

## 1. CURSO

CS311. Programación Competitiva (Obligatorio)

## 2. INFORMACIÓN GENERAL

2.1 Créditos	:	4
2.2 Horas de teoría	:	2 (Semanal)
2.3 Horas de práctica	:	2 (Semanal)
2.4 Duración del periodo	:	16 semanas
2.5 Condición	:	Obligatorio
2.6 Modalidad	:	Presencial
2.7 Prerrequisitos	:	CS212. Análisis y Diseño de Algoritmos. (5 <sup>to</sup> Sem)

## 3. PROFESORES

Atención previa coordinación con el profesor

## 4. INTRODUCCIÓN AL CURSO

La Programación Competitiva combina retos de solucionar problemas con el añadido de poder competir con otras personas. Enseña a los participantes a pensar más rápido y desarrollar habilidades para resolver problemas, que son de gran demanda en la industria. Este curso enseñará la resolución de problemas algorítmicos de manera rápida combinando la teoría de algoritmos y estructuras de datos con la práctica la solución de los problemas.

## 5. OBJETIVOS

- Que el alumno utilice técnicas de estructuras de datos y algoritmos complejos.
- Que el alumno aplique los conceptos aprendidos para la aplicación sobre un problema real.
- Que el alumno investigue la posibilidad de crear un nuevo algoritmo y/o técnica nueva para resolver un problema real.

## 6. COMPETENCIAS

- a) Aplicar conocimientos de computación y de matemáticas apropiadas para la disciplina. (**Usar**)
- b) Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución. (**Usar**)

## 7. COMPETENCIAS ESPECÍFICAS

- a4) Aplicar técnicas eficientes de resolución de problemas computacionales. ( )
- b1) Identificar y aplicar de forma eficiente diversas estrategias algorítmicas y estructuras de datos para la solución de un problema dadas ciertas restricciones de espacio y tiempo. ( )

## 8. TEMAS

<b>Unidad 1: Introducción (20)</b>	
<b>Competencias esperadas: a,b,h</b>	
<b>Temas</b>	<b>Objetivos de Aprendizaje</b>
<ul style="list-style-type: none"> <li>• Introducción a la Programación competitiva</li> <li>• Modelo computacional</li> <li>• Complejidad algorítmica</li> <li>• Problemas sobre búsqueda y ordenamiento</li> <li>• Recursión y recurrencia</li> <li>• Estrategia divide y conquista</li> </ul>	<ul style="list-style-type: none"> <li>• Reconocer y sabes como usar los recursos del modelo de computación RAM (Random Access Machine). [Usar]</li> <li>• Determinar el tiempo y espacio de complejidad de algoritmos. [Usar]</li> <li>• Determinar relaciones de recurrencia para algoritmos recursivos.[Usar]</li> <li>• Resolver problemas de búsqueda y ordenamiento.[Usar]</li> <li>• Aprender a seleccionar los algoritmos adecuados para problemas de tipo divide y conquista.[Usar]</li> <li>• Diseñar nuevos algoritmos para la resolución de problemas.[Usar]</li> </ul>
<b>Lecturas :</b> [Cor+09], [Hal13], [Kul19], [Mig03], [Laa17], [ALP12]	

<b>Unidad 2: Estructuras de datos (20)</b>	
<b>Competencias esperadas: a,b,h</b>	
<b>Temas</b>	<b>Objetivos de Aprendizaje</b>
<ul style="list-style-type: none"> <li>• Problemas sobre arrays y strings</li> <li>• Problemas sobre listas enlazadas</li> <li>• Problemas sobre pilas, colas</li> <li>• Problemas sobre arboles</li> <li>• Problemas sobre Hash tables</li> <li>• Problemas sobre Heaps</li> </ul>	<ul style="list-style-type: none"> <li>• Reconocer las distintas estructuras de datos sus complejidades usos y restricciones. [Usar]</li> <li>• Identificar el tipo de estructura de datos adecuado a la resolución del problema. [Usar]</li> <li>• Reconocer tipos de problemas asociado a operaciones sobre estructuras de datos como búsqueda, inserción, eliminación y actualización.[Usar]</li> </ul>
<b>Lecturas :</b> [Cor+09], [Hal13], [Kul19], [Mig03], [Laa17], [ALP12]	

<b>Unidad 3: Paradigmas de diseño (20)</b>	
<b>Competencias esperadas: a,b,h</b>	
<b>Temas</b>	<b>Objetivos de Aprendizaje</b>
<ul style="list-style-type: none"> <li>• Fuerza bruta</li> <li>• Divide y conquista</li> <li>• Backtracking</li> <li>• Greedy</li> <li>• Programación Dinamica</li> </ul>	<ul style="list-style-type: none"> <li>• Aprender los distintos paradigmas de resolución de problemas.[Usar]</li> <li>• Aprender a seleccionar los algoritmos adecuados para distintos problemas según el tipo de paradigma.[Usar]</li> </ul>
<b>Lecturas :</b> [Cor+09], [Hal13], [Kul19], [Mig03], [Laa17], [ALP12]	

<b>Unidad 4: Gráfos (20)</b>	
<b>Competencias esperadas: a,b,h</b>	
<b>Temas</b>	<b>Objetivos de Aprendizaje</b>
<ul style="list-style-type: none"> <li>• Recorrido de gráfos</li> <li>• Aplicaciones y problemas sobre gráfos</li> <li>• Camino mas corto</li> <li>• Redes y flujos</li> </ul>	<ul style="list-style-type: none"> <li>• Identificar problemas clasificados como problemas de grafos. [Usar]</li> <li>• Aprender a seleccionar los algoritmos adecuados para problemas de grafos (recorrido, MST, camino mas costo, redes y flujos) y conocer sus soluciones eficientes. [Usar]</li> </ul>
<b>Lecturas :</b> [Cor+09], [Hal13], [Kul19], [Mig03], [Laa17], [ALP12]	

<b>Unidad 5: Tópicos avanzados (20)</b>	
<b>Competencias esperadas: a,b,h</b>	
<b>Temas</b>	<b>Objetivos de Aprendizaje</b>
<ul style="list-style-type: none"> <li>• Teoria de números</li> <li>• Probabilidad y combinaciones</li> <li>• Algoritmos para manejos de strings (tries, string hashing, z-algorithm)</li> <li>• Geometria y sweep line algorithms, segment trees</li> </ul>	<ul style="list-style-type: none"> <li>• Aprender a elegir los algoritmos adecuados para problemas sobre teoria de números y matemáticas ya que son importantes en programación competitiva. [Usar]</li> <li>• Aprender a seleccionar los algoritmos adecuados para problemas sobre probabilidades y combinaciones, manejos de strings y geometría computacional. [Usar]</li> </ul>
<b>Lecturas :</b> [Cor+09], [Hal13], [Kul19], [Mig03], [Laa17], [ALP12]	

<b>Unidad 6: Problemas de dominio especifico (20)</b>	
<b>Competencias esperadas: a,b,h</b>	
<b>Temas</b>	<b>Objetivos de Aprendizaje</b>
<ul style="list-style-type: none"> <li>• Latencia y rendimiento</li> <li>• Paralelismo</li> <li>• Redes</li> <li>• Almacenamiento</li> <li>• Alta disponibilidad</li> <li>• Caching</li> <li>• Proxies</li> <li>• Equilibradores de carga</li> <li>• Almacenamiento clave-valor</li> <li>• Replicar y compartir</li> <li>• Elección del líder</li> <li>• Limitación de la tasa</li> <li>• Registro y monitoreo</li> </ul>	<ul style="list-style-type: none"> <li>• Aprender a diseñar sistemas para diferentes problemas de dominio especifico aplicando conocimiento sobre redes, computación distribuida, alta disponibilidad, almacenamiento y arquitectura de sistemas. [Usar]</li> </ul>
<b>Lecturas :</b> [Cor+09], [Hal13], [Kul19], [Mig03], [Laa17], [ALP12]	

## 9. PLAN DE TRABAJO

### 9.1 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.

### 9.2 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.

### 9.3 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.

## 10. SISTEMA DE EVALUACIÓN

\*\*\*\*\* EVALUATION MISSING \*\*\*\*\*

## 11. BIBLIOGRAFÍA BÁSICA

- [ALP12] A. Aziz, T.H. Lee, and A. Prakash. *Elements of Programming Interviews: The Insiders' Guide*. ElementsOfProgrammingInterviews.com, 2012. ISBN: 9781479274833. URL: <https://books.google.com.pe/books?id=y6FLBQAAQBAJ>.
- [Cor+09] T. H. Cormen et al. *Introduction to Algorithms*. MIT Press, 2009.
- [Hal13] Steven Halim. *Competitive Programming*. 3 rd. Lulu, 2013.
- [Kul19] Alexander S. Kulikov. *Learning Algorithms Through Programming and Puzzle Solving*. Active Learning Technologies, 2019.
- [Laa17] Antti Laaksonen. *Guide to Competitive Programming: Learning and Improving Algorithms Through Contests*. Springer, 2017.
- [Mig03] Steve Skiena Miguel A. Revilla. *Programming Challenges: The Programming Contest Training Manual*. Springer, May 2003. ISBN: 978-0387001630.