UNIVERSIDAD DE EXTREMADURA

Plan Docente de la asignatura

I. Descripción y contextualización

Identificación y características de la asignatura				
Denominación y código	**Estructuras de Datos y Algoritmos – A3**			
Curso y Titulación	2º de Ingeniería Informática			
Área	Lenguajes y Sistemas Informáticos			
Departamento	*Ingeniería de Sistemas Informáticos y Telemáticos*			
Tipo y ctos. LRU	troncal		(6T + 3P)	
Coeficientes	Practicidad: 3 alto		Agrupamiento: 3 medio	
Duración ECTS (créditos)	**Anual**		**8,18 ECTS (204,5 horas)**	
Distribución ECTS (rangos)	Grupo grande: 30%	Seminario-Lab.: 15%	Tutoría ECTS: 5%	No presenciales: 55%
	60h	30h	0	112-113
Descriptores (según BOE)	RD 1459/1990 - I.I. RD 1460 / 1990 - I.T.I.S. RD 1461 / 1990 - I.T.I.G. Denominación: Estructura de Datos y de la Información Contenidos: Tipos abstractos de datos. Estructuras de datos y algoritmos de manipulación. Estructuras de la información: ficheros y bases de datos Créditos: 9 Área de Conocimiento: Lenguajes y Sistemas InformáticosI			
Coordinador-Profesor/es	Pablo García Rodríguez, María Luisa Durán Martín-Merás Juan Hernández Núñez			
Tutorías complementarias (1)	Despacho Sala 3	Ext. 7260	pablogr@unex.es	
	Martes de 16-22			
Tutorías complementarias (2)	Despacho I-1	Ext. 7808	mlduran@unex.es	
	Lunes y Martes 10.30 – 13.30			
	Despacho I-7	Ext. 7257	juanher@unex.es	

Tutorías complementarias (3)	Despacho I-7	Ext. 7257	juanher@unex.es
	Primer cuatrimestre: Lunes y Martes de 9.30 a 11.30; Viernes de 10-11 Segundo Cuatrimestre: Lunes, Martes Miércoles, de 9-11		

*Contextualización profesional**

Dentro de los contenidos específicos del título se encuentra la subcategoría Programación, en cuyo estudio se centra esta materia. En el Libro Blanco esta subcategoría debe tratar:

- Fundamentos y metodología de la programación
- Algoritmia
- Computabilidad
- Lenguajes de programación
- Paradigmas de programación
- Estructuras de datos

Son competencias fundamentales de esta asignatura:

- Fundamentos y metodología de la programación
- Algoritmia
- Computabilidad
- Estructuras de datos
- Paradigmas de programación

En el borrador del título recientemente publicado, el bloque de Programación, con un total de 27 créditos ECTS, incluye los siguientes conocimientos, capacidades y destrezas: "Comprender los fundamentos teóricos de programación: analizar la computabilidad y complejidad algorítmica. Conocer y utilizar lenguajes estructurados para el desarrollo de sistemas software. Conocer las estructuras de datos básicas, sus aplicaciones y propiedades. Determinar las estructuras de datos más adecuadas. Conocer y utilizar los principales paradigmas de programación. Saber emplear técnicas de verificación y validación de programas."

Esta asignatura desarrolla de forma intensa la mayoría de los conocimientos y destrezas anteriores, que se aplicarán en asignaturas posteriores. La aplicación de estos conocimientos en otras asignaturas contribuye a que sean afianzados e incorporados como base en el desarrollo de trabajos de programación.

Esta materia debe ser tratada independientemente del perfil profesional que pretenda alcanzarse, al ser una materia de segundo curso y las destrezas y competencias que se adquieren con el desarrollo de esta materia son de utilidad común para desarrollar tareas de programación de cualquier índole profesional. Los objetivos fundamentales son que el alumno sepa analizar, diseñar e implementar programas, se pretende que el alumno mantenga una visión crítica que deberá desarrollar posteriormente. También le dotaremos de las herramientas necesarias para la construcción de los programas. Cualquier perfil profesional que desarrolle el estudiante posteriormente se verá influido por su conocimiento de las técnicas básicas y avanzadas de programación y su capacidad para analizar y resolver problemas, presentar y valorar alternativas, competencias desarrolladas ampliamente en esta asignatura.

Al ser una asignatura de programación de segundo curso, sienta sus bases sobre la asignatura "Elementos de programación" de primer curso. Esta es la razón por la que ambas asignaturas mantienen una serie de competencias compartidas que una vez introducidas en el primer curso son desarrolladas posteriormente en esta asignatura.

Competencias compartidas:

- Computabilidad
- Lenguajes de programación
- Paradigmas de programación
- Estructuras de datos

Esta asignatura también se relaciona con muchas de las asignaturas restantes que forman parte de los contenidos específicos de la Ingeniería Informática, ya que en casi todas se usa la programación de ordenadores para conseguir los objetivos formativos propuestos.

*Contextualización curricular**

Las competencias genéricas y específicas del Título que están relacionadas de una forma directa con los contenidos de esta materia se pueden resumir en las siguientes:

1. Dirigir y coordinar el proyecto de desarrollo y mantenimiento de aplicaciones, supervisando las funciones y recursos de análisis funcional, orgánico y programación, asegurando la adecuada explotación de las aplicaciones.
2. Dominar todas las etapas de la vida de un proyecto (análisis de concepción, análisis técnico, programación, pruebas, documentación y formación de usuarios).
3. Control y seguimiento de plazos, indicadores económicos y de calidad.
4. Analizar y recoger nuevas técnicas y herramientas del mercado estudiando su viabilidad y necesidad. Posibilidad de contratar recursos externos.
5. Control y Gestión del Desarrollo del Proyecto Informático.
6. Realizar el análisis y el diseño detallado de las aplicaciones informáticas.
7. Definir la estructura modular y de datos para llevar a cabo las aplicaciones informáticas que cumplan con las especificaciones funcionales y restricciones del lenguaje de programación.
8. Definición y descripción de procedimientos e interfaz de usuario.
9. Realizar pruebas que verifiquen la validez funcional, la integridad de los datos y el rendimiento de las aplicaciones informáticas.
10. Elaborar y mantener documentación descriptiva de la génesis, producción y operatividad de las aplicaciones informáticas.
11. Participar en el diseño de nuevos sistemas informáticos como consecuencia de la información de áreas de la empresa que utilizan para el desarrollo de sus tareas, métodos y procesos manuales.
12. Integrar sistemas informáticos existentes susceptibles de inter-relacionarse.
13. Escuchar y asesorar a los usuarios en la resolución de los problemas que se les plantean con el uso de los sistemas informáticos.
14. Mantenerse al día en Técnicas, Métodos y Herramientas de Análisis y Diseño.
15. Creación de los tests de pruebas para verificar que los Sistemas Informáticos cumplen los requisitos y especificaciones de Análisis y Diseño.
16. Dirección del arranque o “lanzamiento” de un nuevo sistema.
17. Ayudar al Área de Estudios en la resolución de los fallos que se producen en los Sistemas en Producción.
18. Supervisar, controlar y dar validez a los procesos de desarrollo.
19. Asesorar a los programadores en los problemas que se les plantean con la programación de los sistemas.
20. Dirigir el equipo de trabajo compuesto por Analistas Funcionales, Analistas de aplicaciones, Programadores.

*Contextualización personal**

Itinerarios de procedencia y requisitos formativos de los alumnos:

Esta es una asignatura anual de segundo curso que se plantea como la continuación de la asignatura de primer curso "Elementos de programación" de tal forma que dicha asignatura debe presentarse como prerrequisito de ésta. Ambas materias sientan las bases de la programación, una herramienta básica en la formación de un Ingeniero en Informática.

Existe mucha interrelación entre esta asignatura y el resto de las asignaturas del primer y segundo curso, así como con el resto de asignaturas de cursos posteriores que utilizan la programación de ordenadores como herramienta.

Asignaturas como *Organización de computadores* de segundo curso contribuyen a que el alumno pueda entender mejor el funcionamiento de los algoritmos en cuanto a su ejecución en el ordenador y de esta forma se facilita el análisis de costes computacionales, tanto en tiempo como en espacio.

La asignatura *Laboratorio de programación II* está estrechamente relacionada con *Estructuras de datos y algoritmos*, ya que comparten objetivos formativos. De hecho, se considera que ambas asignaturas deben ser co_rrequisitos, en *Laboratorio de programación II* se llevan a la práctica casi todos los conocimientos teóricos que se aportan en *Estructuras de datos y algoritmos.*

Las asignaturas *Matemática discreta* y *Estadística* de segundo curso van proporcionando al alumno herramientas matemáticas que son de aplicación directa e inmediata en *Estructuras de datos y algoritmos*. Por tanto estas dos asignaturas también deberían marcarse como co_rrequisitos de *Estructuras de datos y algoritmos*

Por tanto se podría afirmar que si el alumnos ha superado las asignaturas de primer curso, especialmente *Elementos de programación* y *Laboratorio de Programación I* y se encuentra cursando las asignaturas de segundo indicadas anteriormente como co_rrequisitos, está en condiciones de superar esta asignatura sin mayores dificultades.

Con todas estas asignaturas, el estudiante debería obtener una versión general del campo de actuación del Ingeniero en Informática y de las bases teóricas que le ayudarán en el desarrollo posterior de otras materias.

Otras consideraciones de interés

De forma general, nos gustaría destacar el bajo nivel que demuestran los alumnos en la expresión escrita y oral (además una gran deficiencia en conocimientos elementales relacionados con las matemáticas), necesidades esenciales para un futuro Ingeniero en Informática. Por tanto, será necesario incidir en estas competencias transversales a lo largo de su formación.

Es muy importante que estos alumnos, de segundo (y primer) curso, tengan más información acerca de la titulación en la que se encuentran, puesto que rara vez sus expectativas corresponden con los objetivos de la titulación (con las consecuencias que este hecho conlleva). En las asignaturas de programación muchos alumnos se dan cuenta de que estas asignaturas no tienen ninguna relación con sus espectativas sobre la titulación de Ingenería Informática (normalmente son todo lo relacionado con internet, navegación web, chats, foros, juegos, etc.) y dado el trabajo que estas asignaturas conllevan, muchos de los alumnos piensan que no podrán superarlas. Es indispensable que los alumnos tengan una guía detallada de la universidad, de la titulación y de cada una de las materias, con sus objetivos, antes de realizar la matrícula

II. Objetivos

Relacionados con competencias académicas y disciplinares	*Vinculación*
Descripción	*CET*[i]
1. Aprender y comprender el concepto de eficiencia temporal y espacial de un algoritmo.	1, 2, 4, 6, 13, 14, 15, 18
2. Capacitar al alumno en el análisis de la eficiencia de algoritmos, tanto iterativos como recursivos.	1, 2, 4, 6, 13, 14, 15, 18
3. Capacitar al alumno en la especificación formal de tipos abstractos de datos utilizando notaciones formales	1, 2, 7, 8, 10
4. Capacitar al alumno en la verificación formal de algoritmos	6, 9, 14, 15, 18
5. Determinar las estructuras de datos adecuadas para almacenamiento de información en memoria principal de diferentes problemas	7
6. Conocer diferentes alternativas de implementación de estructuras de datos, razonando sobre la eficiencia espacial y temporal de tales estructuras	6, 7, 17
7. Familiarizar al alumno con un conjunto de tipos abstractos de datos de frecuente aplicación, capacitando al alumno para identificar y aplicar ese conjunto de TADs de manera adecuada en una aplicación concreta.	6, 7, 15
8. Conocer diversas técnicas de diseño de algoritmos y razonar sobre su aplicabilidad y adecuación a la hora de resolver un problema.	2, 6, 11, 14
9. Dotar al alumno de capacidad analítica para comparar distintas soluciones alternativas que permiten resolver un mismo problema.	1 - 18

Relacionados con otras competencias personales y profesionales	*Vinculación*
Descripción	*CET*
10. Desarrollar la capacidad de abstracción del alumno	1 - 18
11. Dotar al alumno de una sistemática de actuación ante el planteamiento de un determinado problema que se desee resolver	1 - 20

III. Contenidos

*Selección y estructuración de conocimientos generales**

1. Conceptos Fundamentales. Abstracciones

- Repaso de Programación Estructurada; Programación modular, Diseño Descendente y Refinamientos Sucesivos.
- Abstracción de datos. Motivación y conceptos. Tipos Abstractos de Datos

2. Complejidad de algoritmos iterativos

- Objetivo del análisis de algoritmos
- Análisis a priori y análisis a posteriori
- Eficiencia de un algoritmo: Eficiencia espacial y eficiencia temporal
- Noción de Complejidad
- Determinación del tiempo de ejecución de un algoritmo: casos peor, mejor, medio
- Medidas significativas de problemas comunes
- Reglas para el cálculo del tiempo de ejecución de un algoritmo
- Notación asintótica
  - Cota superior (O). Propiedades
  - Cota inferior (Ω). Propiedades
  - Orden exacto (Θ). Propiedades
- Utilidades y significado de las cotas superior e inferior
- Ordenes de Complejidad
- Medidas Frecuentes. Ejemplos
- Consideraciones sobre la eficiencia de un programa
- Tablas comparativas
- Análisis de algoritmos de ordenación
  - Determinación de las medidas significativas
  - Algoritmo de la burbuja
  - Algoritmo de inserción directa
  - Algoritmo de inserción binaria
  - Algoritmo de selección directa
  - Estudio comparativo de los distintos algoritmos. Tablas de tiempos de ejecución
- Análisis de algoritmos de búsqueda
  - Determinación de las medidas significativas
  - Algoritmo de búsqueda lineal
  - Algoritmo de búsqueda binaria
  - Determinación de valores máximos y mínimos en un conjunto de datos
- Ejercicios

3. Complejidad de algoritmos recursivos

- Introducción. Repaso del concepto de recursividad y terminación recursiva
- Diseño de problemas utilizando recursividad
  - El caso trivial
  - Condición de terminación
  - Convergencia de las llamadas recursivas al caso trivial

- Corrección de algoritmos recursivos. Inducción matemática
- Ventajas e inconvenientes de la recursividad
- Complejidad de algoritmos recursivos
- Ecuaciones de recurrencia
- Expansión y soluciones de ecuaciones de recurrencias
- Ejercicios

4. Verificación formal de programas

- Introducción
- Especificación formal con pre- y postcondiciones
- Verificación a posteriori
- Verificación formal de algoritmos iterativos
- Verificación formal de algoritmos recursivos

5. Especificaciones algebraicas

- Especificación algebraica de un TAD
- Notación de Guttag
- Algunas notas sobre la especificación algebraica de TADs
- Ejemplos
- TADs lineales
    - Listas
    - Implementación de estructuras de datos con punteros
    - Pilas (listas LIFO)
    - Colas (listas FIFO)

6. Árboles

- Descripción y terminología fundamental
- Especificación algebraica del TAD árbol binario
- Especificación de la clase árbol binario
- Ejemplos de uso
- Recorridos de árboles binarios
    - Implementaciones de la clase árbol binario
    - Implementación mediante tablas. Extensión a árboles de grado $k > 2$
    - Implementación mediante apuntadores. Extensión a árboles de grado $k > 2$
    - Comparación de las implementaciones: tiempo vs. espacio
- Ejemplo de aplicación y ejercicios
- Árboles binarios ordenados (o árboles de búsqueda)
    - Definición y propiedades
    - Especificación de la clase ABO
    - Ejemplos de uso
    - Implementación de la clase ABO
    - Propiedad de equilibrio
    - Definición
    - Compromisos y consecuencias del equilibrio
    - Ejemplo de aplicación y ejercicios
- Árboles AVL

- Árboles generales
  - Introducción
  - Consecuencias de representaciones basadas en la generalización de las representaciones de árboles binarios
  - Representación de árboles generales mediante árboles binarios
- Otros tipos de árboles: implementación y aplicaciones
- Ejemplo de aplicación y ejercicios

7. Tablas, conjuntos y colas de prioridad

- Tablas
  - Descripción
  - Especificaciones algebraicas
  - Implementaciones de tablas
- TAD Conjunto
  - Descripción
  - Especificación algebraica
  - Especificación de la clase Conjunto
  - Implementaciones de la clase Conjunto
- TAD Cola de prioridad
- Montículos

8. Grafos

- Introducción
- Definiciones, conceptos generales y terminología básica
- Especificación algebraica del TAD Grafo
- Especificación de la clase Grafo
- Implementaciones de la clase árbol grafo
  - La clase conjunto
  - Matriz de adyacencia
  - Listas de adyacencia
  - Comparación de las implementaciones: tiempo vs. espacio
- Recorridos de Grafos
  - Profundidad. Propiedades
  - Anchura. Propiedades
- Conectividad. Componentes conexas
- Algoritmos de búsqueda de caminos en grafos
  - Algoritmo de multiplicación de matrices
  - Algoritmo de Warshall
  - Algoritmo de Floyd
  - Algoritmo de Dijkstra
- Árboles de recubrimiento de coste mínimo. Algoritmos de Prim y Kruskal
- Ejercicios

9. Especificación, uso y diseño de TADs

- Ejercicios

10. Esquemas algorítmicos

- Presentación de las distintas técnicas de diseño de algoritmos
- Divide y vencerás
  - Introducción
  - Esquema General
  - Ejemplos de aplicación
    - Búsqueda binaria
    - Torres de Hanoi
    - Ordenación por mezcla
    - Otros casos de estudio
- Algoritmos voraces
  - Introducción
  - Esquema General
  - Ejemplos de aplicación
    - Algoritmo de Kruskal
    - Algoritmo de Dijkstra
    - Algoritmo de Prim
    - Otros casos de estudio
- Vuelta atrás (backtracking)
  - Introducción
  - Esquema general: una solución, todas las soluciones
  - Ejemplos de aplicación:
    - Problema de las n-reinas
    - Combinaciones en las quinielas
    - Problema del laberinto
    - Otros casos de estudio
- Ramificación y poda
  - Introducción
  - Esquema General
  - Ejemplos de aplicación
- Programación dinámica
  - Introducción
  - Elementos de programación Dinámica
  - Principio del óptimo
  - Redundancia en el cálculo de subproblemas: cálculo de los números de Fibonacci
  - Ejemplos de aplicación
    - Algoritmo de Floyd
    - Algoritmo de Warshall

Secuenciación de bloques temáticos y temas
A. ANÁLISIS DE ALGORITMOS
1. Conceptos fundamentales. Abstracciones
2. Complejidad de algoritmos iterativos
3. Complejidad de algoritmos recursivos
4. Verificación formal de algoritmos
B. ESTRUCTURAS DE DATOS
5. Especificaciones algebraicas
6. Árboles
7. Tablas, conjuntos y colas de prioridad
8. Grafos
9. Especificación, uso y diseño de TADs
C. DISEÑO DE ALGORITMOS
10. Esquemas algoritmicos

Interrelación			
Requisitos (Rq) y redundancias (Rd)		Tema	*Procedencia*
Conocimientos algorítmicos y de estructuras de datos en su implementación	Rq/Rd	Todos	Laboratorio de programación II (2º curso)
Conceptos básicos de programación, así como de algoritmia y estructuras de datos, sobre todo en lo que se refiere a implementaciones	Rq	Todos	Laboratorio de programación I (1er curso)
Conceptos básicos de programación, así como de algoritmia y estructuras de datos	Rq	Todos	Elementos de programación (1er curso)
Estructuras de datos avanzadas	Rd	Todos	Estructuras de almacenamiento de la información (3er curso)

IV. Metodología docente y plan de trabajo del estudiante

Actividades de enseñanza-aprendizaje				*Vinculación*	
Descripción y secuenciación de actividades	*Tipo*[ii]		*D*[iii]	*Tema*	*Objet.*
1. Presentación asignatura. Teoría de conceptos generales. Abstracciones	GG	T	3	1	3, 7, 10
2. Teoría de la complejidad de algoritmos iterativos	GG	T	7	1	1, 2, 6, 9, 11
3. Ejercicios de consolidación del tema teórico anterior	S	P	4	1	
4. Estudio posterior del tema teórico y resolución de problemas	NP	P	9	2	
5. Teoría de la complejidad de algoritmos recursivos	GG	T	6	2	1, 2, 6, 9, 11
6. Ejercicios de consolidación del tema teórico anterior	S	P	3	2	
7. Estudio posterior del tema teórico y resolución de problemas	NP	P	8	3	
8. Teoría sobre la verificación formal de algoritmos	GG	T	6	3	4, 10
9. Ejercicios de consolidación del tema teórico anterior	S	P	3	3	
10. Estudio posterior del tema teórico y resolución de problemas	NP	P	8	4	
11. Teoría sobre especificaciones algebraicas	GG	T	7	4	3, 7, 10
12. Ejercicios de consolidación del tema teórico anterior	S	P	3	4	
13. Estudio posterior del tema teórico y resolución de problemas	NP	P	10	5	
14. Teoría relativa a árboles	GG	T	7	5	5, 6, 7
15. Ejercicios de consolidación del tema teórico anterior	S	P	2	5	
16. Estudio posterior del tema teórico y resolución de problemas	NP	P	9	6	
17. Teoría de tablas, conjuntos y colas de prioridad	GG	T	5	6	5, 6, 7
18. Ejercicios de consolidación del tema teórico anterior	S	P	2	6	
19. Estudio posterior del tema teórico y resolución de problemas	NP	P	7	7	
20. Teoría de grafos	GG	T	8	7	5, 6, 7
21. Ejercicios de consolidación del tema teórico anterior	S	P	4	7	
22. Estudio posterior del tema teórico y resolución de problemas	NP	P	8	8	
23. Ejercicios sobre especificación, uso y diseño de TAD's	S	P	3	8	3, 7, 9, 10
24. Estudio posterior sobre los últimos ejercicios resueltos en clase	NP	P	5	8	
25. Teoría de esquemas algorítmicos	GG	T	7	9	8, 10, 11
26. Ejercicios de consolidación del tema teórico anterior	S	P	3	9	
27. Estudio posterior del tema teórico y resolución de problemas	NP	P	8	9	
28. Realización de problemas por parte del alumno	NP	P	10	Todos	
29. Problemas en general de todo el curso	S	P	4	Todos	
30. Preparación del examen final	NP	P	12	Todos	
31. Examen test teórico	GG	C-E	1	Todos	
32. Examen problemas	GG	C-E	3	Todos	

Distribución del tiempo (ECTS)			*Dedicación del alumno*		*Dedicación del profesor*	
Distribución de actividades		*Nº alumnos*	*H. presenciales*	*H. no presenc.*	*H. presenciales*	*H. no presenc.*
Grupo grande (Más de 20 alumnos)	Coordinac./evaluac. (I)	30	4	-	4	10
	Teóricas (II y III)	30	56	-	56	60
	Prácticas (IV, V y VI)	-	-	-	-	-
	Subtotal	30	60	-	60	70
Seminario-Laboratorio (6-20 alumnos)	Coordinac./evaluac. (I)	-	-	-	-	10
	Teóricas (II y III)	-	-	-	-	-
	Prácticas (IV, V y VI)	10	30	113	90	100
	Subtotal	10	30	113	90	110
Tutoría ECTS (1-5 alumnos)	Coordinac./evaluac. (I)	-	-	-	-	-
	Teóricas (II y III)	-	-	-	-	-
	Prácticas (IV, V y VI)	-	-	-	-	-
	Subtotal	-	-	-	-	-
Tutoría comp. y preparación de ex. (VII)		-	-	-	-	-
Totales			90	113	150	180

*Otras consideraciones metodológicas**

Recursos y metodología de trabajo en las actividades presenciales

Las actividades expositivas cumplen la función de vertebrar el proceso de aprendizaje del alumno durante el desarrollo de la asignatura, habitualmente esta actividad expositiva se acompaña de equipos electrónicos de visualización además del uso tradicional de la pizarra.

Recursos y metodología de trabajo en las actividades semi-presenciales y no presenciales

Los seminarios en pequeño grupo seguirán una metodología de *aprendizaje Basado en Problemas* basándose sobre todo en la discusión y debate por parte de los alumnos acerca de diferentes soluciones para un mismo problema.
Además se dispone del *Aula Virtual* donde también es posible abrir diferentes foros de discusión para diferentes problemas donde se aplican de forma práctica los conceptos teóricos de la asignatura.

Recursos y metodología de trabajo para los alumnos que no han alcanzado los requisitos

Los alumnos que no alcanzan los requisitos pueden someter a discusión las soluciones de problemas con el profesor utilizando las horas de tutorías tradicionales.

Recursos y metodología de trabajo para desarrollar competencias transversales

Considerando como competencias transversales, la habilidad para la toma de decisiones, o la capacidad de exponer ideas y soluciones por escrito, se pueden desarrollar en las actividades de grupo pequeño, tal y como se explica en el apartado de *metodología de trabajo en las actividades semi-presenciales y no presenciales.*

V. Evaluación

*Criterios de evaluación**	*Vinculación**	
Descripción	*Objetivos*	*CC*[iv]
1. Adquisición de conocimientos relacionados con la titulación y la asignatura	todos	40%
2. Resolución de problemas	todos	45%
3. Estructuración clara, concisa y estructurada de los ejercicios y trabajos a presentar	todos	10%
4. Participación activa en clase resolviendo problemas planteados	todos	5%

Actividades e instrumentos de evaluación		
Seminarios ECTS	▪ Valoración del trabajo continuo del alumno en las tutorías ▪ Resolución de problemas en el día a día, planteados durante las clases prácticas ▪ Participación activa en clase del alumno resolviendo problemas planteados ▪ Resolución de problemas por parte del alumno durante las clases o en sesiones específicas	25% NR
Examen teórico	▪ Una parte de teoría, que consistirá en un examen de tipo test. Para poder aprobar la asignatura habrá que obtener una puntuación mínima igual o superior al 40% de la puntuación máxima de ese test.	25% R
Examen práctico	▪ Una parte de problemas. Habrá que sacar una nota mínima de 3 puntos sobre 10 en cada problema para que se pueda calcular la nota media.	50% R

VI. Bibliografía

Bibliografía de apoyo seleccionada

[Aho 88] A. Aho; J. Hopcroft; J. Ullman. *Estructuras de datos y algoritmos*. Editorial Addison_Wesley Americana, 1988.

[Brassard 97] G. Brassard; P. Bratley. *Fundamentos de Algoritmia.* Prentice Hall, 1997.

[Franch 01] X. Franch Gutiérrez. *Estructuras de datos. Especificación, diseño e implementación, 3ª edición.* Edicions UPC, 2001.

[Horowitz 95] E. Horowitz; S. Sahni. *Fundamentals of data structures in C++*. Editorial Computer Science Press, 1995.

[Peña 98] R. Peña Martí. *Diseño de programas. Formalismo y abstracción. 2ª ed.* Prentice-Hall, 1998.

*Bibliografía o documentación de lectura obligatoria**

[Martí 04] Narciso Martí; Yolanda Ortega; José A. Verdejo. *Estructuras de datos y métodos algorítmicos. Ejercicios resueltos.* Editorial Pearson – Prentice Hall (Prentice Práctica), 2004.

[Guerequeta 00] R. Guerequeta; A. Vallecillo. *Técnicas de Diseño de Algoritmos, 2ª edición.* Servicio de Publicaciones de la Universidad de Málaga, 2000.
(http://polaris.lcc.uma.es/~av/Libro/).

*Bibliografía o documentación de ampliación, sitios web...**

Consultar la página web de la asignatura en AVUEX y en la web oficial de la Escuela Politécnica

---

Códigos del Plan Docente

i *CET*. Competencias Específicas del Título (véase el apartado de Contextualización curricular)

ii *Tipos de actividades*. GG (Grupo Grande); S (Seminario o Laboratorio); Tut (Tutoría ECTS); No presenciales (NP); C-E, I (Coordinación o evaluación); T, II (Teórica de carácter expositivo o de aprendizaje a partir de documentos); T, III (Teórica de discusión); P, IV (Prácticas basadas en la solución de problemas); P, V (Prácticas basadas en la observación, experimentación, aplicación de destrezas, estudio de casos...); P, VI (Prácticas con proyectos o trabajos dirigidos); T-P, VII (Otras teórico-prácticas).

iii *D.* Duración en sesiones de 1 hora de trabajo presencial o no presencial (considerando en cada hora 50-55 minutos de trabajo neto y 5-10 de descanso).

iv *CC: Criterios de Calificación* (ponderación del criterio de evaluación en la calificación cuantitativa final).
v *NR*: actividad "no recuperable" o que no permite evaluación extraordinaria.
(*) Apartados no obligatorios.