

Plan Docente de la Asignatura

Guía Docente

Curs acadèmic:	2008-09
Nom de l'assignatura:	Programació I
Codi assignatura:	12406
Estudis:	Enginyeria en Informàtica / Enginyeria tècnica en Informàtica de Sistemes
Crèdits:	5
Crèdits ECTS:	4
Hores de dedicació:	100
Professorat:	Jesús Ibáñez, Vanesa Daza, Amaury Hazan, Mar Pérez, Nir Lipovetzky
Grup:	1 i 2

Guía Docente

1. Datos descriptivos de la asignatura

- Curso académico: 2008-09
- Nombre de la asignatura: Programació I Código: 12406
- Tipo: Troncal
- Estudios: Ingeniería en Informàtica e Ingeniería Técnica en Informàtica de Sistemas
- Número de créditos: 5 Número ECTS: 4
- Número total de horas de dedicación: 100
- Temporalización:
 - o Curso: 1er curso
 - o Tipo: Trimestral
 - o Periodo: 1er Trimestre
- Coordinación: Jesús Ibáñez
- Profesorado: Jesús Ibáñez, Vanesa Daza, Amaury Hazan, Mar Pérez, Nir Lipovetzky
- Departamento: Technologies de la Informació i les Comunicacions
- Grupo: 1, 2
- Lengua de docencia: Catalán y castellano
- Edificio en que se imparte la asignatura: Edificio França
- Horario:
 - Grupo 1:
 - Lunes: 10:30-12:30
 - Martes: 12:30-14:30
 - Miércoles: 8:30-10:30
 - Grupo 2
 - Miércoles: 10:30-12:30
 - Jueves: 8:30-10:30
 - Viernes: 8:30-10:30

2. Presentación de la asignatura

Programación I está integrada en un bloque de asignaturas sobre algorítmica y programación que se llevan a cabo en el primer curso de los estudios, tanto de Ingeniería en Informática como de Ingeniería Técnica en Informática de Sistemas. Para esta primera asignatura del bloque se presupone que los estudiantes no tienen conocimientos previos sobre algorítmica y programación. En ella se establecen las bases de la algorítmica, de las estructuras de datos y de la programación en el lenguaje C. En las asignaturas de Programación II y Estructuras de Datos y de la Información I (segundo trimestre) y Programación III (tercer trimestre) se profundizará en las competencias aquí adquiridas, de forma que al final del primer curso el estudiante ha de ser capaz de desarrollar programas de tamaño considerable con las estructuras de datos adecuadas, tanto en forma imperativa como orientada a objetos. Así mismo, es importante destacar que las bases adquiridas en Programación I son imprescindibles para poder implementar la parte práctica de numerosas asignaturas a lo largo de los estudios.

Las actividades de aprendizaje se han dividido fundamentalmente en función de su tipología en varias categorías:

- sesiones de teoría: los profesores presentan una serie de conceptos y técnicas, además de ejemplos de su utilización. Los estudiantes deberán repasar fuera del aula los apuntes y sus notas para acabar de asimilar los contenidos.
- sesiones de ejercicios (seminarios): los estudiantes deben resolver una serie de ejercicios de pequeño tamaño, poniendo en práctica los conceptos y técnicas presentados en las sesiones de teoría. Estas actividades se comienzan en aulas de ordenadores, y deben completarse fuera del aula.
- sesiones de prácticas: los estudiantes deben resolver unos problemas de un tamaño mayor que los ejercicios antes citados, de forma que deben decidir qué conceptos y técnicas deben utilizar en cada caso. Estas actividades se comienzan en aulas de ordenadores, y deben completarse fuera del aula.
- ejercicios de autoevaluación: ejercicios para que el estudiante, al final de cada unidad didáctica, compruebe que ha asimilado los conceptos y técnicas presentados. Esta actividad la realiza el estudiante fuera del aula.

3. Prerrequisitos para el seguimiento del itinerario formativo

Esta asignatura no requiere ningún conocimiento previo de programación ni algorítmica. Para la realización de algunos ejercicios sí se requieren los conocimientos de matemáticas de nivel de bachillerato.

4. Competencias de la asignatura

El objetivo fundamental de esta asignatura es que los alumnos adquieran las bases de la algorítmica y estructuras de datos, así como que sean capaces de desarrollar fluidamente programas de tamaño medio utilizando el lenguaje C.

En este capítulo se detalla qué es lo que se espera que los estudiantes hayan aprendido al acabar la asignatura. En primer lugar, las competencias generales se refieren a habilidades no directamente relacionadas con la programación en sí, sino al ámbito profesional de un informático. Las competencias específicas son las referidas a aspectos propios de la asignatura.

4.1 Competencias generales

Instrumentales

CG1: Capacidad de síntesis

El estudiante ha de ser capaz de escribir soluciones con los elementos esenciales, de forma simple, elegante y lo más eficiente posible.

CG2: Capacidad de análisis

El estudiante ha de ser capaz de, a partir de un problema concreto, analizarlo y proponer soluciones adecuadas a dicho problema.

Sistémicas

CG3: Capacidad para aplicar el conocimiento en la práctica

El estudiante ha de ser capaz de aplicar los conocimientos adquiridos para resolver problemas concretos, eligiendo la técnica que mejor se ajuste a cada caso.

CG4: Interés por la calidad

El estudiante ha de ser capaz de que su código sea, además de eficiente, fácil de leer y mantener. Así mismo es importante que se documente correctamente, tanto dentro del propio código como en una memoria.

4.2 Competencias específicas

CE1: Capacidad para trabajar con un compilador y debugador

El estudiante ha de ser capaz de trabajar con las herramientas adecuadas para la programación: un compilador y un debugador. Esta competencia es primordial para el correcto desarrollo de las demás.

CE2: Dominio de los tipos de datos estáticos básicos y compuestos

El estudiante ha de ser capaz de distinguir los diferentes tipos de datos estáticos básicos y compuestos y de decidir el tipo adecuado para cada situación concreta.

CE3: Dominio de las estructuras de control

El estudiante ha de ser capaz de distinguir las diferentes estructuras de control compuestas y de decidir las más adecuadas para resolver problemas concretos.

CE4: Capacidad de resolución de problemas mediante diseño descendente y dominio de la utilización de funciones

El estudiante ha de ser capaz de resolver problemas de una complejidad considerable utilizando las técnicas del diseño descendente. En particular, el estudiante ha de comprender el funcionamiento de las llamadas a funciones y pasos de parámetros; y ser capaz de dividir un problema en las unidades adecuadas.

CE5: Dominio de los tipos de datos dinámicos y de la gestión dinámica de memoria

El estudiante ha de comprender el mecanismo de gestión de memoria, así como el uso de punteros y control dinámico de estructuras de datos. Se incluye también el manejo de ficheros de texto.

CE6: Documentación y estructuración de código

El estudiante ha de adquirir el hábito de estructurar y documentar el código de forma adecuada con la finalidad de facilitar su posterior lectura.

CE7: Capacidad de lectura (rápida) de código en C

El estudiante ha de ser capaz de comprender código escrito por otros programadores, de forma relativamente rápida.

5. Contenidos**Bloque 1: Introducción y conceptos generales**

Conceptos	Procedimientos
- Breve historia de la programación y sus lenguajes y paradigmas - Mecanismos de compilación e interpretación - Diferencia entre programa y algoritmo	

Bloque 2: Tipos de datos básicos

Conceptos	Procedimientos
- Variables y constantes - Los diferentes tipos de datos básicos: - o Tipos numéricos - o Caracteres - o Booleanos	- Declaración de constantes y variables de los diferentes tipos

Bloque 3: Expresiones, sentencias y estructuras de control

Conceptos	Procedimientos
- Formación de expresiones - Introducción a la lógica booleana - Sentencias o instrucciones: - o Asignaciones - o Operaciones de entrada y salida - o El orden de precedencia de los operadores - Las estructuras de control: - o Las estructuras condicionales - o Las estructuras iterativas	- Evaluación de expresiones - Resolución de pequeños problemas utilizando las sentencias y estructuras de control adecuadas

Bloque 4: La descomposición funcional y el diseño descendente

Conceptos	Procedimientos
- Diseño descendente - Declaración de funciones - Definición de parámetros - Llamadas a funciones y paso de parámetros - El tipo void	- Descomposición de problemas en subproblemas - Resolución de pequeños problemas definiendo funciones de forma adecuada

Bloque 5: Tipos de datos compuestos estáticos

Conceptos	Procedimientos
- Arrays unidimensionales - Arrays multidimensionales - Strings o cadenas de caracteres - Estructuras (structs)	- Resolución de pequeños problemas sobre las operaciones típicas con cada uno de los tipos de datos

Bloque 6: Declaración de tipos propios

Conceptos	Procedimientos
- Los tipos de las estructuras - Definición de tipos mediante typedef - Conversiones de tipos	- Resolución de pequeños problemas de definición de tipos de datos propios

Bloque 7: Los punteros y la gestión dinámica de memoria

Conceptos	Procedimientos
- Declaración de punteros - Operaciones de dirección e indirección - Asignación de valores a punteros - El puntero nulo - Paso de parámetros por referencia - Arrays y punteros en C	- Resolución de pequeños problemas sobre las operaciones típicas con punteros - Resolución de pequeños problemas típicos de arrays utilizando punteros

Bloque 8: La descomposición funcional y el diseño descendente (segunda parte)

Conceptos	Procedimientos
- Paso de parámetros por referencia - La función main y sus argumentos - Visibilidad y alcance - Las librerías y el pre-procesador	- Resolución de problemas mediante descomposición utilizando funciones con parámetros por referencia

Bloque 9: Ficheros de texto

Conceptos	Procedimientos
- El tipo fichero (FILE) - Operaciones con ficheros: - o Abrir - o Cerrar - o Escribir - o Leer - Entrada y salida estándar	- Resolución de pequeños problemas sobre las operaciones típicas con ficheros de texto

6. Metodología de aprendizaje

El proceso habitual de aprendizaje en una unidad didáctica comienza con una sesión de teoría en el que se presentan ciertos fundamentos teórico-prácticos. Esta actividad se realiza en un grupo de entre 60 y 90 estudiantes. El estudiante deberá posteriormente completar esta actividad con una lectura detenida de los apuntes. Por ejemplo, una sesión típica de teoría de 1,5 horas de duración, convenientemente aprovechadas, necesitará un trabajo personal fuera del aula de 0,5 horas.

A continuación se lleva a cabo una sesión de ejercicios, en los cuales el estudiante pone en práctica los conceptos y técnicas presentados en la sesión de teoría, mediante la implementación de programas para solucionar pequeños problemas. El objetivo es que el estudiante consolide los fundamentos para posteriormente poder llevar a cabo problemas de mayor tamaño. Esta actividad se realiza individualmente, en un grupo de alrededor de 15 estudiantes. Cada actividad de este tipo está programada con una duración de 4 horas, de las cuales 1 se realiza con soporte del profesor. No obstante, en función del número final de alumnos, dos grupos de estas sesiones pueden juntarse en uno solo, de forma que la sesión con soporte del profesor sea de 2 horas. Los primeros ejercicios de la sesión, para los cuales se ofrecen las soluciones, deberán resolverse antes de llegar al aula. El profesor puede solicitar la entrega de uno de los ejercicios planteados al final de cada sesión.

El siguiente paso es el de la sesión de prácticas. En ella se proponen unos problemas de mayor tamaño, especialmente en las prácticas parcial y final, que requieren un diseño previo de la solución a implementar y que deben integrar diferentes conceptos y técnicas. En la práctica final se reúnen todas las competencias específicas que el estudiante debe adquirir en esta asignatura. Esta actividad se realiza por parejas, en un grupo de alrededor de 30 estudiantes, y debe continuarse fuera del aula.

El último paso de la unidad didáctica es el de la resolución de unos ejercicios de autoevaluación mediante los cuales el estudiante puede comprobar si ha adquirido las competencias que posteriormente se evaluarán mediante los exámenes parcial y final.

Algunas unidades didácticas presentan varios ciclos como el anteriormente presentado, o tienen más de una sesión de prácticas o de ejercicios.

En cada sesión de ejercicios y de prácticas se dedica un tiempo para discutir los principales problemas que se han presentado en la sesión anterior.

6.1 Unidades didácticas

Unidad didáctica 1: Primeros pasos: introducción, tipos de datos básicos y estructuras de control

Bloques de contenido	Actividades de aprendizaje			
1: Introducción y conceptos generales	Sesiones de Teoría	Sesiones de Ejercicios	Sesiones de Prácticas	Autoevaluación
2: Tipos de datos básicos 3: Expresiones, sentencias y estructuras de control	T1, T2 y T3	E1	P1 y P2	A1

Dedicación total Unidad Didáctica 1: 17:30 horas (8:00 en el aula, 9:30 fuera)

Detalle de las actividades:

Sesión de teoría T1. 2 horas (1:30 en el aula, 0:30 fuera): breve historia de la programación y explicando los modelos de compilación e interpretación.

Sesiones de teoría T2 y T3. 4 horas (por cada sesión 1:30 en el aula, 0:30 fuera): explicación de los tipos de datos básicos y principales ejemplos de utilización. Explicación de la formación de expresiones y sentencias básicas con los principales ejemplos de su utilización. Estructuras de control condicionales e iterativas y principales ejemplos de su utilización.

Sesión de ejercicios E1. 4 horas (1:00 en el aula, 3:00 fuera): ejercicios sobre formación de expresiones y estructuras de control condicionales e iterativas.

Sesión de prácticas P1. 2 horas (1:00 en el aula, 1:00 fuera): se explicará cómo instalar el compilador, y cómo editar, compilar, debugar y ejecutar un programa. Se

dará a los alumnos pequeños programas con errores para que se habitúen a los típicos mensajes de error del compilador.

Sesión de prácticas P2. 4:30 horas (1:30 en el aula, 3:00 fuera): el estudiante ha de resolver (programando en C) unos pequeños problemas, utilizando las estructuras de control condicionales e iterativas.

Autoevaluación A1. 1 hora (fuera del aula): resolución de ejercicios de autoevaluación sobre formación de expresiones y estructuras de control

Unidad didáctica 2: Funciones y diseño descendente

Bloques de contenido	Actividades de aprendizaje			
4: La descomposición funcional y el diseño descendente	Sesiones de Teoría	Sesiones de Ejercicios	Sesiones de Prácticas	Autoevaluación
	T4	E2	P3	A2

Dedicación total Unidad Didáctica 2: 11:30 horas (4:00 en el aula, 7:30 fuera)

Detalle de las actividades:

Sesión de teoría T4. 2 horas (1:30 en el aula, 0:30 fuera): explicación de las bases del diseño descendente así como la definición de funciones y el mecanismo de llamada y paso de parámetros por valor, con ejemplos de utilización.

Sesión de ejercicios E2. 4 horas (1:00 en el aula, 3:00 fuera): ejercicios sobre diseño descendente y paso de parámetros por valor.

Sesión de prácticas P3. 4:30 horas (1:30 en el aula, 3:00 fuera): el estudiante ha de resolver (programando en C) unos pequeños problemas, utilizando convenientemente funciones y haciendo un pequeño diseño descendente.

Autoevaluación A2. 1 hora (fuera del aula): Resolución de ejercicios de autoevaluación sobre paso de parámetros y visibilidad.

Unidad didáctica 3: Tipos de datos compuestos estáticos y declaración de tipos

Bloques de contenido	Actividades de aprendizaje			
5: Tipos de datos compuestos estáticos 6: Declaración de tipos propios	Sesiones de Teoría	Sesiones de Ejercicios	Sesiones de Prácticas	Autoevaluación
	T5 y T6	E3 y E4	P parcial (P4 y P5) Revisión P parcial	A3

Dedicación total Unidad Didáctica 3: 24 horas (8 en el aula, 16 fuera)

Detalle de las actividades:

Sesiones de teoría T5 y T6. 4 horas (por cada sesión 1:30 en el aula, 0:30 fuera): explicación de los tipos de datos compuestos (arrays, cadenas y estructuras) con ejemplos de utilización. Explicación de la definición de tipos de datos propios por parte del programador, con ejemplos de utilización

Sesiones de ejercicios E3 y E4. 8 horas (por cada sesión 1 en el aula, 3 fuera): ejercicios sobre arrays, cadenas y estructuras, y de definición de tipos de datos propios
Práctica parcial. 11 horas (3 en el aula, 8 fuera): el estudiante ha de resolver (programando en C) unos pequeños problemas, utilizando arrays, cadenas y estructuras, en algunos casos definiendo sus propios tipos

Autoevaluación A3. 1 hora (fuera del aula): Resolución de ejercicios de autoevaluación sobre tipos de datos compuestos

Unidad didáctica 4: Punteros y paso de parámetros por referencia

Bloques de contenido	Actividades de aprendizaje			
7: Los punteros y la gestión dinámica de memoria	Sesiones de Teoría	Sesiones de Ejercicios	Sesiones de Prácticas	Autoevaluación
8: La descomposición funcional y el diseño descendente (segunda parte)	T7 y T8	E5 y E6	(*)	A4

Dedicación total Unidad Didáctica 4: 14 horas (6 en el aula, 8 fuera)

Detalle de las actividades:

Sesiones de teoría T7 y T8. 5 horas (por cada sesión 2:00 en el aula, 0:30 fuera): explicación del funcionamiento de los punteros con ejemplos de utilización. Explicación del paso de parámetros por referencia con ejemplos de utilización

Sesiones de ejercicios E5 y E6. 8 horas (por cada sesión 1:00 en el aula, 3:00 fuera): ejercicios sobre diseño descendente y paso de parámetros por valor. Incluye discusión sobre la mejor solución de diseño descendente en ciertos problemas

Autoevaluación A4. 1 hora (fuera del aula): Resolución de ejercicios de autoevaluación sobre punteros y paso de parámetros por referencia

(*) La parte práctica de esta unidad didáctica está incluida en la práctica final (ver unidad didáctica 5)

Unidad didáctica 5: Ficheros de texto y práctica final

Bloques de contenido	Actividades de aprendizaje			
9: Ficheros de texto	Sesiones de Teoría	Sesiones de Ejercicios	Sesiones de Prácticas	Autoevaluación
	T9, T10 y T11	E7	P final (P6, E8 y P7)	A5

Dedicación total Unidad Didáctica 5: 33 horas (10 en el aula, 23 fuera)

Detalle de las actividades:

Sesión de teoría T9. 2 horas (1:30 en el aula, 0:30 fuera): explicación de los ficheros de texto y ejemplos de utilización

Sesiones de teoría T10. 2 horas (en el aula): aspectos relacionados con la práctica final

Sesiones de teoría T11. 2 horas (1:30 en el aula, 0:30 fuera): repaso general y dudas

Sesión de ejercicios E7. 4 horas (1:00 en el aula, 3:00 fuera): Ejercicios sobre ficheros de texto

Práctica final. 20 horas (en total 4:00 en el aula, 18:00 fuera): el estudiante ha de resolver (programando en C) un problema de tamaño considerable que integre todos los conceptos y técnicas de la asignatura. Incluye la confección de una memoria explicativa de su trabajo. También se incluye un control de seguimiento por parte de los profesores (en el aula)

Autoevaluación A5. 1 hora (fuera del aula): Resolución de ejercicios de autoevaluación sobre ficheros

Dedicación total de la asignatura: 100 horas (36 en el aula, 64 fuera)

6.2 Organización temporal del trabajo en el aula

T: sesiones de teoría (*)

E: sesiones de ejercicios (*)

P: sesiones de prácticas (*)

(*) se incluye sólo el día en el que se debe llevar a cabo el trabajo en el aula, pero no el trabajo personal a realizar fuera

GRUPO 1

	Lunes 10:30-12:30	Martes 12:30-14:30	Miércoles 8:30-10:30
Semana 1			
Semana 2	T1 (1,5h)	T2 (1,5h)	P1 (1h)
Semana 3	T3 (1,5h)	E1 (1h)	P2 (1,5h)
Semana 4	T4 (1,5h)	E2 (1h)	P3 (1,5h)
Semana 5	T5 (1,5h)	E3 (1h)	T6 (1,5h)
Semana 6	E4 (1h)	P4 (P Parcial) (1,5)	P5 (P Parcial) (1,5h)
Semana 7	T7 (2h)	T8 (2h)	E5 (1h)
Semana 8	E6 (1h)	T9 (1,5h)	E7 (1h)
Semana 9	T10 (2h) (explicación P final, en 2 grupos)		P6 (P Final) (1,5h)
Semana 10	E8 (Seguimiento P Final) (1h)		P7 (P Final) (1,5h)
Semana 11	T11 (Repaso final) (1,5h)		

Entrega Práctica Parcial: 5 de noviembre

Entrega Práctica Final: 5 de diciembre

GRUPO 2

	Miércoles 10:30-12:30	Jueves 8:30-10:30	Viernes 8:30-10:30
Semana 1		T1 (1,5h)	T2 (1,5h)
Semana 2	P1 (1h)	T3 (1,5h)	E1 (1h)
Semana 3	P2 (1,5h)	T4 (1,5h)	E2 (1h)
Semana 4	P3 (1,5h)	T5 (1,5h)	E3 (1h)
Semana 5	T6 (1,5h)	E4 (1h)	P4 (P Parcial) (1,5)
Semana 6	P5 (P Parcial) (1,5h)	T7 (2h)	T8 (2h)
Semana 7	E5 (1h)	E6 (1h)	T9 (1,5h)
Semana 8	E7 (1h)		T10 (2h) (explicación P final, en 2 grupos)
Semana 9	P6 (P Final) (1,5h)		E8 (Seguimiento P Final) (1h)
Semana 10	P7 (P Final) (1,5h)	T11 (Repaso final) (1,5h)	
Semana 11			

Entrega Práctica Parcial: 5 de noviembre

Entrega Práctica Final: 5 de diciembre

7. Evaluación

7.1 Criterios de evaluación generales

Esta asignatura consta de dos partes principales:

- Fundamentos teórico-prácticos, que engloba las actividades relacionadas con las sesiones de teoría y de ejercicios
- Prácticas

Cada una supone un 50% de la nota final, aunque es necesario aprobar ambas para superar la asignatura.

Fundamentos teórico-prácticos

Habrán dos exámenes, uno a mitad de la asignatura y otro al final. El primero supone un 25% de la nota y se centra en el bloque de ejercicios realizados en las primeras sesiones de ejercicios. El segundo examen supone el 75% restante y abarca cuestiones teórico-prácticas de todo el conjunto de la asignatura.

En las sesiones de ejercicios los profesores irán revisando el trabajo de los estudiantes a lo largo del trimestre, y podrán pedir la entrega de un ejercicio al final de cada sesión. En función de dichas revisiones y entregas, se podrá subir (pero nunca bajar) la nota de la parte de fundamentos teórico-prácticos hasta 1 punto.

Además, se han programado varias actividades de auto-evaluación, una al final de cada unidad didáctica, con el objetivo de que el estudiante vaya valorando su propio progreso. Estas actividades constan de una serie de preguntas test de respuesta múltiple, de un nivel similar a las de los exámenes. Así mismo, en las actividades de ejercicios el estudiante puede también evaluar su progreso, comparando sus soluciones con los resultados publicados. En todo caso, estas actividades de auto-evaluación no inciden en la nota.

Prácticas

La nota de esta parte se obtiene a partir de dos prácticas entregables, una a mitad de la asignatura y otra al final. La práctica parcial tiene un peso del 25%, mientras que la final supone un 75%.

En cuanto a la nota de la práctica final, a parte de la valoración global, los profesores irán revisando el progreso en el trabajo de los estudiantes, a lo largo de las sesiones de prácticas y ejercicios dedicados a la práctica final. En el caso en que algún grupo no hiciese dicha revisión, o que ésta no fuese satisfactoria, tendrá que hacer una defensa de su práctica con posterioridad a su entrega, durante el período de exámenes.

A parte de las dos prácticas entregables, las otras prácticas servirán a los estudiantes para evaluar su propio progreso, comparando sus ejercicios con los resultados publicados. Estas actividades no inciden en la nota.

Recuperación de Septiembre

En el caso de no aprobar ambas partes en la convocatoria de diciembre, sólo se guardará para septiembre una parte aprobada en el caso de que la nota de la otra parte sea igual o superior al 3. En caso contrario, habrá que volver a examinarse y entregar una nueva práctica (con un enunciado diferente).

7.2 Concreción

Práctica parcial

Se evaluará la resolución de una práctica de tamaño reducido en la que el estudiante podrá mostrar el grado de asimilación de los conceptos y competencias adquiridos aproximadamente a la mitad de la asignatura. En particular se evaluarán las competencias CE2, CE3 y ciertos aspectos de la CE4 y la CE6 (estructuración de código comentado). La práctica consistirá en la resolución de un problema, incluyendo su análisis y la programación de una solución adecuada en lenguaje C. El estudiante deberá decidir las estructuras de datos estáticas así como el flujo de operaciones de la solución y realizar un sencillo diseño descendente. Así mismo se valorarán las cuatro competencias generales definidas.

Como se ha indicado anteriormente, parte de la nota de la práctica parcial vendrá de la evaluación anónima que los propios estudiantes realizarán de las prácticas de otros grupos.

Práctica final

Se evaluará la resolución de una práctica de un tamaño mayor al de la parcial en la que el estudiante podrá mostrar el grado de asimilación de los conceptos y competencias adquiridos durante toda la asignatura. En particular se evaluarán las competencias CE2, CE3, CE4, CE5 y CE6. La práctica consistirá en la resolución de un problema considerablemente complejo, incluyendo su análisis, la elección de las estructuras de datos (estáticas y dinámicas) adecuadas, el diseño descendente, la programación de una solución adecuada en lenguaje C y su correcta documentación en una memoria en la que también se justificarán las decisiones adoptadas. Así mismo se valorarán las cuatro competencias generales definidas.

Como se ha indicado anteriormente, a lo largo de las sesiones dedicadas a la práctica final, los profesores revisarán el trabajo de los estudiantes, especialmente en lo referente a las decisiones importantes sobre estructuración en funciones y estructuras de datos a utilizar. En el caso de no haber realizado dicha revisión, deberá hacerse a lo largo del período de exámenes.

Examen parcial

Se evaluará la comprensión y aplicación de los conceptos y técnicas adquiridos durante la primera mitad de la asignatura. En concreto se evaluarán las competencias CE2, CE3 y CE4. El método de evaluación consiste en un examen tipo test de alrededor de 10 preguntas con 4 opciones por cada pregunta con sólo una respuesta válida. Las preguntas estarán extraídas (con ligeras modificaciones) de las propuestas para las sesiones de ejercicios y de las actividades de auto-evaluación.

Examen final

Se evaluará la comprensión y aplicación de los conceptos adquiridos a lo largo de toda la asignatura en pequeños programas. En concreto se evaluarán las competencias CE1, CE2, CE3, CE4, CE5 y especialmente la CE7. El método de evaluación consiste en un examen tipo test de alrededor de 20 preguntas con 4 opciones por cada pregunta con sólo una respuesta válida. La evaluación tendrá lugar durante el período de exámenes.

8. Fuentes de información y recursos didácticos

8.1 Fuentes de información para el aprendizaje. Bibliografía básica

- Brian W. Kernighan, Dennis M. Ritchie: *El lenguaje de programación C*. Segunda edición. Prentice-Hall. ISBN: 968-880-205-0

8.2 Fuentes de información para el aprendizaje. Bibliografía complementaria

Otros libros sobre C:

- Herbert Schildt: *C Manual de referencia*. Mc Graw Hill. 84-481-0335-1
- James L. Antonakos, Kenneth C. Mansfield Jr.: *Programación estructurada en C*. Prentice-Hall. ISBN: 84-89660-23-9
- Marco A. Peña, José M. Cela: *Introducción a la programación en C*. Edicions UPF. ISBN: 84-8301-429-7
- Luis Joyanes, Ignacio Zahonero: *Programación en C*. Mc Graw Hill. ISBN: 84-481-3013-8
- Félix García, Jesús Carretero, Javier Fernández, Alejandro Calderón: *El lenguaje de programación C. Diseño e implementación de programas*. Prentice-Hall. ISBN: 84-205-3178-2
- P.J. Plauger, Jim Brodie: *C Estándar. Guía de referencia rápida para programadores*. Anaya. ISBN: 84-7614-264-1

Otros libros “clásicos” sobre algoritmia (más avanzados):

- Alfred V. Aho, Jeffrey D. Ullman, John E. Hopcroft: *Estructuras de datos y algoritmos*. Addison Wesley, 1988. ISBN: 968-444-345-5
- Niklaus Wirth: *Algoritmos + estructuras de datos = programas*. Ediciones del Castillo, 1980. ISBN: 84-219-0172-9
- Niklaus Wirth: *Algoritmia y estructuras de datos*. Prentice Hall, 1987. ISBN: 968-880-113-5
- D.E. Knuth: *El arte de programar ordenadores* (3 vols). Editoria Reverté. ISBN: 84-291-2661-9
- Terrence W. Pratt, Marvin V. Zelkowitz: *Lenguajes de programación. Diseño e implementación*. Prentice Hall, 3ª edic., 1997. ISBN: 970-17-0046-5
- G. Brassard, P. Bratley: *Fundamentos de algoritmia*. Prentice-Hall. ISBN: 84-89660-00-X

8.3 Recursos didácticos. Material docente de la asignatura

Web de la asignatura: <http://www.tecn.upf.es/~jibanez/programacio1/>

- o Apuntes
- o Enunciados para las sesiones de ejercicios
- o Enunciados para las sesiones de prácticas
- o Ejercicios de autoevaluación