National University of Engineering (UNI)
School of Artificial Intelligence
Syllabus 2027-I

1. COURSE

CS2S1. Operating systems (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS2S1. Operating systems
2.2 Semester	:	5th Semester.
2.3 Credits	:	4
2.4 Hours	:	2 HT; 4 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS221. Computer Systems Architecture. (4th Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. GOALS

- 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.

6. COMPETENCES

AG-C03) Individual and Team Work: Performs effectively as an individual and as a member or leader in diverse teams. (Usage)

AG-C09) Design and Development of Solutions: Designs, implements, and evaluates solutions for complex computing problems. (Usage)

7. TOPICS

Unit 1: Role and Purpose of Operating Systems (8 hours)	
Competences Expected: AG-C03,AG-C09	
Topics	**Learning Outcomes**

• 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.
  – Unauthorized access to files on an unencrypted drive can be achieved by moving the media to another computer.
  – Operating systems enforced security can be defeated by infiltrating the boot layer before the operating system is loaded.
  – Process isolation can be subverted by inadequate authorization checking at API boundaries.
  – Vulnerabilities in system firmware can provide attack vectors that bypass the operating system entirely.
  – Improper isolation of virtual machine memory, computing, and hardware can expose the host system to attacks from guest systems.
  – 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
	• Understand the objectives and functions of modern operating systems [Explain]
	• Evaluate the design issues in different usage scenarios (e.g., real time OS, mobile, server) [Evaluate]
	• Understand the functions of a contemporary operating system with respect to convenience, efficiency, and the ability to evolve [Explain]
	• Understand how evolution and stability are desirable and mutually antagonistic in operating systems function [Explain]

Readings : [**Silberschatz2018**], [**Tanenbaum2014**]	

Unit 2: Shell Scripting (8 hours)	
Competences Expected: AG-C03,AG-C09	
Topics	**Learning Outcomes**
• Error/exception handling • Piping • System commands: – Interface with operating systems • Environment variables • File abstraction and operators • Data structures, such as arrays and lists. • Regular expressions • Programs and processes • Workflow	• Create and execute automated scripts to manage various system tasks [Create] • Solve various text processing problems through scripting [Solve]
Readings : [**Robbins2005**], [**Silberschatz2018**]	

Unit 3: Scheduling (16 hours)	
Competences Expected: AG-C03,AG-C09	
Topics	**Learning Outcomes**
• 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 • 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.	• 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] • Explain the relationships between scheduling algorithms and application domains [Explain] • Explain the distinctions among types of processor scheduler such as short-term, medium-term, long-term, and I/O [Explain] • Evaluate a problem or solution to determine appropriateness for asymmetric and/or symmetric multiprocessing [Evaluate] • Evaluate a problem or solution to determine appropriateness as a process vs threads [Evaluate] • List some contexts benefitting from preemption and deadline scheduling [List] • 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]
Readings : [**Silberschatz2018**], [**Stallings2017**]	

Unit 4: Memory Management (16 hours)	
Competences Expected: AG-C03,AG-C09	
Topics	**Learning Outcomes**
• 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 – 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 • Virtual memory: leveraging virtual memory hardware for OS services and efficiency	• Explain memory hierarchy and cost-performance tradeoffs [Explain] • Summarize the principles of virtual memory as applied to caching and paging [Summarize] • Evaluate the tradeoffs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed [Evaluate] • Describe the reason for and use of cache memory (performance and proximity, how caches complicate isolation and virtual machine abstraction) [Describe] • 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] • Explain how hardware is utilized for efficient virtualization [Explain]
Readings : [**Silberschatz2018**], [**ArpaciDusseau2018**]	

Unit 5: File Systems API and Implementation (12 hours)	
Competences Expected: AG-C03,AG-C09	
Topics	**Learning Outcomes**
• 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	• Explain the choices to be made in designing file systems [Explain] • Evaluate different approaches to file organization, recognizing the strengths and weaknesses of each [Evaluate] • Apply software constructs appropriately given knowledge of the file system implementation [Apply]
Readings : [**Silberschatz2018**], [**Stallings2017**]	

Unit 6: Protection and Safety (12 hours)	
Competences Expected: AG-C03,AG-C09	
Topics	**Learning Outcomes**

• 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
  – Rings of protection (history from Multics to virtualized x86)
  – 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
	• Understand the requirement for protection and security mechanisms in operating systems [Explain]
	• List and describe the attack vectors that leverage OS vulnerabilities [List]
	• Understand the mechanisms available in an OS to control access to resources [Explain]
	• Summarize the features and limitations of an operating system that impact protection and security [Summarize]

Readings : [**Silberschatz2018**], [**Tanenbaum2014**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY