reclamation of shared data; reference counts and alternatives (c) Bulk data placement and transfer; reducing message traffic and improving locality; overlapping data transfer and computation; impact of data layout such as array-of-structs vs struct-of-arrays (d) Emulating shared memory: distributed shared memory, Remote Direct Memory Access (RDMA) (e) Data store consistency: Atomicity, linearizability, transactionality, coherence, causal ordering, conflict resolution, eventual consistency, blockchains (f) Faults, partitioning, and partial failures; voting; protocols such as Paxos and Raft. (g) Design tradeoffs among consistency, availability, partition (fault) tolerance; impossibility of meeting all at once (h) Security and trust: Byzantine failures, proof of work and alternatives

Learning Outcomes **I:** Give an example of an ordering of accesses among concurrent activities (e.g., program with a data race) that is not sequentially consistent [Create] **II:** Write a program that illustrates memory-access or message reordering [Write] **III:** Describe the relative merits of optimistic versus conservative concurrency control under different rates of contention among updates [Describe] **IV:** Give an example of a scenario in which an attempted optimistic update may never complete [Create] **V:** Modify a concurrent system to use a more scalable, reliable, or available data store [Create] **VI:** Using an existing platform supporting replicated data stores, write a program that maintains a key-value mapping even when one or more hosts fail [Write]

Bibliography: [van Steen and Tanenbaum, 2023, Kleppmann, 2017]

5.10 Coordination, 7 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Dependencies (a) Initiation or progress of one activity may be dependent on other activities, so as to avoid race conditions, ensure termination, or meet other requirements (b) Ensuring progress by avoiding dependency cycles, using monotonic conditions, removing inessential dependencies **II:** Control constructs and design patterns (a) Completion-based: Barriers, joins, including termination control (b) Data-enabled: Queues, producer-consumer designs (c) Condition-based: Polling, retrying, backoffs, helping, suspension, signaling, timeouts (d) Reactive: Enabling and triggering continuations

Learning Outcomes **I:** Show how to ensure that a program correctly terminates when all of a set of concurrent tasks have completed [Design] **II:** Write a function that efficiently counts events such as sensor inputs or networking packet receptions [Write] **III:** Write a filter/map/reduce program in multiple styles [Write] **IV:** Write a program in which the termination of one set of parallel actions is followed by another [Write] **V:** Write a service that creates a thread (or other procedural form of activation) to return a requested web page to each new client [Write]

Bibliography: [Herlihy et al., 2020, van Steen and Tanenbaum, 2023]

5.11 Coordination: Synchronization and Atomicity, 3 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Atomicity (a) Atomic instructions, enforced local access orderings (b) Locks and mutual exclusion; lock granularity (c) Using locks in a specific language; maintaining liveness without introducing races (d) Deadlock avoidance: Ordering, coarsening, randomized retries; backoffs, encapsulation via lock managers (e) Common errors: Failing to lock or unlock when necessary, holding locks while invoking unknown operations (f) Avoiding locks: replication, read-only, ownership, and non-blocking constructions

Learning Outcomes I: Show how to avoid or repair a race error in a given program [Analyze]

Bibliography: [Herlihy et al., 2020]

5.12 Coordination: Advanced Properties, 4 hr(s)

Competences: ABET-2,ABET-3

Topics: I: One or more of the following properties and extensions (a) Progress properties including lock-free, wait-free, fairness, priority scheduling, interactions with consistency, reliability (b) Performance with respect to contention, granularity, convoying, scaling (c) Non-blocking data structures and algorithms (d) Ownership and resource control (e) Lock variants and alternatives: sequence locks, read-write locks; Read-Copy-Update (RCU), reentrancy; tickets; controlling spinning versus blocking (f) Transaction-based control: Optimistic and conservative (g) Distributed locking: reliability (h) Alternatives to barriers: Clocks; counters, virtual clocks; dataflow and continuations; futures and RPC; consensus-based, gathering results with reducers and collectors (i) Speculation, selection, cancellation; observability and security consequences (j) Resource control using semaphores and condition variables (k) Control flow: Scheduling computations, series-parallel loops with (possibly elected) leaders, pipelines and streams, nested parallelism (l) Exceptions and failures. Handlers, detection, timeouts, fault tolerance, voting

Learning Outcomes I: Write a program that speculatively searches for a solution by multiple activities, terminating others when one is found [Write] **II:** Write a program in which a numerical exception (such as divide by zero) in one activity causes termination of others [Write] **III:** Write a program for multiple parties to agree upon the current time of day; discuss its limitations compared to protocols such as network transfer protocol (NTP) [Write]

Bibliography: [Herlihy et al., 2020]

5.13 Evaluation, 7 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Safety and liveness requirements in terms of temporal logic constructs to express "always" and "eventually" See also: Parallel and Distributed Computing **II:** Identifying, testing for, and repairing violations, including common forms of errors such as failure to ensure necessary ordering (race errors), atomicity (including check-then-act errors), and termination (livelock) **III:** Performance requirements metrics for throughput, responsiveness, latency, availability, energy consumption, scalability, resource usage, communication costs, waiting and rate control, fairness; service level agreements. See also: Latency, Cache and Memory Hierarchy, Virtualization and Isolation **IV:** Performance impact of design and implementation choices, including granularity, overhead, consensus costs, and energy consumption. See also: Sustainable Design and Pervasive Computing, Environmental Footprint of Computing Systems, Systemic Effects and Social Context **V:** Estimating scalability limitations, for example using Amdahl's Law or Universal Scalability Law. See also: Events, Tools and Experimentation, Performance Metrics and Benchmarks, Analytical Performance Models

Learning Outcomes I: Revise a specification to enable parallelism and distribution without violating other essential properties or features [Redesign] **II:** Explain how concurrent notions of safety and liveness extend their sequential counterparts [Explain] **III:** Specify a set of invariants that must hold at each bulk-parallel step of a computation [Analyze] **IV:** Write a test program that can reveal a data race error; for example, missing an update when two activities both try to increment a variable [Write] **V:** In a given context, explain the extent to which introducing parallelism in an otherwise sequential program would be expected to improve throughput and/or reduce latency, and how it may impact energy efficiency [Explain] **VI:** Show how scaling and efficiency change for sample problems without and with the assumption of problem size changing with the number of processors; further explain whether and how scalability would change under relaxations of sequential dependencies [Design]

Bibliography: [Sterling et al., 2024, Pacheco and Malensek, 2021]

5.14 Evaluation: Formal Methods, 3 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Formal verification and analysis methods (a) Extensions to formal sequential requirements such as linearizability (b) Protocol, session, and transactional specifications (c) Use of tools such as Unified Modelling Language (UML), Temporal Logic of Actions (TLA), program logics (d) Security analysis: safety and liveness in the presence of hostile or buggy behaviors by other parties; required properties of communication mechanisms (for example lack of cross-layer leakage), input screening, rate limiting. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles (e) Static analysis applied to correctness, throughput, latency, resources, energy. See also: Sustainable Design and Pervasive Computing, Environmental Footprint of Computing Systems, Systemic Effects and Social Context (f) Directed Acyclic Graph (DAG) model analysis of algorithmic efficiency (work, span, critical paths)

Learning Outcomes I: Specify and measure behavior when a service is requested by unexpectedly many clients [Analyze] **II:** Identify and repair a performance problem due to sequential bottlenecks [Analyze] **III:** Empirically compare throughput of two implementations of a common design (perhaps using an existing test harness framework) [Compare]

Bibliography: [Sterling et al., 2024, Pacheco and Malensek, 2021]

5.15 Evaluation: Testing and Measurement, 2 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Testing tools and measurement techniques (a) Testing and debugging; tools such as race detectors, fuzzers, lock dependency checkers, unit/stress/torture tests, visualizations, continuous integration, continuous deployment, and test generators (b) Measuring and comparing throughput, overhead, waiting, contention, communication, data movement, locality, resource usage, behavior in the presence of excessive numbers of events, clients, or threads. See also: Events, Tools and Experimentation, Performance Metrics and Benchmarks, Analytical Performance Models (c) Application domain specific analyses and evaluation techniques

Learning Outcomes I: Identify and repair a performance problem due to communication or data latency [Analyze] **II:** Identify and repair a performance problem due to resource management overhead [Analyze] **III:** Identify and repair a reliability or availability problem [Analyze]

Bibliography: [Sterling et al., 2024, Pacheco and Malensek, 2021]

5.16 Algorithms, 4 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Expressing and implementing algorithms in given languages and frameworks, to initiate activities (for example threads), use shared memory constructs, and channel, socket, and/or remote procedure call APIs. See also: Parallel and Distributed Computing. (a) Data parallel examples including map/reduce. (b) Using channel, socket, and/or RPC APIs in a given language, with program control for sending (usually procedural) vs receiving. (usually reactive or RPC-based). (c) Using locks, barriers, and/or synchronizers to maintain liveness without introducing races.

Learning Outcomes I: Implement a parallel/distributed component based on a known algorithm [Implement] **II:** Write a data-parallel program that for example computes the average of an array of numbers [Write] **III:** Write a producer-consumer program in which one component generates numbers, and another computes their average. Measure speedups when the numbers are small scalars versus large multi-precision values [Write]

Bibliography: [Pacheco and Malensek, 2021, Herlihy et al., 2020]

5.17 Algorithms: Application Domains Survey, 4 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Survey of common application domains across multicore, reactive, data parallel, cluster, cloud, open distributed systems, and frameworks (with reference to the following table). (a) Multicore: Typical Execution agents: Threads. Typical Communication mechanisms: Shared memory, Atomics, locks. Typical Algorithmic domains: Resource management, data processing. Typical Engineering goals: Throughput, latency, energy. (b) Reactive: Typical Execution agents: Handlers, threads. Typical Communication mechanisms: I/O Channels. Typical Algorithmic domains: Services, real-time. Typical Engineering goals: Latency. (c) Data parallel: Typical Execution agents: GPU, SIMD, accelerators, hybrid. Typical Communication mechanisms: Heterogeneous memory. Typical Algorithmic domains: Linear algebra, graphics, data analysis. Typical Engineering goals: Throughput, energy. (d) Cluster: Typical Execution agents: Managed hosts. Typical Communication mechanisms: Sockets, channels. Typical Algorithmic domains: Simulation, data analysis. Typical Engineering goals: Throughput. (e) Cloud: Typical Execution agents: Provisioned hosts. Typical Communication mechanisms: Service APIs. Typical Algorithmic domains: Web applications. Typical Engineering goals: Scalability. (f) Open distributed: Typical Execution agents: Autonomous hosts. Typical Communication mechanisms: Sockets, Data stores. Typical Algorithmic domains: Fault tolerant data stores and services. Typical Engineering goals: Reliability.

Learning Outcomes I: Extend an event-driven sequential program by establishing a new activity in an event handler (for example a new thread in a GUI action handler) [Design] **II:** Improve the performance of a sequential component by introducing parallelism and/or distribution [Create] **III:** Choose among different parallel/distributed designs for components of a given system [Assess]

Bibliography: [Pacheco and Malensek, 2021, Herlihy et al., 2020]

5.18 Algorithms: Algorithmic Domains, 4 hr(s)

Competences: ABET-2,ABET-3

Topics: I: One of more of the following algorithmic domains. See also: Algorithmic Strategies: (a) Linear algebra: Vector and matrix operations, numerical precision/stability, applications in data analytics and machine learning. (b) Data processing: sorting, searching and retrieval, concurrent data structures. (c) Graphs, search, and combinatorics: Marking, edge-parallelization, bounding, speculation, network-based analytics. (d) Modeling and simulation: differential equations; randomization, N-body problems, genetic algorithms. (e) Computational logic: satisfiability (SAT), concurrent logic programming. (f) Graphics and computational geometry: Transforms, rendering, ray-tracing. (g) Resource management: Allocating, placing, recycling and scheduling

processors, memory, channels, and hosts; exclusive vs shared resources; static, dynamic and elastic algorithms; Real-time constraints; Batching, prioritization, partitioning; decentralization via work-stealing and related techniques. (h) Services: Implementing web APIs, electronic currency, transaction systems, multiplayer games.

Learning Outcomes I: Design, implement, analyze, and evaluate a component or application for X operating in a given context, where X is in one of the listed domains, for example, a genetic algorithm for factory floor design [Design] **II:** Critique the design and implementation of an existing component or application, or one developed by classmates [Critique] **III:** Compare the performance and energy efficiency of multiple implementations of a similar design, for example, multicore versus clustered versus GPU [Compare]

Bibliography: [Pacheco and Malensek, 2021, Herlihy et al., 2020]

6. Bibliography

[Corporation, 2024] Corporation, N. (2024). Cuda c++ programming guide. `https://docs.nvidia.com/cuda/cuda-c-programming-guide/`. Documentación oficial de NVIDIA.

[Herlihy et al., 2020] Herlihy, M., Shavit, N., Luchangco, V., and Spear, M. (2020). *The Art of Multiprocessor Programming*. Morgan Kaufmann, 2nd edition.

[Kirk and mei W. Hwu, 2016] Kirk, D. B. and mei W. Hwu, W. (2016). *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann, Cambridge, MA, 3rd edition.

[Kleppmann, 2017] Kleppmann, M. (2017). *Designing Data-Intensive Applications*. O'Reilly Media, 1st edition.

[mei W. Hwu et al., 2022] mei W. Hwu, W., Kirk, D. B., and Hajj, I. E. (2022). *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann, 4th edition.

[Pacheco and Malensek, 2021] Pacheco, P. S. and Malensek, M. (2021). *An Introduction to Parallel Programming*. Morgan Kaufmann, 2nd edition.

[Sterling et al., 2024] Sterling, T., Brodowicz, M., and Anderson, M. (2024). *High Performance Computing: Modern Systems and Practices*. Morgan Kaufmann, 2nd edition.

[van Steen and Tanenbaum, 2023] van Steen, M. and Tanenbaum, A. S. (2023). *Distributed Systems*. Maarten van Steen, 4th edition.

[Corporation, 2024] Corporation, N. (2024). Cuda c++ programming guide. `https://docs.nvidia.com/cuda/cuda-c-programming-guide/`. Documentación oficial de NVIDIA.

[Herlihy et al., 2020] Herlihy, M., Shavit, N., Luchangco, V., and Spear, M. (2020). *The Art of Multiprocessor Programming*. Morgan Kaufmann, 2nd edition.

[Kirk and mei W. Hwu, 2016] Kirk, D. B. and mei W. Hwu, W. (2016). *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann, Cambridge, MA, 3rd edition.

[Kleppmann, 2017] Kleppmann, M. (2017). *Designing Data-Intensive Applications*. O'Reilly Media, 1st edition.

[mei W. Hwu et al., 2022] mei W. Hwu, W., Kirk, D. B., and Hajj, I. E. (2022). *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann, 4th edition.

[Pacheco and Malensek, 2021] Pacheco, P. S. and Malensek, M. (2021). *An Introduction to Parallel Programming*. Morgan Kaufmann, 2nd edition.

[Sterling et al., 2024] Sterling, T., Brodowicz, M., and Anderson, M. (2024). *High Performance Computing: Modern Systems and Practices*. Morgan Kaufmann, 2nd edition.

[van Steen and Tanenbaum, 2023] van Steen, M. and Tanenbaum, A. S. (2023). *Distributed Systems*. Maarten van Steen, 4th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA213 Partial Differential Equations (Mandatory)

1. General information

- **Semester**: 7th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA212 Ordinary Differential Equations (6th Sem-Pg. 409)
  - MA211 Calculus IV (5th Sem-Pg. 390)

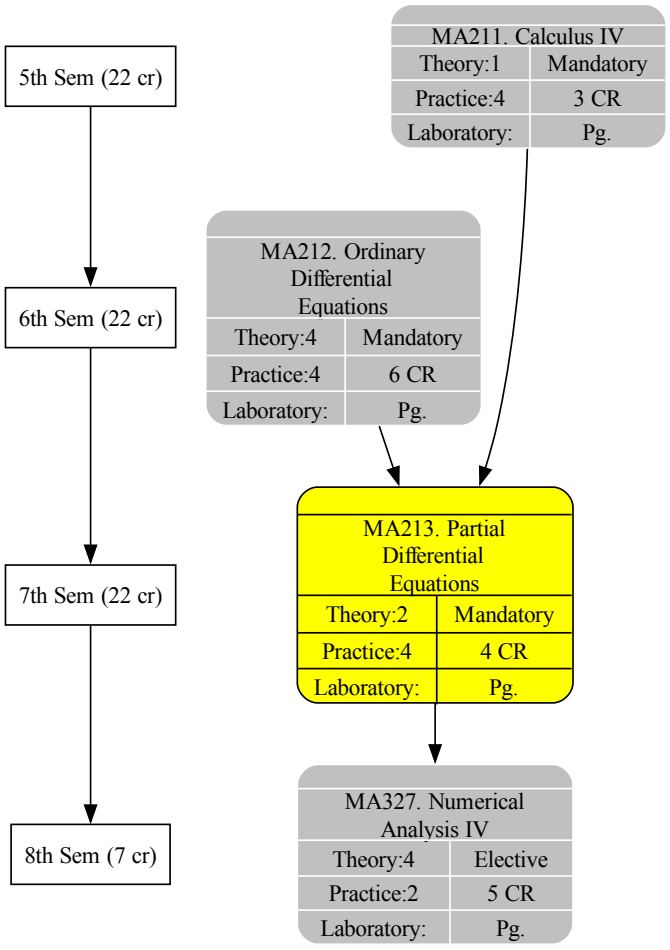


Figure 7.39: Connection Map. MA213 Partial Differential Equations

2. Course Rationale

The Partial Differential Equations course studies phenomena that depend on multiple variables, such as heat propagation, waves, and electrical potentials. The course introduces Fourier series and separation of variables methods to solve the classical equations of mathematical physics (Heat, Wave, and Laplace equations). It is a key component for advanced numerical simulation and continuum mechanics analysis.

3. Goals of the Course

- Use Fourier series to represent periodic functions and solve boundary value problems.
- Solve linear partial differential equations using the separation of variables method.
- Classify second-order PDEs and understand their physical behavior.
- Apply finite difference numerical methods for solving PDEs.

4. General Competences (*Outcomes*)

5. Content

5.1 Fourier Series and Sturm-Liouville Problems, 24 hr(s)

Competences:

Topics: **I:** Orthogonal functions and sine and cosine Fourier series. **II:** Complex Fourier series and Parseval's identity. **III:** Eigenvalue problems and the Sturm-Liouville theorem. **IV:** Application of series to solving ODEs with boundary conditions.

Learning Outcomes **I:** Implement basic ODE solvers (Euler, Runge-Kutta) [Usage] **II:** Analyze the stability and convergence properties of ODE numerical methods [Assessment] **III:** Select an appropriate numerical method for a given ODE based on stability and accuracy requirements [Assessment] **IV:** Solve systems of ODEs and convert higher-order ODEs to first-order systems [Usage]

Bibliography: [Boyce et al., 2017, Zill, 2018]

5.2 Diffusion Equations (Heat), 24 hr(s)

Competences:

Topics: **I:** Derivation of the heat equation in one dimension. **II:** Separation of variables method for Dirichlet and Neumann conditions. **III:** Heat problems in rods with insulated ends or radiation. **IV:** Heat equation in two and three dimensions (Cartesian coordinates).

Learning Outcomes **I:** Implement root-finding algorithms (bisection, Newton, secant) and analyze their convergence [Usage] **II:** Classify the order of convergence of iterative methods [Assessment] **III:** Apply gradient descent to solve simple univariate optimization problems [Usage] **IV:** Design appropriate stopping criteria for iterative methods [Usage]

Bibliography: [Boyce et al., 2017, Zill, 2018]

5.3 Wave and Laplace Equations, 24 hr(s)

Competences:

Topics: **I:** The one-dimensional wave equation: Vibrating string. **II:** D'Alembert's solution and traveling waves. **III:** Laplace's equation in Cartesian and polar coordinates (Dirichlet Problem). **IV:** Steady-state applications and electrostatics.

Learning Outcomes **I:** Construct interpolating polynomials using different methods [Usage] **II:** Implement cubic spline interpolation [Usage] **III:** Apply least squares method for data fitting [Usage] **IV:** Explain Runge's phenomenon and strategies to mitigate it [Familiarity] **V:** Analyze the error of an interpolation scheme [Assessment]

Bibliography: [Boyce et al., 2017, Zill, 2018]

5.4 Numerical Methods for PDE and Transforms, 24 hr(s)

Competences:

Topics: **I:** Introduction to the finite difference method for parabolic and elliptic PDEs. **II:** Use of the Fourier Transform to solve PDEs in infinite domains. **III:** Stability and convergence of basic numerical schemes. **IV:** Modeling of real-world problems using scientific computing software.

Learning Outcomes **I:** Compute derivatives of functions using finite difference schemes [Usage] **II:** Apply Richardson extrapolation to improve the accuracy of numerical approximations [Usage] **III:** Implement numerical integration rules (Trapezoidal, Simpson, Gaussian) [Usage] **IV:** Compare the accuracy and efficiency of different numerical integration methods [Assessment] **V:** Analyze the error for numerical differentiation and integration formulas [Assessment]

Bibliography: [Boyce et al., 2017, Zill, 2018]

6. Bibliography

[Boyce et al., 2017] Boyce, W. E., DiPrima, R. C., and Meade, D. B. (2017). *Ecuaciones Diferenciales y Problemas con Valores en la Frontera*. Limusa Wiley, 11th edition.

[Zill, 2018] Zill, D. G. (2018). *Ecuaciones Diferenciales con Aplicaciones de Modelado*. Cengage Learning, 11th edition.

[Boyce et al., 2017] Boyce, W. E., DiPrima, R. C., and Meade, D. B. (2017). *Ecuaciones Diferenciales y Problemas con Valores en la Frontera*. Limusa Wiley, 11th edition.

[Zill, 2018] Zill, D. G. (2018). *Ecuaciones Diferenciales con Aplicaciones de Modelado*. Cengage Learning, 11th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA326 Numerical Analysis III (Mandatory)

1. General information

- **Semester**: 7th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA325 Numerical Analysis II (6th Sem-Pg. 412)

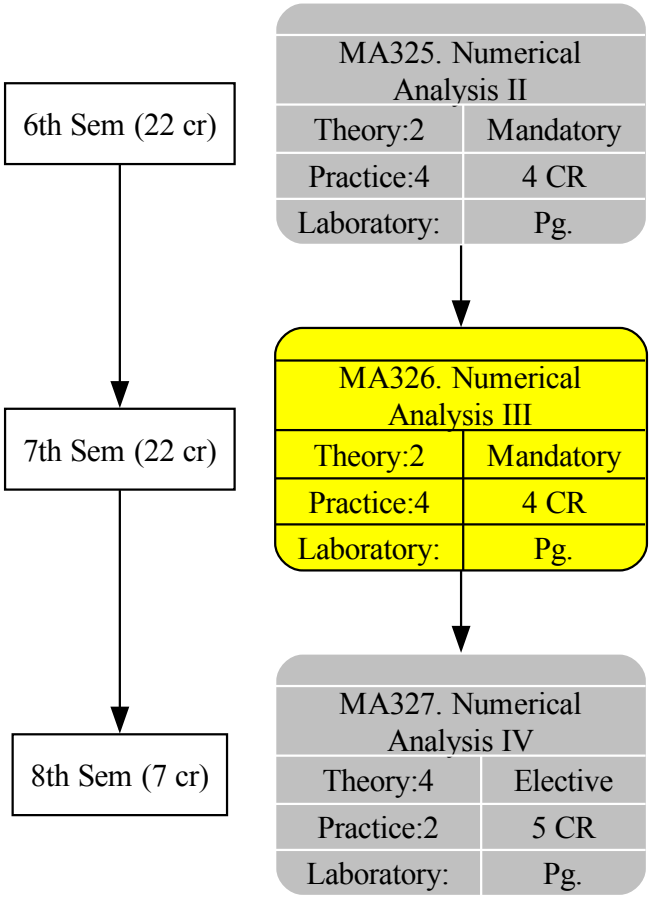


Figure 7.40: Connection Map. MA326 Numerical Analysis III

2. Course Rationale

Numerical Analysis 3 focuses on the computational solution of Ordinary Differential Equations (ODEs). In computational mathematics, many systems cannot be solved analytically; therefore, understanding the convergence, stability, and efficiency of numerical integrators is vital. This course covers from basic one-step methods to advanced Runge-Kutta schemes and the specialized treatment of stiff equations, which are critical in simulating chemical reactions, electrical circuits, and structural mechanics.

3. Goals of the Course

- Understand the fundamental theory of numerical methods for Initial Value Problems (IVPs).
- Master the construction and analysis of high-order Runge-Kutta methods.
- Evaluate the stability and convergence of numerical schemes.
- Implement specialized algorithms for Stiff Differential Equations.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Numerical Solution of ODEs: Foundations, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Review of Existence and Uniqueness (Lipschitz continuity) **II:** Elementary methods: Euler, Heun, and Taylor series methods **III:** Error analysis: Local truncation error and Global error **IV:** Consistency, Convergence, and Zero-stability

Learning Outcomes **I:** Formulate discrete approximations for continuous differential systems [Usage] **II:** Derive the local truncation error for basic one-step methods [Assessment] **III:** Analyze the impact of step size on numerical stability [Familiarity]

Bibliography: [Burden et al., 2016, Heath, 2002]

5.2 Runge-Kutta Methods, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** General structure of Runge-Kutta (RK) methods **II:** Butcher Tableaus and order conditions **III:** Explicit and Implicit Runge-Kutta methods **IV:** Adaptive step-size control: Runge-Kutta-Fehlberg (RK45)

Learning Outcomes **I:** Construct high-order RK methods using Butcher Tableaus [Usage] **II:** Implement adaptive algorithms to optimize computational cost [Assessment] **III:** Prove the order of convergence for specific RK schemes [Familiarity]

Bibliography: [Quarteroni et al., 2007, Butcher, 2016]

5.3 Linear Multi-step Methods, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Adams-Bashforth and Adams-Moulton formulas **II:** Predictor-Corrector methods **III:** General Theory of Linear Multi-step methods (Dahlquist theorems) **IV:** Comparison between RK and Multi-step methods

Learning Outcomes **I:** Develop multi-step codes for efficient long-term integration [Usage] **II:** Verify the root condition for zero-stability [Assessment] **III:** Select appropriate methods based on the trade-off between memory and accuracy [Familiarity]

Bibliography: [Burden et al., 2016, Quarteroni et al., 2007]

5.4 Stiff Equations and Absolute Stability, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Definition and characterization of Stiff Systems **II:** Linear Stability Analysis: The test equation and Stability Regions **III:** A-stability and L-stability **IV:** Backward Differentiation Formulas (BDF) and Implicit methods for stiffness

Learning Outcomes **I:** Identify stiffness in physical systems through eigenvalue analysis [Assessment] **II:** Plot and interpret stability regions in the complex plane [Usage] **III:** Apply implicit solvers to overcome step-size restrictions in stiff problems [Assessment]

Bibliography: [Heath, 2002, Butcher, 2016]

6. Bibliography

[Burden et al., 2016] Burden, R. L., Faires, J. D., and Burden, A. M. (2016). *Numerical Analysis*. Cengage Learning, 10th edition.

[Butcher, 2016] Butcher, J. C. (2016). *Numerical Methods for Ordinary Differential Equations*. Wiley, 3rd edition.

[Heath, 2002] Heath, M. T. (2002). *Scientific Computing: An Introductory Survey*. McGraw-Hill, 2nd edition.

[Quarteroni et al., 2007] Quarteroni, A., Sacco, R., and Saleri, F. (2007). *Numerical Mathematics*. Springer, 2nd edition.

[Burden et al., 2016] Burden, R. L., Faires, J. D., and Burden, A. M. (2016). *Numerical Analysis*. Cengage Learning, 10th edition.

[Butcher, 2016] Butcher, J. C. (2016). *Numerical Methods for Ordinary Differential Equations*. Wiley, 3rd edition.

[Heath, 2002] Heath, M. T. (2002). *Scientific Computing: An Introductory Survey*. McGraw-Hill, 2nd edition.

[Quarteroni et al., 2007] Quarteroni, A., Sacco, R., and Saleri, F. (2007). *Numerical Mathematics*. Springer, 2nd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA361 Mathematical Modeling (Mandatory)

1. General information

- **Semester**: 7th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - MA325 Numerical Analysis II (6th Sem-Pg. 412)

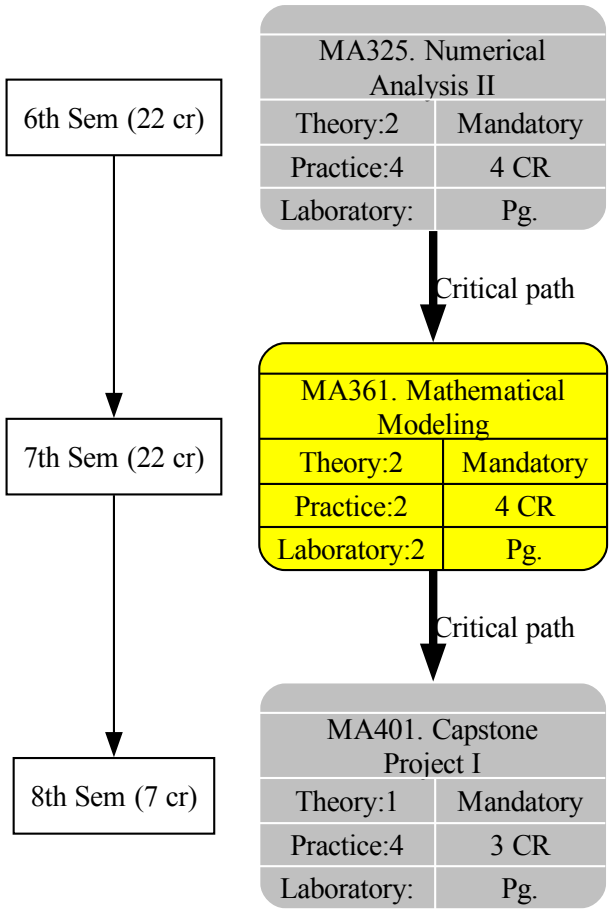


Figure 7.41: Connection Map. MA361 Mathematical Modeling

2. Course Rationale

This course introduces the fundamental principles of mathematical modeling to represent, analyze, and predict the behavior of systems in various areas such as science, engineering, and business. It emphasizes the formulation of models from real-world problems, their solution using mathematical and computational tools, and the critical interpretation of results. The course develops abstraction skills and the application of advanced mathematics to solve complex real-world problems.

3. Goals of the Course

- Understand the fundamental principles and stages of the mathematical modeling process.
- Formulate mathematical models from concrete problems in different scientific and technological contexts.
- Apply analytical and numerical methods to solve the proposed models.
- Critically interpret simulation results and validate models with real data.
- Develop skills to effectively communicate models and their conclusions.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Mathematical Model Formulation, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Modeling process and cycle **II:** Continuous vs. discrete models **III:** Deterministic vs. stochastic models **IV:** Assumptions identification and simplification techniques **V:** Dimensional analysis and scaling

Learning Outcomes **I:** Describe the complete modeling cycle from problem definition to solution interpretation [Familiarity] **II:** Classify models as continuous/discrete and deterministic/stochastic [Assessment] **III:** Identify and justify appropriate assumptions for model simplification [Assessment] **IV:** Apply dimensional analysis to reduce complexity of physical models [Usage]

Bibliography: [Giordano et al., 2013, Lin and Segel, 2009]

5.2 Continuous Systems Simulation, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Ordinary Differential Equations (ODEs) models **II:** Partial Differential Equations (PDEs) models **III:** Finite difference methods for PDEs **IV:** Solver selection and implementation **V:** Stiff systems and adaptive methods

Learning Outcomes **I:** Formulate ODE and PDE models for physical systems [Usage] **II:** Implement finite difference methods for simple PDEs [Usage] **III:** Select appropriate numerical solvers for different types of ODEs/PDEs [Assessment] **IV:** Handle stiff systems using appropriate numerical methods [Usage]

Bibliography: [Braun et al., 2013, Giordano et al., 2013]

5.3 Discrete Systems Simulation, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Queueing theory and models **II:** Cellular automata and agent-based models **III:** Monte Carlo simulation methods **IV:** Markov chains and processes **V:** Discrete-event simulation

Learning Outcomes **I:** Formulate and analyze simple queueing models [Usage] **II:** Implement cellular automata and agent-based simulations [Usage] **III:** Apply Monte Carlo methods for probabilistic simulation [Usage] **IV:** Analyze Markov chains for state transitions and steady-state behavior [Assessment] **V:** Design discrete-event simulations for system analysis [Usage]

Bibliography: [Giordano et al., 2013, Meerschaert, 2013]

5.4 Sensitivity Analysis and Robustness, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Local sensitivity analysis (partial derivatives) **II:** Global sensitivity analysis (Sobol indices) **III:** Morris method and screening techniques **IV:** Robustness analysis and worst-case scenarios **V:** Uncertainty propagation

Learning Outcomes **I:** Perform local sensitivity analysis using derivative-based methods [Usage] **II:** Apply global sensitivity analysis techniques (e.g., Sobol indices) [Usage] **III:** Use Morris method for factor screening [Usage] **IV:** Assess model robustness under parameter variations [Assessment] **V:** Quantify uncertainty propagation through models [Assessment]

Bibliography: [Banks et al., 2010, Giordano et al., 2013]

5.5 Model Validation and Verification, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Verification: numerical code correctness **II:** Validation: comparison with experimental data **III:** Uncertainty Quantification (UQ) in models **IV:** Model calibration and parameter estimation **V:** Goodness-of-fit tests and metrics

Learning Outcomes **I:** Distinguish between verification and validation in modeling [Familiarity] **II:** Apply verification techniques to ensure code correctness [Usage] **III:** Validate models using experimental data and statistical tests [Usage] **IV:** Calibrate model parameters using optimization techniques [Usage] **V:** Apply uncertainty quantification methods to assess model reliability [Usage]

Bibliography: [Giordano et al., 2013, Lin and Segel, 2009, Meerschaert, 2013]

6. Bibliography

[Banks et al., 2010] Banks, J., II, J. S. C., Nelson, B. L., and Nicol, D. M. (2010). *Discrete-Event System Simulation*. Prentice Hall, 5th edition.

[Braun et al., 2013] Braun, M., Coleman, C. S., and Drew, D. A. (2013). *Differential Equation Models*. Springer, 1st edition.

[Giordano et al., 2013] Giordano, F. R., Fox, W. P., and Horton, S. B. (2013). *A First Course in Mathematical Modeling*. Cengage Learning, 5th edition.

[Lin and Segel, 2009] Lin, C. C. and Segel, L. A. (2009). *Mathematics Applied to Deterministic Problems in the Natural Sciences*. Society for Industrial and Applied Mathematics, 1st edition.

[Meerschaert, 2013] Meerschaert, M. M. (2013). *Mathematical Modeling*. Academic Press, 4th edition.

[Banks et al., 2010] Banks, J., II, J. S. C., Nelson, B. L., and Nicol, D. M. (2010). *Discrete-Event System Simulation*. Prentice Hall, 5th edition.

[Braun et al., 2013] Braun, M., Coleman, C. S., and Drew, D. A. (2013). *Differential Equation Models*. Springer, 1st edition.

[Giordano et al., 2013] Giordano, F. R., Fox, W. P., and Horton, S. B. (2013). *A First Course in Mathematical Modeling*. Cengage Learning, 5th edition.

[Lin and Segel, 2009] Lin, C. C. and Segel, L. A. (2009). *Mathematics Applied to Deterministic Problems in the Natural Sciences*. Society for Industrial and Applied Mathematics, 1st edition.

[Meerschaert, 2013] Meerschaert, M. M. (2013). *Mathematical Modeling*. Academic Press, 4th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

ID203 Technical and professional English V (Mandatory)

1. General information

- **Semester**: 7th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 6 hours;
- **Prerrequisites**:
  - ID202 Technical and professional English IV (6th Sem-Pg. 423)

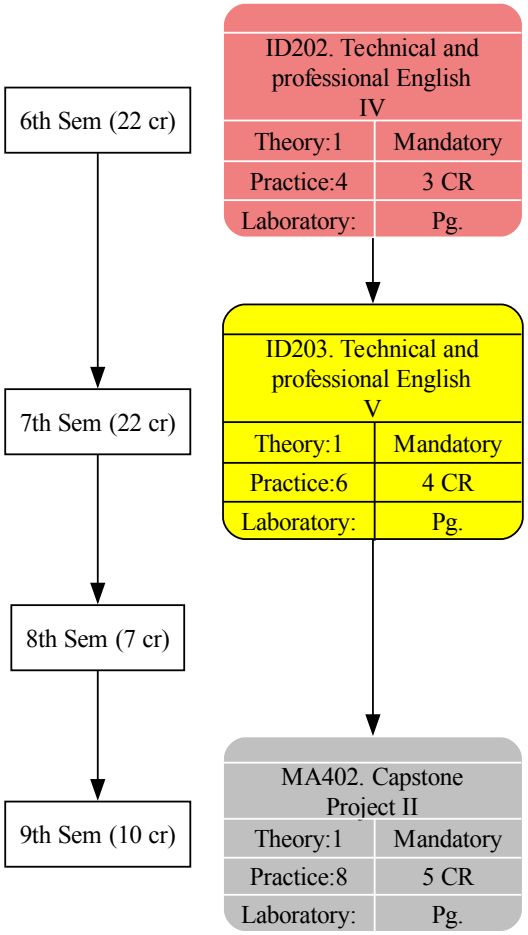


Figure 7.42: Connection Map. ID203 Technical and professional English V

2. Course Rationale

A fundamental part of the integral formation of a professional is the ability to communicate in a foreign language in addition to the native language itself. It not only broadens its cultural horizon but also allows a more humane and comprehensive view of life. In the case of foreign languages, undoubtedly English is the most practical because it is spoken around the world. There is no country where it is not spoken. In careers related to tourist services, English is perhaps the most important practical tool that the student must master from the outset as part of his / her integral education

3. Goals of the Course

- Increase the ability and fluency of speaking and understanding the English language.
- That the students interact with greater emphasis in the creation of dialogues.

4. General Competences (*Outcomes*)

ABET-4) An ability to communicate effectively with a range of audiences. (Usage)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Familiarity)

5. Content

5.1 It's a wonderful world, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Auxiliary verbs **II:** Verb tenses **III:** Negative Questions and Prayers **IV:** Short answers **V:** Word formation **VI:** Colloquial expressions **VII:** Error correction

Learning Outcomes **I:** At the end of the first unit, each student, understanding the grammar of auxiliaries and different types of sentences, is able to express a greater number of expressions of time and also use prepositions to describe varied places and times. He is also able to analyze and express ideas about word formation.

Bibliography: [Soars and John, 2002g, Soars and John, 2002i, Soars and John, 2002h, Cambridge, 2006, MacGrew, 1999]

5.2 Happiness!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Simple present **II:** Present continuous **III:** Passive Voice in Present **IV:** Verbs for sports and free time **V:** Types of numbers and **VI:** Inventions / Modern World **VII:** Corrección de errores

Learning Outcomes **I:** At the end of the second unit, students have identified how to express sports and leisure activities. It uses all kinds of numerical expressions. Express situations and states related to present forms. Explain and apply vocabulary of outdoor activities.

Bibliography: [Soars and John, 2002g, Soars and John, 2002i, Soars and John, 2002h, Cambridge, 2006, MacGrew, 1999]

5.3 Telling tales!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: **I:** Simple Past Time **II:** Past Continuous **III:** Passive Voice in Past **IV:** Vocabulary of Art and Literature **V:** Expressions to give and ask opinions **VI:** Stories and stories

Learning Outcomes **I:** At the end of the third unit, students having recognized the characteristics of the past passive forms, they use these make descriptions of various types. Describe art and literature and give indications of opinion. They will use conjunctions to unite type ideas.

Bibliography: [Soars and John, 2002g, Soars and John, 2002i, Soars and John, 2002h, Cambridge, 2006, MacGrew, 1999]

5.4 Doing the right thing!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: I: Mode Auxiliary Verbs I. **II:** Affirmative, Negative and Modals Questions **III:** Use of nationalities and other adjectives **IV:** Expressions of orders and offers **V:** Guide to Good Manners **VI:** Form Fill **VII:** Phonetic symbols

Learning Outcomes I: At the conclusion of the fourth unit, the students having identified the idea of expressing ideas of modes of actions that happen at the moment or that are related at any time, structure sentences in the Present. They express ideas of nationalities and make requests and offers varied.

Bibliography: [Soars and John, 2002g, Soars and John, 2002i, Soars and John, 2002h, Cambridge, 2006, MacGrew, 1999]

5.5 On the move!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: I: Future with Will **II:** Future Time Prayers with going to **III:** Use of might for future **IV:** Climate Expressions **V:** Vocabulary of the climate **VI:** Expressions for hotels and transportation **VII:** E-mails

Learning Outcomes I: At the end of the fifth unit, students, from the understanding of future time, will elaborate sentences using the necessary elements. They will also assimilate the need to express ideas of the climate. They will acquire vocabulary to describe use of public transportation. Expressions will be presented to order at hotels.

Bibliography: [Soars and John, 2002g, Soars and John, 2002i, Soars and John, 2002h, Cambridge, 2006, MacGrew, 1999]

5.6 I just love it!, 0 hr(s)

Competences: ABET-4,ABET-6

Topics: I: Questions with Shapes Like **II:** Patrones Verbales II **III:** Vocabulario de Comida, Lugares y ocupaciones **IV:** Palabras que van unidas en contexto **V:** Expresiones para vistas y sonidos **VI:** Composición de Impresiones personales

Learning Outcomes I: At the end of the sixth unit, students having learned the basics of structuring questions with like and with verbal patterns work applied to appropriate contexts. They emphasize the difference between meals, places and people. Describe sights and sounds. They use expressions to compare daily life in different places. They assume the idea of different lifestyles.

Bibliography: [Soars and John, 2002g, Soars and John, 2002i, Soars and John, 2002h, Cambridge, 2006, MacGrew, 1999]

6. Bibliography

[Cambridge, 2006] Cambridge (2006). *Diccionario Inglés-Espanol Cambridge*. Editorial Oxford.

[MacGrew, 1999] MacGrew, J. (1999). *Focus on Grammar Basic*. Editorial Oxford.

[Soars and John, 2002g] Soars, L. and John (2002g). *American Headway N 3 Student Book*. Editorial Oxford.

[Soars and John, 2002h] Soars, L. and John (2002h). *American Headway N 3 Teachers Book*. Editorial Oxford.

[Soars and John, 2002i] Soars, L. and John (2002i). *American Headway N 3 Work Book*. Editorial Oxford.

Eighth Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

AI263 Introduction to Machine Learning (Mandatory)

1. General information

- **Semester**: 8th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS261 Artificial Intelligence (7th Sem-Pg. 427)

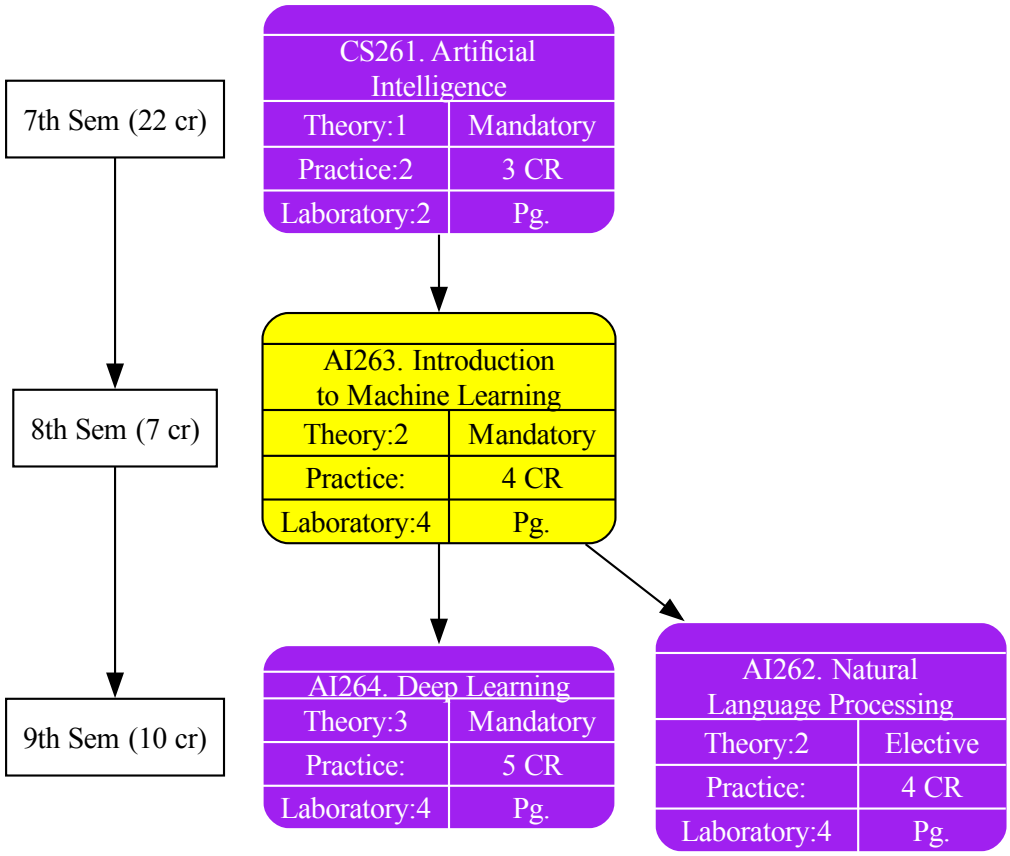


Figure 7.43: Connection Map. AI263 Introduction to Machine Learning

2. Course Rationale

This course introduces the fundamentals of machine learning, covering classical and modern algorithms for classification, regression, and clustering problems. It focuses on practical implementation using scikit-learn and TensorFlow, with applications in computer vision and natural language processing.

3. Goals of the Course

- Understand the mathematical principles behind ML algorithms.

- Implement complete ML pipelines with Python.
- Evaluate and optimize models using standard metrics.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 ML Fundamentals, 15 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Supervised vs unsupervised learning **II:** Overfitting and regularization **III:** Cross-validation and learning curves

Learning Outcomes **I:** Explain the bias-variance tradeoff [Familiarity] **II:** Implement k-fold cross-validation [Usage]

Bibliography: [Bishop, 2006, Goodfellow et al., 2016]

5.2 Linear Models, 15 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Linear and logistic regression **II:** Support Vector Machines (SVM) **III:** Linear discriminant analysis

Learning Outcomes **I:** Program linear models with scikit-learn [Usage] **II:** Interpret regression coefficients [Assessment]

Bibliography: [Hastie et al., 2009, Murphy, 2012]

5.3 Ensembles and Trees, 15 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Decision trees and random forests **II:** Gradient Boosting (XGBoost, LightGBM) **III:** Stacking and voting

Learning Outcomes **I:** Optimize hyperparameters with GridSearch [Usage] **II:** Visualize decision trees [Assessment]

Bibliography: [Hastie et al., 2009, Géron, 2022]

5.4 Basic Neural Networks, 15 hr(s)

Competences: ABET-1,ABET-2

Topics: **I:** Multilayer perceptrons (MLP) **II:** Backpropagation and optimizers **III:** Introduction to Keras/TensorFlow

Learning Outcomes **I:** Build simple neural networks [Usage] **II:** Monitor training with TensorBoard [Assessment]

Bibliography: [Goodfellow et al., 2016, Chollet, 2021]

6. Bibliography

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

[Chollet, 2021] Chollet, F. (2021). *Deep Learning with Python*. Manning, 2nd edition.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Géron, 2022] Géron, A. (2022). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly, 3rd edition.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer, 2nd edition.

[Murphy, 2012] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

[Chollet, 2021] Chollet, F. (2021). *Deep Learning with Python*. Manning, 2nd edition.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Géron, 2022] Géron, A. (2022). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly, 3rd edition.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer, 2nd edition.

[Murphy, 2012] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA401 Capstone Project I (Mandatory)

1. General information

- **Semester**: 8th Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA361 Mathematical Modeling (7th Sem-Pg. 449)
  - MA371 Functional analysis I (6th Sem-Pg. 420)
  - 140cr

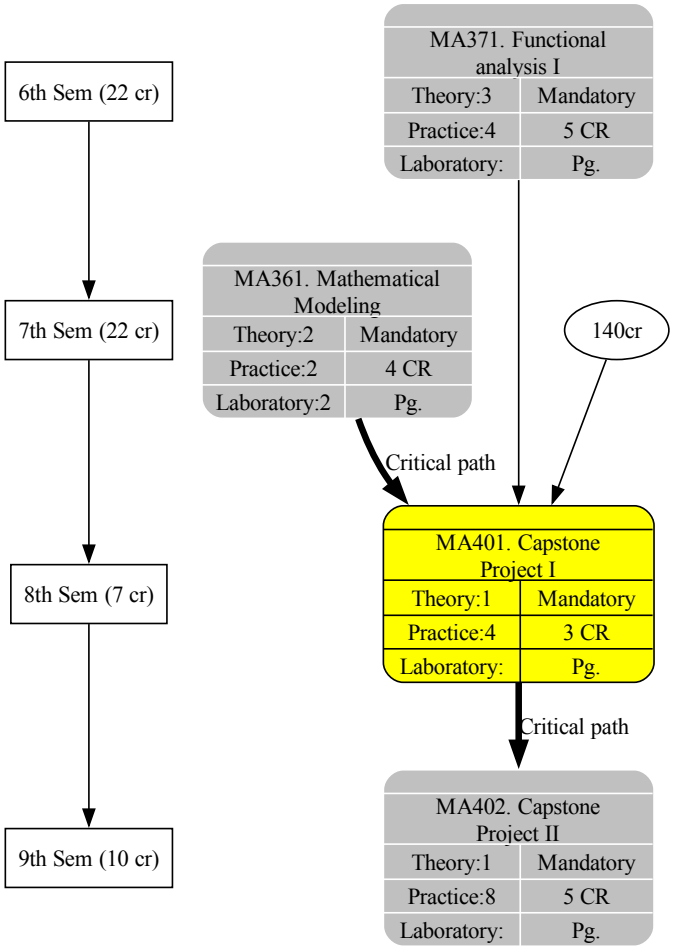


Figure 7.44: Connection Map. MA401 Capstone Project I

2. Course Rationale

This course represents the first semester of the capstone project in computational mathematics. Students develop an applied research project or technological development that integrates and applies the knowledge acquired during the degree program. The course emphasizes problem formulation, literature review, methodological design, and initial development of computational solutions under

the supervision of a faculty advisor. Projects must demonstrate research capability, innovation, and application of computational mathematics to real-world problems.

3. Goals of the Course

- Formulate and delimit a research problem in computational mathematics with theoretical or applied relevance.
- Conduct a systematic literature review and state-of-the-art analysis related to the research problem.
- Design an appropriate methodology to address the problem, selecting suitable computational techniques and tools.
- Develop the initial phases of the project, including prototypes and preliminary analyses.
- Effectively communicate project progress through technical reports and oral presentations.
- Demonstrate autonomous work capability and project management under tutorial supervision.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Assessment)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Assessment)

ABET-4) An ability to communicate effectively with a range of audiences. (Assessment)

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts. (Assessment)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Assessment)

5. Content

5.1 Project Formulation and Planning, 12 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: **I:** Identification and delimitation of research problems **II:** Systematic literature review and state-of-the-art analysis **III:** Formulation of research objectives and hypotheses **IV:** Methodological design and work plan **V:** Development of project proposal and schedule

Learning Outcomes **I:** Formulate research problems with social impact [Assessment] **II:** Design management plans for scientific projects [Usage] **III:** Conduct problem studies using research methods [Usage] **IV:** Evaluate technical and scientific feasibility of projects [Assessment]

Bibliography: [Babbie, 2016, Kothari, 2004]

5.2 Methodological Development and Tools, 12 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: **I:** Selection and justification of computational methods **II:** Design of experiments and validation protocols **III:** Computational tools for project development **IV:** Version management and technical documentation **V:** Ethical considerations in computational research

Learning Outcomes **I:** Apply ethical principles in professional research [Usage] **II:** Work effectively in multidisciplinary teams [Usage] **III:** Communicate technical progress effectively [Usage] **IV:** Select appropriate tools for the project [Assessment]

Bibliography: [Oates, 2006, Pressman, 2014]

5.3 Implementation and Initial Development, 12 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: I: Development of prototypes and proof of concepts **II:** Implementation of algorithms and computational methods **III:** Preliminary analysis of results and validation **IV:** Iteration and refinement of solutions **V:** Technical documentation of development

Learning Outcomes I: Learn independently and continuously [Usage] **II:** Conduct studies using research methods [Usage] **III:** Implement mathematical processes on computer [Usage] **IV:** Adapt solutions based on preliminary results [Assessment]

Bibliography: [Sommerville, 2015, McConnell, 2004]

6. Bibliography

[Babbie, 2016] Babbie, E. R. (2016). *The Basics of Social Research*. Cengage Learning, 7th edition.

[Kothari, 2004] Kothari, C. R. (2004). *Research Methodology: Methods and Techniques*. New Age International, 2nd edition.

[McConnell, 2004] McConnell, S. (2004). *Code Complete: A Practical Handbook of Software Construction*. Microsoft Press, 2nd edition.

[Oates, 2006] Oates, B. J. (2006). *Researching Information Systems and Computing*. SAGE Publications, 1st edition.

[Pressman, 2014] Pressman, R. S. (2014). *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 8th edition.

[Sommerville, 2015] Sommerville, I. (2015). *Software Engineering*. Pearson, 10th edition.

[Babbie, 2016] Babbie, E. R. (2016). *The Basics of Social Research*. Cengage Learning, 7th edition.

[Kothari, 2004] Kothari, C. R. (2004). *Research Methodology: Methods and Techniques*. New Age International, 2nd edition.

[McConnell, 2004] McConnell, S. (2004). *Code Complete: A Practical Handbook of Software Construction*. Microsoft Press, 2nd edition.

[Oates, 2006] Oates, B. J. (2006). *Researching Information Systems and Computing*. SAGE Publications, 1st edition.

[Pressman, 2014] Pressman, R. S. (2014). *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 8th edition.

[Sommerville, 2015] Sommerville, I. (2015). *Software Engineering*. Pearson, 10th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS2S1 Operating systems (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS210 Algorithms and Data Structures (4th Sem-Pg. 367)

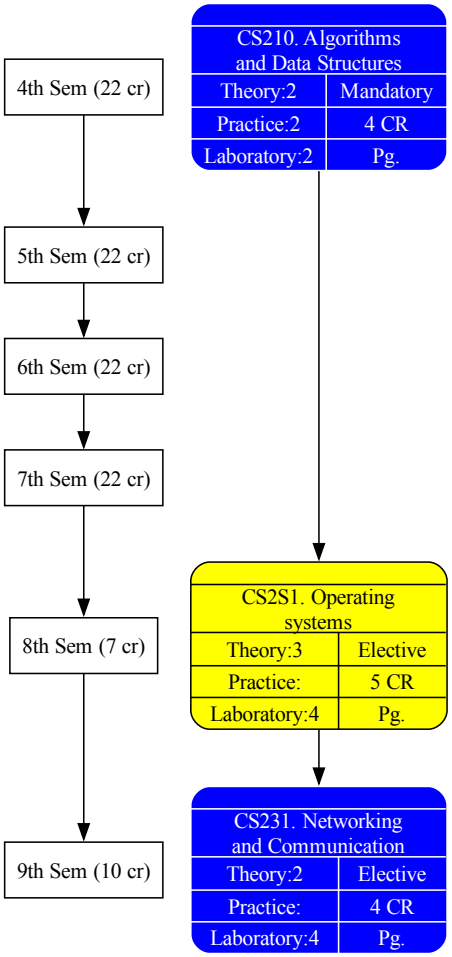


Figure 7.45: Connection Map. CS2S1 Operating systems

2. Course Rationale

The operating system is a collection of services needed to safely interface the hardware with applications. This course explores the mechanisms and policies needed to virtualize computation, memory, and I/O. Students will understand core concepts such as concurrency, persistence, and trust boundaries, which are fundamental for developing robust and efficient software systems.

3. Goals of the Course

- Understand the role and purpose of modern operating systems.

- Master process scheduling and the lifecycle of execution units.
- Analyze memory management, including physical and virtual memory virtualization.
- Evaluate storage systems and file management techniques.
- Comprehend the basics of security and protection within an OS environment.

4. General Competences (*Outcomes*)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Usage)

5. Content

5.1 Role and Purpose of Operating Systems, 8 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Operating systems mediate between general purpose hardware and application-specific software. **II:** Universal operating system functions (e.g., process, user and device interfaces, persistence of data) **III:** Extended and/or specialized operating system functions (e.g., embedded systems, server types such as file, web, multimedia, boot loaders and boot security) **IV:** Design issues (e.g., efficiency, robustness, flexibility, portability, security, compatibility, power, safety, tradeoffs between error checking and performance, flexibility and performance, and security and performance). See also: Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering **V:** Influences of security, networking, multimedia, parallel and distributed computing **VI:** Overarching concern of security/protection: Neglecting to consider security at every layer creates an opportunity to inappropriately access resources. (a) Unauthorized access to files on an unencrypted drive can be achieved by moving the media to another computer. (b) Operating systems enforced security can be defeated by infiltrating the boot layer before the operating system is loaded. (c) Process isolation can be subverted by inadequate authorization checking at API boundaries. (d) Vulnerabilities in system firmware can provide attack vectors that bypass the operating system entirely. (e) Improper isolation of virtual machine memory, computing, and hardware can expose the host system to attacks from guest systems. (f) The operating system may need to mitigate exploitation of hardware and firmware vulnerabilities, leading to potential performance reductions (e.g., Spectre and Meltdown mitigations). **VII:** Exposure of operating systems functions in shells and systems programming. See also: Shell Scripting

Learning Outcomes **I:** Understand the objectives and functions of modern operating systems [Explain] **II:** Evaluate the design issues in different usage scenarios (e.g., real time OS, mobile, server) [Evaluate] **III:** Understand the functions of a contemporary operating system with respect to convenience, efficiency, and the ability to evolve [Explain] **IV:** Understand how evolution and stability are desirable and mutually antagonistic in operating systems function [Explain]

Bibliography: [Silberschatz et al., 2018, Tanenbaum and Bos, 2014]

5.2 Shell Scripting, 8 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Error/exception handling **II:** Piping **III:** System commands: (a) Interface with operating systems **IV:** Environment variables **V:** File abstraction and operators **VI:** Data structures, such as arrays and lists. **VII:** Regular expressions **VIII:** Programs and processes **IX:** Workflow

Learning Outcomes **I:** Create and execute automated scripts to manage various system tasks [Create] **II:** Solve various text processing problems through scripting [Solve]

Bibliography: [Robbins and Beebe, 2005, Silberschatz et al., 2018]

5.3 Scheduling, 16 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Preemptive and non-preemptive scheduling **II:** Schedulers and policies (e.g., first come, first serve, shortest job first, priority, round robin, multilevel). See also: Resource Management **III:** Concepts of Symmetric Multi-Processor (SMP) scheduling and cache coherence. See also: Memory Hierarchy **IV:** Timers (e.g., building many timers out of finite hardware timers). See also: Assembly Level Machine Organization **V:** Fairness and starvation **VI:** Subtopics of operating systems such as energy-aware scheduling and real-time scheduling. See also: Performance and Energy Efficiency, Embedded Characteristics and Architectures, Real-Time, Memory and Verification, Systems, Programming and Sensors, Mobile Platforms: Capabilities and Asynchronous Computing, Mobile Platforms: Architecture and Constraints, Mobile Platforms: Development Fundamentals **VII:** Cooperative scheduling, such as Linux futexes and userland scheduling.

Learning Outcomes **I:** Compare and contrast the common algorithms used for both preemptive and non-preemptive scheduling of tasks in operating systems, such as priority, performance comparison, and fair-share schemes [Compare] **II:** Explain the relationships between scheduling algorithms and application domains [Explain] **III:** Explain the distinctions among types of processor scheduler such as short-term, medium-term, long-term, and I/O [Explain] **IV:** Evaluate a problem or solution to determine appropriateness for asymmetric and/or symmetric multiprocessing [Evaluate] **V:** Evaluate a problem or solution to determine appropriateness as a process vs threads [Evaluate] **VI:** List some contexts benefitting from preemption and deadline scheduling [List] **VII:** Explain the ways that the logic embodied in scheduling algorithms are applicable to other operating systems mechanisms, such as first come first serve or priority to disk I/O, network scheduling, project scheduling, and problems beyond computing [Explain]

Bibliography: [Silberschatz et al., 2018, Stallings, 2017]

5.4 Memory Management, 16 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Review of physical memory, address translation and memory management hardware See also: Memory Hierarchy, Discrete Mathematics **II:** Impact of memory hierarchy including cache concept, cache lookup, and per-CPU caching on operating system mechanisms and policy See also: Memory Hierarchy, Latency, Cache and Memory Hierarchy, Virtualization and Isolation **III:** Logical and physical addressing, address space virtualization. See also: Memory Hierarchy, Discrete Mathematics **IV:** Concepts of paging, page replacement, thrashing and allocation of pages and frames **V:** Allocation/deallocation/storage techniques (algorithms and data structure) performance and flexibility (a) Arenas, slab allocators, free lists, size classes, heterogeneously sized pages (huge pages) **VI:** Memory caching and cache coherence and the effect of flushing the cache to avoid speculative execution vulnerabilities. See also: Functional Organization, Memory Hierarchy, Latency, Cache and Memory Hierarchy, Virtualization and Isolation **VII:** Security mechanisms and concepts in memory management including sandboxing, protection, isolation, and relevant vectors of attack. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles **VIII:** Virtual memory: leveraging virtual memory hardware for OS services and efficiency

Learning Outcomes **I:** Explain memory hierarchy and cost-performance tradeoffs [Explain] **II:** Summarize the principles of virtual memory as applied to caching and paging [Summarize] **III:** Evaluate the tradeoffs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed [Evaluate] **IV:** Describe the reason for and use of cache memory (performance and proximity, how caches complicate isolation and virtual machine abstraction) [Describe] **V:** Code/Develop efficient programs that consider the effects of page replacement and frame

allocation on the performance of a process and the system in which it executes [Create] **VI:** Explain how hardware is utilized for efficient virtualization [Explain]

Bibliography: [Silberschatz et al., 2018, Arpaci-Dusseau and Arpaci-Dusseau, 2018]

5.5 File Systems API and Implementation, 12 hr(s)

Competences: ABET-2,ABET-6

Topics: I: Concept of a file including data, metadata, operations, and access-mode **II:** File system mounting **III:** File access control **IV:** File sharing **V:** Basic file allocation methods, including linked allocation table **VI:** File system structures comprising file allocation including various directory structures and methods for uniquely identifying files (e.g., name, identified or metadata storage location) **VII:** Allocation/deallocation/storage techniques (algorithms and data structure) impact on performance and flexibility (i.e., internal and external fragmentation and compaction) **VIII:** Free space management such as using bit tables vs linking **IX:** Implementation of directories to segment and track file location

Learning Outcomes I: Explain the choices to be made in designing file systems [Explain] **II:** Evaluate different approaches to file organization, recognizing the strengths and weaknesses of each [Evaluate] **III:** Apply software constructs appropriately given knowledge of the file system implementation [Apply]

Bibliography: [Silberschatz et al., 2018, Stallings, 2017]

5.6 Protection and Safety, 12 hr(s)

Competences: ABET-2,ABET-6

Topics: I: Overview of operating system security mechanisms. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles **II:** Attacks and antagonism (scheduling, etc.). See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles **III:** Review of major vulnerabilities in real operating systems. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles **IV:** Operating systems mitigation strategies such as backups. See also: System Reliability **V:** Policy/mechanism separation. See also: Risk and Data Management, Security Governance, Management and Policy **VI:** Security methods and devices. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles (a) Rings of protection (history from Multics to virtualized x86) (b) x86_64 rings -1 and -2 (hypervisor and ME/PSP) **VII:** Protection, access control, and authentication. See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles, Protocols, Applications and Advanced Cryptography, Cryptographic Algorithms, Cryptography Fundamentals

Learning Outcomes I: Understand the requirement for protection and security mechanisms in operating systems [Explain] **II:** List and describe the attack vectors that leverage OS vulnerabilities [List] **III:** Understand the mechanisms available in an OS to control access to resources [Explain] **IV:** Summarize the features and limitations of an operating system that impact protection and security [Summarize]

Bibliography: [Silberschatz et al., 2018, Tanenbaum and Bos, 2014]

6. Bibliography

[Arpaci-Dusseau and Arpaci-Dusseau, 2018] Arpaci-Dusseau, R. H. and Arpaci-Dusseau, A. C. (2018). *Operating Systems: Three Easy Pieces*. Arpaci-Dusseau Books, 1.0 edition.

[Robbins and Beebe, 2005] Robbins, A. and Beebe, N. H. (2005). *Classic Shell Scripting*. O'Reilly Media.

[Silberschatz et al., 2018] Silberschatz, A., Galvin, P. B., and Gagne, G. (2018). *Operating System Concepts*. Wiley, 10th edition.

[Stallings, 2017] Stallings, W. (2017). *Operating Systems: Internals and Design Principles*. Pearson, 9th edition.

[Tanenbaum and Bos, 2014] Tanenbaum, A. S. and Bos, H. (2014). *Modern Operating Systems*. Pearson, 4th edition.

[Arpaci-Dusseau and Arpaci-Dusseau, 2018] Arpaci-Dusseau, R. H. and Arpaci-Dusseau, A. C. (2018). *Operating Systems: Three Easy Pieces*. Arpaci-Dusseau Books, 1.0 edition.

[Robbins and Beebe, 2005] Robbins, A. and Beebe, N. H. (2005). *Classic Shell Scripting*. O'Reilly Media.

[Silberschatz et al., 2018] Silberschatz, A., Galvin, P. B., and Gagne, G. (2018). *Operating System Concepts*. Wiley, 10th edition.

[Stallings, 2017] Stallings, W. (2017). *Operating Systems: Internals and Design Principles*. Pearson, 9th edition.

[Tanenbaum and Bos, 2014] Tanenbaum, A. S. and Bos, H. (2014). *Modern Operating Systems*. Pearson, 4th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS3I1 Computer Security (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS210 Algorithms and Data Structures (4th Sem-Pg. 367)

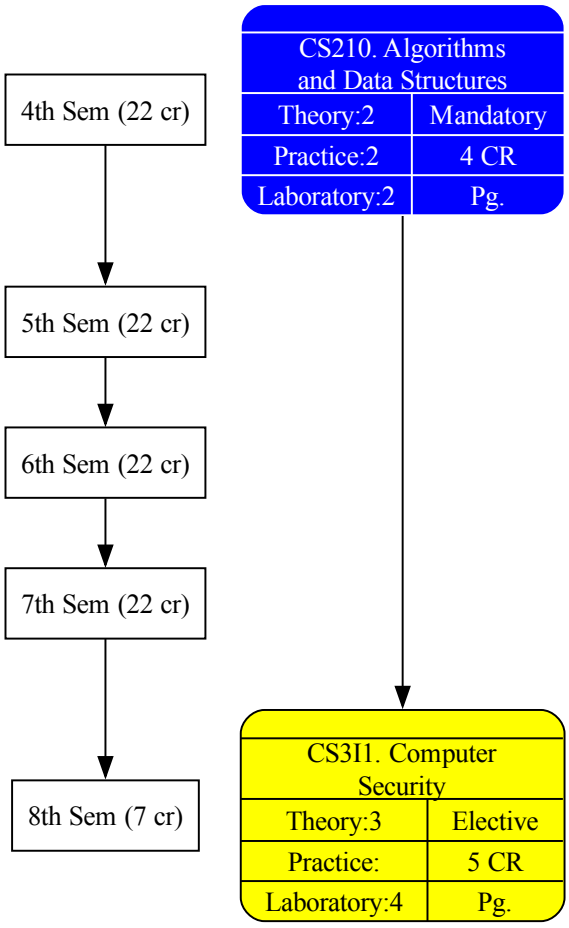


Figure 7.46: Connection Map. CS3I1 Computer Security

2. Course Rationale

In an increasingly digital world, security is no longer an optional feature but a fundamental requirement. This course provides a comprehensive introduction to the principles and practices of cybersecurity, covering everything from theoretical foundations to secure coding and digital forensics. Students will gain the skills necessary to design, implement, and govern secure systems while understanding the ethical and professional responsibilities involved in protecting information and infrastructure.

3. Goals of the Course

- Understand the fundamental principles and mathematical bases of security.
- Apply secure coding techniques and cryptographic methods to software.
- Analyze system vulnerabilities and perform forensic investigations.
- Evaluate security governance and professional ethics in cybersecurity.

4. General Competences (*Outcomes*)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Security Mindset and Principles, 6 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Developing a security mindset incorporating crosscutting concepts: confidentiality, integrity, availability, risk assessment, systems thinking, adversarial thinking, human-centered thinking. **II:** Vulnerabilities, threats, attack surfaces, and attack vectors Protection and Safety. **III:** Denial of Service (DoS) and Distributed Denial of Service (DDoS) Protection and Safety. **IV:** Principles and practices of protection, e.g., least privilege, open design, fail-safe defaults, defense in depth, and zero trust; and how they can be implemented Principles of Operating System, Protection and Safety, Coding Practices, Large-Scale Construction and Process, Computer Crimes and Attacks, Security Laws, Policies and Responsibility. **V:** Optimization considerations between security, privacy, performance, and other design goals System Design, Software Development Practices, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices. **VI:** Concepts of trust and trustworthiness.

Learning Outcomes **I:** Evaluate a system for possible attacks that can be launched by an adversary [Evaluate] **II:** Design and develop approaches to protect a system from a set of identified threats [Design] **III:** Describe how harm to user privacy can be avoided [Describe] **IV:** Develop a system that incorporates various principles of security and privacy [Create]

Bibliography: [Pfleeger et al., 2015, Stallings, 2017]

5.2 Cryptography Fundamentals, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Differences between algorithmic, applied, and mathematical views of cryptography. **II:** Mathematical preliminaries: modular arithmetic, Euclidean algorithm, probabilistic independence, linear algebra basics, number theory, finite fields, complexity, asymptotic analysis Discrete Mathematics, Linear Algebra. **III:** Basic cryptography: symmetric key and public key cryptography Foundational Data Structures, Foundational Algorithms, Advanced Algorithms, Discrete Mathematics. **IV:** Classical cryptosystems, such as shift, substitution, transposition ciphers, code books, and machines Discrete Mathematics. **V:** Kerckhoff's principle and use of vetted libraries Coding Practices, Large-Scale Construction and Process.

Learning Outcomes **I:** Explain the role of cryptography in supporting security and privacy [Explain] **II:** Discuss the risks of inventing one's own cryptographic methods [Debate] **III:** Discuss the importance of prime numbers in cryptography and explain their use in cryptographic algorithms [Debate] **IV:** Implement and cryptanalyze classical ciphers [Implement]

Bibliography: [Stallings, 2017]

5.3 Injection and Input Validation, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Common vulnerabilities and weaknesses. **II:** SQL injection and other injection attacks. **III:** Cross-site scripting techniques and mitigations. **IV:** Input validation and data sanitization Protection and Safety, Basic Programming Concepts and Constructs, Programming Practices: Testing, Debugging, and Documentation, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.

Learning Outcomes **I:** Identify underlying problems in given examples of an enumeration of common weaknesses and explain how they can be circumvented [Analyze] **II:** Apply input validation and data sanitization techniques to enhance security of a program [Apply] **III:** Evaluate and prevent SQL injections attacks on a database application [Evaluate] **IV:** Evaluate and prevent cross-site scripting attacks against a website [Evaluate] **V:** Construct a program that tests for all input handling errors [Create]

Bibliography: [Pfleeger et al., 2015]

5.4 Malware Analysis and Advanced Security, 6 hr(s)

Competences: ABET-2

Topics: **I:** Offensive security techniques as a defense. **II:** AI-assisted malware detection techniques. **III:** Ransomware: creation, prevention, and mitigation. **IV:** Secure use of third-party components Coding Practices, Large-Scale Construction and Process, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices. **V:** Malware: varieties, creation, reverse engineering, and defense against them Language Translation and Execution, Systems Execution and Memory Model. **VI:** Assurance: testing (including fuzzing and penetration testing), verification, and validation Protection and Safety, Basic Programming Concepts and Constructs, Programming Practices: Testing, Debugging, and Documentation, Coding Practices, Large-Scale Construction and Process, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices. **VII:** Static and dynamic analyses Program Analysis and Analyzers, Protection and Safety, Evaluation, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices. **VIII:** Secure compilers and secure code generation Language Translation and Execution, Run-time Behavior and Systems.

Learning Outcomes **I:** Describe different kinds of malicious software [Describe] **II:** Explain the risks of misusing interfaces with third-party code and how to correctly use third-party code [Explain] **III:** Discuss the need to update software to fix security vulnerabilities and the lifecycle management of the fix [Debate] **IV:** Apply static and dynamic tools to identify programming faults [Apply] **V:** Evaluate a system for the existence of malware and remove it [Evaluate] **VI:** Implement preventive techniques to reduce the occurrence of ransomware [Implement]

Bibliography: [Sikorski and Honig, 2012, Eilam, 2005, National Institute of Standards and Technology, 2020]

5.5 Threat Analysis and Security Engineering, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Security engineering goals: building systems that remain dependable despite errors, accidents, or malicious adversaries Coding Practices, Large-Scale Construction and Process, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices, Computer Crimes and Attacks, Security Laws, Policies and Responsibility. **II:** Problem analysis and situational analysis to address system security Performance Testing and Benchmarking, Testing Tools and Analysis,

Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices. **III:** Security analysis, covering security requirements analysis; security controls analysis; threat analysis; and vulnerability analysis Program Analysis and Analyzers, Evaluation. **IV:** Security attack domains and attack surfaces, e.g., communications and networking, hardware, physical, social engineering, software, and supply chain Network Security. **V:** Security attack modes, techniques, and tactics, e.g., authentication abuse; brute force; buffer manipulation; code injection; content insertion; denial of service; eavesdropping; function bypass; impersonation; integrity attack; interception; phishing; protocol analysis; privilege abuse; spoofing; and traffic injection Network Security, Protection and Safety, Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices.

Learning Outcomes I: Create a threat model for a system or system design [Create] **II:** Apply situational analysis to develop secure solutions under a specified scenario [Apply] **III:** Identify and mitigate security vulnerabilities and weaknesses in a system [Analyze]

Bibliography: [Anderson, 2020]

5.6 Digital Forensics Techniques, 6 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Basic principles and methodologies for digital forensics. **II:** System design for forensics. **III:** Forensics in different situations: operating systems, file systems, application forensics, web forensics, network forensics, mobile device forensics, use of database auditing Network Security. **IV:** Attacks on forensics and preventing such attacks.

Learning Outcomes I: Explain what a digital investigation is and how it can be implemented Computer Crimes and Attacks, Security Laws, Policies and Responsibility [Explain] **II:** Design and implement software to support forensics [Design] **III:** Extract data from a hard drive to comply with the law Computer Crimes and Attacks, Security Laws, Policies and Responsibility [Extract] **IV:** Recover data based on a given search term from an imaged system [Analyze] **V:** Reconstruct data and events from an application history, or a web artifact, or a cloud database, or a mobile device. Mobile Platforms: Capabilities and Asynchronous Computing, Mobile Platforms: Architecture and Constraints, Mobile Platforms: Development Fundamentals, Cloud Architecture and Data Management, Frontend, Frameworks and Web Standards [Apply] **VI:** Capture and analyze network traffic. Network Security [Analyze] **VII:** Develop approaches to address the challenges associated with mobile device forensics [Create] **VIII:** Apply forensics tools to investigate security breaches [Apply] **IX:** Identify and mitigate anti-forensic methods [Analyze]

Bibliography: [Pfleeger et al., 2015]

5.7 Security Governance, Management and Policy, 6 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Protecting critical assets from threats. **II:** Security governance: organizational objectives and general risk assessment. **III:** Security management: achieve and maintain appropriate levels of confidentiality, integrity, availability, accountability, authenticity, and reliability Performance Testing and Benchmarking, Testing Tools and Analysis, Verification and validation Concepts, Test Planning and Types, Advanced Testing Practices. **IV:** Security policy: organizational policies, issue-specific policies, system-specific policies.

Learning Outcomes I: Describe critical assets and how they can be protected [Describe] **II:** Differentiate between security governance, management, and controls, giving examples of each [Differentiate] **III:** Describe a technical control and implement it to mitigate specific threats [Describe]

Bibliography: [Anderson, 2020]

5.8 Security Ethics, Law and Society, 6 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Societal impacts on breakdowns in security and privacy Digital Equity and Accessibility, Social Impact and AI, Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Computer Crimes and Attacks, Security Laws, Policies and Responsibility. **II:** Applicability of laws and regulations on security and privacy Computer Crimes and Attacks, Security Laws, Policies and Responsibility. **III:** Professional ethical considerations when designing secure systems and maintaining privacy; ethical hacking Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Ethics Codes and Legal Accountability, Professional Responsibility and Public Policy, Professional Values, Nature and Continuing Development, Ethical Dissent and Cultural Context, Workplace Ethics, Computer Crimes and Attacks, Security Laws, Policies and Responsibility. **IV:** Security and privacy implications of malicious AI/machine learning actors, e.g., identifying deep fakes Machine Learning Formulation and Evaluation, Deep Learning, RL and Ethics, Introduction to AI: Concepts and Intelligent Behavior, Problem Characteristics and Agent Nature, Machine Learning Fundamentals, Machine Learning Data, Representations and Evaluation, Neural Networks and Ethics, Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Computer Crimes and Attacks, Security Laws, Policies and Responsibility. **V:** Societal impacts of Internet of Things (IoT) devices and other emerging technologies on security and privacy Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation, Computer Crimes and Attacks, Security Laws, Policies and Responsibility.

Learning Outcomes **I:** Calculate the impact of a breakdown in security of a given system [Calculate] **II:** Construct a system that conforms to security laws [Create] **III:** Apply a set of privacy regulations to design a system that protects privacy [Apply] **IV:** Evaluate the legal ramifications of a system not corresponding to applicable laws and regulations [Evaluate]

Bibliography: [Anderson, 2020, Pfleeger et al., 2015]

6. Bibliography

[Anderson, 2020] Anderson, R. (2020). *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley, 3rd edition.

[Eilam, 2005] Eilam, E. (2005). *Reversing: Secrets of Reverse Engineering*. Wiley, Indianapolis, IN.

[National Institute of Standards and Technology, 2020] National Institute of Standards and Technology (2020). Security and privacy controls for information systems and organizations. Technical Report NIST SP 800-53 Rev. 5, National Institute of Standards and Technology, Gaithersburg, MD. Disponible en: `https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final`.

[Pfleeger et al., 2015] Pfleeger, C. P., Pfleeger, S. L., and Margulies, J. (2015). *Security in Computing*. Prentice Hall, 5th edition.

[Sikorski and Honig, 2012] Sikorski, M. and Honig, A. (2012). *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software*. No Starch Press, San Francisco, CA.

[Stallings, 2017] Stallings, W. (2017). *Computer Security: Principles and Practice*. Pearson, 4th edition.

[Anderson, 2020] Anderson, R. (2020). *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley, 3rd edition.

[Eilam, 2005] Eilam, E. (2005). *Reversing: Secrets of Reverse Engineering*. Wiley, Indianapolis, IN.

[National Institute of Standards and Technology, 2020] National Institute of Standards and Technology (2020). Security and privacy controls for information systems and organizations. Technical Report NIST SP 800-53 Rev. 5, National Institute of Standards and Technology, Gaithersburg, MD. Disponible en: `https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final`.

[Pfleeger et al., 2015] Pfleeger, C. P., Pfleeger, S. L., and Margulies, J. (2015). *Security in Computing*. Prentice Hall, 5th edition.

[Sikorski and Honig, 2012] Sikorski, M. and Honig, A. (2012). *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software*. No Starch Press, San Francisco, CA.

[Stallings, 2017] Stallings, W. (2017). *Computer Security: Principles and Practice*. Pearson, 4th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

QC251 Quantum Mechanics (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 4 hours; **Practice**: 2 hours;
- **Prerrequisites**:
  - FI101 General Physics (2nd Sem-Pg. 340)
  - MA371 Functional analysis I (6th Sem-Pg. 420)

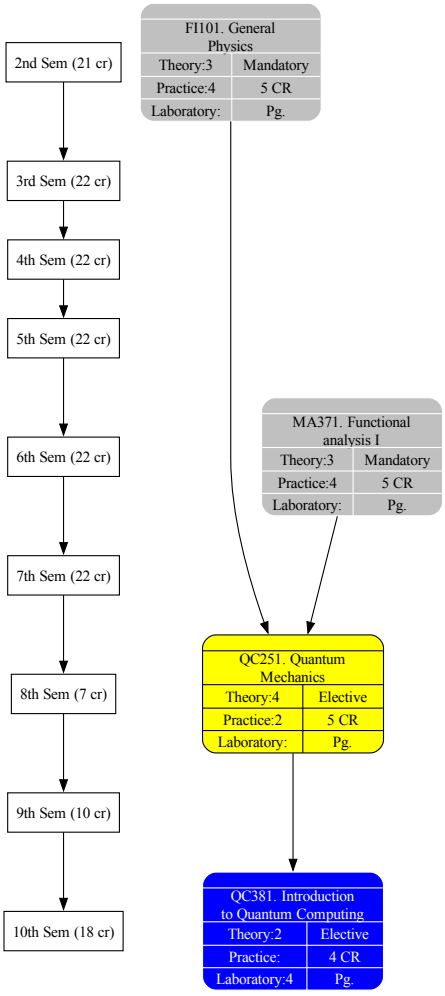


Figure 7.47: Connection Map. QC251 Quantum Mechanics

2. Course Rationale

Quantum mechanics is fundamental for understanding the physical principles that govern atomic and subatomic systems, which form the basis of modern computational technologies such as quantum computing, quantum cryptography, semiconductors, and solid-state devices. This course provides the mathematical and conceptual foundations of quantum mechanics, with special emphasis on its applications in computer science, including quantum algorithms, quantum simulation, and quantum hardware development.

3. Goals of the Course

- Understand the fundamental postulates of quantum mechanics and their mathematical formulation.
- Apply quantum theory to solve basic problems in few-level quantum systems.
- Relate quantum concepts to applications in quantum computing and information technologies.
- Develop the ability to simulate simple quantum systems using computational tools.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Fundamentals of Quantum Mechanics, 12 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Crisis of classical physics: blackbody radiation, photoelectric effect. **II:** Wave-particle duality and complementarity principle. **III:** Postulates of quantum mechanics: wave function, operators, and expectation values. **IV:** Time-dependent and time-independent Schrödinger equation. **V:** Born's probabilistic interpretation.

Learning Outcomes **I:** Explain the limitations of classical physics that motivated the development of quantum mechanics. [Familiarity] **II:** State the fundamental postulates of quantum mechanics. [Familiarity] **III:** Interpret the physical meaning of the wave function and its relation to probabilities. [Usage] **IV:** Formulate simple problems using the Schrödinger equation. [Assessment]

Bibliography: [Griffiths and Schroeter, 2018, Nielsen and Chuang, 2010]

5.2 Simple Quantum Systems, 14 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Particle in a one-dimensional and three-dimensional box. **II:** Quantum harmonic oscillator: algebraic solution and Hermite polynomials. **III:** Quantum tunneling through potential barriers. **IV:** Finite potential well and bound states. **V:** Ladder operators for the harmonic oscillator.

Learning Outcomes **I:** Solve the Schrödinger equation analytically for simple quantum systems. [Familiarity] **II:** Calculate energy levels and wave functions for symmetric potential wells. [Usage] **III:** Explain the quantum tunneling phenomenon and its technological applications. [Assessment] **IV:** Use ladder operators to manipulate quantum states. [Assessment]

Bibliography: [Griffiths and Schroeter, 2018, McIntyre et al., 2012]

5.3 Linear Algebra in Quantum Mechanics, 10 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Hilbert spaces and Dirac notation. **II:** Linear operators, Hermitian and unitary operators. **III:** Eigenvalues and eigenvectors in quantum systems. **IV:** Matrix representations of quantum operators. **V:** Tensor product and composite spaces.

Learning Outcomes I: Use Dirac notation to represent quantum states. [Familiarity] **II:** Differentiate between Hermitian, unitary, and projection operators. [Usage] **III:** Construct matrix representations of simple quantum operators. [Usage] **IV:** Apply the tensor product to describe composite quantum systems. [Assessment]

Bibliography: [Shankar, 2012, Nielsen and Chuang, 2010]

5.4 Angular Momentum and Spin, 8 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Orbital angular momentum: operators and eigenfunctions. **II:** Spin: the intrinsic degree of freedom. **III:** Pauli matrices and spin-1/2 representation. **IV:** Coupling of angular momenta. **V:** Identical particles and Pauli exclusion principle.

Learning Outcomes I: Calculate angular momentum eigenvalues for simple systems. [Familiarity] **II:** Manipulate spin states using Pauli matrices. [Usage] **III:** Apply the Pauli exclusion principle to fermion systems. [Assessment] **IV:** Explain the importance of spin in technologies such as magnetic resonance. [Assessment]

Bibliography: [Griffiths and Schroeter, 2018, McIntyre et al., 2012]

5.5 Perturbation Theory, 10 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Time-independent perturbations: nondegenerate and degenerate theory. **II:** Corrections to energy levels and wave functions. **III:** Time-dependent perturbations: Fermi's golden rule. **IV:** Radiation-induced transitions. **V:** Variational method for ground states.

Learning Outcomes I: Apply nondegenerate perturbation theory to simple quantum systems. [Familiarity] **II:** Calculate energy level corrections due to small perturbations. [Usage] **III:** Use Fermi's golden rule to calculate transition rates. [Assessment] **IV:** Apply the variational method to estimate ground state energies. [Assessment]

Bibliography: [Griffiths and Schroeter, 2018, Shankar, 2012]

5.6 Applications in Quantum Computing, 10 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Qubits: representation and basic operations. **II:** Quantum gates and elementary quantum circuits. **III:** Quantum entanglement and no-cloning theorem. **IV:** Simple quantum algorithms: Deutsch-Jozsa and quantum teleportation. **V:** Simulation of quantum systems on classical computers. **VI:** Implementation of quantum algorithms using frameworks like Qiskit or Cirq.

Learning Outcomes I: Represent and manipulate qubits using Dirac notation. [Familiarity] **II:** Explain the concept of quantum entanglement and its applications. [Usage] **III:** Implement simple quantum circuits using quantum programming frameworks. [Usage] **IV:** Simulate few-qubit quantum systems in computational environments. [Assessment] **V:** Describe the fundamental advantages of quantum computing over classical computing. [Assessment]

Bibliography: [Nielsen and Chuang, 2010, Rieffel and Polak, 2011]

6. Bibliography

[Griffiths and Schroeter, 2018] Griffiths, D. J. and Schroeter, D. F. (2018). *Introduction to Quantum Mechanics*. Cambridge University Press, 3rd edition. Texto estándar para cursos introductorios de mecánica cuántica, con explicaciones claras y problemas bien seleccionados.

[McIntyre et al., 2012] McIntyre, D. H., Manogue, C. A., and Tate, J. (2012). *Quantum Mechanics: A Paradigms Approach*. Pearson. Enfoque moderno con aplicaciones experimentales y conexiones con tecnologías actuales.

[Nielsen and Chuang, 2010] Nielsen, M. A. and Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition. La biblia de la computación cuántica, conecta teoría cuántica con aplicaciones en computación e información.

[Rieffel and Polak, 2011] Rieffel, E. G. and Polak, W. H. (2011). *Quantum Computing: A Gentle Introduction*. MIT Press. Introducción accesible a la computación cuántica, ideal para estudiantes de ciencias de la computación.

[Shankar, 2012] Shankar, R. (2012). *Principles of Quantum Mechanics*. Springer, 2nd edition. Excelente para el formalismo matemático y tratamiento riguroso de los fundamentos.

[Griffiths and Schroeter, 2018] Griffiths, D. J. and Schroeter, D. F. (2018). *Introduction to Quantum Mechanics*. Cambridge University Press, 3rd edition. Texto estándar para cursos introductorios de mecánica cuántica, con explicaciones claras y problemas bien seleccionados.

[McIntyre et al., 2012] McIntyre, D. H., Manogue, C. A., and Tate, J. (2012). *Quantum Mechanics: A Paradigms Approach*. Pearson. Enfoque moderno con aplicaciones experimentales y conexiones con tecnologías actuales.

[Nielsen and Chuang, 2010] Nielsen, M. A. and Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition. La biblia de la computación cuántica, conecta teoría cuántica con aplicaciones en computación e información.

[Rieffel and Polak, 2011] Rieffel, E. G. and Polak, W. H. (2011). *Quantum Computing: A Gentle Introduction*. MIT Press. Introducción accesible a la computación cuántica, ideal para estudiantes de ciencias de la computación.

[Shankar, 2012] Shankar, R. (2012). *Principles of Quantum Mechanics*. Springer, 2nd edition. Excelente para el formalismo matemático y tratamiento riguroso de los fundamentos.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS253 Exploratory data analysis (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA351 Statistics and Probabilities (6th Sem-Pg. 415)

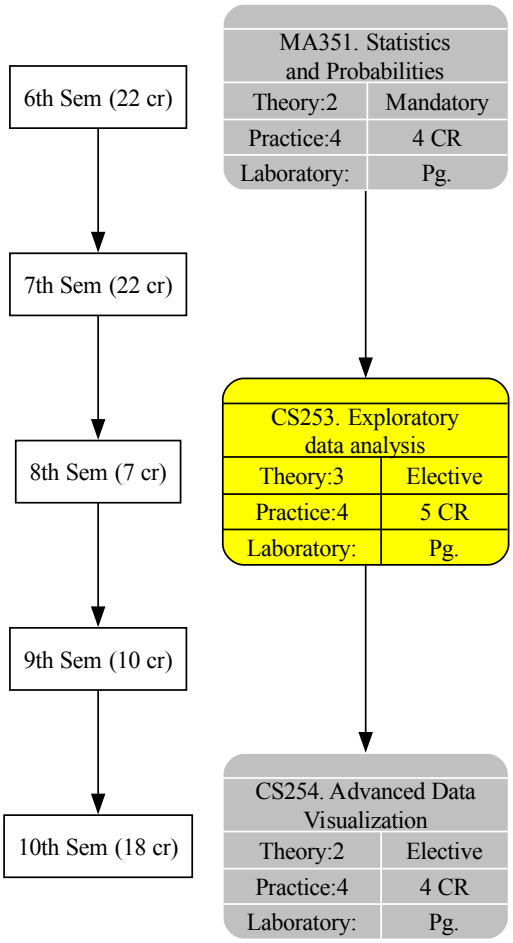


Figure 7.48: Connection Map. CS253 Exploratory data analysis

2. Course Rationale

Exploratory Data Analysis (EDA) is a critical stage in any data science project. This course provides the tools to understand the structure, detect anomalies, and extract initial patterns from data before formal modeling. It focuses on the use of descriptive statistics, data cleaning, and preliminary visualization to validate assumptions and guide the selection of complex models, ensuring the integrity and quality of scientific findings.

3. Goals of the Course

- Apply data cleaning and preprocessing techniques to ensure data quality.
- Use descriptive statistical methods to summarize complex datasets.
- Identify patterns, outliers, and relationships between variables through visual analysis.
- Formulate hypotheses based on the initial exploration of the data.
- Communicate preliminary findings effectively through technical reports.

4. General Competences (*Outcomes*)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Foundations of Exploratory Data Analysis, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Philosophy and objectives of exploratory analysis. **II:** Data types: structured, semi-structured, and unstructured. **III:** Measurement scales: nominal, ordinal, interval, and ratio. **IV:** Workflow in data exploration.

Learning Outcomes I: Identify the importance of EDA in the data life cycle [Familiarity]. **II:** Classify variables according to their measurement scale and type [Usage].

Bibliography: [Tukey, 1977, Peng, 2016]

5.2 Data Cleaning and Preparation, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: "Tidy Data" concept and table manipulation. **II:** Handling missing values: imputation versus removal. **III:** Detection and treatment of outliers. **IV:** Normalization and standardization of numerical variables.

Learning Outcomes I: Implement cleaning routines for noisy datasets [Usage]. **II:** Evaluate the impact of different imputation strategies [Assessment].

Bibliography: [Wickham, 2014, McKinney, 2022]

5.3 Univariate Descriptive Statistics, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Measures of central tendency, dispersion, and shape. **II:** Frequency distributions and density functions. **III:** Skewness and kurtosis analysis in real variables. **IV:** Univariate visualization: Histograms, Boxplots, and Kernel Density Estimation (KDE).

Learning Outcomes I: Numerically summarize the distribution of a variable [Usage]. **II:** Interpret density plots to identify biases in the data [Assessment].

Bibliography: [Healy, 2018, Tukey, 1977]

5.4 Bivariate and Multivariate Exploration, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Pearson, Spearman, and Kendall correlation. **II:** Contingency tables and measures of categorical association. **III:** Scatter plots, correlation matrices, and heatmaps. **IV:** Conceptual introduction to dimensionality reduction (PCA).

Learning Outcomes I: Quantify the relationship between multiple variables in a dataset [Usage]. **II:** Use heatmaps to detect collinearity in datasets [Assessment].

Bibliography: [Healy, 2018, Wickham and Grolemund, 2016]

5.5 Computational Tools for Exploration, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Computational ecosystems for analysis (Python/Pandas or R/Tidyverse). **II:** Efficient manipulation of large-volume DataFrames. **III:** Automatic generation of data profiling reports.

Learning Outcomes I: Implement automated EDA processes through programming [Usage]. **II:** Optimize data processing for efficient analysis [Assessment].

Bibliography: [McKinney, 2022, Wickham and Grolemund, 2016]

5.6 Exploratory Hypothesis Testing, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Normality tests and variance assumptions. **II:** Group comparison: T-test and ANOVA in an exploratory context. **III:** Non-parametric methods and resampling (Bootstrapping).

Learning Outcomes I: Validate statistical assumptions necessary for subsequent modeling [Usage]. **II:** Design exploratory experiments to confirm visual findings [Assessment].

Bibliography: [Peng, 2016, Healy, 2018]

5.7 Data Ethics and Responsibility, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Identification of selection and confirmation biases in data. **II:** Data privacy: Anonymization and masking techniques. **III:** Ethics in communicating preliminary findings and associated risks.

Learning Outcomes I: Identify potential ethical biases in real-world datasets [Familiarity]. **II:** Apply basic anonymization techniques during analysis [Usage].

Bibliography: [Healy, 2018]

5.8 Result Communication and Data Storytelling, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Structure of a technical data quality report. **II:** Storytelling based on exploratory discovery. **III:** Selection of key visualizations for decision-making.

Learning Outcomes I: Write effective technical reports on data quality [Assessment]. **II:** Perform presentations of exploratory findings to technical audiences [Assessment].

Bibliography: [Wickham and Grolemund, 2016, Healy, 2018]

6. Bibliography

[Healy, 2018] Healy, K. (2018). *Data Visualization: A Practical Introduction*. Princeton University Press.

[McKinney, 2022] McKinney, W. (2022). *Python for Data Analysis*. O'Reilly Media, 3rd edition.

[Peng,] Peng, R. D. *Exploratory Data Analysis with R*. Leanpub.

[Tukey, 1977] Tukey, J. W. (1977). *Exploratory Data Analysis*. Addison-Wesley.

[Wickham, 2014] Wickham, H. (2014). Tidy data. *Journal of Statistical Software*, 59(10):1–23.

[Wickham and Grolemund, 2016] Wickham, H. and Grolemund, G. (2016). *R for Data Science*. O'Reilly Media.

[Healy, 2018] Healy, K. (2018). *Data Visualization: A Practical Introduction*. Princeton University Press.

[McKinney, 2022] McKinney, W. (2022). *Python for Data Analysis*. O'Reilly Media, 3rd edition.

[Peng, 2016] Peng, R. D. (2016). *Exploratory Data Analysis with R*. Leanpub.

[Tukey, 1977] Tukey, J. W. (1977). *Exploratory Data Analysis*. Addison-Wesley.

[Wickham, 2014] Wickham, H. (2014). Tidy data. *Journal of Statistical Software*, 59(10):1–23.

[Wickham and Grolemund, 2016] Wickham, H. and Grolemund, G. (2016). *R for Data Science*. O'Reilly Media.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS370 Big Data and Analytics (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS271 Data Management (4th Sem-Pg. 372)

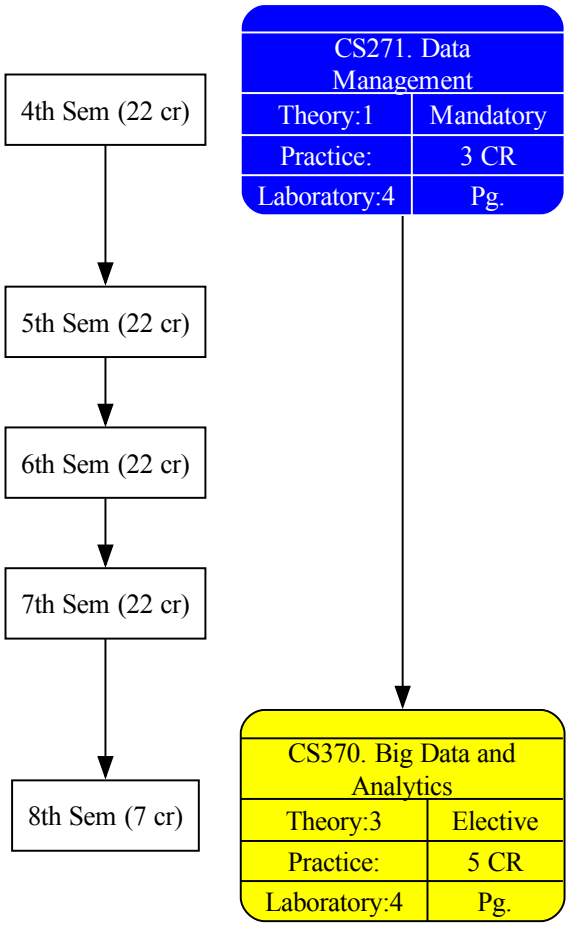


Figure 7.49: Connection Map. CS370 Big Data and Analytics

2. Course Rationale

In the current era, the volume, velocity, and variety of data exceed the capabilities of traditional storage and processing systems. This course introduces the fundamental concepts and technologies of Big Data. It covers distributed file systems, parallel processing frameworks, NoSQL databases, and real-time data streaming. Students will learn to design scalable architectures to extract value from massive datasets.

3. Goals of the Course

- Understand the 5 V's of Big Data and their impact on system design.
- Design and implement scalable solutions using the MapReduce paradigm.
- Master the use of distributed file systems like HDFS.
- Analyze different architectural patterns for Big Data (Lambda and Kappa).
- Apply stream processing techniques for real-time data analysis.

4. General Competences (*Outcomes*)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Core Database System Concepts, 12 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Purpose and advantages of database systems **II:** Components of database systems **III:** Design of core DBMS functions (e.g., query mechanisms, transaction management, buffer management, access methods) **IV:** Database architecture, data independence, and data abstraction **V:** Transaction management **VI:** Normalization **VII:** Approaches for managing large volumes of data (e.g., NoSQL database systems, use of MapReduce) **VIII:** How to support CRUD-only applications **IX:** Distributed databases/cloud-based systems **X:** Structured, semi-structured, and unstructured data **XI:** Use of a declarative query language **XII:** Systems supporting structured and/or stream content

Learning Outcomes **I:** Identify at least four advantages that using a database system provides [Analyze] **II:** Enumerate the components of a (relational) database system [Enumerate] **III:** Follow a query as it is processed by the components of a (relational) database system [Analyze] **IV:** Defend the value of data independence [Defend] **V:** Compose a simple select-project-join query in SQL [Compose] **VI:** Enumerate the four properties of a correct transaction manager [Enumerate] **VII:** Describe the advantages for eliminating duplicate repeated data [Describe] **VIII:** Outline how MapReduce uses parallelism to process data efficiently [Outline] **IX:** Evaluate the differences between structured and semi/unstructured databases [Evaluate]

Bibliography: [White, 2015, Karau et al., 2017]

5.2 Distributed Databases/Cloud Computing, 12 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Distributed DBMS: (a) Distributed data storage (b) Distributed query processing (c) Distributed transaction model (d) Homogeneous and heterogeneous solutions (e) Client-server distributed databases **II:** Parallel DBMS: (a) Parallel DBMS architectures: shared memory, shared disk, shared nothing (b) Speedup and scale-up, e.g., use of the MapReduce processing model (c) Data replication and weak consistency models

Learning Outcomes **I:** Describe the key components of a distributed DBMS, including distributed data storage, query processing, and transaction management [Describe] **II:** Analyze the tradeoffs between parallel DBMS architectures: shared memory, shared disk, and shared nothing [Analyze] **III:** Describe data replication strategies and weak consistency models in distributed database systems [Describe]

Bibliography: [White, 2015, Lin and Dyer, 2010]

5.3 NoSQL Systems, 12 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Why NoSQL? (e.g., Impedance mismatch between Application [CRUD] and RDBMS) **II:** Key-Value and Document data model **III:** Storage systems (e.g., Key-Value systems, Data Lakes) **IV:** Distribution Models (Sharding and Replication) **V:** Graph Databases **VI:** Consistency Models (Update and Read, Quorum consistency, CAP theorem) **VII:** Processing model (e.g., Map-Reduce, multi-stage map-reduce, incremental map-reduce) **VIII:** Case Studies: Cloud storage system (e.g., S3); Graph databases; "When not to use NoSQL"

Learning Outcomes I: Develop a use case for the use of NoSQL over RDBMS [Create] **II:** Describe the defining characteristics behind Key-Value and Document-based data models [Describe]

Bibliography: [Sadalage and Fowler, 2012, Kleppmann, 2017]

5.4 Data Analytics, 12 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Exploratory data techniques (motivation, representation, descriptive statistics, visualizations) **II:** Data science lifecycle: business understanding, data understanding, data preparation, modeling, evaluation, deployment, and user acceptance **III:** Data mining and machine learning algorithms: e.g., classification, clustering, association, regression **IV:** Data acquisition and governance **V:** Data security and privacy considerations **VI:** Data fairness and bias **VII:** Data visualization techniques and their use in data analytics **VIII:** Entity Resolution

Learning Outcomes I: Describe several data exploration approaches, including visualization, to understanding unfamiliar datasets [Describe] **II:** Apply several data exploration approaches to understanding unfamiliar datasets [Apply] **III:** Describe basic machine learning/data mining algorithms and when they are appropriate for use [Describe] **IV:** Apply several machine learning/data mining algorithms [Apply] **V:** Describe legal and ethical considerations in acquiring, using, and modifying datasets [Describe] **VI:** Describe issues of fairness and bias in data collection and usage [Describe]

Bibliography: [Matei et al., 2012, Bill and Matei, 2018, Kleppmann, 2017, Psaltis, 2017]

6. Bibliography

[Bill and Matei, 2018] Bill, C. and Matei, Z. (2018). *Spark: The Definitive Guide: Big Data Processing Made Simple*. O'Reilly Media.

[Karau et al., 2017] Karau, H., Konwinski, A., Wendell, P., and Zaharia, M. (2017). *Learning Spark: Lightning-Fast Big Data Analysis*. O'Reilly Media.

[Kleppmann, 2017] Kleppmann, M. (2017). *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*. O'Reilly Media.

[Lin and Dyer, 2010] Lin, J. and Dyer, C. (2010). *Data-Intensive Text Processing with MapReduce*. Morgan and Claypool Publishers.

[Matei et al., 2012] Matei, Z., Mosharaf, C., Tathagata, D., Ankur, D., Justin, M., Murphy, M., J, F. M., Scott, S., and Ion, S. (2012). Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX Association.

[Psaltis, 2017] Psaltis, A. (2017). *Streaming Data: Understanding the real-time pipeline*. Manning Publications.

[Sadalage and Fowler, 2012] Sadalage, P. J. and Fowler, M. (2012). *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*. Addison-Wesley.

[White, 2015] White, T. (2015). *Hadoop: The Definitive Guide*. O'Reilly Media, 4th edition.

[Bill and Matei, 2018] Bill, C. and Matei, Z. (2018). *Spark: The Definitive Guide: Big Data Processing Made Simple*. O'Reilly Media, Sebastopol, CA.

[Karau et al., 2017] Karau, H., Konwinski, A., Wendell, P., and Zaharia, M. (2017). *Learning Spark: Lightning-Fast Big Data Analysis*. O'Reilly Media.

[Kleppmann, 2017] Kleppmann, M. (2017). *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*. O'Reilly Media.

[Lin and Dyer, 2010] Lin, J. and Dyer, C. (2010). *Data-Intensive Text Processing with MapReduce*. Morgan and Claypool Publishers.

[Matei et al., 2012] Matei, Z., Mosharaf, C., Tathagata, D., Ankur, D., Justin, M., Murphy, M., J, F. M., Scott, S., and Ion, S. (2012). Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, San Jose, CA. USENIX Association.

[Psaltis, 2017] Psaltis, A. (2017). *Streaming Data: Understanding the real-time pipeline*. Manning Publications.

[Sadalage and Fowler, 2012] Sadalage, P. J. and Fowler, M. (2012). *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*. Addison-Wesley.

[White, 2015] White, T. (2015). *Hadoop: The Definitive Guide*. O'Reilly Media, 4th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

STA252 Inferential Statistics (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA351 Statistics and Probabilities (6th Sem-Pg. 415)

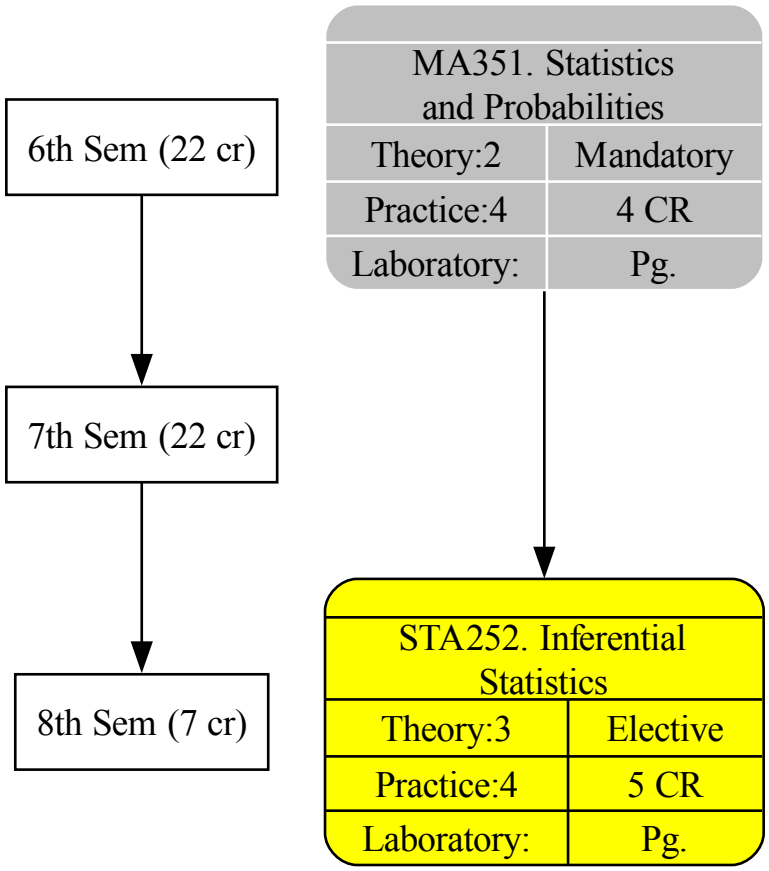


Figure 7.50: Connection Map. STA252 Inferential Statistics

2. Course Rationale

Inferential statistics is crucial in computer science for analyzing data, drawing conclusions, and making evidence-based decisions. This course covers the principles of statistical inference, including point estimation, confidence intervals, and hypothesis testing.

3. Goals of the Course

- Understand the fundamental concepts of inferential statistics.

- Apply point estimation techniques and confidence intervals.
- Conduct hypothesis tests to make data-driven decisions.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Familiarity)

5. Content

5.1 Introduction to Statistical Inference, 6 hr(s)

Competences: ABET-1

Topics: I: Sampling and sampling distributions. **II:** Point estimation. **III:** Confidence intervals.

Learning Outcomes I: Describe the principles of sampling and sampling distributions. [Familiarity] **II:** Calculate point estimates for population parameters. [Usage] **III:** Construct and interpret confidence intervals. [Assessment]

Bibliography: [Devore, 2020, Montgomery and Runger, 2019]

5.2 Hypothesis Testing, 10 hr(s)

Competences: ABET-1

Topics: I: Null and alternative hypotheses. **II:** Type I and Type II errors. **III:** P-value. **IV:** Hypothesis tests for means and proportions.

Learning Outcomes I: Formulate null and alternative hypotheses. [Familiarity] **II:** Calculate and interpret the p-value. [Usage] **III:** Conduct hypothesis tests for means and proportions. [Assessment]

Bibliography: [Devore, 2020, Montgomery and Runger, 2019]

5.3 Analysis of Variance (ANOVA), 8 hr(s)

Competences: ABET-1

Topics: I: One-way ANOVA. **II:** Two-way ANOVA.

Learning Outcomes I: Apply one-way ANOVA to compare means of multiple groups. [Familiarity] **II:** Conduct two-way ANOVA to analyze the influence of two variables. [Usage]

Bibliography: [Devore, 2020, Montgomery and Runger, 2019]

5.4 Linear Regression, 8 hr(s)

Competences: ABET-1

Topics: I: Simple linear regression. **II:** Multiple linear regression.

Learning Outcomes I: Fit a simple linear regression model. [Familiarity] **II:** Interpret the coefficients of a linear regression model. [Usage] **III:** Apply multiple linear regression to model the relationship between multiple variables. [Assessment]

Bibliography: [Devore, 2020, Montgomery and Runger, 2019]

5.5 Non-parametric Tests, 8 hr(s)

Competences: ABET-1

Topics: **I:** Chi-square test. **II:** Mann-Whitney U test. **III:** Kruskal-Wallis test.

Learning Outcomes **I:** Apply non-parametric tests when the assumptions of parametric tests are not met. [Familiarity] **II:** Perform the chi-square test for independence. [Usage] **III:** Use the Mann-Whitney U and Kruskal-Wallis tests to compare groups. [Assessment]

Bibliography: [Devore, 2020]

5.6 Applications in Computing, 8 hr(s)

Competences: ABET-1

Topics: **I:** Data analysis in computational experiments. **II:** Model validation. **III:** Machine learning (e.g., model evaluation).

Learning Outcomes **I:** Apply inferential statistics techniques to analyze data in computational experiments. [Familiarity] **II:** Use statistical methods to validate computational models. [Usage] **III:** Apply inferential statistics in evaluating machine learning models. [Assessment]

Bibliography: [Devore, 2020]

6. Bibliography

[Devore,] Devore, J. L. *Probability and Statistics for Engineering and the Sciences*. Cengage Learning, 10th edition.

[Montgomery and Runger,] Montgomery, D. C. and Runger, G. C. *Applied Statistics and Probability for Engineers*. Wiley, 8th edition.

[Devore, 2020] Devore, J. L. (2020). *Probability and Statistics for Engineering and the Sciences*. Cengage Learning, Boston, 10th edition.

[Montgomery and Runger, 2019] Montgomery, D. C. and Runger, G. C. (2019). *Applied Statistics and Probability for Engineers*. Wiley, Hoboken, 8th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA327 Numerical Analysis IV (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 4 hours; **Practice**: 2 hours;
- **Prerrequisites**:
  - MA326 Numerical Analysis III (7th Sem-Pg. 446)
  - MA213 Partial Differential Equations (7th Sem-Pg. 443)

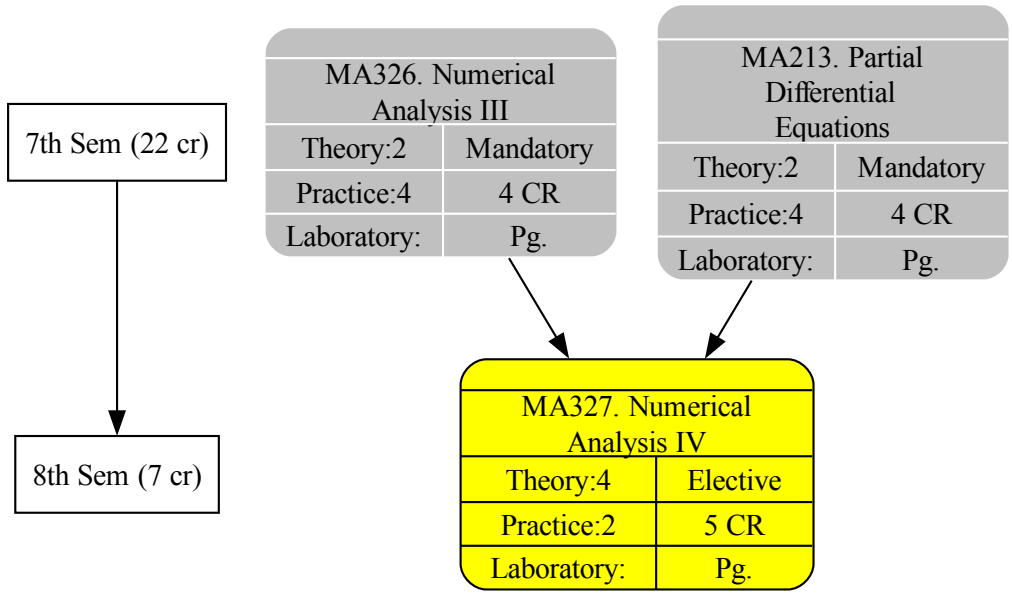


Figure 7.51: Connection Map. MA327 Numerical Analysis IV

2. Course Rationale

Numerical Analysis 4 provides the computational tools necessary to solve Partial Differential Equations (PDEs), which describe a wide range of phenomena in physics, engineering, and finance. The course focuses on the three fundamental types of second-order PDEs: elliptic, parabolic, and hyperbolic. Students will learn to discretize continuous domains, analyze the stability and convergence of numerical schemes, and implement algorithms for complex boundary value problems.

3. Goals of the Course

- Classify partial differential equations and understand their physical implications.
- Develop and implement Finite Difference Methods (FDM) for different PDE types.
- Analyze the stability (Von Neumann), consistency, and convergence of numerical schemes.
- Introduce advanced discretization techniques such as Finite Element and Finite Volume methods.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to

the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Elliptic Equations and Boundary Value Problems, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Classification of second-order PDEs **II:** Finite Difference Method for Laplace and Poisson equations **III:** Boundary conditions: Dirichlet, Neumann, and Robin **IV:** Iterative solvers for large sparse systems (Jacobi, Gauss-Seidel, SOR) **V:** Introduction to Finite Element Method (FEM) in 1D

Learning Outcomes **I:** Discretize steady-state problems using five-point stencils [Usage] **II:** Solve resulting linear systems using optimized iterative methods [Assessment] **III:** Evaluate the effect of grid refinement on the solution accuracy [Familiarity]

Bibliography: [Burden et al., 2016, Quarteroni et al., 2007]

5.2 Parabolic Equations and Evolution Problems, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** The Heat Equation in 1D and 2D **II:** Explicit schemes and stability restrictions (CFL condition) **III:** Implicit schemes: Crank-Nicolson and Backward Euler **IV:** Consistency and Von Neumann stability analysis

Learning Outcomes **I:** Implement time-stepping algorithms for diffusion processes [Usage] **II:** Determine the stability limits of explicit numerical schemes [Assessment] **III:** Apply the Thomas algorithm for tridiagonal systems in implicit methods [Usage]

Bibliography: [Heath, 2002, Quarteroni et al., 2007]

5.3 Hyperbolic Equations and Wave Phenomena, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** First-order advection equations **II:** The Wave Equation and D'Alembert's solution **III:** Upwind, Lax-Wendroff, and Leapfrog schemes **IV:** Dispersion and dissipation errors in wave propagation

Learning Outcomes **I:** Design numerical schemes for transport and wave equations [Usage] **II:** Analyze numerical diffusion and oscillation in high-order schemes [Assessment] **III:** Verify the Courant-Friedrichs-Lewy (CFL) condition for hyperbolic systems [Usage]

Bibliography: [LeVeque, 2007, Heath, 2002]

5.4 Advanced Topics in Numerical PDEs, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Non-linear PDEs and Burgers' equation **II:** Introduction to Finite Volume Method (FVM) for conservation laws **III:** Multidimensional problems and Alternating Direction Implicit (ADI) methods **IV:** Computational implementation using Python/MATLAB

Learning Outcomes **I:** Implement numerical solvers for non-linear conservation laws [Usage] **II:** Generalize 1D methods to complex 2D/3D geometries [Familiarity] **III:** Perform comparative studies of different discretization techniques [Assessment]

Bibliography: [Quarteroni et al., 2007, LeVeque, 2007]

6. Bibliography

[Burden et al., 2016] Burden, R. L., Faires, J. D., and Burden, A. M. (2016). *Numerical Analysis*. Cengage Learning, 10th edition.

[Heath, 2002] Heath, M. T. (2002). *Scientific Computing: An Introductory Survey*. McGraw-Hill, 2nd edition.

[LeVeque, 2007] LeVeque, R. J. (2007). *Finite Difference Methods for Ordinary and Partial Differential Equations*. SIAM.

[Quarteroni et al., 2007] Quarteroni, A., Sacco, R., and Saleri, F. (2007). *Numerical Mathematics*. Springer, 2nd edition.

[Burden et al., 2016] Burden, R. L., Faires, J. D., and Burden, A. M. (2016). *Numerical Analysis*. Cengage Learning, 10th edition.

[Heath, 2002] Heath, M. T. (2002). *Scientific Computing: An Introductory Survey*. McGraw-Hill, 2nd edition.

[LeVeque, 2007] LeVeque, R. J. (2007). *Finite Difference Methods for Ordinary and Partial Differential Equations*. SIAM.

[Quarteroni et al., 2007] Quarteroni, A., Sacco, R., and Saleri, F. (2007). *Numerical Mathematics*. Springer, 2nd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA352 Stochastic processes (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA351 Statistics and Probabilities (6th Sem-Pg. 415)
  - MA210 Calculus III (4th Sem-Pg. 380)
  - CS261 Artificial Intelligence (7th Sem-Pg. 427)

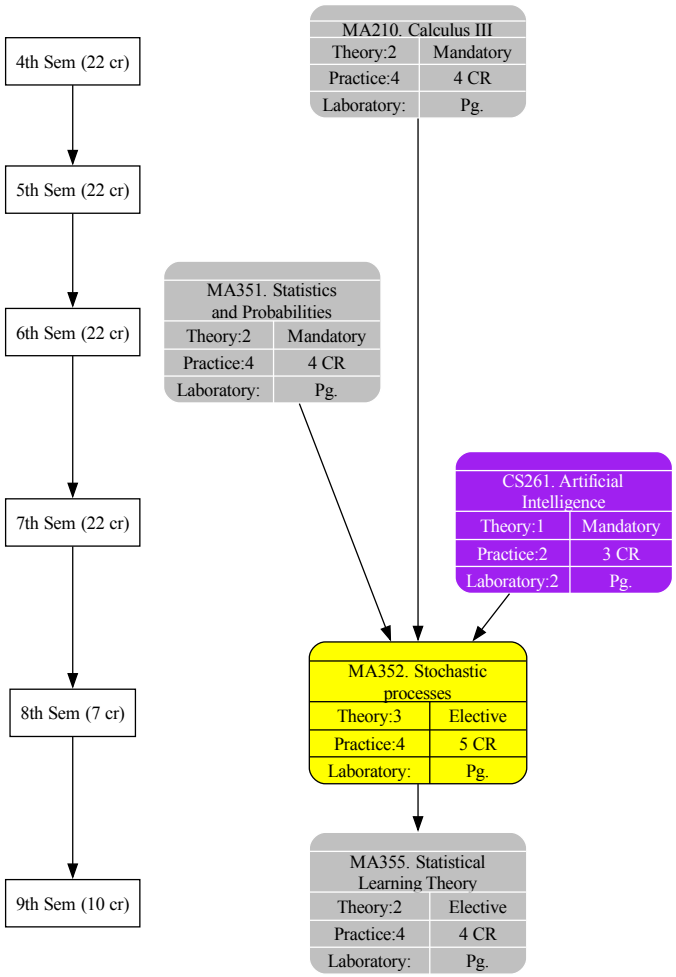


Figure 7.52: Connection Map. MA352 Stochastic processes

2. Course Rationale

This course introduces the theory of stochastic processes, providing mathematical foundations for the study of systems that evolve randomly over time. It develops skills for modeling random phenomena in finance, engineering, and sciences through Markov chains, Poisson processes, and Brownian motion, with applications in simulation and analysis of stochastic systems.

3. Goals of the Course

- Understand fundamental concepts of stochastic processes
- Model systems using discrete and continuous time Markov chains
- Analyze Poisson processes and their applications
- Apply stochastic processes to problems in finance and sciences

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Advanced Probability and Stochastic Processes, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Measure-theoretic probability foundations **II:** Random variables and distributions **III:** Stochastic processes characterization **IV:** Markov chains and Markov processes **V:** Brownian motion and diffusion processes

Learning Outcomes **I:** Apply measure-theoretic concepts to probability spaces [Familiarity] **II:** Characterize different types of stochastic processes [Assessment] **III:** Analyze Markov chains for steady-state and transient behavior [Usage] **IV:** Model systems using continuous-time stochastic processes [Usage]

Bibliography: [Ross, 2019, Karlin and Taylor, 2014]

5.2 Discrete Systems Simulation, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Queueing theory and models **II:** Cellular automata and agent-based models **III:** Monte Carlo simulation methods **IV:** Markov chains and processes **V:** Discrete-event simulation

Learning Outcomes **I:** Formulate and analyze simple queueing models [Usage] **II:** Implement cellular automata and agent-based simulations [Usage] **III:** Apply Monte Carlo methods for probabilistic simulation [Usage] **IV:** Analyze Markov chains for state transitions and steady-state behavior [Assessment] **V:** Design discrete-event simulations for system analysis [Usage]

Bibliography: [Ross, 2019, Karlin and Taylor, 2014]

5.3 Computational Statistical Inference, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Maximum Likelihood Estimation (MLE) **II:** Bayesian inference methods **III:** Markov Chain Monte Carlo (MCMC) methods **IV:** Expectation-Maximization (EM) algorithm **V:** Bootstrap and jackknife resampling

Learning Outcomes **I:** Implement Maximum Likelihood Estimation for various distributions [Usage] **II:** Apply Bayesian methods for parameter estimation [Usage] **III:** Implement MCMC algorithms for posterior sampling [Usage] **IV:** Use EM algorithm for models with latent variables [Usage] **V:** Apply bootstrap methods for confidence intervals [Usage]

Bibliography: [Ross, 2019, Karlin and Taylor, 2014]

5.4 Continuous Systems Simulation, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Ordinary Differential Equations (ODEs) models **II:** Partial Differential Equations (PDEs) models **III:** Finite difference methods for PDEs **IV:** Solver selection and implementation **V:** Stiff systems and adaptive methods

Learning Outcomes **I:** Formulate ODE and PDE models for physical systems [Usage] **II:** Implement finite difference methods for simple PDEs [Usage] **III:** Select appropriate numerical solvers for different types of ODEs/PDEs [Assessment] **IV:** Handle stiff systems using appropriate numerical methods [Usage]

Bibliography: [Ross, 2019, Karlin and Taylor, 2014]

6. Bibliography

[Karlin and Taylor, 2014] Karlin, S. and Taylor, H. M. (2014). *A First Course in Stochastic Processes*. Academic Press, 2nd edition.

[Ross, 2019] Ross, S. M. (2019). *Introduction to Probability Models*. Academic Press, 12th edition.

[Karlin and Taylor, 2014] Karlin, S. and Taylor, H. M. (2014). *A First Course in Stochastic Processes*. Academic Press, 2nd edition.

[Ross, 2019] Ross, S. M. (2019). *Introduction to Probability Models*. Academic Press, 12th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA372 Functional analysis II (Elective)

1. General information

- **Semester**: 8th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA371 Functional analysis I (6th Sem-Pg. 420)

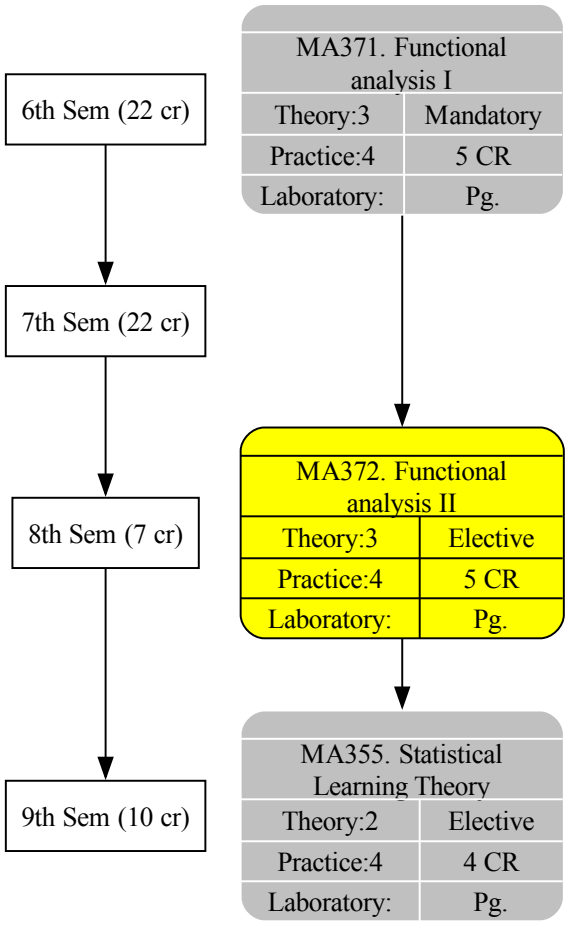


Figure 7.53: Connection Map. MA372 Functional analysis II

2. Course Rationale

This course deepens the theoretical foundations of measure and integration, providing essential tools for advanced analysis in infinite-dimensional spaces. It focuses on the rigorous construction of abstract measures and the operational aspects of the Lebesgue integral. Finally, the course explores the geometric structure and operator theory in Hilbert spaces, which are fundamental for developing computational methods and models in mathematical physics.

3. Goals of the Course

- Understand and construct abstract measures and measure extensions in general spaces.
- Master Lebesgue integration theory and its fundamental convergence theorems.
- Analyze measure duality and decomposition through the Radon-Nikodym theorem.
- Study advanced geometry and operator theory within Hilbert spaces.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Measure Theory, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Algebras, σ-algebras, and measures **II:** Outer measures and the Carathéodory Extension Theorem **III:** Borel measures on the real line **IV:** Signed measures and the Hahn-Jordan decomposition **V:** Complex measures and total variation

Learning Outcomes **I:** Construct measures from outer measures using the Carathéodory criterion [Assessment] **II:** Apply the Jordan decomposition to analyze signed measures [Usage] **III:** Prove regularity properties of Borel measures [Familiarity]

Bibliography: [Folland, 1999, Royden and Fitzpatrick, 2010]

5.2 Integration Theory, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Measurable functions and simple functions **II:** The Lebesgue integral: definition and fundamental properties **III:** Convergence theorems: Monotone Convergence, Dominated Convergence, and Fatou's Lemma **IV:** The Radon-Nikodym Theorem and Lebesgue-Stieltjes property **V:** Differentiation of measures and the Fundamental Theorem of Calculus in measure spaces

Learning Outcomes **I:** Evaluate the convergence of function sequences under the Lebesgue integral [Assessment] **II:** Use the Radon-Nikodym Theorem to find probability densities or measure derivatives [Usage] **III:** Compare Riemann and Lebesgue integrals in terms of applicability [Familiarity]

Bibliography: [Folland, 1999, Brezis, 2011]

5.3 Hilbert Spaces, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Review of inner products and completeness **II:** Orthogonality, orthonormal bases, and the Gram-Schmidt process **III:** Riesz Representation Theorem and linear functionals **IV:** Adjoint, self-adjoint, and unitary operators in Hilbert spaces **V:** Introduction to spectral theory for compact operators

Learning Outcomes **I:** Apply the projection theorem to solve best approximation problems [Usage] **II:** Identify the adjoint of classical linear operators [Assessment] **III:** Analyze the spectral structure of self-adjoint operators [Familiarity]

Bibliography: [Kreyszig, 1989, Conway, 1990]

5.4 Operator Theory, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Bounded and unbounded operators: definitions and domains **II:** The spectrum of an operator: point, continuous, and residual spectrum **III:** Normal operators and the Spectral Theorem for compact operators **IV:** Applications of operator theory in numerical analysis

Learning Outcomes I: Calculate the spectrum of various types of linear operators [Assessment] **II:** Apply the spectral theorem in operator decomposition [Usage] **III:** Solve operational problems using basic functional calculus [Assessment]

Bibliography: [Brezis, 2011, Yosida, 1980]

6. Bibliography

[Brezis, 2011] Brezis, H. (2011). *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Springer Science & Business Media.

[Conway, 1990] Conway, J. B. (1990). *A Course in Functional Analysis*. Springer-Verlag, 2nd edition.

[Folland, 1999] Folland, G. B. (1999). *Real Analysis: Modern Techniques and Their Applications*. Wiley-Interscience, 2nd edition.

[Kreyszig, 1989] Kreyszig, E. (1989). *Introductory Functional Analysis with Applications*. Wiley.

[Royden and Fitzpatrick, 2010] Royden, H. and Fitzpatrick, P. (2010). *Real Analysis*. Pearson, 4th edition.

[Yosida, 1980] Yosida, K. (1980). *Functional Analysis*. Springer-Verlag, 6th edition.

[Brezis, 2011] Brezis, H. (2011). *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Springer Science & Business Media.

[Conway, 1990] Conway, J. B. (1990). *A Course in Functional Analysis*. Springer-Verlag, 2nd edition.

[Folland, 1999] Folland, G. B. (1999). *Real Analysis: Modern Techniques and Their Applications*. Wiley-Interscience, 2nd edition.

[Kreyszig, 1989] Kreyszig, E. (1989). *Introductory Functional Analysis with Applications*. Wiley.

[Royden and Fitzpatrick, 2010] Royden, H. and Fitzpatrick, P. (2010). *Real Analysis*. Pearson, 4th edition.

[Yosida, 1980] Yosida, K. (1980). *Functional Analysis*. Springer-Verlag, 6th edition.

Ninth Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

AI264 Deep Learning (Mandatory)

1. General information

- **Semester**: 9th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - AI263 Introduction to Machine Learning (8th Sem-Pg. 455)

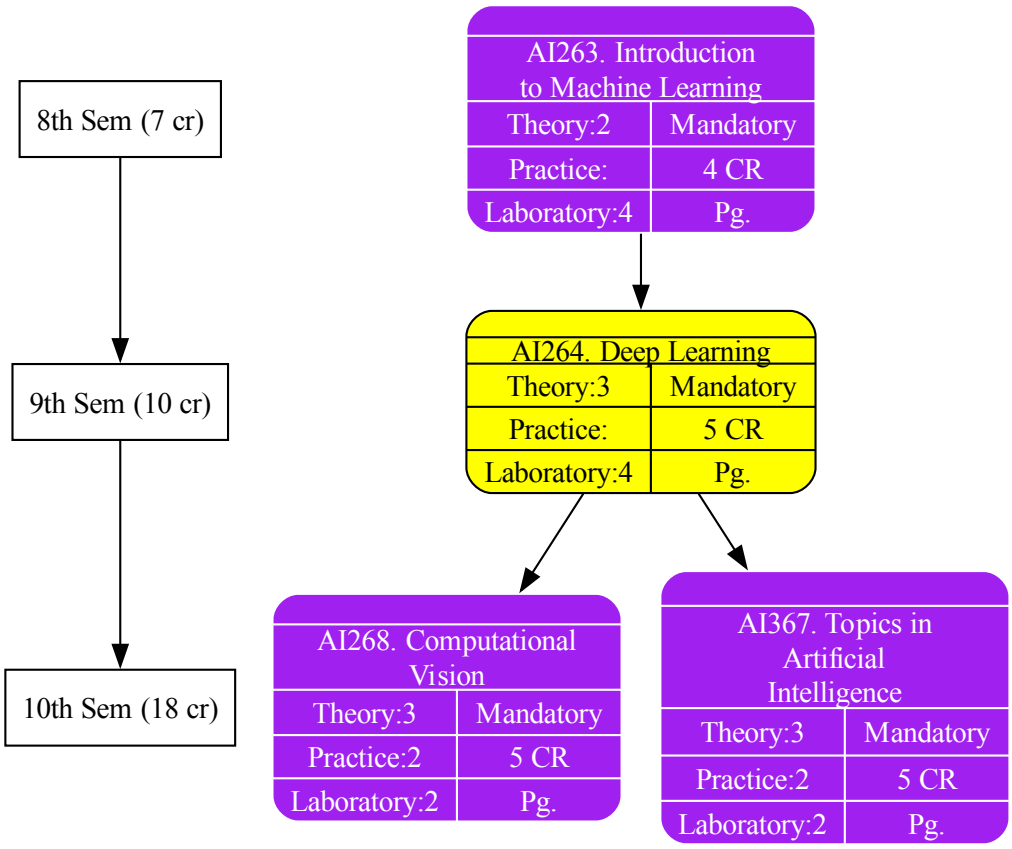


Figure 7.54: Connection Map. AI264 Deep Learning

2. Course Rationale

This course covers modern deep learning fundamentals, including convolutional networks, transformer architectures, and training techniques for advanced AI systems.

3. Goals of the Course

- Implement deep neural networks using modern frameworks (PyTorch/TensorFlow)
- Design architectures for specific domains (vision, language, etc.)
- Analyze ethical limitations of deep models

4. General Competences (*Outcomes*)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Assessment)

5. Content

5.1 Neural Network Training and Optimization, 10 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Modern backpropagation: computational graphs and automatic differentiation **II:** Advanced optimizers: momentum, RMSProp, Adam, and learning-rate schedules

Learning Outcomes I: Implement a deep neural network from scratch, including its backpropagation pass [Apply] **II:** Select and tune an optimizer and its learning-rate schedule for training stability [Apply]

Bibliography: [Goodfellow et al., 2016]

5.2 Regularization Techniques for Deep Networks, 5 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Dropout regularization **II:** Batch normalization

Learning Outcomes I: Diagnose overfitting in a training run using regularization diagnostics [Analyze] **II:** Apply dropout and batch normalization to improve a model's generalization [Apply]

Bibliography: [Goodfellow et al., 2016]

5.3 Convolutional and Residual Network Architectures, 9 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Deep convolutional neural network architectures **II:** Residual connections and ResNets

Learning Outcomes I: Design a deep CNN architecture for a given vision task [Apply] **II:** Adapt a residual architecture to a specific vision domain [Apply]

Bibliography: [He et al., 2016]

5.4 Transformer Architectures and Generative Models, 16 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Transformers and the attention mechanism **II:** Generative models: variational autoencoders (VAEs) and generative adversarial networks (GANs)

Learning Outcomes I: Train a transformer model on a real dataset [Apply] **II:** Analyze and contrast VAE and GAN generative-model architectures [Analyze]

Bibliography: [Vaswani et al., 2017]

6. Bibliography

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *CVPR*, pages 770–778.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *CVPR*, pages 770–778.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA402 Capstone Project II (Mandatory)

1. General information

- **Semester**: 9th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 8 hours;
- **Prerrequisites**:
  - MA401 Capstone Project I (8th Sem-Pg. 458)
  - ID203 Technical and professional English V (7th Sem-Pg. 452)

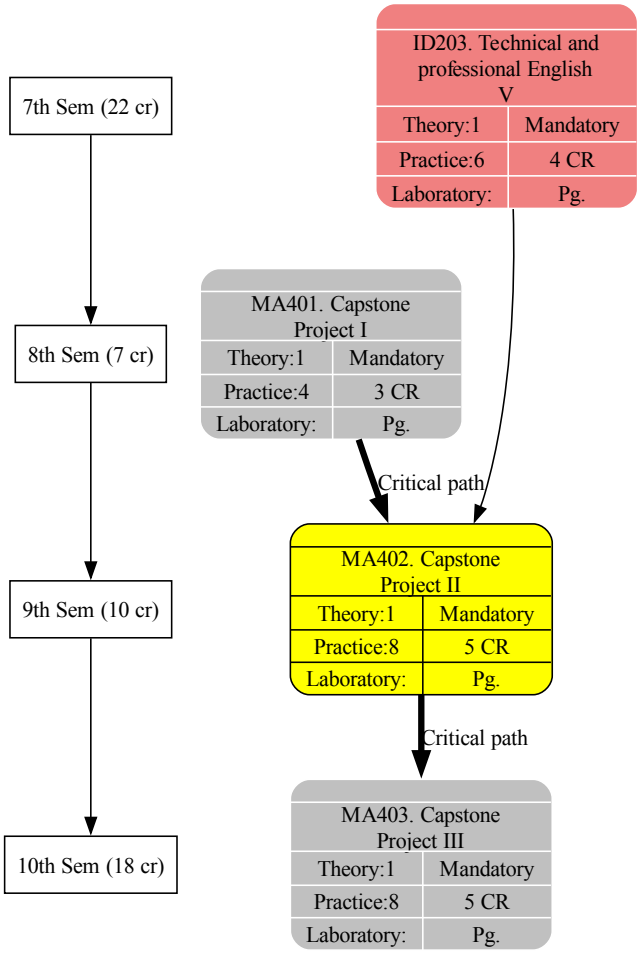


Figure 7.55: Connection Map. MA402 Capstone Project II

2. Course Rationale

This course constitutes the final phase of the capstone project in computational mathematics, where students complete, validate, and present an applied research project or technological development. It emphasizes the complete implementation of computational solutions, rigorous analysis of results, methodological validation, and professional communication of findings. The project must

demonstrate mastery of computational mathematics, innovation capability, and contribution to knowledge or solution of real-world problems, culminating in professional-quality work and a public defense before a specialized jury.

3. Goals of the Course

- Fully implement the designed computational solution, applying advanced computational mathematics techniques.
- Conduct comprehensive analysis of results and rigorous methodological validations.
- Professionally document the entire research and development process of the project.
- Evaluate the impact and limitations of the developed solution, proposing future improvements.
- Publicly defend the project before a specialized jury, demonstrating topic mastery and technical communication skills.
- Demonstrate autonomous work capability, project management, and professional ethics in applied research development.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Assessment)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Assessment)

ABET-4) An ability to communicate effectively with a range of audiences. (Assessment)

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts. (Assessment)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Assessment)

5. Content

5.1 Advanced Implementation and Optimization, 20 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: **I:** Complete implementation of algorithms and computational methods **II:** Optimization and performance improvement of solutions **III:** Technical validation and exhaustive testing **IV:** Handling of edge cases and complex situations **V:** Iterative refinement based on results

Learning Outcomes **I:** Conduct complete studies using research methods [Assessment] **II:** Formally demonstrate properties of solutions [Assessment] **III:** Implement complex mathematical processes on computer [Usage] **IV:** Optimize computational solutions for efficiency [Assessment]

Bibliography: [Steve, 2004, E, 1997]

5.2 Results Analysis and Validation, 20 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: **I:** Rigorous statistical analysis of results **II:** Methodological validation and comparison with benchmarks **III:** Sensitivity and robustness analysis of solutions **IV:** Evaluation of model limitations and assumptions **V:** Critical interpretation of findings and conclusions

Learning Outcomes **I:** Analyze social impact of scientific solutions [Assessment] **II:** Conduct rigorous methodological validations [Assessment] **III:** Recognize formal structure of complex problems [Assessment] **IV:** Interpret results in applied context [Usage]

Bibliography: [C, 2017, P et al., 2005]

5.3 Professional Documentation and Communication, 20 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: **I:** Preparation of professional technical report **II:** Preparation of scientific articles or technical reports **III:** Design of effective presentations for defense **IV:** Communication of results to different audiences **V:** Ethical considerations in results publication

Learning Outcomes **I:** Apply ethical principles in scientific communication [Usage] **II:** Effectively communicate complex results [Assessment] **III:** Learn continuously through feedback [Usage] **IV:** Document processes professionally [Assessment]

Bibliography: [Michael, 1996, R and W, 2008]

5.4 Defense and Final Evaluation, 20 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: **I:** Preparation for public project defense **II:** Presentation strategies and technical argumentation **III:** Handling jury questions and objections **IV:** Critical self-evaluation of own work **V:** Planning of future work and improvements

Learning Outcomes **I:** Perform effectively in public defense [Assessment] **II:** Manage complex scientific projects [Assessment] **III:** Solve advanced mathematical problems [Assessment] **IV:** Respond to criticism constructively [Usage]

Bibliography: [L, 2018, H, 2006]

6. Bibliography

[C, 2017] C, M. D. (2017). *Design and Analysis of Experiments*. John Wiley & Sons, 9th edition.

[E, 1997] E, K. D. (1997). *The Art of Computer Programming*. Addison-Wesley.

[H, 2006] H, A. R. R. (2006). *Dazzle 'em with Style: The Art of Oral Scientific Presentation*. Elsevier Academic Press.

[L, 2018] L, T. K. (2018). *A Manual for Writers of Research Papers, Theses, and Dissertations*. University of Chicago Press.

[Michael, 1996] Michael, A. (1996). *The Craft of Scientific Writing*. Springer.

[P et al., 2005] P, B. G. E., Stuart, H. J., and G, H. W. (2005). *Statistics for Experimenters: Design, Innovation, and Discovery*. Wiley-Interscience.

[R and W, 2008] R, M. J. and W, M. R. (2008). *Successful Scientific Writing: A Step-by-Step Guide for the Biological and Medical Sciences*. Cambridge University Press.

[Steve, 2004] Steve, M. (2004). *Code Complete: A Practical Handbook of Software Construction*. Microsoft Press, 2nd edition.

[C, 2017] C, M. D. (2017). *Design and Analysis of Experiments*. John Wiley & Sons, 9th edition.

[E, 1997] E, K. D. (1997). *The Art of Computer Programming*, volume 1-3. Addison-Wesley, Reading, MA.

[H, 2006] H, A. R. R. (2006). *Dazzle 'em with Style: The Art of Oral Scientific Presentation*. Elsevier Academic Press, Burlington, MA, 2nd edition.

[L, 2018] L, T. K. (2018). *A Manual for Writers of Research Papers, Theses, and Dissertations*. University of Chicago Press, Chicago, IL, 9th edition.

[Michael, 1996] Michael, A. (1996). *The Craft of Scientific Writing*. Springer, New York, 3rd edition.

[P et al., 2005] P, B. G. E., Stuart, H. J., and G, H. W. (2005). *Statistics for Experimenters: Design, Innovation, and Discovery*. Wiley-Interscience, Hoboken, NJ, 2nd edition.

[R and W, 2008] R, M. J. and W, M. R. (2008). *Successful Scientific Writing: A Step-by-Step Guide for the Biological and Medical Sciences*. Cambridge University Press, Cambridge, 3rd edition.

[Steve, 2004] Steve, M. (2004). *Code Complete: A Practical Handbook of Software Construction*. Microsoft Press, Redmond, WA, 2nd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS231 Networking and Communication (Elective)

1. General information

- **Semester**: 9th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - CS2S1 Operating systems (8th Sem-Pg. 461)

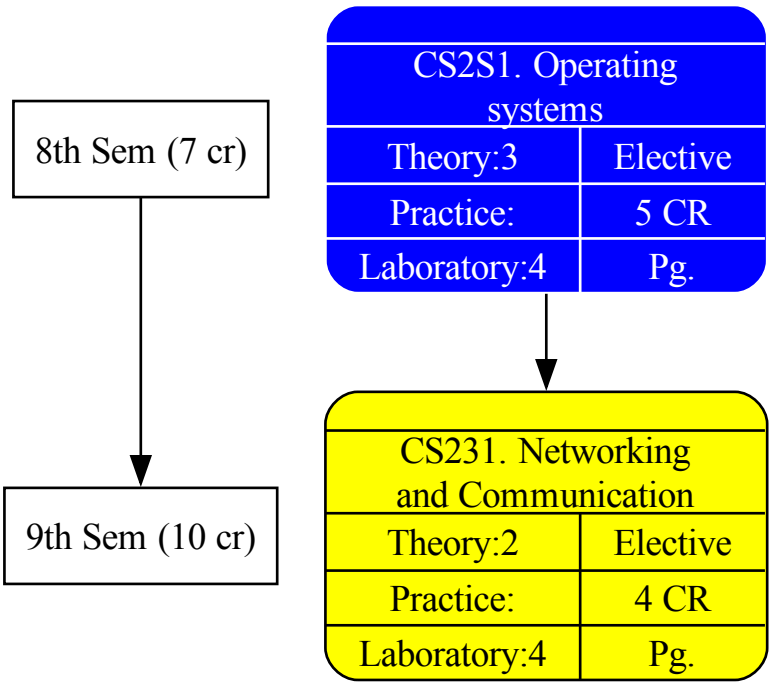


Figure 7.56: Connection Map. CS231 Networking and Communication

2. Course Rationale

Networking and communication are foundational to modern computing systems, distributed applications, and the global internet. This course provides a comprehensive introduction to the full CS2023 Networking and Communication Body of Knowledge—from physical transmission and data link protocols, through reliable transport and routing, to application services, security, mobility, and emerging paradigms such as SDN and quantum networking. Students will develop both theoretical understanding and practical skills applicable to the design and analysis of modern networked systems.

3. Goals of the Course

- Understand the layered architecture of the TCP/IP and OSI models, including the role and responsibilities of each layer.

- Explain single-hop communication mechanisms including MAC protocols, encoding, Ethernet, WiFi, and LAN topologies.
- Master the principles of reliable data transfer, flow control, and congestion control.
- Analyze routing paradigms, forwarding mechanisms, and IP scalability in local and wide area networks.
- Comprehend fundamental application-layer protocols of the Internet (DNS, HTTP, TCP, UDP) and implement socket-based applications.
- Understand network security threats and the countermeasures used to protect networked systems.
- Describe principles of mobile networking including cellular networks, wireless LANs, and device-to-device communication.
- Identify emerging networking paradigms such as SDN, network virtualization, and quantum networking.

4. General Competences (*Outcomes*)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Usage)

5. Content

5.1 Fundamentals of Networks and Communications, 6 hr(s)

Competences: ABET-2,ABET-6

Topics: I: Importance of networking in contemporary computing, and associated challenges. See also: Digital Equity and Accessibility, Social Impact and AI, Civil Liberties and Expression, Data, Surveillance and Privacy Technology, Privacy Philosophical Foundations and Legislation **II:** Organization of the internet (e.g., users, Internet Service Providers, autonomous systems, content providers, content delivery networks) **III:** Switching techniques (e.g., circuit and packet) **IV:** Layers and their roles (application, transport, network, datalink, and physical) **V:** Layering principles (e.g., encapsulation and hourglass model). See also: Digital Systems and Logic, Sequential and Parallel Processing, State Machines **VI:** Network elements (e.g., routers, switches, hubs, access points, and hosts) **VII:** Basic queueing concepts (e.g., relationship with latency, congestion, service levels, etc.)

Learning Outcomes I: Articulate the organization of the internet [Articulate] **II:** List and define the appropriate network terminology [List] **III:** Describe the layered structure of a typical networked architecture [Describe] **IV:** Identify the different types of complexity in a network (edges, core, etc.) [Analyze]

Bibliography: [Kurose and Ross, 2021, Tanenbaum et al., 2021, Peterson and Davie, 2022]

5.2 Single Hop Communication, 6 hr(s)

Competences: ABET-2,ABET-6

Topics: I: Introduction to modulation, bandwidth, and communication media **II:** Encoding and Framing **III:** Medium Access Control (MAC) (e.g., random access and scheduled access) **IV:** Ethernet and WiFi **V:** Switching (e.g., spanning trees, VLANS). **VI:** Local Area Network Topologies (e.g., data center, campus networks).

Learning Outcomes I: Describe some basic aspects of modulation, bandwidth, and communication media [Describe] **II:** Describe in detail a MAC protocol [Describe] **III:** Demonstrate understanding of encoding and framing solution tradeoffs [Demonstrate] **IV:** Describe details of the implementation of Ethernet [Describe] **V:** Describe how switching works [Describe] **VI:** Describe one kind of a LAN topology [Describe]

Bibliography: [Tanenbaum et al., 2021, Peterson and Davie, 2022]

5.3 Networked Applications, 6 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Naming and address schemes (e.g., DNS, and Uniform Resource Identifiers) **II:** Distributed application paradigms (e.g., client/server, peer-to-peer, cloud, edge, and fog). See also: Communication, Coordination **III:** Diversity of networked application demands (e.g., latency, bandwidth, and loss tolerance). See also: Communication, Sustainable Design and Pervasive Computing, Environmental Footprint of Computing Systems, Systemic Effects and Social Context, Digital Equity and Accessibility, Social Impact and AI **IV:** Coverage of application-layer protocols (e.g., HTTP) **V:** Interactions with TCP, UDP, and Socket APIs. See also: Programs: Distribution, Programs: Execution Properties, GPU Programming, Programs: Implementation Mappings, Programs: Declarative Parallelism, Programs: Starting Activities

Learning Outcomes **I:** Define the principles of naming, addressing, resource location [Define] **II:** Analyze the needs of specific networked application demands [Analyze] **III:** Describe the details of one application layer protocol [Describe] **IV:** Implement a simple client-server socket-based application [Implement]

Bibliography: [Kurose and Ross, 2021, Peterson and Davie, 2022]

5.4 Reliability Support, 6 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Unreliable delivery (e.g., UDP) **II:** Principles of reliability (e.g., delivery without loss, duplication, or out of order). See also: System Reliability **III:** Error control (e.g., retransmission, error correction) **IV:** Flow control (e.g., stop and wait, window based) **V:** Congestion control (e.g., implicit and explicit congestion notification) **VI:** TCP and performance issues (e.g., Tahoe, Reno, Vegas, Cubic)

Learning Outcomes **I:** Describe the operation of reliable delivery protocols [Describe] **II:** List the factors that affect the performance of reliable delivery protocols [List] **III:** Describe some TCP reliability design issues [Describe] **IV:** Design and implement a simple reliable protocol [Design]

Bibliography: [Kurose and Ross, 2021, Tanenbaum et al., 2021]

5.5 Routing and Forwarding, 6 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Routing paradigms and hierarchy (e.g., intra/inter domain, centralized and decentralized, source routing, virtual circuits, QoS) **II:** Forwarding methods (e.g., forwarding tables and matching algorithms) **III:** IP and Scalability issues (e.g., NAT, CIDR, BGP, different versions of IP)

Learning Outcomes **I:** Describe various routing paradigms and hierarchies [Describe] **II:** Describe how packets are forwarded in an IP network [Describe] **III:** Describe how the Internet tackles scalability challenges [Describe]

Bibliography: [Kurose and Ross, 2021, Tanenbaum et al., 2021, Peterson and Davie, 2022]

5.6 Network Security, 6 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** General intro about security (Threats, vulnerabilities, and countermeasures). See also: Computer Crimes and Attacks, Security Laws, Policies and Responsibility, AI and Hardware Security, Access Control and Applications, Security Mindset and Principles, Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering **II:** Network specific threats and attack types (e.g., denial of service, spoofing, sniffing and traffic redirection, attacker-in-the-middle, message integrity attacks, routing attacks, ransomware, and traffic analysis). See also: AI and Hardware Security, Access Control and Applications, Security Mindset and Principles, Security Design and Controls Engineering, Threat Analysis and Security Engineering, Trusted Computing and Privacy Engineering **III:** Countermeasures: (a) Cryptography (e.g. SSL, TLS, symmetric/asymmetric) (b) Architectures for secure networks (e.g., secure channels, secure routing protocols, secure DNS, VPNs, DMZ, Zero Trust Network Access, hyper network security, anonymous communication protocols, isolation) (c) Network monitoring, intrusion detection, firewalls, spoofing and DoS protection, honeypots, tracebacks, BGP Sec, RPKI

Learning Outcomes **I:** Describe some of the threat models of network security [Describe] **II:** Describe specific network-based countermeasures [Describe] **III:** Analyze various aspects of network security from a case study [Analyze]

Bibliography: [Stallings, 2022, Kurose and Ross, 2021]

5.7 Mobility Support, 6 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Principles of cellular communication (e.g., 4G, 5G) **II:** Principles of Wireless LANs (mainly 802.11) **III:** Device to device communication (e.g., IoT communication) **IV:** Multi-hop wireless networks (e.g., ad hoc networks, opportunistic, delay tolerant)

Learning Outcomes **I:** Describe some aspects of cellular communication such as registration [Describe] **II:** Describe how 802.11 supports mobile users [Describe] **III:** Describe practical uses of device-to-device communication, as well as multihop [Describe] **IV:** Describe one type of mobile network such as ad hoc [Describe]

Bibliography: [Tanenbaum et al., 2021, Peterson and Davie, 2022]

5.8 Emerging Topics, 6 hr(s)

Competences: ABET-2,ABET-6

Topics: **I:** Middleboxes (e.g., advances in usage of AI, intent-based networking, filtering, deep packet inspection, load balancing, NAT, CDN) **II:** Network Virtualization (e.g., SDN, Data Center Networks) **III:** Quantum Networking (e.g., Intro to the domain, teleportation, security, Quantum Internet) **IV:** Satellite, mmWave, Visible Light

Learning Outcomes **I:** Describe the value of advances in middleboxes in networks [Describe] **II:** Describe the importance of Software Defined Networks [Describe] **III:** Describe some of the added value achieved by using Quantum Networking [Describe]

Bibliography: [Peterson and Davie, 2022, Tanenbaum et al., 2021]

6. Bibliography

[Kurose and Ross, 2021] Kurose, J. and Ross, K. (2021). *Computer Networking: A Top-Down Approach*. Pearson, 8th edition.

[Peterson and Davie, 2022] Peterson, L. L. and Davie, B. S. (2022). *Computer Networks: A Systems Approach*. Morgan Kaufmann, 6th edition.

[Stallings, 2022] Stallings, W. (2022). *Cryptography and Network Security: Principles and Practice*. Pearson, 8th edition.

[Tanenbaum et al., 2021] Tanenbaum, A. S., Feamster, N., and Wetherall, D. J. (2021). *Computer Networks*. Pearson, 6th edition.

[Kurose and Ross, 2021] Kurose, J. and Ross, K. (2021). *Computer Networking: A Top-Down Approach*. Pearson, 8th edition.

[Peterson and Davie, 2022] Peterson, L. L. and Davie, B. S. (2022). *Computer Networks: A Systems Approach*. Morgan Kaufmann, 6th edition.

[Stallings, 2022] Stallings, W. (2022). *Cryptography and Network Security: Principles and Practice*. Pearson, 8th edition.

[Tanenbaum et al., 2021] Tanenbaum, A. S., Feamster, N., and Wetherall, D. J. (2021). *Computer Networks*. Pearson, 6th edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA381 Algebraic Structures (Elective)

1. General information

- **Semester**: 9th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA281 Numbers theory (5th Sem-Pg. 401)

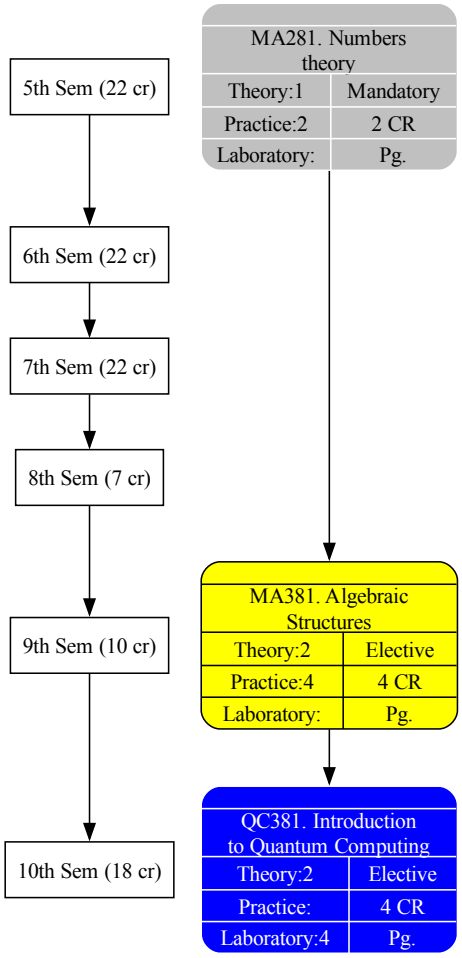


Figure 7.57: Connection Map. MA381 Algebraic Structures

2. Course Rationale

Algebraic Structures provides the formal and abstract foundation essential for advanced mathematics and its computational applications. This course introduces the fundamental structures of modern algebra—groups, rings, and fields—developing the ability to think in terms of axioms, homomorphisms, and invariants. Mastery of these concepts is crucial for understanding the theoretical underpinnings of areas such as coding theory, cryptography, algorithm design, and algebraic geometry. The course emphasizes the interplay between abstract theory and concrete examples, particularly those arising in computational and discrete mathematics.

3. Goals of the Course

- Master the axiomatic definitions and basic properties of groups, rings, and fields.
- Understand and apply the concepts of subgroups, normal subgroups, quotient groups, and the fundamental homomorphism theorems for groups.
- Analyze the structure of rings, including ideals, quotient rings, integral domains, and fields.
- Apply the theory of finite fields to practical problems in coding theory and cryptography.
- Develop the ability to construct rigorous proofs and recognize algebraic structures in various mathematical and computational contexts.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Groups and Subgroups, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Binary operations and the definition of a group. Basic examples: cyclic, symmetric, dihedral, and matrix groups. **II:** Elementary properties of groups: uniqueness of identity and inverses, cancellation laws. **III:** Subgroups and subgroup tests. Centralizer, normalizer, and center of a group. **IV:** Cyclic groups: classification, structure, and subgroups. **V:** Permutation groups: cycle notation, transpositions, and parity.

Learning Outcomes **I:** State the group axioms and verify whether a given set with an operation forms a group [Familiarity]. **II:** Apply subgroup tests and generate subgroups, including cyclic subgroups of a given element [Usage]. **III:** Analyze the structure of finite cyclic groups and decompose permutations into disjoint cycles [Assessment].

Bibliography: [Dummit and Foote, 2004, Gallian, 2021]

5.2 Cosets, Normal Subgroups, and Homomorphisms, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Cosets and Lagrange's Theorem. Applications to the order of elements. **II:** Normal subgroups and their characterization. Quotient (factor) groups. **III:** Group homomorphisms and isomorphisms: definitions, properties, and examples. **IV:** The Fundamental Homomorphism Theorem (First Isomorphism Theorem) for groups. **V:** Introduction to group actions: definition and basic examples (orbit-stabilizer).

Learning Outcomes **I:** Compute cosets and use Lagrange's Theorem to deduce properties of finite groups [Familiarity]. **II:** Determine if a subgroup is normal and construct the corresponding quotient group [Usage]. **III:** Use the Fundamental Homomorphism Theorem to establish isomorphisms between quotient groups and known groups [Assessment].

Bibliography: [Dummit and Foote, 2004, Judson, 2021]

5.3 Rings, Ideals, and Integral Domains, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Definition and basic examples of rings, commutative rings, and rings with unity. **II:** Ring homomorphisms, isomorphisms, and basic properties. **III:** Ideals and quotient rings. The Fundamental Homomorphism Theorem for rings. **IV:** Operations on ideals: sum, product, and intersection. **V:** Integral domains, fields, and characteristic of a ring. Finite integral domains are fields.

Learning Outcomes I: Identify the defining properties of rings, integral domains, and fields from given examples [Familiarity]. **II:** Verify whether a subset of a ring is an ideal and construct the corresponding quotient ring [Usage]. **III:** Apply ring homomorphism theorems to analyze ring structures and prove basic results about integral domains [Assessment].

Bibliography: [Dummit and Foote, 2004, Hungerford, 2014]

5.4 Fields and Applications, 16 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Field extensions: algebraic and transcendental elements. Degree of an extension. **II:** Finite fields (Galois fields): existence, uniqueness, and structure. The field $GF(p^n)$. **III:** Polynomial rings over a field. Irreducible polynomials and construction of finite fields. **IV:** Primitive elements in finite fields and their application in cryptography (e.g., Diffie-Hellman, AES). **V:** Brief introduction to error-correcting codes: Hamming distance and linear codes as vector spaces over finite fields.

Learning Outcomes I: Describe the classification and basic properties of finite fields [Familiarity]. **II:** Construct small finite fields using irreducible polynomials over $\mathbb{Z}_p$ [Usage]. **III:** Explain how the structure of finite fields underpins modern cryptographic protocols and coding theory [Assessment].

Bibliography: [Gallian, 2021, Lidl and Niederreiter, 1994]

6. Bibliography

[Dummit and Foote, 2004] Dummit, D. S. and Foote, R. M. (2004). *Abstract Algebra*. John Wiley & Sons, 3rd edition. Comprehensive and widely-used graduate-level textbook, thorough and rigorous.

[Gallian, 2021] Gallian, J. A. (2021). *Contemporary Abstract Algebra*. CRC Press, 10th edition. Very accessible undergraduate text with numerous applications and historical notes.

[Hungerford, 2014] Hungerford, T. W. (2014). *Abstract Algebra: An Introduction*. Cengage Learning, 3rd edition. Clear introduction focusing on understanding core concepts, suitable for first exposure.

[Judson, 2021] Judson, T. W. (2021). *Abstract Algebra: Theory and Applications*. Orthogonal Publishing, 2021a edition. Open-source textbook with an applied focus, including chapters on cryptography and coding theory. Freely available online.

[Lidl and Niederreiter, 1994] Lidl, R. and Niederreiter, H. (1994). *Introduction to Finite Fields and Their Applications*. Cambridge University Press, 2nd edition. Authoritative reference on the theory and computational aspects of finite fields.

[Dummit and Foote, 2004] Dummit, D. S. and Foote, R. M. (2004). *Abstract Algebra*. John Wiley & Sons, 3rd edition. Comprehensive and widely-used graduate-level textbook, thorough and rigorous.

[Gallian, 2021] Gallian, J. A. (2021). *Contemporary Abstract Algebra*. CRC Press, 10th edition. Very accessible undergraduate text with numerous applications and historical notes.

[Hungerford, 2014] Hungerford, T. W. (2014). *Abstract Algebra: An Introduction*. Cengage Learning, 3rd edition. Clear introduction focusing on understanding core concepts, suitable for first exposure.

[Judson, 2021] Judson, T. W. (2021). *Abstract Algebra: Theory and Applications*. Orthogonal Publishing, 2021a edition. Open-source textbook with an applied focus, including chapters on cryptography and coding theory. Freely available online.

[Lidl and Niederreiter, 1994] Lidl, R. and Niederreiter, H. (1994). *Introduction to Finite Fields and Their Applications*. Cambridge University Press, 2nd edition. Authoritative reference on the theory and computational aspects of finite fields.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

AI262 Natural Language Processing (Elective)

1. General information

- **Semester**: 9th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - AI263 Introduction to Machine Learning (8th Sem-Pg. 455)

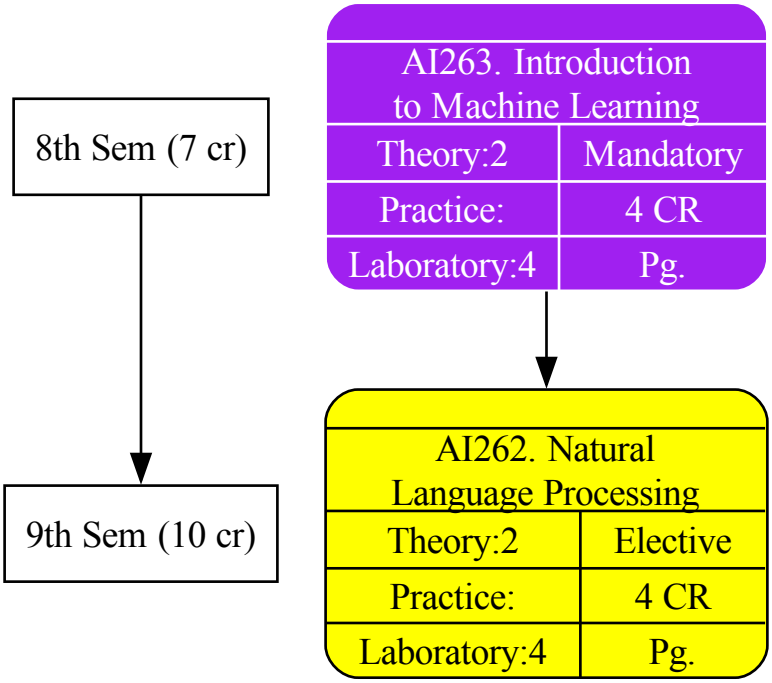


Figure 7.58: Connection Map. AI262 Natural Language Processing

2. Course Rationale

This course introduces the theoretical and practical fundamentals of Natural Language Processing (NLP), covering from classical techniques to modern deep learning models. Students will learn to implement complete NLP pipelines for applications like sentiment analysis, machine translation, and chatbots.

3. Goals of the Course

- Understand computational representation of human language.
- Implement text processing algorithms in Python.
- Apply NLP models to real-world problems using modern frameworks.

4. General Competences (*Outcomes*)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 NLP Fundamentals, 15 hr(s)

Competences: ABET-1,ABET-2

Topics: I: Tokenization and text representation **II:** N-gram models and smoothing **III:** Feature extraction (TF-IDF, word embeddings)

Learning Outcomes I: Preprocess text for NLP models [Usage] **II:** Explain limitations of classical statistical models [Familiarity]

Bibliography: [Jurafsky and Martin, 2021, Manning and Schütze, 1999]

5.2 Language Modeling, 15 hr(s)

Competences: ABET-1,ABET-2

Topics: I: Word2Vec, GloVe and FastText **II:** Recurrent Neural Networks (RNNs) for NLP **III:** Sequence-to-sequence models (seq2seq)

Learning Outcomes I: Train word embeddings [Usage] **II:** Implement basic RNNs for text generation [Assessment]

Bibliography: [Goldberg, 2017, Mikolov et al., 2013]

5.3 Transformers and Modern NLP, 15 hr(s)

Competences: ABET-1,ABET-2

Topics: I: Transformer architecture (self-attention) **II:** BERT and pre-trained models **III:** Fine-tuning for specific tasks

Learning Outcomes I: Apply pre-trained models with HuggingFace [Usage] **II:** Evaluate NLP performance metrics [Assessment]

Bibliography: [Vaswani et al., 2017, Devlin et al., 2019]

5.4 Practical Applications, 15 hr(s)

Competences: ABET-1,ABET-2

Topics: I: Sentiment analysis with transformers **II:** Question-answering systems **III:** Model deployment with Flask/FastAPI

Learning Outcomes I: Build a complete NLP pipeline [Usage] **II:** Optimize models for production [Assessment]

Bibliography: [Jurafsky and Martin, 2021, Chollet, 2021]

6. Bibliography

[Chollet, 2021] Chollet, F. (2021). *Deep Learning with Python*. Manning, 2nd edition.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*.

[Goldberg, 2017] Goldberg, Y. (2017). *Neural Network Methods for Natural Language Processing*. Morgan & Claypool.

[Jurafsky and Martin, 2021] Jurafsky, D. and Martin, J. H. (2021). *Speech and Language Processing*. Pearson, 3rd edition.

[Manning and Schütze, 1999] Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

[Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

[Chollet, 2021] Chollet, F. (2021). *Deep Learning with Python*. Manning, 2nd edition.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*.

[Goldberg, 2017] Goldberg, Y. (2017). *Neural Network Methods for Natural Language Processing*. Morgan & Claypool.

[Jurafsky and Martin, 2021] Jurafsky, D. and Martin, J. H. (2021). *Speech and Language Processing*. Pearson, 3rd edition.

[Manning and Schütze, 1999] Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

[Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS273 Time series analysis (Elective)

1. General information

- **Semester**: 9th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA351 Statistics and Probabilities (6th Sem-Pg. 415)

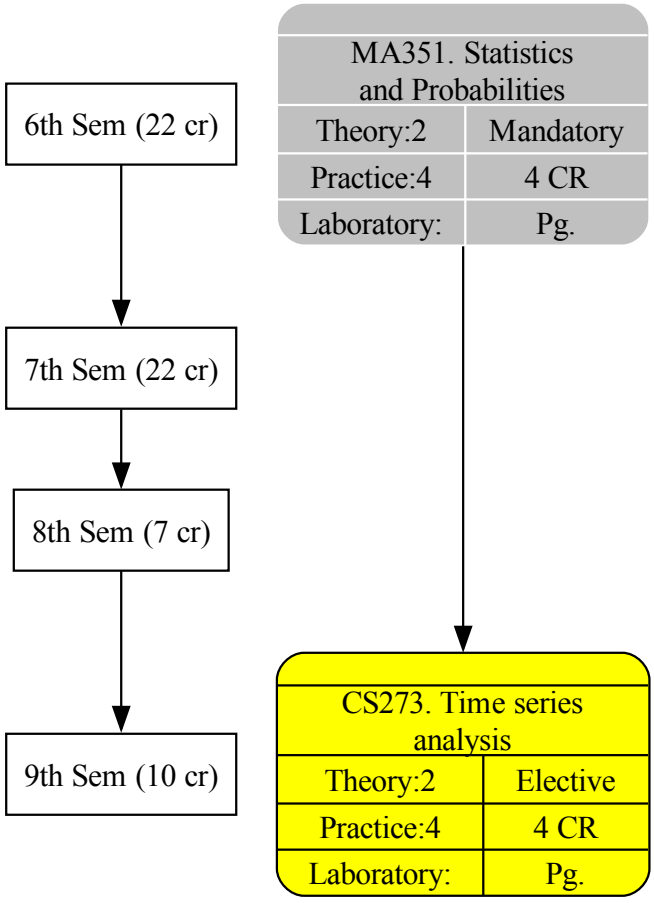


Figure 7.59: Connection Map. CS273 Time series analysis

2. Course Rationale

In various fields such as finance, meteorology, and engineering, data is collected sequentially over time, presenting unique temporal dependencies. This course introduces the theoretical and practical foundations for analyzing, modeling, and forecasting time series. It covers everything from classical decomposition methods to advanced stochastic models (ARIMA, GARCH), allowing the student to extract relevant information from dynamic processes and make decisions based on rigorous predictions.

3. Goals of the Course

- Understand the fundamental components of a time series (trend, seasonality, cycle, and residual).
- Identify and transform series to achieve stationarity.
- Model stochastic processes using Box-Jenkins methodologies.
- Evaluate forecast quality through error metrics and temporal cross-validation.
- Implement advanced predictive models for series with volatility or complex dependencies.

4. General Competences (*Outcomes*)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Introduction and Fundamental Concepts, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Definition and examples of time series. **II:** Series components: Trend, Seasonality, and Noise. **III:** Lag and difference operators. **IV:** Smoothing operations: Simple and weighted moving averages.

Learning Outcomes **I:** Decompose a time series into its basic components [Usage]. **II:** Identify visual patterns of temporal dependence in real data [Familiarity].

Bibliography: [Hyndman and Athanasopoulos, 2021, Box et al., 2015]

5.2 Stationarity and Autocorrelation Functions, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Weak and strong stationarity. **II:** Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF). **III:** White noise and Random Walk. **IV:** Unit root tests (Augmented Dickey-Fuller).

Learning Outcomes **I:** Determine if a series is stationary using statistical tests [Usage]. **II:** Interpret correlograms to identify possible stochastic models [Assessment].

Bibliography: [Shumway and Stoffer, 2017, Box et al., 2015]

5.3 Linear Process Models (AR and MA), 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Autoregressive processes (AR). **II:** Moving Average processes (MA). **III:** Invertibility and stability conditions. **IV:** Parameter estimation and information criteria (AIC, BIC).

Learning Outcomes **I:** Model linear dependencies in stationary time series [Usage]. **II:** Select the optimal model order based on statistical criteria [Assessment].

Bibliography: [Shumway and Stoffer, 2017, Brockwell and Davis, 2016]

5.4 Box-Jenkins Methodology and ARIMA Models, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Integrated models (ARIMA). **II:** Identification, estimation, and diagnostic checking. **III:** Residual analysis and Ljung-Box test. **IV:** Seasonal models (SARIMA).

Learning Outcomes I: Implement the full Box-Jenkins modeling cycle [Usage]. **II:** Validate model adequacy through residual analysis [Assessment].

Bibliography: [Box et al., 2015, Hyndman and Athanasopoulos, 2021]

5.5 Forecasting and Model Evaluation, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Point forecasts and confidence intervals. **II:** Error metrics: MAE, RMSE, MAPE, and MASE. **III:** Time Series Cross-Validation (Time Series Split). **IV:** Forecast combinations.

Learning Outcomes I: Generate future predictions with their respective uncertainty margins [Usage]. **II:** Rigorously compare the performance of multiple predictive models [Assessment].

Bibliography: [Hyndman and Athanasopoulos, 2021, Brockwell and Davis, 2016]

5.6 Volatility Models (ARCH and GARCH), 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Conditional heteroskedasticity in financial series. **II:** ARCH (Autoregressive Conditional Heteroskedasticity) models. **III:** GARCH models and their extensions. **IV:** Risk estimation and Value at Risk (VaR).

Learning Outcomes I: Identify the presence of volatility clustering in financial data [Usage]. **II:** Model conditional variance of complex dynamic processes [Assessment].

Bibliography: [Tsay, 2010, Shumway and Stoffer, 2017]

5.7 Ethics and Responsibility in Forecasting, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Socioeconomic consequences of erroneous forecasts. **II:** Transparency in communicating uncertainty. **III:** Ethical use of sensitive temporal data and privacy. **IV:** Biases in automated prediction algorithms.

Learning Outcomes I: Recognize the ethical impact of decisions based on predictions [Familiarity]. **II:** Communicate model limitations and risks to stakeholders [Usage].

Bibliography: [Hyndman and Athanasopoulos, 2021]

5.8 Implementation and Real-World Applications, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Computational tools (R: 'forecast', 'fable'; Python: 'statsmodels'). **II:** Automation of forecasting workflows. **III:** Applications in energy demand, inventory, and markets. **IV:** Presentation of a final prediction project.

Learning Outcomes I: Develop automated forecasting systems using specialized software [Assessment]. **II:** Solve a real prediction problem by integrating all course stages [Assessment].

Bibliography: [Tsay, 2010, Hyndman and Athanasopoulos, 2021]

6. Bibliography

[Box et al., 2015] Box, G. E. P., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time Series Analysis: Forecasting and Control*. Wiley, 5th edition.

[Brockwell and Davis, 2016] Brockwell, P. J. and Davis, R. A. (2016). *Introduction to Time Series and Forecasting*. Springer, 3rd edition.

[Hyndman and Athanasopoulos, 2021] Hyndman, R. J. and Athanasopoulos, G. (2021). *Forecasting: Principles and Practice*. OTexts.

[Shumway and Stoffer, 2017] Shumway, R. H. and Stoffer, D. S. (2017). *Time Series Analysis and Its Applications: With R Examples*. Springer, 4th edition.

[Tsay, 2010] Tsay, R. S. (2010). *Analysis of Financial Time Series*. Wiley, 3rd edition.

[Box et al., 2015] Box, G. E. P., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time Series Analysis: Forecasting and Control*. Wiley, 5th edition.

[Brockwell and Davis, 2016] Brockwell, P. J. and Davis, R. A. (2016). *Introduction to Time Series and Forecasting*. Springer, 3rd edition.

[Hyndman and Athanasopoulos, 2021] Hyndman, R. J. and Athanasopoulos, G. (2021). *Forecasting: Principles and Practice*. OTexts, 3rd edition.

[Shumway and Stoffer, 2017] Shumway, R. H. and Stoffer, D. S. (2017). *Time Series Analysis and Its Applications: With R Examples*. Springer, 4th edition.

[Tsay, 2010] Tsay, R. S. (2010). *Analysis of Financial Time Series*. Wiley, 3rd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA341 Optimization I (Elective)

1. General information

- **Semester**: 9th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - MA221 Numerical Analysis I (5th Sem-Pg. 393)

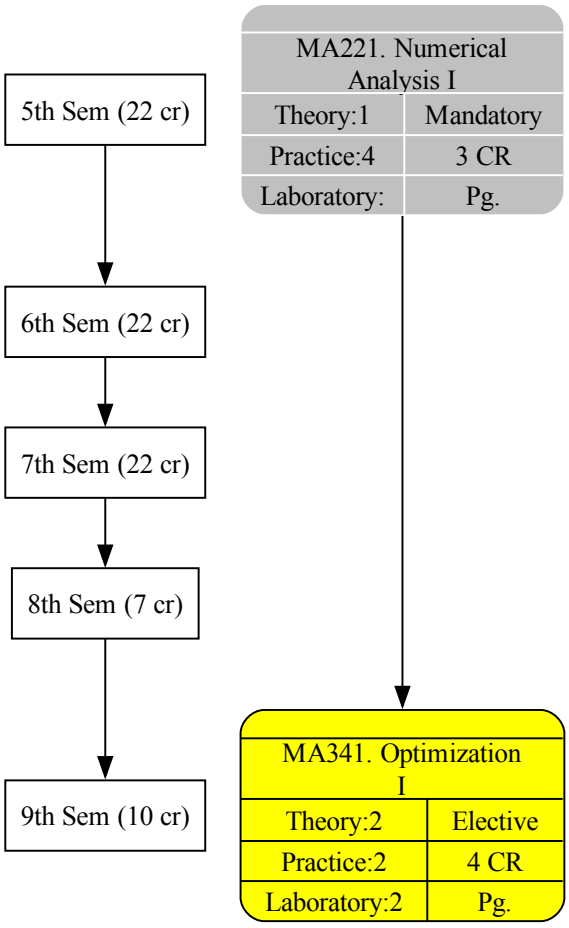


Figure 7.60: Connection Map. MA341 Optimization I

2. Course Rationale

This course introduces the mathematical foundations of optimization, covering both unconstrained and constrained problems. Classical methods of continuous optimization, optimality conditions, numerical algorithms, and applications in various fields are studied. The course provides the theoretical and computational foundations for formulating and solving optimization problems in science, engineering, and economics, developing analytical skills to characterize and find optimal solutions.

3. Goals of the Course

- Understand the fundamental concepts of optimization and classify different types of optimization problems.
- Apply optimality conditions to characterize solutions in continuous optimization problems.
- Implement numerical algorithms to solve unconstrained and constrained optimization problems.
- Analyze the convergence and efficiency of different optimization methods.
- Formulate and solve real-world optimization problems in various applied contexts.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Convex and Non-Convex Optimization, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Convex sets and functions **II:** Optimality conditions **III:** Gradient methods for convex optimization **IV:** Non-convex optimization challenges **V:** Convergence analysis

Learning Outcomes **I:** Characterize convex optimization problems [Assessment] **II:** Apply optimality conditions to verify solutions [Usage] **III:** Implement gradient-based methods for convex problems [Usage] **IV:** Analyze challenges in non-convex optimization [Assessment]

Bibliography: [Nocedal and Wright, 2006, Boyd and Vandenberghe, 2004]

5.2 Constrained Optimization, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Linear programming (simplex method) **II:** Integer programming **III:** Lagrange multipliers and KKT conditions **IV:** Penalty and barrier methods **V:** Duality theory

Learning Outcomes **I:** Solve linear programming problems [Usage] **II:** Formulate and solve integer programming problems [Usage] **III:** Apply KKT conditions to constrained problems [Usage] **IV:** Use penalty methods for constrained optimization [Usage]

Bibliography: [Nocedal and Wright, 2006, Boyd and Vandenberghe, 2004]

5.3 Optimization for Machine Learning, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Stochastic gradient descent **II:** Momentum and adaptive methods (Adam, RMSProp) **III:** Regularization and sparsity **IV:** Distributed optimization for ML **V:** Second-order methods for ML

Learning Outcomes **I:** Implement stochastic gradient descent variants [Usage] **II:** Apply adaptive optimization methods [Usage] **III:** Use regularization for model generalization [Usage] **IV:** Design distributed optimization strategies [Usage]

Bibliography: [Nocedal and Wright, 2006, Bertsekas, 2016]

5.4 Metaheuristics and Bio-Inspired Optimization, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Genetic algorithms and evolutionary strategies **II:** Swarm intelligence (PSO, ant colony) **III:** Simulated annealing **IV:** Tabu search **V:** Hybrid metaheuristics

Learning Outcomes **I:** Implement genetic algorithms [Usage] **II:** Apply swarm intelligence methods [Usage] **III:** Use simulated annealing for global optimization [Usage] **IV:** Design hybrid metaheuristic approaches [Usage]

Bibliography: [Talbi, 2009, Mirjalili, 2019]

6. Bibliography

[Bertsekas, 2016] Bertsekas, D. P. (2016). *Nonlinear Programming*. Athena Scientific, 3rd edition.

[Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, 1st edition.

[Mirjalili, 2019] Mirjalili, S. (2019). *Evolutionary Algorithms and Neural Networks: Theory and Applications*. Springer, 1st edition.

[Nocedal and Wright, 2006] Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer, 2nd edition.

[Talbi, 2009] Talbi, E.-G. (2009). *Metaheuristics: From Design to Implementation*. Wiley, 1st edition.

[Bertsekas, 2016] Bertsekas, D. P. (2016). *Nonlinear Programming*. Athena Scientific, 3rd edition.

[Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, 1st edition.

[Mirjalili, 2019] Mirjalili, S. (2019). *Evolutionary Algorithms and Neural Networks: Theory and Applications*. Springer, 1st edition.

[Nocedal and Wright, 2006] Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer, 2nd edition.

[Talbi, 2009] Talbi, E.-G. (2009). *Metaheuristics: From Design to Implementation*. Wiley, 1st edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA355 Statistical Learning Theory (Elective)

1. General information

- **Semester**: 9th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - MA352 Stochastic processes (8th Sem-Pg. 490)
  - MA372 Functional analysis II (8th Sem-Pg. 493)

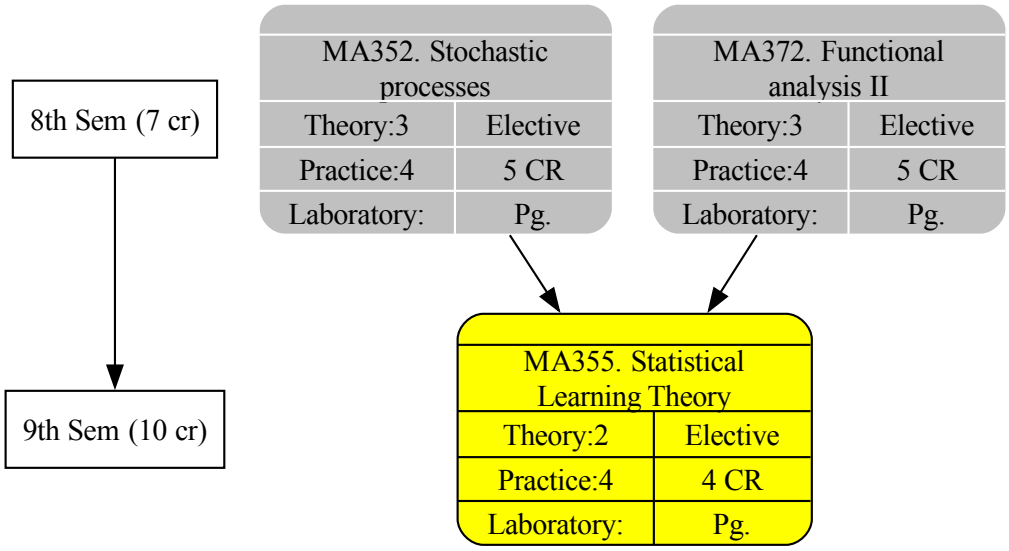


Figure 7.61: Connection Map. MA355 Statistical Learning Theory

2. Course Rationale

This course provides the theoretical foundations of statistical learning, covering the mathematical principles that support machine learning algorithms. It focuses on the theoretical analysis of generalization capability, model complexity, and performance guarantees for various learning methods. The course connects theory with practice through the study of generalization bounds, bias-variance tradeoff, and foundations of computational learning theory.

3. Goals of the Course

- Understand the fundamentals of statistical learning theory and its probabilistic foundations.
- Analyze the bias-variance tradeoff in the selection and evaluation of machine learning models.
- Apply Vapnik-Chervonenkis (VC) theory to analyze model complexity and generalization capability.
- Develop and demonstrate generalization bounds for different learning algorithms.
- Connect theoretical principles with the practical implementation of statistical learning algorithms.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Foundations of Statistical Learning, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Bias-variance tradeoff **II:** Vapnik-Chervonenkis (VC) theory **III:** Empirical risk minimization **IV:** Generalization bounds **V:** Learning theory fundamentals

Learning Outcomes I: Analyze the bias-variance tradeoff in model selection [Assessment] **II:** Apply VC theory to analyze model complexity [Usage] **III:** Implement empirical risk minimization [Usage] **IV:** Interpret generalization bounds [Familiarity]

Bibliography: [Vapnik, 1998, Hastie et al., 2009]

5.2 Supervised Learning Methods, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Linear models and extensions **II:** Support Vector Machines (SVM) **III:** Decision trees and ensemble methods **IV:** Kernel methods **V:** Probabilistic supervised learning

Learning Outcomes I: Implement and analyze linear models [Usage] **II:** Apply SVM for classification and regression [Usage] **III:** Construct ensemble methods (bagging, boosting) [Usage] **IV:** Use kernel methods for nonlinear problems [Usage]

Bibliography: [Hastie et al., 2009, James et al., 2013]

5.3 Unsupervised Learning Methods, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Clustering algorithms (k-means, hierarchical) **II:** Dimensionality reduction (PCA, t-SNE) **III:** Density estimation **IV:** Manifold learning **V:** Autoencoders and deep unsupervised learning

Learning Outcomes I: Apply clustering algorithms to data exploration [Usage] **II:** Implement PCA and other dimensionality reduction methods [Usage] **III:** Use density estimation for unsupervised learning [Usage] **IV:** Analyze manifold learning techniques [Assessment]

Bibliography: [Murphy, 2012, Bishop, 2006]

5.4 Model Evaluation and Selection, 20 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Cross-validation strategies **II:** Performance metrics for classification and regression **III:** Model comparison and statistical tests **IV:** Hyperparameter optimization **V:** Learning curves and diagnostic tools

Learning Outcomes I: Implement cross-validation for model assessment [Usage] **II:** Select appropriate performance metrics [Assessment] **III:** Compare models using statistical tests [Usage] **IV:** Optimize hyperparameters using systematic methods [Usage]

Bibliography: [Hastie et al., 2009, James et al., 2013]

6. Bibliography

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer, 1st edition.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd edition.

[James et al., 2013] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer, 1st edition.

[Murphy, 2012] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press, 1st edition.

[Vapnik, 1998] Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley, 1st edition.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer, 1st edition.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd edition.

[James et al., 2013] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer, 1st edition.

[Murphy, 2012] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press, 1st edition.

[Vapnik, 1998] Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley, 1st edition.

Tenth Semester

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS400 Pre-professional internships (Mandatory)

1. General information

- **Semester**: 10^{th} Sem. **Credits**: 3
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - 110Cr

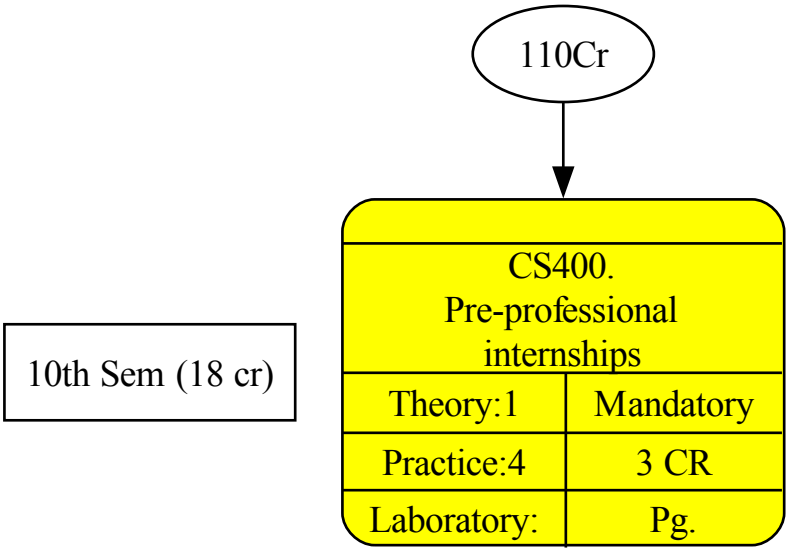


Figure 7.62: Connection Map. CS400 Pre-professional internships

2. Course Rationale

This course allows students to apply the knowledge acquired during their academic training in a real work environment, under the supervision of a company and the university. Pre-professional internships are essential to develop professional, ethical, and technical competencies, ensuring that the student gains experience in projects related to the degree program.

3. Goals of the Course

- Evaluate student performance in a real work environment, applying Computer Science knowledge.
- Develop professional, ethical, and teamwork skills in a business context.
- Ensure that the activities performed are aligned with the degree program objectives.

4. General Competences (*Outcomes*)

5. Content

5.1 Performance Evaluation in Work Environment, 48 hr(s)

Competences:

Topics: **I:** Student integration into a professional work team. **II:** Application of technical knowledge in real projects. **III:** Evaluation of ethical and professional competencies in the work environment. **IV:** Presentation of reports and feedback from the company and university.

Learning Outcomes **I:** Demonstrate technical skills in a real work environment [Assessment]. **II:** Apply ethical and professional principles in their activities [Assessment]. **III:** Work effectively in teams and communicate professionally [Assessment].

Bibliography: [de Educación de la ACM, 2022, Society, 2021]

6. Bibliography

[de Educación de la ACM, 2022] de Educación de la ACM, C. (2022). Guía para prácticas pre-profesionales exitosas en ciencia de la computación. Technical report, Association for Computing Machinery. Documento oficial con estándares para prácticas en computación.

[Society, 2021] Society, I. C. (2021). *Prácticas Profesionales en Computación: Ética y Gestión*. IEEE Press. Manual de buenas prácticas para entornos laborales en TI.

[de Educación de la ACM, 2022] de Educación de la ACM, C. (2022). Guía para prácticas pre-profesionales exitosas en ciencia de la computación. Technical report, Association for Computing Machinery. Documento oficial con estándares para prácticas en computación.

[Society, 2021] Society, I. C. (2021). *Prácticas Profesionales en Computación: Ética y Gestión*. IEEE Press. Manual de buenas prácticas para entornos laborales en TI.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

AI268 Computational Vision (Mandatory)

1. General information

- **Semester**: 10th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - AI264 Deep Learning (9th Sem-Pg. 496)

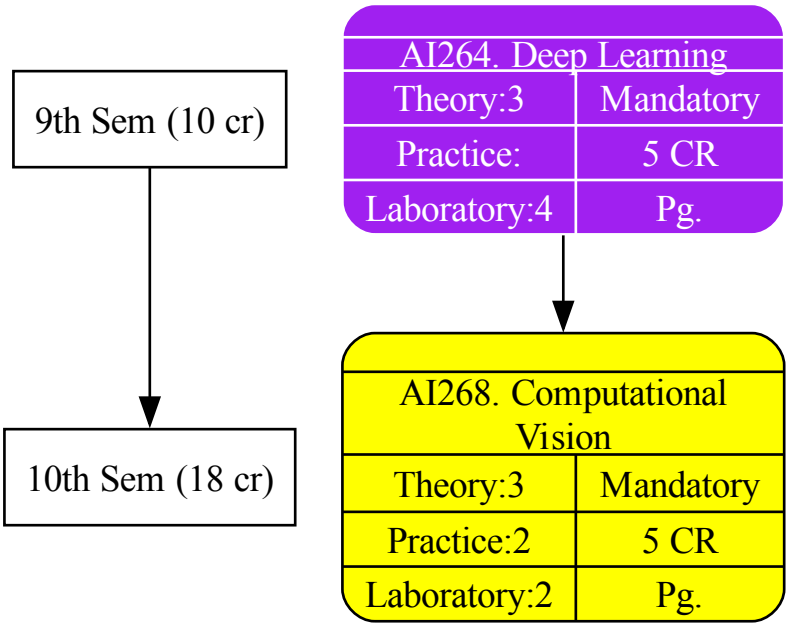


Figure 7.63: Connection Map. AI268 Computational Vision

2. Course Rationale

Computer Vision is a key area of Artificial Intelligence that enables machines to interpret and understand the visual world. This course covers fundamental techniques for image processing, feature extraction, and object recognition. With the advent of deep learning, computer vision has become essential for autonomous vehicles, medical imaging, and augmented reality. Students will learn the mathematical foundations and algorithmic approaches to solve real-world visual perception problems.

3. Goals of the Course

- Understand the fundamentals of image formation and digital representation.
- Master image processing techniques such as filtering and edge detection.
- Apply feature extraction methods to identify key patterns in images.
- Implement object detection and recognition using deep learning models.
- Analyze 3D vision and motion estimation techniques.

4. General Competences (*Outcomes*)

5. Content

5.1 Introduction to AI: Concepts and Intelligent Behavior, 5 hr(s)

Competences:

Topics: **I:** Overview of AI problems, Examples of successful recent AI applications **II:** Definitions of agents with examples (e.g., reactive, deliberative) **III:** What is intelligent behavior?: (a) The Turing test and its flaws (b) Multimodal input and output (c) Simulation of intelligent behavior (d) Rational versus non-rational reasoning **IV:** Overview of AI Applications, growth, and impact (economic, societal, ethics) **V:** Philosophical issues **VI:** History of AI

Learning Outcomes **I:** Describe the Turing test and the "Chinese Room" thought experiment [Explain] **II:** Differentiate between optimal reasoning/behavior and human-like reasoning/behavior [Evaluate] **III:** Differentiate the terms: AI, machine learning, and deep learning [Explain]

Bibliography: [Szeliski, 2022, Gonzalez and Woods, 2018]

5.2 Problem Characteristics and Agent Nature, 5 hr(s)

Competences:

Topics: **I:** Problem characteristics: (a) Fully versus partially observable (b) Single versus multi-agent (c) Deterministic versus stochastic (d) Static versus dynamic (e) Discrete versus continuous **II:** Nature of agents: (a) Autonomous, semi-autonomous, mixed-initiative autonomy (b) Reflexive, goal-based, and utility-based (c) Decision making under uncertainty and with incomplete information (d) The importance of perception and environmental interactions (e) Learning-based agents (f) Embodied agents: i. sensors, dynamics, effectors **III:** Practice identifying problem characteristics in example environments **IV:** Additional depth on nature of agents with examples **V:** Additional depth on AI Applications, Growth, and Impact (economic, societal, ethics, security)

Learning Outcomes **I:** Enumerate the characteristics of a specific problem [Apply]

Bibliography: [Szeliski, 2022, Gonzalez and Woods, 2018]

5.3 Perception and Computer Vision, 12 hr(s)

Competences:

Topics: **I:** Computer vision: (a) Image acquisition, representation, processing, and properties (b) Shape representation, object recognition, and segmentation (c) Motion analysis (d) Generative models **II:** Audio and speech recognition **III:** Touch and proprioception **IV:** Other modalities (e.g., olfaction) **V:** Modularity in recognition **VI:** Approaches to pattern recognition: (a) Classification algorithms and measures of classification quality (b) Statistical techniques (c) Deep learning techniques

Learning Outcomes **I:** Summarize the importance of image and object recognition in AI and indicate several significant applications of this technology [Summarize] **II:** List at least three image-segmentation approaches, such as thresholding, edge-based and region-based algorithms, along with their defining characteristics, strengths, and weaknesses [List] **III:** Implement 2d object recognition based on contour-based and/or region-based shape representations [Apply] **IV:** Distinguish the goals of sound-recognition, speech-recognition, and speaker-recognition and identify how the raw audio signal will be handled differently in each of these cases [Distinguish] **V:** Provide at least two examples of a transformation of a data source from one sensory domain to another, e.g., tactile data interpreted as single-band 2d images [Generate] **VI:** Implement a feature-extraction algorithm on real data, e.g., an edge or corner detector for images or vectors of Fourier coefficients describing a short slice of audio signal [Apply] **VII:** Implement an

algorithm combining features into higher-level percepts, e.g., a contour or polygon from visual primitives or phoneme hypotheses from an audio signal [Apply] **VIII:** Implement a classification algorithm that segments input percepts into output categories and quantitatively evaluates the resulting classification [Apply] **IX:** Evaluate the performance of the underlying feature-extraction, relative to at least one alternative possible approach (whether implemented or not) in its contribution to the classification task (8), above [Evaluate] **X:** Describe at least three classification approaches, their prerequisites for applicability, their strengths, and their shortcomings [Describe] **XI:** Implement and evaluate a deep learning solution to problems in computer vision, such as object or scene recognition [Apply]

Bibliography: [Szeliski, 2022, Prince, 2012]

5.4 Machine Learning Fundamentals, 5 hr(s)

Competences:

Topics: I: Definition and examples of a broad variety of machine learning tasks: (a) Supervised learning: i. Classification ii. Regression (b) Reinforcement learning (c) Unsupervised learning: i. Clustering **II:** Fundamental ideas: (a) No free lunch theorem: no one learner can solve all problems; representational design decisions have consequences. (b) Sources of error and undecidability in machine learning **III:** A simple statistical-based supervised learning such as linear regression or decision trees: (a) Focus on how they work without going into mathematical or optimization details; enough to understand and use existing implementations correctly **IV:** The overfitting problem/controlling solution complexity (regularization, pruning - intuition only): (a) The bias (underfitting) - variance (overfitting) tradeoff

Learning Outcomes I: Describe the differences among the three main styles of learning (supervised, reinforcement, and unsupervised) and determine which is appropriate to a particular problem domain [Explain] **II:** Differentiate the terms of AI, machine learning, and deep learning [Evaluate] **III:** Frame an application as a classification problem, including the available input features and output to be predicted (e.g., identifying alphabetic characters from pixel grid input) [Apply] **IV:** Identify overfitting in the context of a problem and learning curves and describe solutions to overfitting [Evaluate] **V:** Explain how machine learning works as an optimization/search process [Explain]

Bibliography: [Goodfellow et al., 2016, Russell and Norvig, 2020]

5.5 Machine Learning Data, Representations and Evaluation, 5 hr(s)

Competences:

Topics: I: Working with Data: (a) Data preprocessing: i. Importance and pitfalls of preprocessing choices (b) Handling missing values (imputing, flag-as-missing): i. Implications of imputing vs flag-as-missing (c) Encoding categorical variables, encoding real-valued data (d) Normalization/standardization (e) Emphasis on real data, not textbook examples **II:** Representations: (a) Hypothesis spaces and complexity (b) Simple basis feature expansion, such as squaring univariate features (c) Learned feature representations **III:** Machine learning evaluation: (a) Separation of train, validation, and test sets (b) Performance metrics for classifiers (c) Estimation of test performance on held-out data (d) Tuning the parameters of a machine learning model with a validation set (e) Importance of understanding what a model is doing, where its pitfalls/shortcomings are, and the implications of its decisions

Learning Outcomes I: Apply two or more simple statistical learning algorithms to a classification task and measure the classifiers' accuracy [Apply] **II:** Implement a statistical learning algorithm and the corresponding optimization process to train the classifier and obtain a prediction on new data [Apply] **III:** Explain proper ML evaluation procedures, including the differences between training and testing performance, and what can go wrong with the evaluation process leading to inaccurate reporting of ML performance [Explain] **IV:** Compare two machine learning

algorithms on a dataset, implementing the data preprocessing and evaluation methodology (e.g., metrics and handling of train/test splits) from scratch [Apply]

Bibliography: [Goodfellow et al., 2016, Russell and Norvig, 2020]

5.6 Neural Networks and Ethics, 4 hr(s)

Competences:

Topics: **I:** Basic neural networks: (a) Fundamentals of understanding how neural networks work and their training process, without details of the calculations (b) Basic introduction to generative neural networks (e.g., large language models) **II:** Ethics for Machine Learning: (a) Focus on real data, real scenarios, and case studies (b) Dataset/algorithmic/evaluation bias and unintended consequences

Learning Outcomes **I:** Describe the neural network training process and resulting learned representations [Explain]

Bibliography: [Goodfellow et al., 2016, Russell and Norvig, 2020]

5.7 Machine Learning Formulation and Evaluation, 3 hr(s)

Competences:

Topics: **I:** Formulation of simple machine learning as an optimization problem, such as least squares linear regression or logistic regression: (a) Objective function (b) Gradient descent (c) Regularization to avoid overfitting (mathematical formulation) **II:** Ensembles of models: (a) Simple weighted majority combination **III:** Performance evaluation: (a) Other metrics for classification (e.g., error, precision, recall) (b) Performance metrics for regressors (c) Confusion matrix (d) Cross-validation: i. Parameter tuning (grid/random search, via cross-validation)

Learning Outcomes **I:** Compare and contrast several learning techniques (e.g., decision trees, logistic regression, naive Bayes, neural networks, and belief networks), providing examples of when each strategy is superior [Compare] **II:** Evaluate the performance of a simple learning system on a real-world dataset [Evaluate] **III:** Characterize the state of the art in learning theory, including its achievements and shortcomings [Explain] **IV:** Explain the problem of overfitting, along with techniques for detecting and managing the problem [Explain] **V:** Explain the triple tradeoff among the size of a hypothesis space, the size of the training set, and performance accuracy [Explain]

Bibliography: [Goodfellow et al., 2016]

5.8 Deep Learning, RL and Ethics, 3 hr(s)

Competences:

Topics: **I:** Deep learning: (a) Deep feed-forward networks (intuition only, no mathematics) (b) Convolutional neural networks (intuition only, no mathematics) (c) Visualization of learned feature representations from deep nets (d) Other architectures (generative NN, recurrent NN, transformers, etc.) **II:** Overview of reinforcement learning methods **III:** Two or more applications of machine learning algorithms: (a) E.g., medicine and health, economics, vision, natural language, robotics, game play **IV:** Ethics for Machine Learning: (a) Continued focus on real data, real scenarios, and case studies (b) Privacy (c) Fairness (d) Intellectual property (e) Explainability

Learning Outcomes **I:** Visualize the training progress of a neural network through learning curves in a well-established toolkit (e.g., TensorBoard) and visualize the learned features of the network [Apply] **II:** Given a real-world application of machine learning, describe ethical issues regarding the choices of data, preprocessing steps, algorithm selection, and visualization/presentation of results [Apply]

Bibliography: [Goodfellow et al., 2016]

5.9 Perception and Computer Vision, 12 hr(s)

Competences:

Topics: **I:** Computer vision: (a) Image acquisition, representation, processing, and properties (b) Shape representation, object recognition, and segmentation (c) Motion analysis (d) Generative models **II:** Audio and speech recognition **III:** Touch and proprioception **IV:** Other modalities (e.g., olfaction) **V:** Modularity in recognition **VI:** Approaches to pattern recognition: (a) Classification algorithms and measures of classification quality (b) Statistical techniques (c) Deep learning techniques

Learning Outcomes **I:** Summarize the importance of image and object recognition in AI and indicate several significant applications of this technology [Summarize] **II:** List at least three image-segmentation approaches, such as thresholding, edge-based and region-based algorithms, along with their defining characteristics, strengths, and weaknesses [List] **III:** Implement 2d object recognition based on contour-based and/or region-based shape representations [Apply] **IV:** Distinguish the goals of sound-recognition, speech-recognition, and speaker-recognition and identify how the raw audio signal will be handled differently in each of these cases [Distinguish] **V:** Provide at least two examples of a transformation of a data source from one sensory domain to another, e.g., tactile data interpreted as single-band 2d images [Generate] **VI:** Implement a feature-extraction algorithm on real data, e.g., an edge or corner detector for images or vectors of Fourier coefficients describing a short slice of audio signal [Apply] **VII:** Implement an algorithm combining features into higher-level percepts, e.g., a contour or polygon from visual primitives or phoneme hypotheses from an audio signal [Apply] **VIII:** Implement a classification algorithm that segments input percepts into output categories and quantitatively evaluates the resulting classification [Apply] **IX:** Evaluate the performance of the underlying feature-extraction, relative to at least one alternative possible approach (whether implemented or not) in its contribution to the classification task (8), above [Evaluate] **X:** Describe at least three classification approaches, their prerequisites for applicability, their strengths, and their shortcomings [Describe] **XI:** Implement and evaluate a deep learning solution to problems in computer vision, such as object or scene recognition [Apply]

Bibliography: [Szeliski, 2022]

6. Bibliography

[Gonzalez and Woods, 2018] Gonzalez, R. C. and Woods, R. E. (2018). *Digital Image Processing*. Pearson, 4th edition.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Prince, 2012] Prince, S. J. D. (2012). *Computer Vision: Models, Learning, and Inference*. Cambridge University Press.

[Russell and Norvig, 2020] Russell, S. and Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition.

[Szeliski, 2022] Szeliski, R. (2022). *Computer Vision: Algorithms and Applications*. Springer, 2nd edition.

[Gonzalez and Woods, 2018] Gonzalez, R. C. and Woods, R. E. (2018). *Digital Image Processing*. Pearson, 4th edition.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Prince, 2012] Prince, S. J. D. (2012). *Computer Vision: Models, Learning, and Inference*. Cambridge University Press.

[Russell and Norvig, 2020] Russell, S. and Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition.

[Szeliski, 2022] Szeliski, R. (2022). *Computer Vision: Algorithms and Applications*. Springer, 2nd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

AI367 Topics in Artificial Intelligence (Mandatory)

1. General information

- **Semester**: 10th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 3 hours; **Practice**: 2 hours; **Laboratory**: 2 hours;
- **Prerrequisites**:
  - AI264 Deep Learning (9th Sem-Pg. 496)

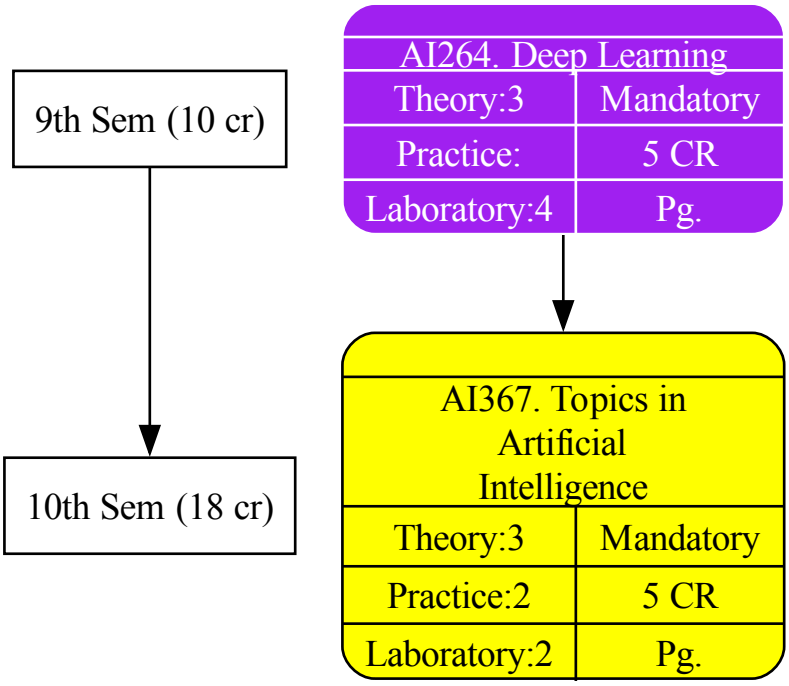


Figure 7.64: Connection Map. AI367 Topics in Artificial Intelligence

2. Course Rationale

Artificial Intelligence is a rapidly evolving field. This course explores advanced and contemporary topics that are currently shaping the industry and research. Focus is placed on Generative AI, Large Language Models (LLMs), and Advanced Reinforcement Learning. Students will analyze the architectures behind modern AI systems, such as Transformers, and discuss the technical and societal challenges of deploying autonomous intelligent agents at scale.

3. Goals of the Course

- Understand the architecture and training process of Large Language Models.
- Master the principles of Generative AI for text and image synthesis.
- Implement advanced Reinforcement Learning algorithms for decision making.
- Apply prompt engineering and fine-tuning techniques to existing models.
- Critically evaluate the latest research papers and trends in AI.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

5. Content

5.1 Machine Learning Fundamentals, 6 hr(s)

Competences: ABET-1

Topics: **I:** Definition and examples of a broad variety of machine learning tasks: (a) Supervised learning: i. Classification ii. Regression (b) Reinforcement learning (c) Unsupervised learning: i. Clustering **II:** Fundamental ideas: (a) No free lunch theorem: no one learner can solve all problems; representational design decisions have consequences. (b) Sources of error and undecidability in machine learning **III:** A simple statistical-based supervised learning such as linear regression or decision trees: (a) Focus on how they work without going into mathematical or optimization details; enough to understand and use existing implementations correctly **IV:** The overfitting problem/controlling solution complexity (regularization, pruning - intuition only): (a) The bias (underfitting) - variance (overfitting) tradeoff

Learning Outcomes **I:** Describe the differences among the three main styles of learning (supervised, reinforcement, and unsupervised) and determine which is appropriate to a particular problem domain [Explain] **II:** Differentiate the terms of AI, machine learning, and deep learning [Evaluate] **III:** Frame an application as a classification problem, including the available input features and output to be predicted (e.g., identifying alphabetic characters from pixel grid input) [Apply] **IV:** Identify overfitting in the context of a problem and learning curves and describe solutions to overfitting [Evaluate] **V:** Explain how machine learning works as an optimization/search process [Explain]

Bibliography: [Goodfellow et al., 2016, Foster, 2022]

5.2 Machine Learning Data, Representations and Evaluation, 5 hr(s)

Competences: ABET-1

Topics: **I:** Working with Data: (a) Data preprocessing: i. Importance and pitfalls of preprocessing choices (b) Handling missing values (imputing, flag-as-missing): i. Implications of imputing vs flag-as-missing (c) Encoding categorical variables, encoding real-valued data (d) Normalization/standardization (e) Emphasis on real data, not textbook examples **II:** Representations: (a) Hypothesis spaces and complexity (b) Simple basis feature expansion, such as squaring univariate features (c) Learned feature representations **III:** Machine learning evaluation: (a) Separation of train, validation, and test sets (b) Performance metrics for classifiers (c) Estimation of test performance on held-out data (d) Tuning the parameters of a machine learning model with a validation set (e) Importance of understanding what a model is doing, where its pitfalls/shortcomings are, and the implications of its decisions

Learning Outcomes **I:** Apply two or more simple statistical learning algorithms to a classification task and measure the classifiers' accuracy [Apply] **II:** Implement a statistical learning algorithm and the corresponding optimization process to train the classifier and obtain a prediction on new data [Apply] **III:** Explain proper ML evaluation procedures, including the differences between training and testing performance, and what can go wrong with the evaluation process leading to inaccurate reporting of ML performance [Explain] **IV:** Compare two machine learning algorithms on a dataset, implementing the data preprocessing and evaluation methodology (e.g., metrics and handling of train/test splits) from scratch [Apply]

Bibliography: [Goodfellow et al., 2016, Foster, 2022]

5.3 Neural Networks and Ethics, 5 hr(s)

Competences: ABET-1

Topics: **I:** Basic neural networks: (a) Fundamentals of understanding how neural networks work and their training process, without details of the calculations (b) Basic introduction to generative neural networks (e.g., large language models) **II:** Ethics for Machine Learning: (a) Focus on real data, real scenarios, and case studies (b) Dataset/algorithmic/evaluation bias and unintended consequences

Learning Outcomes **I:** Describe the neural network training process and resulting learned representations [Explain]

Bibliography: [Goodfellow et al., 2016, Foster, 2022]

5.4 Machine Learning Formulation and Evaluation, 3 hr(s)

Competences:

Topics: **I:** Formulation of simple machine learning as an optimization problem, such as least squares linear regression or logistic regression: (a) Objective function (b) Gradient descent (c) Regularization to avoid overfitting (mathematical formulation) **II:** Ensembles of models: (a) Simple weighted majority combination **III:** Performance evaluation: (a) Other metrics for classification (e.g., error, precision, recall) (b) Performance metrics for regressors (c) Confusion matrix (d) Cross-validation: i. Parameter tuning (grid/random search, via cross-validation)

Learning Outcomes **I:** Compare and contrast several learning techniques (e.g., decision trees, logistic regression, naive Bayes, neural networks, and belief networks), providing examples of when each strategy is superior [Compare] **II:** Evaluate the performance of a simple learning system on a real-world dataset [Evaluate] **III:** Characterize the state of the art in learning theory, including its achievements and shortcomings [Explain] **IV:** Explain the problem of overfitting, along with techniques for detecting and managing the problem [Explain] **V:** Explain the triple tradeoff among the size of a hypothesis space, the size of the training set, and performance accuracy [Explain]

Bibliography: [Goodfellow et al., 2016]

5.5 Deep Learning, RL and Ethics, 3 hr(s)

Competences:

Topics: **I:** Deep learning: (a) Deep feed-forward networks (intuition only, no mathematics) (b) Convolutional neural networks (intuition only, no mathematics) (c) Visualization of learned feature representations from deep nets (d) Other architectures (generative NN, recurrent NN, transformers, etc.) **II:** Overview of reinforcement learning methods **III:** Two or more applications of machine learning algorithms: (a) E.g., medicine and health, economics, vision, natural language, robotics, game play **IV:** Ethics for Machine Learning: (a) Continued focus on real data, real scenarios, and case studies (b) Privacy (c) Fairness (d) Intellectual property (e) Explainability

Learning Outcomes **I:** Visualize the training progress of a neural network through learning curves in a well-established toolkit (e.g., TensorBoard) and visualize the learned features of the network [Apply] **II:** Given a real-world application of machine learning, describe ethical issues regarding the choices of data, preprocessing steps, algorithm selection, and visualization/presentation of results [Apply]

Bibliography: [Goodfellow et al., 2016]

5.6 Natural Language Processing, 20 hr(s)

Competences: ABET-1

Topics: **I:** Deterministic and stochastic grammars **II:** Parsing algorithms: (a) CFGs and chart parsers (e.g., CYK) (b) Probabilistic CFGs and weighted CYK **III:** Representing meaning/Semantics: (a) Logic-based knowledge representations (b) Semantic roles (c) Temporal representations (d) Beliefs, desires, and intentions **IV:** Corpus-based methods **V:** N-grams and HMMs **VI:** Smoothing and backoff **VII:** Examples of use: POS tagging and morphology **VIII:** Information retrieval: (a) Vector space model: i. TF & IDF (b) Precision and recall **IX:** Information extraction **X:** Language translation **XI:** Text classification, categorization: (a) Bag of words model **XII:** Deep learning for NLP: (a) RNNs (b) Transformers (c) Multimodal embeddings (e.g., images + text) (d) Generative language models

Learning Outcomes **I:** Define and contrast deterministic and stochastic grammars, providing examples to show the adequacy of each [Define] **II:** Simulate, apply, or implement classic and stochastic algorithms for parsing natural language [Apply] **III:** Identify the challenges of representing meaning [Analyze] **IV:** List the advantages of using standard corpora. Identify examples of current corpora for a variety of NLP tasks [List] **V:** Identify techniques for information retrieval, language translation, and text classification [Analyze] **VI:** Implement a TF/IDF transform, use it to extract features from a corpus, and train an off-the-shelf machine learning algorithm using those features to do text classification [Apply]

Bibliography: [Tunstall et al., 2022, Russell and Norvig, 2020]

5.7 Machine Learning Fundamentals, 6 hr(s)

Competences: ABET-1

Topics: **I:** Definition and examples of a broad variety of machine learning tasks: (a) Supervised learning: i. Classification ii. Regression (b) Reinforcement learning (c) Unsupervised learning: i. Clustering **II:** Fundamental ideas: (a) No free lunch theorem: no one learner can solve all problems; representational design decisions have consequences. (b) Sources of error and undecidability in machine learning **III:** A simple statistical-based supervised learning such as linear regression or decision trees: (a) Focus on how they work without going into mathematical or optimization details; enough to understand and use existing implementations correctly **IV:** The overfitting problem/controlling solution complexity (regularization, pruning - intuition only): (a) The bias (underfitting) - variance (overfitting) tradeoff

Learning Outcomes **I:** Describe the differences among the three main styles of learning (supervised, reinforcement, and unsupervised) and determine which is appropriate to a particular problem domain [Explain] **II:** Differentiate the terms of AI, machine learning, and deep learning [Evaluate] **III:** Frame an application as a classification problem, including the available input features and output to be predicted (e.g., identifying alphabetic characters from pixel grid input) [Apply] **IV:** Identify overfitting in the context of a problem and learning curves and describe solutions to overfitting [Evaluate] **V:** Explain how machine learning works as an optimization/search process [Explain]

Bibliography: [Sutton and Barto, 2018]

5.8 Machine Learning Data, Representations and Evaluation, 5 hr(s)

Competences: ABET-1

Topics: **I:** Working with Data: (a) Data preprocessing: i. Importance and pitfalls of preprocessing choices (b) Handling missing values (imputing, flag-as-missing): i. Implications of imputing vs flag-as-missing (c) Encoding categorical variables, encoding real-valued data (d) Normalization/standardization (e) Emphasis on real data, not textbook examples **II:** Representations:

(a) Hypothesis spaces and complexity (b) Simple basis feature expansion, such as squaring univariate features (c) Learned feature representations **III:** Machine learning evaluation: (a) Separation of train, validation, and test sets (b) Performance metrics for classifiers (c) Estimation of test performance on held-out data (d) Tuning the parameters of a machine learning model with a validation set (e) Importance of understanding what a model is doing, where its pitfalls/shortcomings are, and the implications of its decisions

Learning Outcomes **I:** Apply two or more simple statistical learning algorithms to a classification task and measure the classifiers' accuracy [Apply] **II:** Implement a statistical learning algorithm and the corresponding optimization process to train the classifier and obtain a prediction on new data [Apply] **III:** Explain proper ML evaluation procedures, including the differences between training and testing performance, and what can go wrong with the evaluation process leading to inaccurate reporting of ML performance [Explain] **IV:** Compare two machine learning algorithms on a dataset, implementing the data preprocessing and evaluation methodology (e.g., metrics and handling of train/test splits) from scratch [Apply]

Bibliography: [Sutton and Barto, 2018]

5.9 Neural Networks and Ethics, 5 hr(s)

Competences: ABET-1

Topics: **I:** Basic neural networks: (a) Fundamentals of understanding how neural networks work and their training process, without details of the calculations (b) Basic introduction to generative neural networks (e.g., large language models) **II:** Ethics for Machine Learning: (a) Focus on real data, real scenarios, and case studies (b) Dataset/algorithmic/evaluation bias and unintended consequences

Learning Outcomes **I:** Describe the neural network training process and resulting learned representations [Explain]

Bibliography: [Sutton and Barto, 2018]

5.10 AI Applications and Societal Impact, 6 hr(s)

Competences: ABET-1

Topics: **I:** At least one application of AI to a specific problem and field, such as medicine, health, sustainability, social media, economics, education, robotics, etc. (choose at least one for the CS Core): (a) Formulating and evaluating a specific application as an AI problem: i. How to deal with underspecified or ill-posed problems (b) Data availability/scarcity and cleanliness: i. Basic data cleaning and preprocessing ii. Data set bias (c) Algorithmic bias (d) Evaluation bias (e) Assessment of societal implications of the application **II:** Deployed deep generative models: (a) High-level overview of deep image generative models (e.g., as of 2023, DALL-E, Midjourney, Stable Diffusion, etc.), their uses, and their shortcomings/pitfalls. (b) High-level overview of large language models (e.g., as of 2023, ChatGPT, Bard, etc.), their uses, and their shortcomings/pitfalls. **III:** Overview of societal impact of AI: (a) Ethics (b) Fairness (c) Trust/explainability (d) Privacy and usage of training data (e) Human autonomy and oversight/regulations/legal requirements (f) Sustainability

Learning Outcomes **I:** Given a real-world application domain and problem, formulate an AI solution to it, identifying proper data/input, preprocessing, representations, AI techniques, and evaluation metrics/methodology [Apply]

Bibliography: [Russell and Norvig, 2020, Bostrom, 2014]

5.11 Additional AI Applications and Societal Impact, 6 hr(s)

Competences: ABET-1

Topics: **I:** One or more additional applications of AI to a broad set of problems and diverse fields, such as medicine, health, sustainability, social media, economics, education, robotics, etc. (choose a different area from that chosen for the CS Core): (a) Formulating and evaluating a specific application as an AI problem: i. How to deal with underspecified or ill-posed problems (b) Data availability/scarcity and cleanliness: i. Basic data cleaning and preprocessing ii. Data set bias (c) Algorithmic bias (d) Evaluation bias (e) Assessment of societal implications of the application **II:** Additional depth on deployed deep generative models: (a) Introduction to how deep image generative models work, (e.g., as of 2023, DALL-E, Midjourney, Stable Diffusion) including discussion of attention (b) Introduction to how large language models work, (e.g., as of 2023, ChatGPT, Bard) including discussion of attention (c) Idea of foundational models, how to use them, and the benefits/issues with training them from big data **III:** Analysis and discussion of the societal impact of AI: (a) Ethics (b) Fairness (c) Trust/explainability (d) Privacy and usage of training data (e) Human autonomy and oversight/regulations/legal requirements (f) Sustainability

Learning Outcomes **I:** Analyze the societal impact of one or more specific real-world AI applications, identifying issues regarding ethics, fairness, bias, trust, and explainability [Evaluate] **II:** Describe some of the failure modes of current deep generative models for language or images, and how this could affect their use in an application [Describe]

Bibliography: [Russell and Norvig, 2020, Bostrom, 2014]

6. Bibliography

[Bostrom, 2014] Bostrom, N. (2014). *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press.

[Foster, 2022] Foster, D. (2022). *Generative Deep Learning*. O'Reilly Media, 2nd edition.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Russell and Norvig, 2020] Russell, S. and Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition.

[Sutton and Barto, 2018] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition.

[Tunstall et al., 2022] Tunstall, L., von Werra, L., and Wolf, T. (2022). *Natural Language Processing with Transformers*. O'Reilly Media.

[Bostrom, 2014] Bostrom, N. (2014). *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press.

[Foster, 2022] Foster, D. (2022). *Generative Deep Learning*. O'Reilly Media, 2nd edition.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Russell and Norvig, 2020] Russell, S. and Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition.

[Sutton and Barto, 2018] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition.

[Tunstall et al., 2022] Tunstall, L., von Werra, L., and Wolf, T. (2022). *Natural Language Processing with Transformers*. O'Reilly Media.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

MA403 Capstone Project III (Mandatory)

1. General information

- **Semester**: 10th Sem. **Credits**: 5
- **Hour of this course**: **Theory**: 1 hours; **Practice**: 8 hours;
- **Prerrequisites**:
  - MA402 Capstone Project II (9th Sem-Pg. 499)

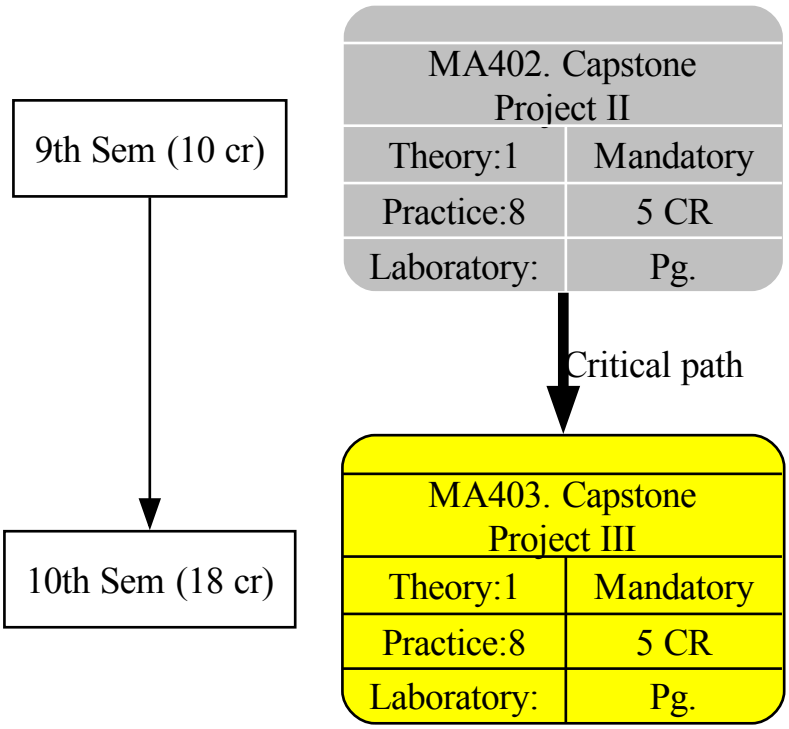


Figure 7.65: Connection Map. MA403 Capstone Project III

2. Course Rationale

The Undergraduate Thesis in Computational Mathematics (MA403) course is the culmination of the student's academic training and research process. This course focuses on the consolidation of the final thesis report, ensuring compliance with scientific standards and academic integrity. The student must demonstrate a total integration of mathematical knowledge and computational skills to solve a research problem, finalizing with the formal presentation of the manuscript and technical preparation for the defense before the academic and professional community.

3. Goals of the Course

- Fully implement the designed computational solution, applying advanced computational mathematics techniques.
- Perform exhaustive analysis of results and rigorous methodological validations.

- Professionally document the entire research and development process of the project.
- Evaluate the impact and limitations of the developed solution, proposing future improvements.
- Publicly defend the project before a specialized jury, demonstrating mastery of the subject and technical communication skills.
- Demonstrate capacity for autonomous work, project management, and professional ethics in the development of applied research.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Assessment)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Assessment)

ABET-4) An ability to communicate effectively with a range of audiences. (Assessment)

ABET-5) An ability to understand ethical and professional responsibilities and the impact of technical and/or scientific solutions in global, economic, environmental, and societal contexts. (Assessment)

ABET-6) An ability to function effectively on teams that establish goals, plan tasks, meet deadlines, and analyze risk and uncertainty. (Assessment)

5. Content

5.1 Technical Consolidation and Final Optimization, 20 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: I: Final review of algorithmic implementation and computational efficiency **II:** Code optimization and scalability of the solution **III:** Formal verification and validation of numerical results **IV:** Software repository management and scientific reproducibility

Learning Outcomes I: Conduct complete studies using research methods [Assessment] **II:** Formally demonstrate properties of solutions [Assessment] **III:** Implement complex mathematical processes on a computer [Usage] **IV:** Optimize computational solutions for efficiency [Assessment]

Bibliography: [Steve, 2004, E, 1997]

5.2 Methodological Validation and Critical Analysis, 20 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: I: Comparative analysis of results against the state of the art **II:** In-depth discussion on the impact and applications of the model **III:** Final statistical validation and error analysis **IV:** Evaluation of the original contribution of the work

Learning Outcomes I: Analyze social impact of scientific solutions [Assessment] **II:** Conduct rigorous methodological validations [Assessment] **III:** Recognize formal structure of complex problems [Assessment] **IV:** Interpret results in an applied context [Usage]

Bibliography: [C, 2017, P et al., 2005]

5.3 Academic Writing and Thesis Report, 20 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: **I:** Final structure of the thesis manuscript according to university standards **II:** Scientific writing techniques for mathematics and computing **III:** Citation management, bibliography, and academic integrity **IV:** Peer review and editorial correction processes

Learning Outcomes **I:** Apply ethical principles in scientific communication [Usage] **II:** Effectively communicate complex results [Assessment] **III:** Learn continuously through feedback [Usage] **IV:** Document processes professionally [Assessment]

Bibliography: [Michael, 1996, R and W, 2008]

5.4 Defense and Professional Projection, 20 hr(s)

Competences: ABET-1,ABET-2,ABET-4,ABET-5,ABET-6

Topics: **I:** Preparation of the public defense protocol **II:** Technical speaking and argumentation before academic peers **III:** Advanced data visualization for high-impact presentations **IV:** Publication plan for results in specialized forums

Learning Outcomes **I:** Perform effectively in public defense [Assessment] **II:** Manage complex scientific projects [Assessment] **III:** Solve advanced mathematical problems [Assessment] **IV:** Respond to criticism constructively [Usage]

Bibliography: [L, 2018, H, 2006]

6. Bibliography

[C, 2017] C, M. D. (2017). *Design and Analysis of Experiments*. John Wiley & Sons, 9th edition.

[E, 1997] E, K. D. (1997). *The Art of Computer Programming*. Addison-Wesley.

[H, 2006] H, A. R. R. (2006). *Dazzle 'em with Style: The Art of Oral Scientific Presentation*. Elsevier Academic Press.

[L, 2018] L, T. K. (2018). *A Manual for Writers of Research Papers, Theses, and Dissertations*. University of Chicago Press.

[Michael, 1996] Michael, A. (1996). *The Craft of Scientific Writing*. Springer.

[P et al., 2005] P, B. G. E., Stuart, H. J., and G, H. W. (2005). *Statistics for Experimenters: Design, Innovation, and Discovery*. Wiley-Interscience.

[R and W, 2008] R, M. J. and W, M. R. (2008). *Successful Scientific Writing: A Step-by-Step Guide for the Biological and Medical Sciences*. Cambridge University Press.

[Steve, 2004] Steve, M. (2004). *Code Complete: A Practical Handbook of Software Construction*. Microsoft Press, 2nd edition.

[C, 2017] C, M. D. (2017). *Design and Analysis of Experiments*. John Wiley & Sons, 9th edition.

[E, 1997] E, K. D. (1997). *The Art of Computer Programming*, volume 1-3. Addison-Wesley, Reading, MA.

[H, 2006] H, A. R. R. (2006). *Dazzle 'em with Style: The Art of Oral Scientific Presentation*. Elsevier Academic Press, Burlington, MA, 2nd edition.

[L, 2018] L, T. K. (2018). *A Manual for Writers of Research Papers, Theses, and Dissertations*. University of Chicago Press, Chicago, IL, 9th edition.

[Michael, 1996] Michael, A. (1996). *The Craft of Scientific Writing*. Springer, New York, 3rd edition.

[P et al., 2005] P, B. G. E., Stuart, H. J., and G, H. W. (2005). *Statistics for Experimenters: Design, Innovation, and Discovery*. Wiley-Interscience, Hoboken, NJ, 2nd edition.

[R and W, 2008] R, M. J. and W, M. R. (2008). *Successful Scientific Writing: A Step-by-Step Guide for the Biological and Medical Sciences*. Cambridge University Press, Cambridge, 3rd edition.

[Steve, 2004] Steve, M. (2004). *Code Complete: A Practical Handbook of Software Construction*. Microsoft Press, Redmond, WA, 2nd edition.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

QC381 Introduction to Quantum Computing (Elective)

1. General information

- **Semester**: 10th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Laboratory**: 4 hours;
- **Prerrequisites**:
  - QC251 Quantum Mechanics (8th Sem-Pg. 472)
  - MA381 Algebraic Structures (9th Sem-Pg. 508)

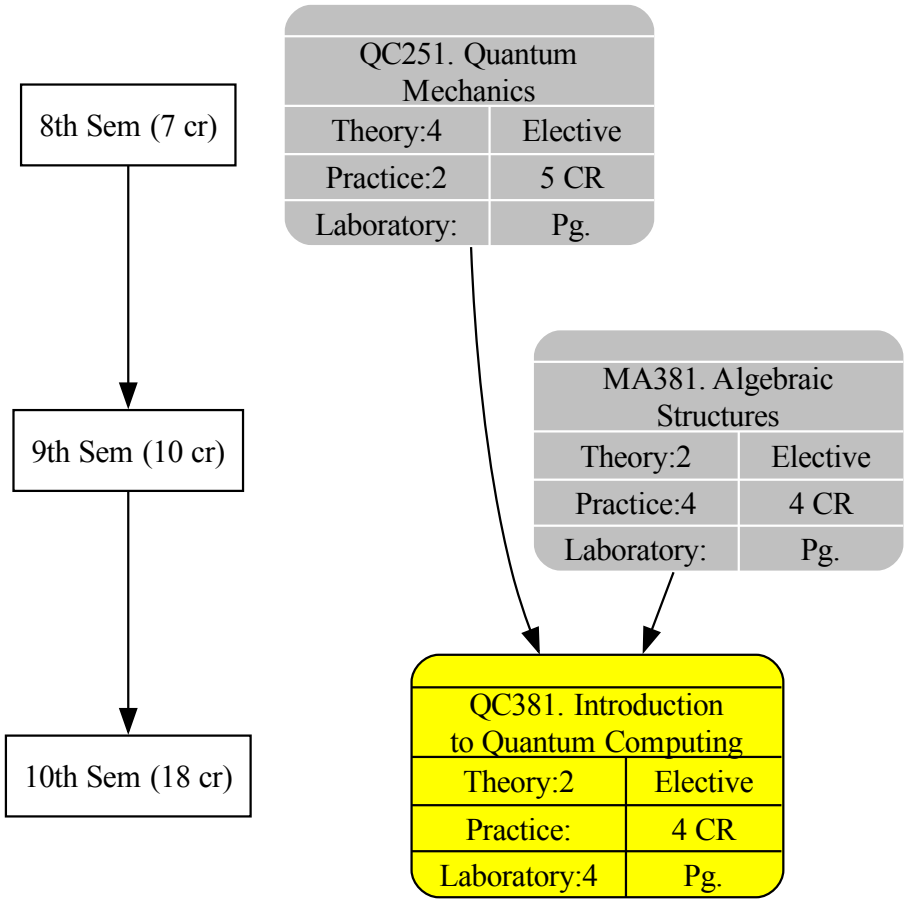


Figure 7.66: Connection Map. QC381 Introduction to Quantum Computing

2. Course Rationale

This course introduces the fundamental principles of quantum computing, including qubits, superposition, entanglement, quantum algorithms, and hardware. Students will contrast classical and quantum models, exploring applications in cryptography, optimization, physical simulation, and quantum machine learning, using frameworks like Qiskit or Cirq. Includes a practical module on hardware architectures (superconductors, ion traps) and their impact on algorithm design.

3. Goals of the Course

- Understand the postulates of quantum mechanics applied to computing.
- Implement basic quantum circuits and optimization algorithms.
- Analyze the ethical, technical, and commercial impact of quantum computing.
- Evaluate hardware limitations in real-world applications.

4. General Competences (*Outcomes*)

ABET-1) An ability to identify, formulate, and solve broadly defined technical or scientific problems by applying knowledge of mathematics and science and/or technical topics to areas relevant to the discipline. (Usage)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

5. Content

5.1 Foundational concepts, 10 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Qubits and Bloch sphere representation. **II:** Postulates of quantum mechanics. **III:** Quantum gates (Hadamard, CNOT, Toffoli).

Learning Outcomes **I:** Mathematically model a qubit [Usage]. **II:** Simulate basic operations in Qiskit [Assessment].

Bibliography: [Nielsen and Chuang, 2010, Team, 2023]

5.2 Quantum Architectures, 10 hr(s)

Competences: ABET-1,ABET-3

Topics: **I:** Principles: (a) The wave-particle duality principle (b) The uncertainty principle in the double-slit experiment (c) What is a Qubit? Superposition, interference, and measurement. Photons as qubits (d) Systems of two qubits, Entanglement, Bell states, The No-Signaling theorem **II:** Axioms of QM: superposition principle, measurement axiom, unitary evolution **III:** Single qubit gates for the circuit model of quantum computation: X, Z, H **IV:** Two qubit gates and tensor products, working with matrices **V:** The No-Cloning Theorem. The Quantum Teleportation protocol **VI:** Algorithms: (a) Simple quantum algorithms: Bernstein-Vazirani, Simon's algorithm (b) Implementing Deutsch-Josza with Mach-Zehnder Interferometers (c) Quantum factoring (Shor's Algorithm) (d) Quantum search (Grover's Algorithm) **VII:** Implementation aspects: (a) The physical implementation of qubits (b) Classical control of a Quantum Processing Unit (QPU) (c) Error mitigation and control, NISQ and beyond (d) Measurement approaches **VIII:** Emerging Applications: (a) Post-quantum encryption (b) The Quantum Internet (c) Adiabatic quantum computation (AQC) and quantum annealing

Learning Outcomes **I:** Discuss how a quantum object produced as a particle propagates like a wave and is detected as a particle with a probability distribution corresponding to the wave [Debate] **II:** Discuss the quantum-level nature that is inherently probabilistic [Debate] **III:** Express your view on entanglement that can be used to create non-classical correlations, but there is no way to use quantum entanglement to send messages faster than the speed of light [Evaluate] **IV:** Describe quantum parallelism and the role of constructive vs destructive interference in quantum algorithms given the probabilistic nature of measurement(s) [Describe] **V:** Analyze a code snippet providing the role of quantum Fourier transform (QFT) in Shor's algorithm [Analyze] **VI:** Write a program to implement Shor's algorithm in a simulator, highlighting the classical components and aspects of Shor's algorithm [Write] **VII:** Enumerate the specifics of each qubit modality (e.g., trapped ion, superconducting, silicon spin, photonic, quantum dot,

neutral atom, topological, color center, electron-on-helium) [Enumerate] **VIII:** Contrast AQC with the gate model of quantum computation and the problems each is better suited to solve [Contrast]

Bibliography: [Möttönen and Vartiainen, 2023, IBM Research, 2023]

5.3 Algorithmic Strategies, 12 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Quantum computing **II:** Deutsch-Jozsa algorithm. **III:** Grover's search (applied to combinatorial optimization). **IV:** Shor's factorization and its impact on RSA.

Learning Outcomes I: Implement Grover for SAT problems [Usage].

Bibliography: [Preskill, 2018, Team, 2023]

5.4 Formal Languages and Automata, 10 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Quantum computation: (a) Postulates of quantum mechanics: i. State space ii. State evolution iii. State composition iv. State measurement (b) Column vector representations of qubits (c) Matrix representations of quantum operations (d) Simple quantum gates (e.g., XNOT, CNOT) **II:** Quantum Approximate Optimization Algorithm (QAOA). **III:** Logistics/finance applications (Portfolio Optimization).

Learning Outcomes I: For a quantum system give examples that explain the following postulates: (a) State Space - system state represented as a unit vector in Hilbert space (b) State Evolution - the use of unitary operators to evolve system state (c) State Composition - the use of tensor product to compose systems states (d) State Measurement - the probabilistic output of measuring a system state. [Familiarity]. **II:** Design a QAOA for routing problems [Assessment].

Bibliography: [Schuld and Petruccione, 2021, Bravyi et al., 2022]

5.5 Ethics and Post-Quantum Security, 6 hr(s)

Competences: ABET-1,ABET-3

Topics: I: Post-quantum cryptography (NIST PQC Standardization). **II:** Quantum supremacy: technical/ethical debate. **III:** Patents and quantum geopolitics.

Learning Outcomes I: Propose post-quantum migration strategies [Familiarity].

Bibliography: [Team, 2019]

6. Bibliography

[Bravyi et al., 2022] Bravyi, S., Dial, O., Gambetta, J. M., Gil, D., and Nazario, Z. (2022). Quantum algorithms for fixed qubit architectures. *Nature Reviews Physics*, 4(8):499–512. Survey on quantum algorithms for optimization and simulation with near-term devices.

[IBM Research, 2023] IBM Research (2023). Ibm quantum hardware roadmap. Technical report, IBM Research. Accedido: 2026-02-13. La página original ha sido movida o actualizada.

[Möttönen and Vartiainen, 2023] Möttönen, M. and Vartiainen, J. (2023). *Quantum Computing: From Qubits to Quantum Machines*. Springer.

[Nielsen and Chuang, 2010] Nielsen, M. A. and Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press.

[Preskill, 2018] Preskill, J. (2018). Lecture notes on quantum computation. Curso Ph219/CS219, California Institute of Technology. Material disponible en línea. La URL original `http://theory.caltech.edu/~preskill/ph219/` no está operativa (febrero 2026). Se recomienda buscar la versión actual en el sitio web del autor.

[Schuld and Petruccione, 2021] Schuld, M. and Petruccione, F. (2021). *Machine Learning with Quantum Computers*. Springer, Cham, Switzerland, 2nd edition. Comprehensive introduction to quantum machine learning algorithms and implementations.

[Team, 2019] Team, G. A. Q. (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, 574.

[Team, 2023] Team, I. Q. (2023). *Qiskit Textbook*. IBM.

[Bravyi et al., 2022] Bravyi, S., Dial, O., Gambetta, J. M., Gil, D., and Nazario, Z. (2022). Quantum algorithms for fixed qubit architectures. *Nature Reviews Physics*, 4(8):499–512. Survey on quantum algorithms for optimization and simulation with near-term devices.

[IBM Research, 2023] IBM Research (2023). Ibm quantum hardware roadmap. Technical report, IBM Research. Accedido: 2026-02-13. La página original ha sido movida o actualizada.

[Möttönen and Vartiainen, 2023] Möttönen, M. and Vartiainen, J. (2023). *Quantum Computing: From Qubits to Quantum Machines*. Springer.

[Nielsen and Chuang, 2010] Nielsen, M. A. and Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press.

[Preskill, 2018] Preskill, J. (2018). Lecture notes on quantum computation. Curso Ph219/CS219, California Institute of Technology. Material disponible en línea. La URL original `http://theory.caltech.edu/~preskill/ph219/` no está operativa (febrero 2026). Se recomienda buscar la versión actual en el sitio web del autor.

[Schuld and Petruccione, 2021] Schuld, M. and Petruccione, F. (2021). *Machine Learning with Quantum Computers*. Springer, Cham, Switzerland, 2nd edition. Comprehensive introduction to quantum machine learning algorithms and implementations.

[Team, 2019] Team, G. A. Q. (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, 574.

[Team, 2023] Team, I. Q. (2023). *Qiskit Textbook*. IBM.

Universidad Nacional Mayor de San Marcos
Facultad de Ciencias Matemáticas
Escuela Profesional de Computación Científica

CS254 Advanced Data Visualization (Elective)

1. General information

- **Semester**: 10th Sem. **Credits**: 4
- **Hour of this course**: **Theory**: 2 hours; **Practice**: 4 hours;
- **Prerrequisites**:
  - CS253 Exploratory data analysis (8th Sem-Pg. 476)

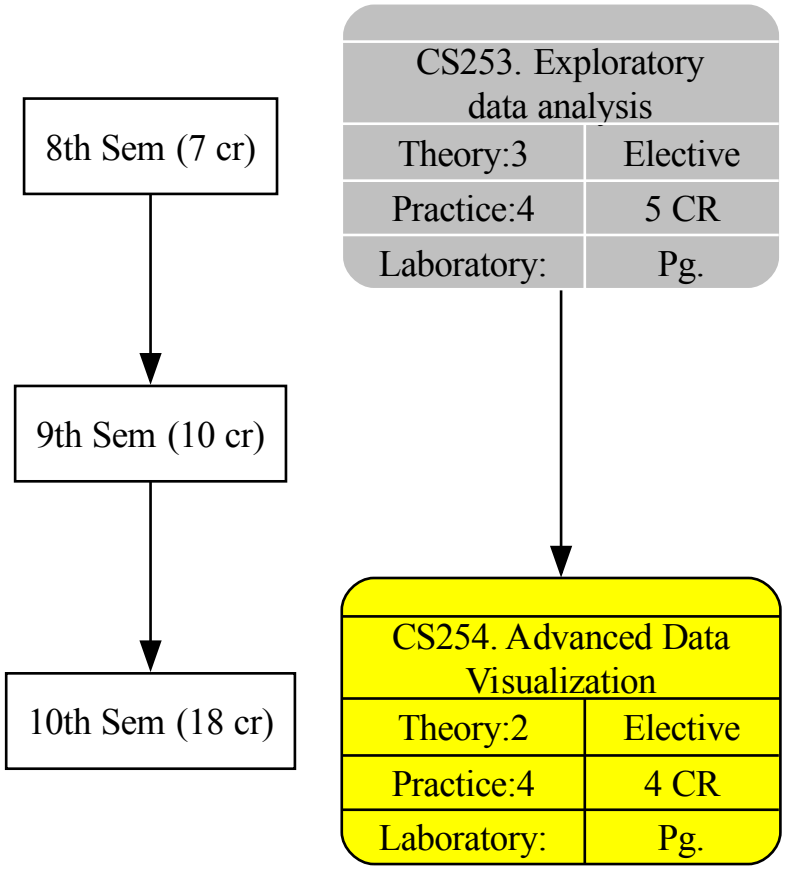


Figure 7.67: Connection Map. CS254 Advanced Data Visualization

2. Course Rationale

In the Big Data era, the ability to communicate complex information visually and effectively is fundamental. This course delves into the principles of visual perception, interactive dashboard design, and the use of advanced tools for data storytelling. It focuses on transforming abstract analysis into visual representations that facilitate strategic decision-making, integrating aesthetics, functionality, and ethics in information management.

3. Goals of the Course

- Apply principles of Gestalt psychology and visual perception to graphic design.
- Develop interactive visualizations using cutting-edge libraries.
- Design data narratives (storytelling) that facilitate the understanding of complex problems.
- Implement visualizations for multidimensional and geospatial data.
- Critically evaluate the effectiveness and ethics of visual representations.

4. General Competences (*Outcomes*)

ABET-3) An ability to develop and conduct experiments or test hypotheses, analyze and interpret data, and use scientific judgment to draw conclusions. (Usage)

ABET-2) An ability to formulate or design a system, process, procedure, or program for the intended purpose. (Usage)

5. Content

5.1 Visual Perception and Design Principles, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Color theory and luminance in visualization. **II:** Gestalt principles applied to information design. **III:** Pre-attentive attributes and cognitive load. **IV:** Visualization criticism: Clarity vs. Decoration (Chartjunk).

Learning Outcomes **I:** Explain how the human visual system processes graphic information [Familiarity]. **II:** Select color palettes and typographies that improve readability [Usage].

Bibliography: [Cairo, 2016, Ware, 2020]

5.2 Interactive and Dynamic Visualization, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Introduction to D3.js and advanced grammar of graphics. **II:** Interactions: Filtering, zoom, brushing, and linking. **III:** Animation to show temporal changes. **IV:** Responsive design for different devices.

Learning Outcomes **I:** Implement interactive elements that allow user exploration [Usage]. **II:** Build functional prototypes of dynamic visualization [Assessment].

Bibliography: [Munzner, 2014, Murray, 2017]

5.3 Spatial and Temporal Data Visualization, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: **I:** Map projections and choropleth maps. **II:** Bubble maps, density maps (heatmaps), and flows. **III:** Advanced time series: Horizon charts and streamgraphs. **IV:** Network and graph visualization.

Learning Outcomes **I:** Represent geographic data avoiding common distortions [Usage]. **II:** Analyze complex temporal patterns using specialized charts [Assessment].

Bibliography: [Wilke, 2019, Cairo, 2016]

5.4 Multivariate and High-Dimensional Data, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Parallel coordinates and radar charts. **II:** Scatterplot matrices (SPLOM). **III:** Projection techniques: t-SNE and UMAP for visualization. **IV:** Hierarchy visualization: Treemaps and sunbursts.

Learning Outcomes I: Reduce dimensionality to visualize hidden structures [Usage]. **II:** Design representations for large-scale hierarchical data [Assessment].

Bibliography: [Munzner, 2014, Wilke, 2019]

5.5 Dashboard Design and Business Intelligence, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Information architecture in control dashboards. **II:** Key Performance Indicators (KPI) and visual alerts. **III:** BI Tools: Tableau, PowerBI, and their extensibility. **IV:** Usability and user experience (UX) in visualization.

Learning Outcomes I: Integrate multiple visualizations into a coherent dashboard [Usage]. **II:** Evaluate the effectiveness of a dashboard through usability testing [Assessment].

Bibliography: [Few, 2013, Cairo, 2019]

5.6 Data Storytelling and Communication, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Narrative structures for data: Scrollytelling. **II:** Noise removal and emphasis on the main message. **III:** Annotations and visual guidance for the viewer. **IV:** The role of the analyst as a scientific communicator.

Learning Outcomes I: Create a compelling story from statistical findings [Usage]. **II:** Design visual presentations for non-technical audiences [Assessment].

Bibliography: [Knaflic, 2015, Cairo, 2019]

5.7 Ethics and Visual Deception, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Misleading charts and axis manipulation. **II:** Cognitive biases in visual interpretation. **III:** Accessibility: Design for color blindness and functional diversity. **IV:** Responsibility in representing sensitive data.

Learning Outcomes I: Identify and correct visualizations that distort reality [Assessment]. **II:** Implement accessibility standards in visual products [Usage].

Bibliography: [Cairo, 2019, Few, 2013]

5.8 Final Project: Visualization Portfolio, 8 hr(s)

Competences: ABET-2,ABET-3

Topics: I: Selection of a complex and real-world dataset. **II:** Iterative design process and feedback. **III:** Technical documentation and aesthetic justification. **IV:** Project defense before a specialized jury.

Learning Outcomes I: Build a complete professional-level visualization solution [Assessment]. **II:** Support technical and design decisions based on theory [Assessment].

Bibliography: [Knaflic, 2015, Murray, 2017]

6. Bibliography

[Cairo, 2016] Cairo, A. (2016). *The Truthful Art: Data, Charts, and Maps for Communication*. New Riders.

[Cairo, 2019] Cairo, A. (2019). *How Charts Lie: Getting Smarter about Visual Information*. W. W. Norton & Company.

[Few, 2013] Few, S. (2013). *Information Dashboard Design: Displaying Data for At-a-Glance Monitoring*. Analytics Press, 2nd edition.

[Knaflic, 2015] Knaflic, C. N. (2015). *Storytelling with Data: A Data Visualization Guide for Business Professionals*. Wiley.

[Munzner, 2014] Munzner, T. (2014). *Visualization Analysis and Design*. A K Peters/CRC Press.

[Murray, 2017] Murray, S. (2017). *Interactive Data Visualization for the Web*. O'Reilly Media, 2nd edition.

[Ware, 2020] Ware, C. (2020). *Information Visualization: Perception for Design*. Morgan Kaufmann, 4th edition.

[Wilke, 2019] Wilke, C. O. (2019). *Fundamentals of Data Visualization*. O'Reilly Media.

[Cairo, 2016] Cairo, A. (2016). *The Truthful Art: Data, Charts, and Maps for Communication*. New Riders.

[Cairo, 2019] Cairo, A. (2019). *How Charts Lie: Getting Smarter about Visual Information*. W. W. Norton & Company.

[Few, 2013] Few, S. (2013). *Information Dashboard Design: Displaying Data for At-a-Glance Monitoring*. Analytics Press, 2nd edition.

[Knaflic, 2015] Knaflic, C. N. (2015). *Storytelling with Data: A Data Visualization Guide for Business Professionals*. Wiley.

[Munzner, 2014] Munzner, T. (2014). *Visualization Analysis and Design*. A K Peters/CRC Press.

[Murray, 2017] Murray, S. (2017). *Interactive Data Visualization for the Web*. O'Reilly Media, 2nd edition.

[Ware, 2020] Ware, C. (2020). *Information Visualization: Perception for Design*. Morgan Kaufmann, 4th edition.

[Wilke, 2019] Wilke, C. O. (2019). *Fundamentals of Data Visualization*. O'Reilly Media.

Chapter 8

Professors & Courses

Taught courses by professor

There are no professors assigned in this document.

Professor assigned per course

- First Semester
  - CS111. Introduction to Programming (Pág 295)
    - There is no professor for this course
  - MA100. Basic Mathematics (Pág 300)
    - There is no professor for this course
  - FG101. Language (Pág 303)
    - There is no professor for this course
  - FG105. Gender and Society (Pág 306)
    - There is no professor for this course
  - FG109. Environment and Sustainable Development (Pág 309)
    - There is no professor for this course
  - FG112. Methods of managing university personnel (Pág 312)
    - There is no professor for this course
  - FG140. Fundamentals of Scientific Research (Pág 315)
    - There is no professor for this course
- Second Semester
  - CS100. Introduction to Computer Science (Pág 318)
    - There is no professor for this course
  - CS112. Objects-oriented programming I (Pág 328)
    - There is no professor for this course
  - AI161. Applied AI (Pág 333)
    - There is no professor for this course
  - MA111. Calculus I (Pág 338)
    - There is no professor for this course
  - FI101. General Physics (Pág 340)
    - There is no professor for this course
- Third Semester
  - CS113. Objects-oriented programming II (Pág 343)
    - There is no professor for this course
  - CS1D1. Discrete Structures I (Pág 349)
    - There is no professor for this course
  - CS2B1. Platform Based Development (Pág 352)
    - There is no professor for this course
  - MA112. Calculus II (Pág 356)

    - There is no professor for this course
  - MA121. Linear Algebra I (Pág 358)
    - There is no professor for this course
  - ID101. Technical and professional English I (Pág 361)
    - There is no professor for this course
- Fourth Semester
  - CS1D2. Discrete Structures II (Pág 365)
    - There is no professor for this course
  - CS210. Algorithms and Data Structures (Pág 367)
    - There is no professor for this course
  - CS271. Data Management (Pág 372)
    - There is no professor for this course
  - MA122. Linear Algebra II (Pág 377)
    - There is no professor for this course
  - MA210. Calculus III (Pág 380)
    - There is no professor for this course
  - ID102. Technical and professional English II (Pág 383)
    - There is no professor for this course
- Fifth Semester
  - CS251. Computer graphics (Pág 387)
    - There is no professor for this course
  - MA211. Calculus IV (Pág 390)
    - There is no professor for this course
  - MA221. Numerical Analysis I (Pág 393)
    - There is no professor for this course
  - MA222. Real Analysis I (Pág 396)
    - There is no professor for this course
  - MA251. Introduction to Statistics (Pág 399)
    - There is no professor for this course
  - MA281. Numbers theory (Pág 401)
    - There is no professor for this course
  - ID201. Technical and professional English III (Pág 405)
    - There is no professor for this course
- Sixth Semester
  - MA212. Ordinary Differential Equations (Pág 409)
    - There is no professor for this course
  - MA325. Numerical Analysis II (Pág 412)
    - There is no professor for this course
  - MA351. Statistics and Probabilities (Pág 415)
    - There is no professor for this course
  - MA371. Functional analysis I (Pág 420)
    - There is no professor for this course
  - ID202. Technical and professional English IV (Pág 423)
    - There is no professor for this course
- Seventh Semester
  - CS261. Artificial Intelligence (Pág 427)
    - There is no professor for this course
  - CS3P1. Parallel and Distributed Computing (Pág 434)
    - There is no professor for this course
  - MA213. Partial Differential Equations (Pág 443)
    - There is no professor for this course
  - MA326. Numerical Analysis III (Pág 446)
    - There is no professor for this course
  - MA361. Mathematical Modeling (Pág 449)
    - There is no professor for this course
  - ID203. Technical and professional English V (Pág 452)

    - There is no professor for this course
- Eighth Semester
  - AI263. Introduction to Machine Learning (Pág 455)
    - There is no professor for this course
  - MA401. Capstone Project I (Pág 458)
    - There is no professor for this course
  - CS2S1. Operating systems (Pág 461)
    - There is no professor for this course
  - CS3I1. Computer Security (Pág 466)
    - There is no professor for this course
  - QC251. Quantum Mechanics (Pág 472)
    - There is no professor for this course
  - CS253. Exploratory data analysis (Pág 476)
    - There is no professor for this course
  - CS370. Big Data and Analytics (Pág 480)
    - There is no professor for this course
  - STA252. Inferential Statistics (Pág 484)
    - There is no professor for this course
  - MA327. Numerical Analysis IV (Pág 487)
    - There is no professor for this course
  - MA352. Stochastic processes (Pág 490)
    - There is no professor for this course
  - MA372. Functional analysis II (Pág 493)
    - There is no professor for this course
- Ninth Semester
  - AI264. Deep Learning (Pág 496)
    - There is no professor for this course
  - MA402. Capstone Project II (Pág 499)
    - There is no professor for this course
  - CS231. Networking and Communication (Pág 503)
    - There is no professor for this course
  - MA381. Algebraic Structures (Pág 508)
    - There is no professor for this course
  - AI262. Natural Language Processing (Pág 512)
    - There is no professor for this course
  - CS273. Time series analysis (Pág 515)
    - There is no professor for this course
  - MA341. Optimization I (Pág 519)
    - There is no professor for this course
  - MA355. Statistical Learning Theory (Pág 522)
    - There is no professor for this course
- Tenth Semester
  - CS400. Pre-professional internships (Pág 525)
    - There is no professor for this course
  - AI268. Computational Vision (Pág 527)
    - There is no professor for this course
  - AI367. Topics in Artificial Intelligence (Pág 533)
    - There is no professor for this course
  - MA403. Capstone Project III (Pág 539)
    - There is no professor for this course
  - QC381. Introduction to Quantum Computing (Pág 543)
    - There is no professor for this course
  - CS254. Advanced Data Visualization (Pág 547)
    - There is no professor for this course

Chapter 9

Equivalences with Other Curriculum Plans

Below are the equivalences of this curriculum with the previous curriculum plan(s). For more details on the proposed curriculum, see section 6.2 (p. 260).

Equivalence from 2026 to 2018

Primer Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CS111	Introducción a la Programación	5	183033	Algorítmica y Fundamentos de Programación	3rd Sem	6
MA100	Matemática Básica	5	CBO105	Matemática Básica	1st Sem	4
FG101	Lenguaje	2	CBO101	Lenguaje	1st Sem	4
FG109	Ambiente y Desarrollo Sostenible	3	CBO202	Medio Ambiente y Desarrollo Sostenible	2nd Sem	4
FG112	Métodos de gestión personal universitaria	2	CBO102	Métodos de Estudio Universitario	1st Sem	3
FG140	Fundamentos de Investigación Científica	3	CBO201	Fundamentos de Investigación Científica	2nd Sem	3

Segundo Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CS100	Introducción a la Ciencia de la Computación	3			2nd Sem	
CS112	Programación Orientada a Objetos I	4			2nd Sem	
AI161	IA Aplicada	4			2nd Sem	
MA111	Cálculo I	5	CBO104	Cálculo I	1st Sem	4
FI101	Física General	5	CBO206	Física General	2nd Sem	4

Tercer Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CS113	Programación Orientada a Objetos II	3			3rd Sem	
CS1D1	Estructuras Discretas I	4	183034	Elementos de Matemática Discreta y Computacional	3rd Sem	4
CS2B1	Desarrollo Basado en Plataformas	4			3rd Sem	
MA112	Cálculo II	4	CBO204	Cálculo II	2nd Sem	4
MA121	Algebra Lineal I	4	183032	Elementos de Álgebra y Geometría	3rd Sem	6

Cuarto Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CS1D2	Estructuras Discretas II	3			4th Sem	
CS210	Algoritmos y Estructuras de Datos	4	183044	Estructura de Datos	4th Sem	6
CS271	Bases de Datos I	3	183053	Base de Datos e Ingeniería de Software	5th Sem	6
MA122	Algebra Lineal II	4	183042	Álgebra	4th Sem	6
MA210	Cálculo III	4	183031	Cálculo III	3rd Sem	6

Quinto Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CS251	Computación Gráfica	3	183072	Computación Gráfica	7th Sem	6
MA211	Cálculo IV	3	183041	Cálculo IV	4th Sem	6
MA221	Análisis Numérico I	3	183054	Métodos Numéricos y Programación I	5th Sem	5
MA222	Análisis Real I	5	183051	Análisis Real	5th Sem	6
MA251	Introducción a la Estadística	3			5th Sem	
MA281	Teoría de números	2			5th Sem	

Sexto Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
MA212	Ecuaciones Diferenciales Ordinarias	6	183052	Ecuaciones Diferenciales Ordinarias	5th Sem	6
MA325	Análisis Numérico II	4	183063	Métodos Númericos y Programación II	6th Sem	6
MA371	Análisis funcional I	5	183061	Análisis Funcional I	6th Sem	5

Séptimo Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CS261	Inteligencia Artificial	3	183074	Inteligencia Artificial	7th Sem	5
CS3P1	Computación Paralela y Distribuída	3			7th Sem	
MA213	Ecuaciones Diferenciales Parciales	4	183081	Ecuaciones Diferenciales Parciales I	8th Sem	6
MA326	Análisis Numérico III	4	183093	Métodos Númericos y Programación III	9th Sem	6
MA361	Modelado Matemático	4	183065	Técnicas de Modelamiento	6th Sem	5

Octavo Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
AI263	Introducción al Aprendizaje de Máquina	4			8th Sem	
MA401	Proyecto de final de carrera I	3	183084	Proyecto de Investigación	8th Sem	4
CS2S1	Sistemas Operativos	5	183064	Sistemas Operativos	6th Sem	5
CS3I1	Seguridad en Computación	5			8th Sem	
QC251	Mecánica Cuántica	5	183123	Mecánica Cuántica	8th Sem	5
CS253	Análisis exploratorio de datos	5			8th Sem	
CS370	Big Data y Analítica de Datos	5			8th Sem	
STA252	Estadística Inferencial	5	183062	Probabilidad e Inferencia Estadística	6th Sem	5
MA327	Análisis Numérico IV	5			8th Sem	
MA352	Procesos estocásticos	5			8th Sem	
MA372	Análisis funcional II	5	183071	Análisis Funcional II	7th Sem	5

Noveno Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
AI264	Aprendizaje Profundo	5			9^{th} Sem	
MA402	Proyecto de final de carrera II	5	183094	Desarrollo del Proyecto de Investigación	9^{th} Sem	5
CS231	Redes y Comunicación	4	183083	Redes, Arquitecturas y Comunicaciones	8^{th} Sem	6
MA381	Estructuras Algebraicas	4			9^{th} Sem	
AI262	Procesamiento de Lenguaje Natural	4			9^{th} Sem	
CS273	Análisis de series temporales	4			9^{th} Sem	
MA341	Optimización I	4	183091	Optimización	9^{th} Sem	5
MA355	Teoría de Aprendizaje Estadístico	4			9^{th} Sem	

Décimo Semestre – Plan2026			Plan 2018			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CS400	Prácticas Pre-profesionales	3	183103	Práctica Preprofesional	10^{th} Sem	4
AI268	Visión Computacional	5			10^{th} Sem	
AI367	Tópicos en Inteligencia Artificial	5			10^{th} Sem	
MA403	Proyecto de final de carrera III	5	183104	Presentación y Sustentación del Proyecto de Investigación	10^{th} Sem	6
QC381	Introduction a la Computación Cuántica	4			10^{th} Sem	
CS254	Visualización de Datos Avanzada	4			10^{th} Sem	

Equivalence from 2018 to 2026

Primer Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CBO105	Matemática Básica	4	MA100	Matemática Básica	1st Sem	5
CBO101	Lenguaje	4	FG101	Lenguaje	1st Sem	2
CBO102	Métodos de Estudio Universitario	3	FG112	Métodos de gestión personal universitaria	1st Sem	2
CBO104	Cálculo I	4	MA111	Cálculo I	2nd Sem	5
CBE004	Género y Sociedad					

Segundo Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
CBO202	Medio Ambiente y Desarrollo Sostenible	4	FG109	Ambiente y Desarrollo Sostenible	1st Sem	3
CBO201	Fundamentos de Investigación Científica	3	FG140	Fundamentos de Investigación Científica	1st Sem	3
			AI161	IA Aplicada	2nd Sem	4
CBO206	Física General	4	FI101	Física General	2nd Sem	5
CBO204	Cálculo II	4	MA112	Cálculo II	3rd Sem	4

Tercer Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
183033	Algorítmica y Fundamentos de Programación	6	CS111	Introducción a la Programación	1st Sem	5
183034	Elementos de Matemática Discreta y Computacional	4	CS1D1	Estructuras Discretas I	3rd Sem	4
			CS2B1	Desarrollo Basado en Plataformas	3rd Sem	4
183032	Elementos de Álgebra y Geometría	6	MA121	Algebra Lineal I	3rd Sem	4
183031	Cálculo III	6	MA210	Cálculo III	4th Sem	4
183035	Inglés I					

Cuarto Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
			CS1D2	Estructuras Discretas II	4th Sem	3
183044	Estructura de Datos	6	CS210	Algoritmos y Estructuras de Datos	4th Sem	4
183042	Álgebra	6	MA122	Algebra Lineal II	4th Sem	4
183041	Cálculo IV	6	MA211	Cálculo IV	5th Sem	3
183045	Inglés II					

Quinto Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
183053	Base de Datos e Ingeniería de Software	6	CS271	Bases de Datos I	4th Sem	3
183054	Métodos Numéricos y Programación I	5	MA221	Análisis Numérico I	5th Sem	3
183051	Análisis Real	6	MA222	Análisis Real I	5th Sem	5
			MA251	Introducción a la Estadística	5th Sem	3
183052	Ecuaciones Diferenciales Ordinarias	6	MA212	Ecuaciones Diferenciales Ordinarias	6th Sem	6
183055	Inglés III					

Sexto Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
183063	Métodos Númericos y Programación II	6	MA325	Análisis Numérico II	6th Sem	4
183061	Análisis Funcional I	5	MA371	Análisis funcional I	6th Sem	5
183065	Técnicas de Modelamiento	5	MA361	Modelado Matemático	7th Sem	4
183064	Sistemas Operativos	5	CS2S1	Sistemas Operativos	8th Sem	5
183062	Probabilidad e Inferencia Estadística	5	STA252	Estadística Inferencial	8th Sem	5

Séptimo Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
183072	Computación Gráfica	6	CS251	Computación Gráfica	5th Sem	3
183074	Inteligencia Artificial	5	CS261	Inteligencia Artificial	7th Sem	3
183071	Análisis Funcional II	5	MA372	Análisis funcional II	8th Sem	5

Octavo Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
183081	Ecuaciones Diferenciales Parciales I	6	MA213	Ecuaciones Diferenciales Parciales	7^{th} Sem	4
183084	Proyecto de Investigación	4	MA401	Proyecto de final de carrera I	8^{th} Sem	3
183123	Mecánica Cuántica	5	QC251	Mecánica Cuántica	8^{th} Sem	5
			MA352	Procesos estocásticos	8^{th} Sem	5
183083	Redes, Arquitecturas y Comunicaciones	6	CS231	Redes y Comunicación	9^{th} Sem	4

Noveno Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
183093	Métodos Númericos y Programación III	6	MA326	Análisis Numérico III	7^{th} Sem	4
183094	Desarrollo del Proyecto de Investigación	5	MA402	Proyecto de final de carrera II	9^{th} Sem	5
183091	Optimización	5	MA341	Optimización I	9^{th} Sem	4
			MA355	Teoría de Aprendizaje Estadístico	9^{th} Sem	4

Décimo Semestre – Plan 2018			Plan2026			
Código	**Curso**	**Cr**	**Código**	**Curso**	**Sem**	**Cr**
183103	Práctica Preprofesional	4	CS400	Prácticas Pre-profesionales	10^{th} Sem	3
183104	Presentación y Sustentación del Proyecto de Investigación	6	MA403	Proyecto de final de carrera III	10^{th} Sem	5
			CS254	Visualización de Datos Avanzada	10^{th} Sem	4

Chapter 10

Laboratories

CS111. Introduction to Programming (Mandatory) 1st Sem, Lab: 4 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.
- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

CS112. Objects-oriented programming I (Mandatory) 2nd Sem, Lab: 4 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.
- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

AI161. Applied AI (Mandatory) 2nd Sem, Lab: 4 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.
- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

CS113. Objects-oriented programming II (Mandatory) 3rd Sem, Lab: 2 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.
- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

CS1D1. Discrete Structures I (Mandatory) 3rd Sem, Lab: 2 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.
- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

CS2B1. Platform Based Development (Mandatory) 3rd Sem, Lab: 2 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

CS1D2. Discrete Structures II (Mandatory) 4th Sem, Lab: 2 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.
- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

CS210. Algorithms and Data Structures (Mandatory) 4th Sem, Lab: 2 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

CS271. Data Management (Mandatory) 4th Sem, Lab: 4 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.

  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

CS251. Computer graphics (Mandatory) 5th Sem, Lab: 4 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

CS261. Artificial Intelligence (Mandatory) 7th Sem, Lab: 2 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.
- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

CS3P1. Parallel and Distributed Computing (Mandatory) 7th Sem, Lab: 2 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

MA361. Mathematical Modeling (Mandatory) 7th Sem, Lab: 2 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

AI263. Introduction to Machine Learning (Mandatory) 8th Sem, Lab: 4 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.
- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

CS2S1. Operating systems (Elective) 8th Sem, Lab: 4 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

CS3I1. Computer Security (Elective) 8th Sem, Lab: 4 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

CS370. Big Data and Analytics (Elective) 8th Sem, Lab: 4 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

AI264. Deep Learning (Mandatory) 9th Sem, Lab: 4 hr(s)

Basic Computing Lab:

- **Hardware**: 30 workstations (Intel Core i5, 8GB RAM, 256GB SSD, integrated graphics/sound, Gigabit Ethernet).
- **OS**: Dual-boot Linux (Ubuntu LTS) / Windows 11 Pro.
- **Programming**: Python, Java, C++ (C++11/14/17), C#, Prolog, Lex/Yacc.

- **Tools**: Git, VS Code, Eclipse, Jupyter Notebook.

CS231. Networking and Communication (Elective) 9th Sem, Lab: 4 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

AI262. Natural Language Processing (Elective) 9th Sem, Lab: 4 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

MA341. Optimization I (Elective) 9th Sem, Lab: 2 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

AI268. Computational Vision (Mandatory) 10th Sem, Lab: 2 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

AI367. Topics in Artificial Intelligence (Mandatory) 10th Sem, Lab: 2 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.

QC381. Introduction to Quantum Computing (Elective) 10th Sem, Lab: 4 hr(s)

Advanced Computing Lab:

- **Hardware**: 30 high-end PCs (Intel Core i7, 16GB RAM, 512GB NVMe SSD, NVIDIA GTX 1660, Gigabit Ethernet).
- **OS**: Linux (Fedora) / Windows 11 Pro.
- **Programming**: C++20, Python (SciPy stack), Perl, R.
- **Software**:
  - *Development*: J2EE, .NET 6, MATLAB, GCC/Clang.
  - *DB/Cloud*: PostgreSQL, MongoDB, Docker.
  - *Specialized*: Rational Rose, VTK, LEDA, 8086 emulators.