Asignatura: Diseño y Análisis de Algoritmos
Código: 18766
Centro: Escuela Politécnica Superior
Titulación: Grado en Ingeniería Informática
Nivel: Grado
Tipo: Formación obligatoria
Nº de créditos: 6

GUÍA DOCENTE DE DISEÑO Y ANÁLISIS DE ALGORITMOS

La presente guía docente corresponde a la asignatura **Diseño y Análisis de Algoritmos**, aprobada para el curso lectivo 2014-2015 en Junta de Centro y publicada en su versión definitiva en la página web de la Escuela Politécnica Superior. Esta guía docente aprobada y publicada antes del periodo de matrícula tiene el carácter de contrato con el estudiante.

ASIGNATURA

DISEÑO Y ANÁLISIS DE ALGORITMOS (DAA)

1.1. Código

18766 del Grado en Ingeniería Informática

1.2. Materia

Computación e Inteligencia Artificial

1.3. Tipo

Formación optativa

1.4. Nivel

Grado

1.5. Curso

4º

1.6. Semestre

1º

1.7. Número de créditos

6 créditos ECTS

1.8. Requisitos previos

Para un buen aprovechamiento del curso, es recomendable haber cursado las materias Programación I, Programación II, Estructuras Discretas y Lógicas y Análisis de Algoritmos.

También se supone unos conocimientos matemáticos básicos al nivel de los cursos Algebra, Cálculo I y Cálculo II.

En relación a las clases prácticas, el estudiante deber tener una competencia media-alta en el lenguaje C y en el manejo de algún entorno gráfico de programación, como Visual C++, Anjuta o similar, y sus herramientas de depuración de código.

Asimismo, es recomendable que el estudiante disponga de un dominio del idioma inglés que le permita como mínimo leer la bibliografía de consulta.

1.9. Requisitos mínimos de asistencia a las sesiones presenciales

Se considerarán dos itinerarios, Presencial y No Presencial, tanto para teoría como para prácticas.

El itinerario presencial requerirá una asistencia mínima del 85% de clases teóricas.

En cualquier caso, **la asistencia a clases de teoría o problemas se considera esencial para la superación de la asignatura**, ya que dicha asistencia supone

- La toma de contacto explicada con el material de la asignatura, con la que el estudiante obtiene una primera y en general suficiente comprensión de dicho material. Dicha comprensión es **muy difícil y costosa de obtener por otras vías**.
- Una parte muy importante, **casi un 30%, del tiempo que el estudiante debe dedicar a la asignatura.**

Por todo ello es sumamente recomendable asistir a dichas clases, indicando la experiencia que **no hacerlo aumenta de manera considerable la dificultad de superación de la asignatura**.

1.10. Datos del equipo docente

Dr. José R. Dorronsoro
Departamento de Ingeniería Informática
Escuela Politécnica Superior
Despacho - Módulo: B-358 Edificio B – 3ª Planta
Teléfono: +34 91 497 2329
Correo electrónico: jose.dorronsoro@uam.es
Página web: https://moodle.uam.es/user/view.php?id=12586&course=1
Horario de atención al alumnado: Petición de cita previa por correo electrónico.

1.11. Objetivos del curso

Las **competencias comunes a la rama de Informática** que el estudiante adquiere con la asignatura Diseño y Análisis de Algoritmos son:

- C6. Conocimiento y aplicación de los procedimientos algorítmicos básicos de las tecnologías informáticas para diseñar soluciones a problemas, analizando la idoneidad y complejidad de los algoritmos propuestos.

- C7. Conocimiento, diseño y utilización de forma eficiente de los tipos y estructuras de datos más adecuados a la resolución de un problema.

Las **competencias de tecnología específica** que el estudiante adquiere con la asignatura Diseño y Análisis de Algoritmos son:

- CC1. Capacidad para tener un conocimiento profundo de los principios fundamentales y modelos de la computación y saberlos aplicar para interpretar, seleccionar, valorar, modelar, y crear nuevos conceptos, teorías, usos y desarrollos tecnológicos relacionados con la informática.
- CC3. Capacidad para evaluar la complejidad computacional de un problema, conocer estrategias algorítmicas que puedan conducir a su resolución y recomendar, desarrollar e implementar aquella que garantice el mejor rendimiento de acuerdo con los requisitos establecidos.
- CC4. Capacidad para conocer los fundamentos, paradigmas y técnicas propias de los sistemas inteligentes y analizar, diseñar y construir sistemas, servicios y aplicaciones informáticas que utilicen dichas técnicas en cualquier ámbito de aplicación.
- CC5. Capacidad para adquirir, obtener, formalizar y representar el conocimiento humano en una forma computable para la resolución de problemas mediante un sistema informático en cualquier ámbito de aplicación, particularmente los relacionados con aspectos de computación, percepción y actuación en ambientes o entornos inteligentes.
- IS1. Capacidad para desarrollar, mantener y evaluar servicios y sistemas software que satisfagan todos los requisitos del usuario y se comporten de forma fiable y eficiente, sean asequibles de desarrollar y mantener y cumplan normas de calidad, aplicando las teorías, principios, métodos y prácticas de la Ingeniería del Software.
- IS4. Capacidad de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales.

Los **resultados del aprendizaje** que el estudiante adquiere con la asignatura Diseño y Análisis de Algoritmos son:

- Algoritmos en Grafos: conocimientos avanzados de las estructuras de datos, problemas de distancias mínimas, de árboles abarcadores mínimos, de exploración de grafos o de problemas de flujos, así como sus aplicaciones a

problemas de conexión de grafos, secuenciación de proteínas o segmentación de imágenes.

- Resolución de problemas mediante programación dinámica: conceptos básicos, enfoque general de dicha técnica y su aplicación a problemas de optimización o de criptografía de clave pública, así como un análisis de su rendimiento.

- Resolución de problemas mediante algoritmos codiciosos: su aplicación a problemas de optimización, su especificidad frente a la programación dinámica y su aplicación a problemas de codificación y de empaquetamiento.

- Resolución de problemas mediante algoritmos recursivos: las implicaciones de una solución recursiva, el análisis de su coste, y su aplicación en problemas concretos de multiplicación de matrices, polinomios y números o de análisis de señales.

Los **objetivos generales** son que el estudiante al acabar el curso haya alcanzado una profundización en su conocimiento de algoritmos avanzados y en la aplicación de técnicas básicas de resolución de problemas computacionales. En concreto, y sobre una selección de problemas de interés en computación teórica y práctica, 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 los lenguajes C y/o Python de los algoritmos.
7. Análisis del rendimiento en el caso peor.

Como resultado de lo anterior, el estudiante debe lograr

- Tener un buen conocimiento general de los principios de diseño de algoritmos de tipo codiciosos, recursivos o de programación dinámica.

- Tener un buen conocimiento de un amplio repertorio de cuestiones y problemas planteados sobre grafos.

- Tener un buen conocimiento de ejemplos relevantes de problemas concretos de las categorías anteriores y de los elementos generales de los principales algoritmos de resolución de cada problema individual.

- Identificar para dichos problemas las estructuras de datos más adecuadas para el mismo, reflejar sobre ellas el enfoque algorítmico a aplicar e identificar los elementos determinantes de su rendimiento.

- Ser capaz, para un nuevo problema, de analizar sus elementos característicos, proponer un enfoque algorítmico general para su resolución, desarrollar una solución adecuada y estimar su coste computacional.
- Comunicar de manera clara, estructurada y concisa los resultados de su trabajo.
- Participar activamente en los análisis y discusiones de grupo que se establezcan al hilo del desarrollo de las actividades propuestas, cooperar con otros compañeros en el desarrollo de trabajos conjuntos y comunicar con propiedad y corrección sus reflexiones sobre los resultados de su trabajo.

Objetivos específicos

El curso se estructura en las siguientes partes y unidades, de las que se dan sus objetivos específicos:

Parte I. Algoritmos sobre grafos

Unidad 1. Revisión de Algoritmos elementales en grafos:

Conocer los conceptos básicos del trabajo en grafos, repasar los problemas de distancia mínima y su resolución. Considerar un enfoque alternativo mediante programación dinámica.

Unidad 2. Árboles abarcadores mínimos.

Revisar el planteamiento general y repasar el algoritmo de Prim. Plantear el algoritmo de Kruskal con carácter general, desarrollar el TAD Conjunto Disjunto e implementar sobre dicho TAD el algoritmo de Kruskal. Analizar la corrección de algoritmos de Prim y Kruskal

Unidad 3. Aplicaciones de Búsqueda en Profundidad.

Repasar los conceptos básicos de búsqueda en profundidad. Aplicar dicha búsqueda a problemas de conexión básica y de biconexión. Determinar la existencia y buscar circuitos eulerianos sobre grafos dirigidos y no dirigidos y su aplicación a la secuenciación de genes. Conocer el concepto de circuitos hamiltonianos y su aplicación al problema del viajante. Tener un primer contacto con los problemas P, NP. Algoritmos aproximados

Unidad 4. Flujos en grafos.

Conocer el planteamiento general del problema de flujos óptimos en grafos y los conceptos fundamentales. Conocer y aplicar el procedimiento general de Ford-Fulkerson de identificación de un flujo máximo y un corte mínimo. Conocer ejemplos

de aplicación de dichas técnicas. Tener un primer contacto con extensiones recientes de la teoría.

Parte II: Técnicas generales de diseño de algoritmos

Unidad 1.Programación dinámica

Conocer las propiedades básicas de los problemas resolubles mediante programación dinámica (PD). Conocer y resolver algunos problemas concretos: el problema de la suma y criptografía de clave pública, la ordenación óptima en el producto de matrices o la construcción de árboles binarios de búsqueda óptimos.

Unidad 2.Recursividad

Repasar los conceptos básicos de la resolución de problemas mediante recursividad, así como cuestiones práctica sobre su implementación. Aplicar técnicas recursivas al problema general de selección. Multiplicar de manera recursiva matrices, números y polinomios y su aplicación a la Transformada Rápida de Fourier.

Unidad 3.Algoritmos codiciosos.

Conocer las propiedades básicas de los problemas resolubles por optimización codiciosa y su diferenciación de las soluciones PD. Resolver de manera codiciosa el problema de empaquetamiento. Derivar de manera codiciosa el algoritmo de Huffman y efectuar una introducción al concepto de entropía y al problema de codificación y compresión.

1.12. Contenidos del programa

El programa detallado del curso es:

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

3.6. Introducción a la complejidad algorítmica. Problemas P y NP. Algoritmos aproximados.

4. Flujos en grafos

4.1. Flujos, flujos máximo, cortes, caminos aumentadores

4.2. Flujos máximos y cortes mínimos

4.3. El algoritmo de Ford-Fulkerson y variantes

4.4. Aplicaciones y extensiones

Parte II: Técnicas generales de diseño de algoritmos

1. Programación dinámica

1.1. El problema de la suma y la criptografía de clave pública 4

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

1.13. Referencias de consulta

Bibliografía disponible a través del catálogo de la biblioteca (pincha aquí)

No hay un manual que se ajuste en su totalidad a los contenidos del curso. En cualquier caso, las referencias inferiores constituyen un buen complemento para el seguimiento del curso.

Referencias básicas:

- Cormen, Leiserson, Rivest, Introduction to algorithms, The MIT Press--Mc Graw Hill.
- Kleinberg, Tardos. Algorithm Design. Pearson.
- Weiss, Data structures and algorithm analysis in C, Benjamin Cummings.

Referencias complementarias:

- Sedgewick, "Algorithms in C", Addison-Wesley.
- Aho, Hopcroft, Ullman, "The design and analysis of algorithms", Addison Wesley.
- Aho, Hopcroft, Ullman, "Data Structures and Algorithms", Addison Wesley.
- Guttag. Introduction to Computation and Programming Using Python.

2. Métodos docentes

La asignatura se organiza en clases teóricas, clases de problemas, prácticas de laboratorio, actividades programadas en grupo y pruebas individualizadas.

En las **clases teóricas** se presentarán los conceptos de manera clara y suficientemente concisa utilizando para ello preferentemente el método de lección magistral pero requiriendo la participación del estudiante y fomentando su iniciativa. El estudiante deberá trabajar de manera autónoma el contenido de cada clase para adquirir una comprensión suficiente que le permita seguir las clases sucesivas y abordar los problemas propuestos.

En las **clases de problemas** se presentarán y resolverán ejercicios y problemas tipo, representativos de lo exigible en las pruebas evaluatorias. El estudiante debe complementar este trabajo realizado en clase con la resolución por su cuenta de ejercicios complementarios que se propondrán a lo largo del curso.

Las clases de problemas se complementarán con **sesiones de trabajo en grupos reducidos**, donde se propondrá la resolución de ciertos problemas significativos, dándose unas primeras pautas sobre los mismos y siguiendo el profesor el avance efectuado por los distintos grupos.

Esta actividad dependerá de la disponibilidad de espacios adecuados para este tipo de trabajo.

En las **prácticas de laboratorio** se efectuarán **tres** prácticas sobre algunos de los algoritmos abordados en clase. Se realizarán utilizando los lenguajes de programación C y Python e incorporarán el uso de otras herramientas, como funciones de medida de tiempo de ejecución, visualizadores de curvas de datos o notebooks de programación.
Dado que es previsible que algunos estudiantes no conozcan Python, se efectuará una introducción al mismo con carácter previo a su uso en prácticas.
A su vez, se fomentará el aprendizaje cooperativo y el trabajo sostenido en el tiempo, inculcándose además el sentido ético que debe primar en el estudio universitario y la vida profesional, persiguiéndose comportamientos fraudulentos como la copia de prácticas.

3. Tiempo de trabajo del estudiante

Recomendaciones generales

- Una aproximación razonable al esfuerzo medio a esperar del estudiante que supere la asignatura se detalla en el **cronograma** de la misma.
- La **asistencia activa** a clase se considera **imprescindible** para la mayoría de los estudiantes. Del cronograma se desprende que dicha asistencia representa unas 38 horas, aproximadamente la mitad del esfuerzo que el estudiante debe hacer para superar los exámenes. Esto es, de no asistir a clase, el estudiante deberá como mínimo efectuar dicho esfuerzo por su cuenta, y ello sin tener en cuenta la dificultad de entender, sin ninguna explicación, materiales de cierta complejidad y dificultad.

- Las clases se impartirán según la fórmula de **lección magistral participativa**, con soporte de trasparencias y explicaciones en pizarra.
- Se recomienda que el estudiante acuda a las mismas provisto de copias impresas de dichas trasparencias así como de los problemas asignados.

En principio la distribución **aproximada** de las diversas actividades del curso será la siguiente:

Resumen		**Horas**	**Porcentaje**	**Tipo**
	Clases teóricas	22	15	Teo./Práct
	Clases de problemas	15	10	Teo./Práct
	Clases prácticas	24	16	Laboratorio
	Tutorías	4	3	Tutoría
	Pruebas escritas	6	4	
No presencial	Estudio semanal material teórico	23	15	Estudio
	Resolución semanal de ejercicios y problemas	20	13	Estudio
	Estudio de material teórico para el examen final	14	9	Estudio
	Ejercicios y problemas para el examen final	22	15	Estudio
	Totales	150	100	

4. Métodos de evaluación y porcentaje en la calificación final

4.1. Consideraciones generales

La asignatura consta de una parte práctica y una teórica. Ambas partes se puntúan sobre 10 puntos. La nota final de la asignatura se obtiene a partir de las notas obtenidas en las partes de teoría y de prácticas como sigue

Calificación=0.4*Prácticas + 0.6*Teoría

Para aprobar la asignatura es obligatorio obtener una nota mayor o igual a 5 puntos, tanto en la parte de teoría como en la práctica. En caso contrario, la nota final que figurará en actas será

Calificación= 0.4*Mín(5, Prácticas) + 0.6*Mín(5,Teoría)

Las notas de teoría y de prácticas se conservan (convalidan) para la convocatoria extraordinaria del mismo curso académico bajo ciertas condiciones.

4.2. Calificación de teoría

La parte de teoría del curso se evaluará mediante tres exámenes parciales, correspondientes cada uno de ellos aproximadamente a una tercera parte del curso y que constituyen el itinerario de evaluación continua. Tendrán una duración de 1 hora con dos preguntas sobre 10 puntos de varios apartados. Dichos exámenes tendrán lugar de acuerdo al siguiente calendario aproximado

- Parcial 1: semana del 20 de octubre de 2014.
- Parcial 2: semana del 1 de diciembre de 2014.
- Parcial 3: primeros de enero de 2015, coincidiendo con la fecha del examen final.

Los parciales 1 y 2 serán liberatorios para aquellos estudiantes que los superen obteniendo además una nota de 4 o superior en cada pregunta.
Caso de no superar alguno de estos dos parciales, el estudiante se examinará de nuevo del mismo (o, en su caso, de ambos) en el examen final, además de, naturalmente, la tercera parte del curso.

La nota final de teoría se calculará como la media de las tres notas de cada parte del curso.

Observaciones importantes:

1. Para aquellos estudiantes que no hayan liberado ninguno de los dos primeros parciales, la nota final será exclusivamente la del examen final de 180'. En este caso, dicho examen **tendrá una primera parte en la que será necesario obtener una nota mínima de 7 para superarlo, salvo que el estudiante haya tenido una nota de 3,5 o superior en ambos parciales**.

2. Aquellos estudiantes que hayan liberado los dos parciales deberán tener una nota de 3,5 o superior en el examen de la tercera parte para superar la asignatura.
3. La nota de teoría sólo se guardará para la convocatoria extraordinaria en el caso de superarse la materia en su totalidad. **No se guardarán notas de parciales individuales.**

4.3. Calificación de prácticas

Según el estudiante haya seguido o no el itinerario presencial en prácticas, podrá haber una **prueba final de prácticas**.
En el primer caso la calificación final de la parte práctica de la asignatura será:

33,3% Nota Pract1 + 33,3% Nota Pract2 + 33,3% Nota Pract3

donde la nota de una práctica corresponderá a su corrección como a los eventuales controles sobre la misma que se vayan haciendo en el laboratorio.

En el segundo caso el estudiante deberá haber completado previamente las prácticas propuestas y la calificación final de la parte práctica de la asignatura tendrá como base una prueba de dos horas, que supondrá efectuar en el laboratorio una modificación de cierto alcance de alguna de las prácticas propuestas así como responder a algunas cuestiones. En este caso la nota de prácticas será la obtenida en la prueba en cuestión.

ATENCIÓN:
Cualquier copia descubierta que se haya realizado a lo largo del curso, tanto en cualquiera de las actividades de teoría como en cualquiera de los apartados de las prácticas, penalizará por igual, tanto a los alumnos que copian como a los copiados. La penalización por copia implica la aplicación de la normativa de la UAM así como la normativa interna de la EPS que sea aplicable.

5. Cronograma

El esfuerzo total del estudiante se estima en 128 horas (incluyendo pruebas), repartidas entre el inicio del curso y la celebración del examen final. Las fechas concretas de la columna **semana** se completarán al inicio del curso.
A continuación se da una aproximación al cronograma de la asignatura, que podría experimentar pequeños cambios según se desarrolle el curso.

Mes		Día	Contenido
Septiembre	8	1	Presentación
		2	Revisión de grafos. Distancias mínimas en grafos.
		3	Algoritmo de Dijkstra Algoritmos codiciosos.

	15	4	AAms. Algoritmo de Prim .Algoritmo de Kruskal
		5	TAD CD I
		6	TAD CD II
	22	7	Corrección de Prim y Kruskal
		8	Problemas (1) 4, 5, 7, 9, 11, 12, 14, 15, 25, 26
		9	Problemas (2) 31, 33, 35, 38, 39, 43, 44
	29	10	Búsqueda en profundidad.
		11	Grafos biconexos y puntos de articulación
		12	Puntos de articulación II
Octubre	6	13	Problemas (3) 31, 33, 39, 40, 41
		14	Circuitos eulerianos en grafos no dirigidos.
	13	15	Circuitos hamiltonianos. El problema del viajante. Introducción a P y NP I.
		16	Introducción a P y NP II. Algoritmos aproximados.
		17	Problemas (4) 43, 50, 52, 55, 56, 57, 58, 59
	20	18	**Parcial 1**
		19	Secuenciación de proteínas. Circuitos eulerianos en grafos no dirigidos. Ensamblado de proteínas.
		20	Problemas (5) 61, 62, 28, 29 Distancia de edición. Máxima subcadena común.
	27	21	Flujos en grafos
		22	Algoritmo de Ford-Fulkerson
Noviembre	3	23	Extensiones y aplicaciones
		24	Repaso de problemas
	10	25	Problema de la mochila
		26	Problema de la suma
		27	Criptografía de clave pública. Algoritmo de Merkle-Hellman.
	17	28	Multiplicación óptima de matrices.
		29	ABdBs óptimos
		30	Problemas (6): 171, 172, 176, 177, 178, 179, 182, 183, 185, 186, 187
	24	31	Algoritmos recursivos, multiplicación de matrices
		32	El problema de selección. Algoritmo Quickselect: caso medio
		33	Algoritmo Quickselect: caso peor. Multiplicación de polinomios
Diciembre	1	34	**Parcial 2**

			Transformada rápida de Fourier I
			Inversión de la TFR.
	8	35	Ejemplos del uso de la TFR.
		36	Problemas (7): 167, 168, 169, 146
		37	Problemas (8): 149, 150, 151, 152
	15	38	Algoritmos codiciosos. Mochila codiciosa. Códigos prefijo.
		39	Codificación Huffman. Entropía. Compresión óptima. Códigos Shannon
		40	Problemas (9): 115, 116, 117, 118, 119, 124, 125, 126, 127 132, 140, 141