



## Universidad Nacional de Ucayali (UNU)

Programa Profesional de  
Ciencia de la Computación  
Sílabo 2023-I

### 1. CURSO

CS212. Análisis y Diseño de Algoritmos (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            | : | Híbrido  |
| 2.7 Prerrequisitos       | : | <ul style="list-style-type: none"><li>• CS210. Algoritmos y Estructuras de Datos. (4<sup>to</sup> Sem)</li><li>• CS211. Teoría de la Computación. (4<sup>to</sup> Sem)</li></ul> |

### 3. PROFESORES

Atención previa coordinación con el profesor

### 4. INTRODUCCIÓN AL CURSO

Un algoritmo es, esencialmente, un conjunto bien definido de reglas o instrucciones que permitan resolver un problema computacional. El estudio teórico del desempeño de los algoritmos y los recursos utilizados por estos, generalmente tiempo y espacio, nos permite evaluar si un algoritmo es adecuado para un resolver un problema específico, compararlo con otros algoritmos para el mismo problema o incluso delimitar la frontera entre lo viable y lo imposible.

Esta materia es tan importante que incluso Donald E. Knuth definió a Ciencia de la Computación como el estudio de algoritmos.

En este curso serán presentadas las técnicas más comunes utilizadas en el análisis y diseño de algoritmos eficientes, con el propósito de aprender los principios fundamentales del diseño, implementación y análisis de algoritmos para la solución de problemas computacionales.

### 5. OBJETIVOS

- Desarrollar la capacidad para evaluar la complejidad y calidad de algoritmos propuestos para un determinado problema.
- Estudiar los algoritmos más representativos, introductorios de las clases más importantes de problemas tratados en computación.
- Desarrollar la capacidad de resolución de problemas algorítmicos utilizando los principios fundamentales de diseño de algoritmos aprendidos.
- Ser capaz de responder a las siguientes preguntas cuando le sea presentado un nuevo algoritmo: ¿Cuán buen desempeño tiene?, ¿Existe una mejor forma de resolver el problema?

### 6. COMPETENCIAS

- a) Aplicar conocimientos de computación y de matemáticas apropiadas para la disciplina. (**Evaluar**)
- b) Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución. (**Evaluar**)
- h) Incorporarse a un proceso de aprendizaje profesional continuo. (**Usar**)
- i) Utilizar técnicas y herramientas actuales necesarias para la práctica de la computación. (**Usar**)

## 7. TEMAS

Unidad 1: Análisis Básico (10)	
Competencias esperadas:	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Diferencias entre el mejor, el esperado y el peor caso de un algoritmo.</li> <li>• Análisis asintótico de complejidad de cotas superior y esperada.</li> <li>• Definición formal de la Notación Big O.</li> <li>• Clases de complejidad como constante, logarítmica, lineal, cuadrática y exponencial.</li> <li>• Uso de la notación Big O.</li> <li>• Relaciones recurrentes.</li> <li>• Análisis de algoritmos iterativos y recursivos.</li> <li>• Teorema Maestro y Árboles Recursivos.</li> </ul>	<ul style="list-style-type: none"> <li>• Explique a que se refiere con “mejor”, “esperado” y “peor” caso de comportamiento de un algoritmo [Evaluar]</li> <li>• En el contexto de a algoritmos específicos, identifique las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos [Evaluar]</li> <li>• Determine informalmente el tiempo y el espacio de complejidad de diferentes algoritmos [Evaluar]</li> <li>• Indique la definición formal de Big O [Evaluar]</li> <li>• Lista y contraste de clases estándares de complejidad [Evaluar]</li> <li>• Use la notación formal de la Big O para dar límites superiores asintóticos en la complejidad de tiempo y espacio de los algoritmos [Evaluar]</li> <li>• Usar la notación formal Big O para dar límites de casos esperados en el tiempo de complejidad de los algoritmos [Evaluar]</li> <li>• Explicar el uso de la notación theta grande, omega grande y o pequeña para describir la cantidad de trabajo hecho por un algoritmo [Evaluar]</li> <li>• Usar relaciones recurrentes para determinar el tiempo de complejidad de algoritmos recursivamente definidos [Evaluar]</li> <li>• Resuelve relaciones de recurrencia básicas, por ejemplo. usando alguna forma del Teorema Maestro [Evaluar]</li> </ul>
<b>Lecturas :</b> [KT05], [DPV06], [RS09], [SF13], [Knu97]	

Unidad 2: Estrategias Algorítmicas (30)	
Competencias esperadas:	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Algoritmos de fuerza bruta.</li> <li>• Algoritmos voraces.</li> <li>• Divide y vencerás.</li> <li>• Programación Dinámica.</li> </ul>	<ul style="list-style-type: none"> <li>• Para cada una de las estrategias (fuerza bruta, algoritmo goloso, divide y vencerás, recursividad en reversa y programación dinámica), identifica un ejemplo práctico en el cual se pueda aplicar [Evaluar]</li> <li>• Utiliza un enfoque voraz para resolver un problema específico y determina si la regla escogida lo guía a una solución óptima [Evaluar]</li> <li>• Usa un algoritmo de divide-y-vencerás para resolver un determinado problema [Evaluar]</li> <li>• Usa programación dinámica para resolver un problema determinado [Evaluar]</li> <li>• Determina el enfoque algorítmico adecuado para un problema [Evaluar]</li> </ul>
<b>Lecturas :</b> [KT05], [DPV06], [RS09], [Als99]	

Unidad 3: Algoritmos y Estructuras de Datos fundamentales (10)	
Competencias esperadas:	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo.</li> <li>• Algoritmos de búsqueda secuencial y binaria.</li> <li>• Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción)</li> <li>• Algoritmos de ordenamiento con peor caso o caso promedio en <math>O(N \lg N)</math> (Quicksort, Heapsort, Mergesort)</li> <li>• Grafos y algoritmos en grafos: <ul style="list-style-type: none"> <li>– Representación de grafos (ej., lista de adyacencia, matriz de adyacencia)</li> <li>– Recorrido en profundidad y amplitud</li> </ul> </li> <li>• Montículos (Heaps)</li> <li>• Grafos y algoritmos en grafos: <ul style="list-style-type: none"> <li>– Problema de corte máximo y mínimo</li> <li>– Búsqueda local</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• Implementar algoritmos numéricos básicos [Evaluar]</li> <li>• Implementar algoritmos de búsqueda simple y explicar las diferencias en sus tiempos de complejidad [Evaluar]</li> <li>• Ser capaz de implementar algoritmos de ordenamiento comunes cuadráticos y <math>O(N \log N)</math> [Evaluar]</li> <li>• Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, búsqueda y hashing [Usar]</li> <li>• Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Familiarizarse]</li> <li>• Resolver problemas usando algoritmos básicos de grafos, incluyendo búsqueda por profundidad y búsqueda por amplitud [Evaluar]</li> <li>• Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección, y para implementar el algoritmo en un contexto específico [Evaluar]</li> <li>• Describir la propiedad del heap y el uso de heaps como una implementación de colas de prioridad [Evaluar]</li> <li>• Resolver problemas usando algoritmos de grafos, incluyendo camino más corto de una sola fuente y camino más corto de todos los pares, y como mínimo un algoritmo de árbol de expansión mínima [Evaluar]</li> </ul>
<b>Lecturas :</b> [KT05], [DPV06], [RS09], [SW11], [GT09]	

Unidad 4: Computabilidad y complejidad básica de autómatas (2)	
Competencias esperadas:	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Introducción a las clases P y NP y al problema P vs. NP.</li> <li>• Introducción y ejemplos de problemas NP- Completos y a clases NP-Completos.</li> </ul>	<ul style="list-style-type: none"> <li>• Define las clases P y NP [Familiarizarse]</li> <li>• Explique el significado de NP-Complejidad [Familiarizarse]</li> </ul>
<b>Lecturas :</b> [KT05], [DPV06], [RS09]	

Unidad 5: Estructuras de Datos Avanzadas y Análisis de Algoritmos (8)	
Competencias esperadas:	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Grafos (ej. Ordenamiento Topológico, encontrando componentes puertemente conectados)</li> <li>• Algoritmos Teórico-Numéricos (Aritmética Modular, Prueba del Número Primo, Factorización Entera)</li> <li>• Algoritmos aleatorios.</li> <li>• Análisis amortizado.</li> <li>• Análisis Probabilístico.</li> </ul>	<ul style="list-style-type: none"> <li>• Entender el mapeamiento de problemas del mundo real a soluciones algorítmicas (ejemplo, problemas de grafos, programas lineares,etc) [Familiarizarse]</li> <li>• Seleccionar y aplicar técnicas de algoritmos avanzadas (ejemplo, randomización, aproximación) para resolver problemas reales [Usar]</li> <li>• Seleccionar y aplicar técnicas avanzadas de análisis (ejemplo, amortizado, probabilístico,etc) para algoritmos [Usar]</li> </ul>
<b>Lecturas :</b> [KT05], [DPV06], [RS09], [Tar83], [Raw92]	

## 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

- [Als99] H. Alsuwaiyel. *Algorithms: Design Techniques and Analysis*. World Scientific, 1999. ISBN: 9789810237400.
- [DPV06] S. Dasgupta, C. Papadimitriou, and U. Vazirani. *Algorithms*. McGraw-Hill Education, 2006. ISBN: 9780073523408.
- [GT09] Michael T. Goodrich and Roberto Tamassia. *Algorithm Design: Foundations, Analysis and Internet Examples*. 2nd. John Wiley & Sons, Inc., 2009. ISBN: 0470088540, 9780470088548.
- [Knu97] D.E. Knuth. *The Art of Computer Programming: Fundamental algorithms Vol 1*. Third Edition. Addison-Wesley, 1997. ISBN: 9780201896831. URL: <http://www-cs-faculty.stanford/~knuth/taocp.html>.
- [KT05] Jon Kleinberg and Eva Tardos. *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., 2005. ISBN: 0321295358.
- [Raw92] G.J.E. Rawlins. *Compared to What?: An Introduction to the Analysis of Algorithms*. Computer Science Press, 1992. ISBN: 9780716782438.
- [RS09] Thomas H. Cormen; Charles E. Leiserson ; Ronald L. Rivest and Clifford Stein. *Introduction to Algorithms, Third Edition*. 3rd. The MIT Press, 2009. ISBN: 0262033844.
- [SF13] R. Sedgewick and P. Flajolet. *An Introduction to the Analysis of Algorithms*. Pearson Education, 2013. ISBN: 9780133373486.
- [SW11] R. Sedgewick and K. Wayne. *Algorithms*. Pearson Education, 2011. ISBN: 9780132762564.
- [Tar83] Robert Endre Tarjan. *Data Structures and Network Algorithms*. Society for Industrial and Applied Mathematics, 1983. ISBN: 0-89871-187-8.