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

1. CURSO

CS311. Programación Competitiva (Obligatorio)

2. INFORMACIÓN GENERAL

2.1 Curso	:	CS311. Programación Competitiva
2.2 Semestre	:	6^{to} Semestre.
2.3 Créditos	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duración del periodo	:	16 semanas
2.6 Condición	:	Obligatorio
2.7 Modalidad de aprendizaje	:	Presencial
2.8 Prerrequisitos	:	CS212. Análisis y Diseño de Algoritmos. (5^{to} Sem)

3. PROFESORES

Atención previa coordinación con el profesor

4. INTRODUCCIÓN AL CURSO

La Programación Competitiva entrena a los estudiantes para resolver problemas algorítmicos complejos bajo restricciones de tiempo y recursos. Este curso cierra la brecha entre el diseño algorítmico teórico y la implementación de alto rendimiento. Cubre técnicas avanzadas en programación dinámica, teoría de grafos y matemáticas, fomentando el pensamiento analítico requerido para entrevistas de ingeniería de software de primer nivel y concursos de programación internacionales como el ICPC.

5. OBJETIVOS

- Dominar técnicas y paradigmas avanzados de programación competitiva.
- Desarrollar la capacidad de analizar rápidamente la complejidad de soluciones potenciales.
- Implementar estructuras de datos eficientes para consultas de rango y actualizaciones dinámicas.
- Aplicar algoritmos matemáticos y geométricos especializados para resolver problemas de concursos.
- Optimizar código para velocidad de ejecución y uso de memoria en un entorno competitivo.

6. RESULTADOS DEL ESTUDIANTE

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

AG-C12) Usa enfoques sistémicos para seleccionar, desarrollar, aplica, integrar y administrar tecnologías seguras de computación para lograr los objetivos del usuario. (Usage)

7. TEMAS

Unidad 1: Estrategias Algorítmicas (12 horas)	
Resultados esperados: AG-C08,AG-C12	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Búsqueda Completa: Retroceso Recursivo y Máscaras de bits. • Estrategias Voraces: Propiedad de elección y subestructura óptima. • Divide y Vencerás: Búsqueda Binaria sobre la respuesta y Búsqueda Ternaria. • Programación Dinámica: Reducción de estado, DP de dígitos y DP en árboles.	• Aplicar técnicas de poda de búsqueda para resolver problemas NP-difíciles con restricciones pequeñas [Usar] • Identificar el paradigma más eficiente para un enunciado de problema dado [Evaluar]
Lecturas : [HH20], [Laa20]	

Unidad 2: Algoritmos Fundamentales (12 horas)	
Resultados esperados: AG-C08,AG-C12	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Árboles de Fenwick (Árboles Binarios Indexados). • Árboles de Segmentos: Propagación Diferida y Árboles de Segmentos Persistentes. • Descomposición de Raíz Cuadrada y Algoritmo de Mo. • Tablas Dispersas para RMQ.	• Implementar estructuras de consulta de rango para actualizaciones de datos en tiempo real [Evaluar] • Analizar las compensaciones entre diferentes estructuras de datos de consulta de rango [Usar]
Lecturas : [HH20], [Ski20]	

Unidad 3: Algoritmos Fundamentales (12 horas)	
Resultados esperados: AG-C08,AG-C12	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Flujo en Redes: Flujo Máximo Corte Mínimo (Dinic y Edmonds-Karp). • Componentes Fuertemente Conexas (Tarjan y Kosaraju). • Emparejamiento Bipartito y Descomposición Pesado-Ligero. • Descomposición por Centroide en árboles.	• Para cada algoritmo en esta unidad explicar paso a paso cómo opera el algoritmo [Explicar] • Para cada enfoque algorítmico (por ejemplo, ordenamiento) en esta unidad aplicar un ejemplo prototípico del enfoque (por ejemplo, ordenamiento por mezcla) [Aplicar] • 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 [Crear] • 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 [Explicar] • Para cada uno de los algoritmos y enfoques algorítmicos en los temas del Núcleo KA: – Explicar un ejemplo prototípico del algoritmo, y – Explicar paso a paso cómo opera el algoritmo. [Explicar]
Lecturas : [Laa20], [HH20]	

Unidad 4: Algoritmos Fundamentales (12 horas)	
Resultados esperados: AG-C08,AG-C12	
Temas	**Aprendizaje esperado (*Learning Outcomes*)**
• Teoría de Números: Criba de Eratóstenes, Euclides Extendido e Inverso Modular. • Combinatoria: Inclusión-Exclusión y Lema de Burnside. • Geometría Computacional: Producto Cruz, Envolvente Convexa (Cadena Monótona) e Intersección de líneas. • Transformada Rápida de Fourier (FFT) para multiplicación de polinomios.	• Resolver problemas combinatorios usando aritmética modular [Usar] • Implementar primitivas geométricas para análisis de relaciones espaciales [Evaluar]
Lecturas : [Ski20], [Laa20]	

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

[HH20] Steven Halim and Felix Halim. *Competitive Programming 4: The Lower Bound of Programming Contests*. Lulu, 2020.

[Laa20] Antti Laaksonen. *Competitive Programmer's Handbook*. Draft, 2020.

[Ski20] Steven S. Skiena. *The Algorithm Design Manual*. 3rd. Springer, 2020.