

## **Análisis de Algoritmos**

### **1. Resumen**

#### **Dónde está?**

En la secuencia

EDI I -> MTP II -> EDI II -> **AA** -> TLF II

#### **Por qué?**

- Disponer de ejemplos de técnicas y enfoques para resolución de problemas mediante programación
- Mejorar la capacidad de análisis de problemas y de eficacia de algoritmos
- Aumentar y profundizar en la comprensión de métodos de utilidad universal
- Evitar reinventar la rueda
- Divertirse programando

#### **Para quién?**

- Estudiantes interesados en la programación
- Estudiantes interesados en la resolución de problemas
- Estudiantes interesados en una carrera en desarrollo de software

#### **Temas afines**

- Complejidad algorítmica
- Investigación operativa (traveling salesman, distancias mínimas, bin packing...)

#### **Otras cosas**

- Seguimiento del estudiante (lista, evaluación continua en prácticas)
- Examen parcial liberatorio bajo ciertas condiciones.

## 2. Objetivos

Es un curso orientado a estudiantes que quieran profundizar en algoritmos y técnicas básicas de resolución de problemas computacionales. En concreto, y sobre una selección de problemas que se pueden considerar como relevantes, se abordarán:

1. Conceptos básicos sobre el problema en cuestión y cuestiones afines.
2. Formulación de los correspondientes algoritmos y su pseudocódigo.
3. Análisis de la corrección de los algoritmos.
4. Análisis y discusión de las estructuras de datos más adecuadas.
5. Desarrollo manual de los algoritmos sobre ejemplos pequeños.
6. Programación en lenguaje C de los algoritmos.
7. Análisis del rendimiento en el caso peor.

## 3. Programa detallado

### Parte I: Algoritmos en Grafos

1. Revisión de Algoritmos elementales en grafos
  - 1.1. *Revisión de conceptos básicos*
  - 1.2. *Problemas de distancia mínima*
  - 1.3. *Revisión del algoritmo de Dijkstra*
2. Árboles abarcadores mínimos
  - 2.1. *Revisión de árboles abarcadores mínimos*
  - 2.2. *Algoritmo de Kruskal*
  - 2.3. *TAD Conjunto Disjunto*
  - 2.4. *Corrección de Prim y Kruskal*
3. Aplicaciones de Búsqueda en Profundidad
  - 3.1. *Introducción*
  - 3.2. *Grafos biconexos*
  - 3.3. *Circuitos eulerianos*

- 3.4. *Secuenciación de genes y caminos eulerianos*
- 3.5. *Circuitos hamiltonianos y el Problema del Viajante*

## **Parte II: Técnicas generales de diseño de algoritmos**

### **1. Programación dinámica**

- 1.1. *El problema de la suma y la criptografía de clave pública*
- 1.2. *Ordenación óptima en el producto de matrices*
- 1.3. *Problemas resolubles por programación dinámica*
- 1.4. *Árboles binarios de búsqueda óptimos*

### **2. Recursividad**

- 2.1. *El problema de Selección*
- 2.2. *Problemas resolubles mediante recursividad*
- 2.3. *Multiplicación de matrices*
- 2.4. *La Transformada Rápida de Fourier*

### **3. Algoritmos codiciosos**

- 3.1. *Secuenciación lineal en colas de trabajos*
- 3.2. *Problemas resolubles mediante algoritmos codiciosos*
- 3.3. *Codificación Huffman*
- 3.4. *Algoritmos codiciosos y de programación dinámica*

### **4. Prácticas**

Se realizarán 3-4 prácticas en lenguaje C, de entrega obligatoria, consistentes en la implementación de algoritmos estudiados en el curso y sus estructuras de datos asociadas, considerándose también otras cuestiones, como la medición de su rendimiento.

## 5. Exámenes

El examen parcial es liberatorio bajo ciertas condiciones.

El examen final consta de 6 a 8 cuestiones sobre:

- Conceptos y resultados teóricos básicos
- Resolución de problemas de análisis, ejecución y aplicación de los algoritmos estudiados.

Se podrá incluir una pregunta relativa a las prácticas.

Nota final: 70% nota teoría + 30% nota prácticas

## 6. Bibliografía

1. Cormen, Leiserson, Rivest, "Introduction to algorithms", The MIT Press--Mc Graw Hill. 1990.
2. Weiss, "Data structures and algorithm analysis in C", Benjamin Cummings, 1993.
3. Sedgewick, "Algorithms in C", Addison-Wesley.
4. Aho, Hopcroft, Ullman, "The design and analysis of algorithms", Addison Wesley.
5. Aho, Hopcroft, Ullman, "Data Structures and Algorithms", Addison Wesley.