



Nombre de la asignatura				Algoritmos Avanzados				Clave de la asignatura			
								C0108071			
Área de formación	Docencia frente a grupo según SATCA				Trabajo de Campo Supervisado según SATCA				Carácter de la asignatura		
	HCS	HPS	TH	C	HTCS	TH	C	TC			
Sustantiva Profesional	2	2	4	4	0	0	0	4	(X) Obligatoria		() Optativa

SERIACIÓN		
Explícita		Implícita
Asignaturas antecedentes	Asignaturas subsecuentes	Conocimientos previos
Ninguna	Ninguna	Aplicación de las estructuras de datos pilas, colas, listas y árboles, así como programar en un lenguaje de alto nivel y conocimientos de matemáticas discretas.

PROPÓSITO DE LA ASIGNATURA	
Calcular la complejidad de un algoritmo (tiempo de ejecución), así como aplicar los diferentes algoritmos de recorridos a un grafo, para resolver problemas de la vida cotidiana utilizando las diferentes técnicas de diseño de algoritmos.	
COMPETENCIAS A DESARROLLAR	
Genéricas	Específicas
Pensamiento crítico y creativo. Capacidad de aplicar los conocimientos en la práctica. Uso de las TIC. Resolución de problemas.	Coordinar equipos de trabajo inter y transdisciplinarios así como multiculturales para la gestión de proyectos computacionales supervisando las funciones y recursos de análisis funcional, orgánico y programación.



UNIDAD No. 1	Notación asintótica	Horas estimadas para cada unidad
		10
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
1.1 Introducción 1.2 Tiempos de Ejecución. 1.3 Concepto de Complejidad. 1.3.1 Algoritmos y Complejidad. 1.3.2 Problemas NP Completos. 1.3.3 Problemas Intratables. 1.3.4 Problemas de Decisión. 1.3.5 Algoritmos No Determinísticos. 1.4 Órdenes de Complejidad. 1.5 Notación Asintótica O, y . 1.5.1 Propiedades de las notaciones asintóticas. 1.5.2 La Velocidad de Crecimiento.	Calcula los tiempos de ejecución de un algoritmo para determinar su eficiencia.	Mapas conceptuales, cuadros o gráficas comparativas. Ejercicios resueltos de la implementación de las notaciones asintóticas.

UNIDAD No. 2	Recurrencias	Horas estimadas para cada unidad
		14
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
2.1. Introducción 2.2. Resolución de Recurrencias 2.3. Método del Teorema Maestro 2.4. Método de la Ecuación Característica 2.5. Cambio de Variable	Resuelve ecuaciones de recurrencia para determinar la complejidad del algoritmo.	Solución de ecuaciones recurrentes lineales y homogéneas. Solución de ecuaciones de recurrencia no lineales.



UNIDAD No. 3	Algoritmos de ordenamiento y búsqueda	Horas estimadas para cada unidad
		12
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
3.1 Algoritmos de ordenamientos $O(n^2)$ 3.1.1 BubbleSort. 3.1.2 SelectionSort 3.1.3 InsertionSort. 3.1.4 ShellSort 3.2 Algoritmo de ordenamiento $O(n \cdot \log(n))$. 3.2.1 HeapSort. 3.2.2 MergeSort 3.2.3 QuickSort. 3.3 Algoritmo de ordenamiento $O(n)$ 3.3.1 Counting sort 3.3.2 Radix sort 3.3.3 Bucket sort	Aplica el método de ordenamiento pertinente en la solución de un problema real.	Programas que aplique los algoritmos de ordenamiento $O(n^2)$. Programas que aplique los algoritmos de ordenamiento $O(n \cdot \log(n))$. Programas donde se apliquen los algoritmos de ordenamiento $O(n)$.



UNIDAD No. 4	Grafos	Horas estimadas para cada unidad
		16
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
4.1. Definiciones Básicas 4.1.1.Representaciones de Grafos 4.1.2.Matriz de Adyacencia 4.1.3.Lista de Adyacencia 4.2. Recorridos de Grafos 4.2.1.Recorridos en Amplitud Primero 4.2.2.Recorridos en Profundidad Primero 4.3. Grafos con Pesos 4.3.1.Árboles 4.3.2.Árbol Cobertor Mínimo 4.3.2.1. Algoritmo de Kruskal 4.3.2.2. Algoritmo de Prim 4.4. Distancias Mínimas en un Grafo Dirigido 4.4.1.Algoritmo de Dijkstra 4.4.2.Algoritmo de Ford 4.4.3.Algoritmo de Floyd-Warshall	Aplica el funcionamiento de algoritmos clásicos sobre grafos aplicados a problemas cotidianos analizando su complejidad computacional.	Programas donde se apliquen los recorridos de Grafos. Programas donde se apliquen los recorridos de Grafos pesados. Programas donde se apliquen los recorridos de Grafos dirigidos.

UNIDAD No. 5	Técnicas de diseño de algoritmos	Horas estimadas para cada unidad
		12
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
5.1. Divide y vencerás. 5.2. Algoritmos voraces. 5.3. Programación dinámica. 5.4. Vuelta atrás (Backtracking). 5.5. Ramificación y poda.	Aplica diferentes técnicas de diseño para la solución de problemas.	Cuadros o gráficas comparativas. Programa donde aplique una técnica de diseño.



Contenidos procedimentales	Contenidos actitudinales
<p>Cálculo y clasificación de un algoritmo. Resolución de recurrencias. Aplicación de métodos de ordenamiento. Solución de problemas utilizando grafos. Aplica diferentes técnicas de diseño.</p>	<p>Responsabilidad en la entrega de sus trabajos. Honestidad en el manejo y presentación de la información. Diligente en el manejo de los datos. Disposición para trabajar en equipo. Rigor en el manejo de la información. Respeto a las propuestas de sus compañeros.</p>

Metodología para la construcción del conocimiento	
Actividades de aprendizaje con el docente	Actividades de aprendizaje autónomo
<p>Exposición. Lecturas específicas. Análisis de ejercicios modelo utilizando un software. Ejercicios modelo utilizando un lenguaje de alto nivel. Caso de uso.</p>	<p>Investigación documental. Elaboración de mapas conceptuales. Desarrollo de programas donde se implementen los diferentes algoritmos en la solución de problemas reales. Solución de ejercicios.</p>

Evidencias de desempeño		
Acreditación	Evaluación	Calificación
<p>Cumplir con lo que establece el Reglamento Escolar vigente.</p> <p>Entrega de evidencias de aprendizaje.</p>	<p>Al final de cada unidad. Al final del curso.</p>	<p>5% Mapas conceptuales, cuadros o gráficas comparativas. 5% Ejercicios resueltos de la implementación de las notaciones asintóticas. 10% Solución de ecuaciones recurrentes lineales y homogéneas. 10% Solución de ecuaciones de recurrencia no lineales. 10% Programas que aplique los algoritmos de ordenamiento $O(n^2)$. 10% Programas que aplique los algoritmos de ordenamiento $O(n \cdot \log(n))$ l. 10% Programas que aplique los algoritmos de ordenamiento $O(n)$. 10% Programas donde se apliquen los recorridos de Grafos. 10% Programas donde se apliquen los recorridos de Grafos pesados. 10% Programas donde se apliquen los recorridos de Grafos dirigidos. 5% Cuadros o gráficas comparativas. 5% Programa donde aplique una técnica de diseño.</p>



FUENTES DE APOYO Y CONSULTA

BÁSICA

1. Cormen, T., et al. (2009). Introduction to Algorithms. USA: The MIT Press.*
2. Du Mortier, G. (2005). Técnicas de programación. Argentina: MP Ediciones.*
3. Guerequeta, R., Vallecillo, A. (2000). Técnicas de diseño de algoritmos. España: Universidad de Málaga.*
4. Sedgewick, R. (1984). Algorithms. USA: Addison Wesley.*
5. Skiena, S. (2008). The algorithm design manual. London: Springer.*

COMPLEMENTARIAS

1. Aho, A. (2007). Estructura de datos y algoritmos. España: Addison Wesley. *
2. Levitin, A. (2006). Introduction to the Design and Analysis of Algorithms. USA: Addison Wesley. *
3. Mañas, J. (1997). Análisis de algoritmos: Complejidad. [en línea] <http://www.lab.dit.upm.es/~lprg/material/apuntes/o/index.html> *

*La bibliografía con antigüedad mayor de cinco años contiene información relevante para el desarrollo de esta asignatura. Cabe destacar que son textos clásicos con ejemplos didácticos de fácil comprensión para el estudiante. Son difíciles de conseguir en el mercado, pero se encuentran en los catálogos de varias bibliotecas.

RESPONSABLE DEL DISEÑO

Elaborado por	Guillermo de los Santos Torres, José Luis Gómez Ramos, Karla Alejandra Zurita Cruz, Oscar Ovando Bautista.
Fecha de elaboración	20 de diciembre de 2016.