



## PROGRAMACIÓN II

### Guía de Aprendizaje – Información al estudiante

#### 1. Datos Descriptivos

<b>Asignatura</b>	Programación II
<b>Materia</b>	Programación
<b>Departamento responsable</b>	DLSIIS
<b>Créditos ECTS</b>	6
<b>Carácter</b>	Obligatoria
<b>Titulación</b>	Graduado en Informática
<b>Curso</b>	2010-2011
<b>Especialidad</b>	No aplica

<b>Curso académico</b>	2010-2011
<b>Semestre en que se imparte</b>	Ambos (Septiembre a enero y febrero a junio)
<b>Semestre principal</b>	segundo
<b>Idioma en que se imparte</b>	castellano
<b>Página Web</b>	moodle



## 2. Profesorado

NOMBRE Y APELLIDO	DESPACHO	Correo electrónico
Ángel Lucas González (Coord.)	2310/CETTICO	agonzalez@fi.upm.es
Jaime Ramírez	5112	jramirez@fi.upm.es
Susana Muñoz	2310	susana@fi.upm.es
Javier Galve	2307	jgalve@fi.upm.es
Julio García	2306	juliog@fi.upm.es

## 3. Conocimientos previos requeridos para poder seguir con normalidad la asignatura

<b>Asignaturas superadas</b>	Programación I
<b>Otros resultados de aprendizaje necesarios</b>	Conocimientos de la sintaxis básica de Java: sentencias básicas, tipos básicos, sentencias de control de flujo y definición de funciones.

## 4. Objetivos de Aprendizaje

COMPETENCIAS ASIGNADAS A LA ASIGNATURA Y SU NIVEL DE ADQUISICIÓN		
Código	Competencia	Nivel
CE-4	Capacidad para describir una solución de forma abstracta.	3
CE-6	Comprender intelectualmente el papel central que tienen los algoritmos y las estructuras de datos, así como una apreciación del mismo;	1
CE-8	Poseer destrezas fundamentales de la programación que permitan la implementación de los algoritmos y las estructuras de datos en el software;	2
CE-9	Poseer las destrezas que se requieren para diseñar e implementar unidades estructurales mayores que utilizan los algoritmos y las estructuras de datos, así como las interfaces por las que se comunican estas unidades;	3

LEYENDA: Nivel de adquisición 1: Conocimiento  
Nivel de adquisición 2: Comprensión  
Nivel de adquisición 3: Aplicación  
Nivel de adquisición 4: Análisis y síntesis

<b>RESULTADOS DE APRENDIZAJE DE LA ASIGNATURA</b>			
<b>Código</b>	<b>Resultado de aprendizaje</b>	<b>Competencias asociadas</b>	<b>Nivel de adquisición</b>
RA1	Traducir especificaciones de tipos abstractos de datos (TADs) a implementaciones Java (p.ej.) correctas.	<b>CE-4, CE-6, CE-8, CE-9</b>	3
RA2	Programar aplicaciones mediante librerías existentes de TADs, iteradores, etc., extendiendo su funcionalidad (con herencia) o adaptándolas a un uso particular (instanciación de genéricos).	<b>CE-8, CE-9</b>	3
RA3	Documentar clases y bibliotecas, tanto de manera pública (hacia el cliente) como privada (hacia el implementador).	CE-4, CE-9	3
RA4	Realizar pruebas para asegurar el correcto funcionamiento de un TAD así como su integración en la aplicación que lo usa.	CE-8, CE-9	3

## 5. Sistema de evaluación de la asignatura

INDICADORES DE LOGRO		
Ref	Indicador	Relacionado con RA
I1	1. Ser capaz de entender el diseño de un TAD con vistas a implementarlo	RA1
I2	2. Ser capaz de especificar en lenguaje natural los contratos asociados a los servicios de un TAD	RA1
I3	3. Apreciar el papel que juega la abstracción y la modelización en el diseño de un TAD	RA1
I4	4. Ser capaz de implementar un TAD lineal (pila, cola, etc.) utilizando programación orientada a objetos.	RA1
I5	5. Ser capaz de implementar una estructura de datos dinámica lineal.	RA1
I6	6. Apreciar la importancia de separar la interfaz de la implementación en un diseño modular.	RA1
I7	7. Ser capaz de localizar y seleccionar las librerías más apropiadas para la aplicación que estamos desarrollando	RA2
I8	8. Ser capaz de implementar una clase extendiendo otra clase ya existente por medio del mecanismo de la herencia	RA2
I9	9. Ser capaz de utilizar una clase genérica en una aplicación	RA2
I10	10. Ser capaz de implementar una clase genérica que implemente un TAD dado	RA2
I11	11. Ser capaz de documentar adecuadamente la interfaz de un TAD, así como su implementación	RA2
I12	12. Ser capaz de utilizar una herramienta para la generación de la documentación del código fuente	RA3
I13	13. Saber utilizar un <i>framework</i> para la automatización de pruebas	RA3
I14	14. Entender el concepto de prueba software y su tipología	RA4
I15	15. Saber diseñar casos de prueba de forma que se asegure una cobertura razonable de los servicios de un TAD	RA4

INDICADORES DE LOGRO		
Ref	Indicador	Relacionado con RA
I16	16. Saber implementar programas que manejen excepciones	RA1

EVALUACION CONTINUA SUMATIVA			
Breve descripción de las actividades evaluables	Momento <sup>1</sup>	Lugar	Peso en la calif.
Examen Parcial 1 (test)	Semana 10	Aula	15%
Examen Parcial 2 (test)	Semana 16	Aula	15%
Ejercicio prácticos (problemas prácticos)		Sala Informática	20%
Proyecto 1	Semanas 9-13	Fuera de clase	25%
Proyecto 2	Semanas 14-16	Fuera de clase	25%
			<b>Total: 100%</b>

EVALUACION CON EXAMEN FINAL SUMATIVA			
Breve descripción de las actividades evaluables	Momento <sup>1</sup>	Lugar	Peso en la calif.
Examen Final (diversos ejercicios)	Fecha estipulada por el centro	Aula	50%
Proyecto 1	Semanas 9-13	Fuera de clase	25%
Proyecto 2	Semanas 14-16	Fuera de clase	25%
			<b>Total: 100%</b>

<sup>1</sup> Las fechas son aproximadas y por lo tanto orientativas

## CRITERIOS DE CALIFICACIÓN

Con el fin de superar esta asignatura, el alumno debe seguir uno de estos dos itinerarios alternativos: **basado en evaluación continua o basado en examen final**. El alumno elegirá al principio del curso el itinerario deseado y una vez hecha la elección no podrá cambiar de itinerario.

1. **Itinerario basado en evaluación continua:** se aplicará un esquema de **evaluación continua combinando exámenes parciales de tipo test, ejercicios prácticos que se realizarán en clase y proyectos** que se realizarán a lo largo del semestre. Siguiendo este esquema, la nota final (NF) de la asignatura se obtendrá a partir de una nota de teoría (NT), una nota de ejercicios prácticos (NE) y una nota de proyectos (NP) mediante la siguiente fórmula:

$$NF = 0.3NT + 0.5NP + 0.2NE, \text{ si } NP \geq 5 \text{ y } NT \geq 4 \text{ y } NE \geq 3$$
$$NF = 0, \text{ e.o.c.}$$

En donde:

**NT. Nota de teoría:** La nota NT será un valor numérico entre 0 y 10.

- Esta nota se obtiene mediante la realización de varios tests o parciales en el periodo lectivo.
- Para poder aprobar la NT durante el periodo lectivo, se deberá obtener al menos un 3 en cada test.
- **La nota de teoría sólo se guarda de un periodo a otro del mismo curso** si esta nota es  $\geq 5$ . En ningún caso las notas se guardarán de un curso a otro.

**NP. Nota de proyectos:** La nota NP será un valor numérico entre 0 y 10.

- Esta nota se obtiene de la media de las calificaciones de varios proyectos a realizar fuera del horario de clase.
- Será obligatorio **obtener al menos un 4 en cada uno de los proyectos** propuestos en un periodo para poder aprobar la NP.
- Los enunciados de los proyectos y sus respectivas fechas de entrega se publicarán durante el curso.
- El código fuente entregado debe compilar en la **versión 1.6 de java**.

**NE. Nota de ejercicios prácticos:** La nota NE será un valor numérico entre 0 y 10.

- Esta nota se obtiene de la media de las calificaciones de varios ejercicios prácticos.
- Los ejercicios prácticos solo se realizarán en las horas de clase estipuladas para ello. **Si un alumno no entrega al menos un 80% de estos ejercicios, no podrá aprobar la asignatura en este itinerario.**

2. **Itinerario basado en un examen final:** en este itinerario el alumno no estará obligado a asistir a clase, y por lo tanto **no tendrá que entregar los ejercicios prácticos que se propongan para ser realizados en horario de clase**. Además, **su nota de teoría la obtendrá mediante la realización de un solo examen final que cubrirá todo la materia que constará de diversos ejercicios teóricos y problemas prácticos**. Por otro lado, el alumno sí que estará obligado a aprobar los proyectos que se propongan para realizar fuera del horario de clase. Siguiendo este esquema, la nota final (NF) de la asignatura se obtendrá a partir de una nota de teoría (NT) y una nota de proyectos (NP) mediante la siguiente fórmula:

$$NF = 0.5NT + 0.5NP, \text{ si } NP \geq 5 \text{ y } NT \geq 4$$
$$NF = 0, \text{ e.o.c.}$$

El Sistema de evaluación mediante sólo prueba final sólo se ofrecerá si así lo exige la Normativa Reguladora de los Sistemas de Evaluación en la UPM que esté vigente en el curso académico 2010-2011, y el procedimiento para optar por este sistema estará sujeto a lo que establezca en su caso Jefatura de Estudios de conformidad con lo que estipule dicha Normativa.

En el caso de que el alumno no apruebe la asignatura siguiendo uno de estos dos itinerarios, dispondrá de la **convocatoria extraordinaria**. En esta convocatoria, **su nota final se obtendrá utilizando la misma fórmula que en el itinerario basado en un examen final**. Si el alumno ya hubiera aprobado (nota igual o superior a 5) los proyectos o la teoría en uno de los dos itinerarios anteriores, se le conservará dicha nota, y solo tendrá que aprobar en esta convocatoria la otra parte de la asignatura (teoría o proyectos) que tuviera suspenso.

## 6. Contenidos y Actividades de Aprendizaje

<b>CONTENIDOS ESPECÍFICOS</b>		
<b>Bloque / Tema / Capítulo</b>	<b>Apartado</b>	<b>Indicadores Relacionados</b>
<b>Tema 1: Introducción a la POO con el lenguaje java</b>	1.1 Definición de clases y objetos	I4, I5
	1.2 Programación modular: paquetes y visibilidad	I3, I4, I6, I7
	1.3 Manejo de excepciones	I16
	1.4 POO avanzada: herencia y genéricos	I7- I10
	1.5 Pruebas de Programas	I13-I15
	1.6 E/S en Java	I4, I7
<b>Tema 2: Tipos abstractos de datos lineales</b>	2.1 Concepto de TAD y Aplicación de los TADs para la resolución de problemas.	I2, I3, I7, I11, I12
	2.2 Diseño e Implementación de un TAD	I1-I5



## 7. Breve descripción de las modalidades organizativas utilizadas y de los métodos de enseñanza empleados

Tabla 7. Modalidades organizativas de la enseñanza








MODALIDADES ORGANIZATIVAS DE LA ENSEÑANZA		
Escenario	Modalidad	Finalidad
	Clases Teóricas	<i>Hablar a los estudiantes</i>
	Seminarios-Talleres	<i>Construir conocimiento a través de la interacción y la actividad de los estudiantes</i>
	Clases Prácticas	<i>Mostrar a los estudiantes cómo deben actuar</i>
	Prácticas Externas	<i>Completar la formación de los alumnos en un contexto profesional</i>
	Tutorías	<i>Atención personalizada a los estudiantes</i>
	Trabajo en grupo	<i>Hacer que los estudiantes aprendan entre ellos</i>
	Trabajo autónomo	<i>Desarrollar la capacidad de autoaprendizaje</i>

Tabla 5. Métodos de enseñanza

MÉTODOS DE ENSEÑANZA		
	Método	Finalidad
	Método Expositivo/Lección Magistral	Transmitir conocimientos y activar procesos cognitivos en el estudiante
	Estudio de Casos	Adquisición de aprendizajes mediante el análisis de casos reales o simulados
	Resolución de Ejercicios y Problemas	Ejercitar, ensayar y poner en práctica los conocimientos previos
	Aprendizaje Basado en Problemas (ABP)	Desarrollar aprendizajes activos a través de la resolución de problemas
	Aprendizaje orientado a Proyectos	Realización de un proyecto para la resolución de un problema, aplicando habilidades y conocimientos adquiridos
	Aprendizaje Cooperativo	Desarrollar aprendizajes activos y significativos de forma cooperativa
	Contrato de Aprendizaje	Desarrollar el aprendizaje autónomo

Se conoce como método expositivo "la presentación de un tema lógicamente estructurado con la finalidad de facilitar información organizada siguiendo criterios adecuados a la finalidad pretendida". Esta metodología -también conocida como lección (lecture)- se centra fundamentalmente en la exposición verbal por parte del profesor de los contenidos sobre la materia objeto de estudio. El término "lección magistral" se suele utilizar para denominar un tipo específico de lección impartida por un profesor en ocasiones especiales.

Análisis intensivo y completo de un hecho, problema o suceso real con la finalidad de conocerlo, interpretarlo, resolverlo, generar hipótesis, contrastar datos, reflexionar, completar conocimientos, diagnosticarlo y, en ocasiones, entrenarse en los posibles procedimientos alternativos de solución.

Situaciones en las que se solicita a los estudiantes que desarrollen las soluciones adecuadas o correctas mediante la ejercitación de rutinas, la aplicación de fórmulas o algoritmos, la aplicación de procedimientos de transformación de la información disponible y la interpretación de los resultados. Se suele utilizar como complemento de la lección magistral.

Método de enseñanza-aprendizaje cuyo punto de partida es un problema que, diseñado por el profesor, el estudiante ha de resolver para desarrollar determinadas competencias previamente definidas.

Método de enseñanza-aprendizaje en el que los estudiantes llevan a cabo la realización de un proyecto en un tiempo determinado para resolver un problema o abordar una tarea mediante la planificación, diseño y realización de una serie de actividades, y todo ello a partir del desarrollo y aplicación de aprendizajes adquiridos y del uso efectivo de recursos.

Enfoque interactivo de organización del trabajo en el aula en el cual los alumnos son responsables de su aprendizaje y del de sus compañeros en una estrategia de corresponsabilidad para alcanzar metas e incentivos grupales. Es tanto un método, a utilizar entre otros, como un enfoque global de la enseñanza, una filosofía.

Un acuerdo establecido entre el profesor y el estudiante para la consecución de unos aprendizajes a través de una propuesta de trabajo autónomo, con una supervisión por parte del profesor y durante un periodo determinado. En el contrato de aprendizaje es básico un acuerdo formalizado, una relación de contraprestación recíproca, una implicación personal y un marco temporal de ejecución.

## 8. Recursos didácticos

<b>RECURSOS DIDÁCTICOS</b>	
<b>BIBLIOGRAFÍA</b>	Material didáctico proporcionado por los profesores
	[Lewis et al 2006]: John Lewis, Joseph Chase Estructura de datos con Java. Diseño de estructuras y algoritmos Addison Wesley
	[Goodrich et al 2005]: M.T. Goodrich, R. Tamassia Data Structures and Algorithms in Java (4th Ed.) John Wiley and Sons. 2005.
	[Horstann et al 2006]: Horstmann Cay S., Gary Cornell Core Java 2 (J2SE 5.0) Volumen I-Fundamentos Prentice Hall, 2006
	[Sun Microsystems, 2006]: Sun Microsystems The Java Tutorials
[Fuertes et al 2007]: José Luis Fuertes, Ángel Lucas González Fundamentos de la programación en Java Koobeht, 2007	
	Sitio Moodle de la asignatura
<b>EQUIPAMIENTO</b>	Laboratorio (sala de ordenadores)
	Software: JDK SE 1.6, JUnit 4.5, Eclipse 3.5
	Aula XXXX



### 9. Cronograma de trabajo de la asignatura

Semana	Actividades en Aula	Actividades en Laboratorio	Trabajo Individual	Trabajo en Grupo	Actividades de Evaluación	Otros
1-3 (18 horas)		<ul style="list-style-type: none"> <li>Definición de Clases y Objetos. 12 horas</li> </ul>	<ul style="list-style-type: none"> <li>Estudio de la asignatura. 6 horas</li> </ul>			
4 (6 horas)		<ul style="list-style-type: none"> <li>Programación modular: paquetes y visibilidad. 4 horas</li> </ul>	<ul style="list-style-type: none"> <li>Estudio de la asignatura. 2 horas</li> </ul>			
5 (6 horas)		<ul style="list-style-type: none"> <li>Manejo de excepciones. 4 horas</li> </ul>	<ul style="list-style-type: none"> <li>Estudio de la asignatura. 2 horas</li> </ul>			
6-8 (22 horas)		<ul style="list-style-type: none"> <li>POO avanzada: herencia. 16 horas</li> </ul>	<ul style="list-style-type: none"> <li>Estudio de la asignatura. 6 horas</li> </ul>			
9 (11 horas)		<ul style="list-style-type: none"> <li>POO avanzada: genéricos</li> <li>Ejercicio del primer proyecto. 4 horas</li> </ul>	<ul style="list-style-type: none"> <li>Estudio de la asignatura. 2 horas</li> <li>Realización del proyecto 1. 5 horas</li> </ul>			
10 (16 horas)		<ul style="list-style-type: none"> <li>Pruebas de Programas. 4 horas</li> </ul>	<ul style="list-style-type: none"> <li>Estudio de la asignatura. 6 horas</li> <li>Realización del proyecto 1. 5 horas</li> </ul>		<ul style="list-style-type: none"> <li>Examen parcial 1. 1 hora</li> </ul>	
11-12 (22 horas)		<ul style="list-style-type: none"> <li>Concepto de TAD y Aplicación de los TADs para la resolución de problemas. 8 horas</li> </ul>	<ul style="list-style-type: none"> <li>Estudio de la asignatura. 4 horas</li> <li>Realización del proyecto 1. 10 horas</li> </ul>			



**POLITÉCNICA**



UNIVERSIDAD POLITÉCNICA DE MADRID  
**FACULTAD DE INFORMÁTICA**  
Campus de Montegancedo  
Boadilla del Monte. 28660 Madrid

13-15 (32 horas)		<ul style="list-style-type: none"><li>• Diseño e Implementación de un TAD. 12 horas</li></ul>	<ul style="list-style-type: none"><li>• Estudio de la asignatura. 6 horas</li><li>• Realización del proyecto 1 y 2. 14 horas</li></ul>			
16 (15 horas)		<ul style="list-style-type: none"><li>• E/S en Java. 4 horas</li></ul>	<ul style="list-style-type: none"><li>• Estudio de la asignatura. 6 horas</li><li>• Realización del proyecto 2. 4 horas</li></ul>		<ul style="list-style-type: none"><li>• Examen parcial 2. 1 hora</li></ul>	
TOTAL 140		<ul style="list-style-type: none"><li>• 64</li></ul>	<ul style="list-style-type: none"><li>• 74</li></ul>		<ul style="list-style-type: none"><li>• 2</li></ul>	

Nota: Para cada actividad se especifica la dedicación en horas que implica para el alumno. Esta distribución de esfuerzos debe entenderse para el "estudiante medio", por lo que si bien puede servir de orientación, no debe tomarse en ningún caso en sentido estricto a la hora de planificar su trabajo. Cada alumno deberá hacer su propia planificación para alcanzar los resultados de aprendizaje descritos en esta Guía y ajustar dicha planificación en un proceso iterativo en función de los resultados intermedios que vaya obteniendo.



**POLITÉCNICA**



UNIVERSIDAD POLITÉCNICA DE MADRID  
**FACULTAD DE INFORMÁTICA**  
Campus de Montegancedo  
Boadilla del Monte. 28660 Madrid