UNIVERSIDAD DEL CAUCA

FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

PROGRAMA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

ASIGNATURA : ALGORITMOS COMPUTACIONALES
CRÉDITOS : 3
MODALIDAD : TEÓRICA-PRÁCTICA
INTENSIDAD : 4 HORAS SEMANALES
PRERREQUISITOS: PROGRAMACIÓN ORIENTADA A OBJETOS
ÁREA : CIENCIAS BASICAS DE INGENIERIA
DEPARTAMENTO : TELEMÁTICA

OBJETIVO

Proporcionar a los estudiantes competencias para resolver problemas a través de soluciones computacionales eficaces y eficientes.

OBJETIVOS ESPECÍFICOS

El estudiante al final del curso estará en capacidad de:

- **Seleccionar, adaptar y utilizar algoritmos típicos para resolver problemas comunes en diferentes dominios de aplicación.**
- **Diseñar tipos abstractos de datos para resolver problemas computacionalmente.**
- **Emplear principios y técnicas básicas de complejidad computacional para analizar y evaluar algoritmos y, consecuentemente, soluciones computacionales.**
- **Utilizar programación orientada a objetos para implementar tipos abstractos de datos y algoritmos computacionales.**

METODOLOGÍA

La asignatura se desarrollará mediante:

- **Clases magistrales orientadas por el profesor.**
- **Clases orientadas por el profesor en las que se resolverán dudas acerca de los contenidos del curso; tales contenidos deben ser previamente enviados por el profesor y desarrollados por los estudiantes.**
- **Prácticas desarrolladas por los estudiantes.**
- **Proyecto final, en el dominio de las telecomunicaciones, desarrollado por los estudiantes.**
- **Consultas soportadas por el profesor y monitores de pregrado/posgrado..**

CONTENIDO

1. **Análisis de algoritmos y problemas**
   - 1.1. **Antecedentes matemáticos.**
   - 1.2. **Análisis de algoritmos y problemas.**
   - 1.3. **Clasificación de funciones por su tasa de crecimiento asintótica.**
   - 1.4. **Búsqueda en un arreglo ordenado.**

2. **Abstracción de datos y estructuras básicas de datos**
   - 2.1. **Especificación de TDA y técnicas de diseño.**
   - 2.2. **TDA elementales: listas y árboles.**
   - 2.3. **Pilas y colas.**
   - 2.4. **TDA para conjuntos dinámicos.**

3. **Recursión e inducción**
   - 3.1. Procedimientos recursivos.
   - 3.2. Demostración por inducción
   - 3.3. Ecuaciones de recurrencia
   - 3.4. Árboles de recursión

   - 3.5. Ordenamiento
   - 3.6. Introducción
   - 3.7. Ordenamiento por inserción
   - 3.8. Divide y vencerás
   - 3.9. Quicksort
   - 3.10. Fusión de sucesiones ordenadas
   - 3.11. Mergesort
   - 3.12. Cotas inferiores para ordenar comparando claves
   - 3.13. Heapsort
   - 3.14. Shellsort
   - 3.15. Ordenamiento por base

4. **Selección y argumentos del adversario**
   - 4.1. Determinación de máximo y mínimo
   - 4.2. Cómo hallar la segunda llave más grande
   - 4.3. El problema de selección
   - 4.4. Una cota inferior para la determinación de la mediana
   - 4.5. Diseño contra un adversario

5. **Conjuntos dinámicos y búsquedas**
   - 5.1. Doblado de arreglos
   - 5.2. Análisis de tiempo amortizado
   - 5.3. Árboles rojinegros
   - 5.4. Hashing
   - 5.5. Relaciones de equivalencia dinámicas y programas Unión-Hallar
   - 5.6. Colas de prioridad con operación de decremento de clave

6. **Grafos y recorridos de grafos**
   - 6.1. Definiciones y representaciones
   - 6.2. Recorrido de grafos
   - 6.3. Búsqueda de primero en profundidad en grafos dirigidos
   - 6.4. Componentes fuertemente conectados de un grafo dirigido
   - 6.5. Búsqueda de primero en profundidad en grafos no dirigido
   - 6.6. Componentes bi-conectados de un grafo no dirigido

7. **Problemas de optimización de grafos y algoritmos codiciosos**
   - 7.1. Algoritmo de Prim
   - 7.2. Caminos más cortos de origen único
   - 7.3. Algoritmo de Kruskal

8. **Cierre transitivo, caminos más cortos de todos los pares**
   - 8.1. Cierre transitivo de una relación binaria
   - 8.2. Algoritmo de Warshall para cierre transitivo
   - 8.3. Caminos más cortos de todos los pares en grafos
   - 8.4. Cálculo del cierre transitivo con operaciones de matrices
   - 8.5. Multiplicación de matrices de bits: algoritmo de Kronrod

9. **Programación dinámica**
   - 9.1. Grafos de sub-problema y su recorrido
   - 9.2. Multiplicación de una sucesión de matrices
   - 9.3. Construcción de árboles de búsqueda binaria óptimos
   - 9.4. División de sucesiones de palabras en líneas
   - 9.5. Desarrollo de un algoritmo de programación dinámica

EVALUACIÓN

El tipo de evaluación y la respectiva ponderación son concertadas el primer día de clase con los estudiantes, teniendo en cuenta el reglamento estudiantil de la Universidad del Cauca. El sistema de evaluación promueve la eficiencia y calidad del proceso de enseñanza - aprendizaje del curso, detectando el nivel de desempeño de los estudiantes con el fin de realizar los correctivos necesarios durante el transcurso del semestre.

BIBLIOGRAFÍA

Jeff Erickson. Algorithms. Department of Computer Science University of Illinois at Urbana-Champaign. 2009. http://jeffe.cs.illinois.edu/teaching/algorithms/2009/everything.pdf

James F. Kurose. Keith W. Ross. Computer Networking: A Top-Down Approach (6th Edition) 6th Edition. http://www-net.cs.umass.edu/kurose-ross-ppt-6e/.

Mark Lutz. Learning Python. 2013. ISBN: 978-1-449-35573-9.

Rudolf Pecinovský. Learn Object Oriented Thinking and Programming. ISBN 978-80-904661-9-7 (PDF). 2013.

Sara Baase, Allen Van Gelder: Computer Algorithms - Introduction to Design and Analysis (3. ed., repr. with corr.). Pearson / Prentice Hall 2000, ISBN 978-0-201-61244-8, pp. I-XIX, 1-688.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein: Introduction to Algorithms (3. ed.). MIT Press 2009, ISBN 978-0-262-03384-8, pp. I-XIX, 1-1292.