Universidad Nacional de Ingeniería (UNI)
Escuela Profesional de
Ciencia de la Computación
Sílabo 2026-I

1. CURSO

CS210. Algoritmos y Estructuras de Datos (Obligatorio)

2. INFORMACIÓN GENERAL

2.1 Curso	:	CS210. Algoritmos y Estructuras de Datos
2.2 Semestre	:	4^{to} Semestre.
2.3 Créditos	:	4
2.4 Horas	:	2 HT; 4 HP;
2.5 Duración del periodo	:	16 semanas
2.6 Condición	:	Obligatorio
2.7 Modalidad de aprendizaje	:	Presencial
2.8 Prerrequisitos	:	CS113. Programación Orientada a Objetos II. (3^{er} Sem)

3. PROFESORES

Atención previa coordinación con el profesor

4. INTRODUCCIÓN AL CURSO

Este curso proporciona un estudio exhaustivo de las estructuras de datos y algoritmos fundamentales con énfasis en su implementación utilizando características avanzadas de C++. Basándose en conocimientos previos de C++, los estudiantes implementarán estructuras de datos genéricas usando plantillas y rasgos, asegurando flexibilidad de tipo y rendimiento. El curso cubre estructuras lineales (vectores, varias listas enlazadas), estructuras de árbol (binario, AVL, B-trees), montículos, tablas hash, tries y estructuras de datos concurrentes. Se pone especial atención en técnicas modernas de C++ incluyendo plantillas variádicas para operaciones genéricas y programación concurrente para estructuras de datos seguras en hilos.

5. OBJETIVOS

- Implementar estructuras de datos genéricas usando plantillas y rasgos.
- Dominar estructuras de árbol avanzadas y árboles balanceados.
- Comprender e implementar estructuras de datos concurrentes.
- Aplicar plantillas variádicas para operaciones genéricas en contenedores.
- Analizar la complejidad algorítmica de todas las estructuras implementadas.
- Diseñar estructuras de datos eficientes para restricciones de problemas específicas.

6. RESULTADOS DEL ESTUDIANTE

AG-C08) Análisis de Problemas: Identifica, formula y analiza problemas complejos de computación. (Usage)

AG-C09) Diseño y Desarrollo de Soluciones: Diseña, implementa y evalúa soluciones para problemas complejos de computación. (Usage)

7. TEMAS

Unidad 1: Fundamentos avanzados para estructuras de datos (8 horas)	
Resultados esperados: AG-C08,AG-C09	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Revisión de metaprogramación con plantillas y rasgos avanzados. • Rasgos de tipos y SFINAE para compilación condicional. • Plantillas variádicas y paquetes de parámetros. • Reenvío perfecto y referencias universales. • Expresiones lambda con capturas genéricas. • Diseño basado en políticas y CRTP (Patrón de Plantilla Curiosamente Recursivo). • Restricciones basadas en conceptos (conceptos C++20 si son aplicables).	• Diseñar rasgos de tipos para personalizar el comportamiento de estructuras de datos [Usar]. • Implementar plantillas variádicas para operaciones genéricas de contenedores [Usar]. • Aplicar diseño basado en políticas para crear estructuras de datos configurables [Evaluar]. • Usar reenvío perfecto para implementar constructores eficientes [Usar].
Lecturas : [Str13], [VJG17]	

Unidad 2: Estructuras de Datos Fundamentales (8 horas)	
Resultados esperados: AG-C08,AG-C09	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Tipo de Dato Abstracto (ADT) y operaciones sobre un ADT: – Operaciones de diccionario (insertar, eliminar, encontrar) • Arreglos: – Numéricos vs no numéricos, cadenas de caracteres – Unidimensionales (vector) vs multidimensionales (matriz) • Registros/Estructuras/Tuplas y Objetos • Listas enlazadas (por razones históricas): – Simples vs Dobles y Lineales vs Circulares • Pilas • Colas y deques: – Cola de prioridad basada en montículo • Tablas/mapas hash: – Resolución de colisiones y complejidad (por ejemplo, sondeo, encadenamiento, rehash) • árboles: – Binarios, n-arios y árboles de búsqueda – Balanceados (por ejemplo, AVL, Rojo-Negro, Montículo) • Conjuntos	• Para cada ADT/Estructura de Datos en esta unidad: – Explicar su definición, propiedades, representación(es) y operaciones de ADT asociadas. – Explicar paso a paso cómo las operaciones de ADT asociadas con la estructura de datos la transforman. [Usar] • Para cada algoritmo en esta unidad explicar paso a paso cómo opera el algoritmo [Usar] • Implementar vector con política de asignación personalizable usando plantillas [Usar]. • Diseñar nodos de lista enlazada genéricos con gestión de memoria basada en rasgos [Usar]. • Comparar complejidad tiempo/espacio de diferentes estructuras lineales [Evaluar]. • Aplicar plantillas variádicas para inicializar contenedores con múltiples elementos [Usar].
Lecturas : [Cor+09], [Str13]	

Unidad 3: Estructuras de Datos Fundamentales (8 horas)	
Resultados esperados: AG-C08,AG-C09	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**

• árboles:
  – Binarios, n-arios y árboles de búsqueda
  – Balanceados (por ejemplo, AVL, Rojo-Negro, Montículo)
• Implementación de plantilla de árbol binario con soporte de iteradores.
• Árbol de búsqueda binaria con operadores de comparación basados en rasgos.
• Algoritmos de recorrido de árbol (iterativo y recursivo).
• Árboles de expresión y sus aplicaciones.
• Estructuras de nodo basadas en plantilla con carga útil configurable.
	• Dados los requisitos para un problema, desarrollar múltiples soluciones usando varias estructuras de datos y algoritmos. Posteriormente, evaluar la idoneidad, fortalezas y debilidades seleccionando un enfoque que satisfaga mejor los requisitos [Evaluar]
	• Explicar factores más allá de la eficiencia computacional que influyen en la elección de algoritmos, como el tiempo de programación, la mantenibilidad y el uso de patrones específicos de la aplicación en los datos de entrada [Familiarizarse]
	• Implementar árbol binario genérico con orden de recorrido configurable [Usar].
	• Diseñar evaluador de árbol de expresión usando patrón visitante [Usar].
	• Analizar complejidades de operaciones de árbol para diferentes estrategias de balanceo [Evaluar].

Lecturas : [Cor+09], [Knu97]	

Unidad 4: Balanced and Multi-way Trees (8 horas)	
Resultados esperados: AG-C08,AG-C09	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Implementación de árbol AVL con rasgo de balanceo basado en plantillas. • Propiedades y operaciones de árbol rojo-negro. • Implementación de plantilla de B-Tree para almacenamiento basado en disco. • Estructuras de árbol 2-3 y árbol 2-3-4. • Implementación de trie digital (árbol de prefijos). • Árboles de sufijos y sus aplicaciones en procesamiento de cadenas. • Estrategias de división/fusión de nodos para B-trees.	• Implementar árboles auto-balanceados usando diseño basado en políticas [Usar]. • Diseñar B-tree para aplicaciones de índice de base de datos [Usar]. • Evaluar compensaciones entre diferentes estructuras de árbol balanceado [Evaluar]. • Implementar trie para coincidencia eficiente de prefijos de cadenas [Usar].
Lecturas : [Cor+09], [Knu97], [SW11]	

Unidad 5: Estructuras de Datos Fundamentales (8 horas)	
Resultados esperados: AG-C08,AG-C09	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• árboles: – Binarios, n-arios y árboles de búsqueda – Balanceados (por ejemplo, AVL, Rojo-Negro, Montículo) • Implementación de montículo binario usando representación de arreglo. • Variantes de montículo mínimo y montículo máximo. • Algoritmos heapify y análisis de complejidad. • Implementación de ADT de cola de prioridad. • Visión general de montículo binomial y montículo de Fibonacci. • Aplicaciones de montículos: algoritmo de Dijkstra, ordenamiento por montículo.	• Explicar la propiedad de montículo y el uso de montículos como una implementación de una cola de prioridad [Familiarizarse] • Implementar montículo genérico con rasgo de comparación configurable [Usar]. • Diseñar cola de prioridad que soporte múltiples estrategias de actualización de prioridad [Usar]. • Analizar complejidad de operaciones de montículo para diferentes tipos de montículo [Evaluar]. • Aplicar montículo en implementaciones de algoritmos de grafos [Usar].
Lecturas : [Cor+09], [SW11]	

Unidad 6: Estructuras de Datos Fundamentales (8 horas)	
Resultados esperados: AG-C08,AG-C09	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Tablas/mapas hash: – Resolución de colisiones y complejidad (por ejemplo, sondeo, encadenamiento, rehash) • Diseño de función hash y propiedades. • Implementación de encadenamiento separado con listas enlazadas. • Direccionamiento abierto: sondeo lineal, sondeo cuadrático, doble hashing. • Hashing perfecto y hashing perfecto mínimo. • Hashing cuco y hashing de rayuela. • Estrategias de redimensionamiento de tabla hash y gestión de factor de carga.	• Explicar cómo se maneja la evitación de colisiones y la resolución de colisiones en tablas hash [Familiarizarse] • Diseñar tabla hash genérica con resolución de colisiones configurable [Usar]. • Implementar funciones hash personalizadas para tipos definidos por el usuario [Usar]. • Evaluar rendimiento de diferentes esquemas de hashing [Evaluar]. • Aplicar tablas hash en la implementación de tabla de símbolos del compilador [Usar].
Lecturas : [Cor+09], [Knu97]	

Unidad 7: Concurrent Data Structures (8 horas)	
Resultados esperados: AG-C08,AG-C09	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Principios de diseño de estructuras de datos seguras para hilos. • Listas enlazadas concurrentes basadas en bloqueos. • Algoritmos sin bloqueo y libres de espera. • Tablas hash concurrentes con bloqueo de grano fino. • Operaciones atómicas y ordenamiento de memoria. • Colas concurrentes (limitadas e ilimitadas). • Pilas concurrentes y técnicas de eliminación.	• Diseñar estructuras de datos seguras para hilos usando mutex y guardas de bloqueo [Usar]. • Implementar lista enlazada sin bloqueo usando operaciones atómicas [Usar]. • Analizar rendimiento de estructuras de datos concurrentes vs secuenciales [Evaluar]. • Aplicar colas concurrentes en patrones productor-consumidor [Usar].
Lecturas : [HS12], [Wil19]	

Unidad 8: Specialized Structures and Applications (8 horas)	
Resultados esperados: AG-C08,AG-C09	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Estructuras de datos persistentes. • Unión-búsqueda de conjuntos disjuntos (union-find) con compresión de trayectoria. • Filtros de Bloom y sus propiedades probabilísticas. • Listas por niveles para búsqueda balanceada probabilística. • Árboles de segmentos y árboles de Fenwick para consultas de rango. • Estructuras de datos espaciales: árboles k-d, quadtrees. • Estructuras de datos inconscientes de la caché.	• Implementar estructuras de datos persistentes con compartición estructural [Usar]. • Diseñar union-find con unión por rango y compresión de trayectoria [Usar]. • Aplicar estructuras especializadas para resolver problemas específicos de dominio [Evaluar]. • Evaluar compensaciones espacio-tiempo en estructuras de datos probabilísticas [Evaluar].
Lecturas : [Oka99], [SW11]	

8. PLAN DE TRABAJO

8.1 Metodología

Se fomenta la participación individual y en equipo para exponer sus ideas, motivándolos con puntos adicionales en las diferentes etapas de la evaluación del curso.

8.2 Sesiones Teóricas

Las sesiones de teoría se llevan a cabo en clases magistrales donde se realizarán actividades que propicien un aprendizaje activo, con dinámicas que permitan a los estudiantes interiorizar los conceptos.

8.3 Sesiones Prácticas

Las sesiones prácticas se llevan en clase donde se desarrollan una serie de ejercicios y/o conceptos prácticos mediante planteamiento de problemas, la resolución de problemas, ejercicios puntuales y/o en contextos aplicativos.

9. SISTEMA DE EVALUACIÓN

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

10. BIBLIOGRAFÍA BÁSICA

[Knu97] Donald E. Knuth. *The Art of Computer Programming, Volume 1: Fundamental Algorithms*. 3rd. Addison-Wesley, 1997.

[Oka99] Chris Okasaki. *Purely Functional Data Structures*. Cambridge University Press, 1999.

[Cor+09] Thomas H. Cormen et al. *Introduction to Algorithms*. 3rd. MIT Press, 2009.

[SW11] Robert Sedgewick and Kevin Wayne. *Algorithms*. 4th. Addison-Wesley, 2011.

[HS12] Maurice Herlihy and Nir Shavit. *The Art of Multiprocessor Programming*. Revised 1st. Morgan Kaufmann, 2012.

[Str13] Bjarne Stroustrup. *The C++ Programming Language*. 4th. Addison-Wesley, 2013.

[VJG17] David Vandevoorde, Nicolai M. Josuttis, and Douglas Gregor. *C++ Templates: The Complete Guide*. 2nd. Addison-Wesley, 2017.

[Wil19] Anthony Williams. *C++ Concurrency in Action*. 2nd. Manning Publications, 2019.