



Universidad
de Alcalá

GUÍA DOCENTE

DESARROLLO DEL SOFTWARE

Grado en Ingeniería de Computadores
Universidad de Alcalá

Curso Académico 2019/2020
Curso 1º – Cuatrimestre 2º

GUÍA DOCENTE

Nombre de la asignatura:	Desarrollo del software
Código:	590000
Titulación en la que se imparte:	Grado en Ingeniería de Computadores
Departamento:	Departamento Ciencias de la Computación
Carácter:	Básica
Créditos ECTS:	6
Curso y cuatrimestre:	1er. Curso / 2º Cuatrimestre
Profesorado:	Otón Tortosa, Salvador Consultar en la página web del departamento
Horario de Tutoría:	El horario de Tutorías se indicará el primer día de clase.
Idioma en el que se imparte:	Español

1. PRESENTACIÓN

La asignatura Desarrollo del Software pretende introducir a los alumnos en el paradigma de programación orientada a objetos. La asignatura presentará los conceptos teóricos asociados a este paradigma como clase, objeto, relaciones de herencia y asociación, clases abstractas y polimorfismo, etc., así como describir las técnicas de análisis y diseño orientado a objetos. Estos conocimientos teóricos se llevarán a la práctica mediante el lenguaje de programación orientado a objetos Java.

Prerrequisitos y Recomendaciones

Es obligatorio haber cursado la asignatura de Fundamentos de Programación.

1.b PRESENTATION

The Software Development subject aims to introduce students to the object-oriented programming paradigm. The subject will present the theoretical concepts associated with this paradigm as a class, object, inheritance and association relationships, abstract classes and polymorphism, etc., as well as describe the techniques of object-oriented analysis and design. This theoretical knowledge will be implemented through the Java object-oriented programming language.

2. COMPETENCIAS

Competencias generales:

CG4 Capacidad para definir, evaluar y seleccionar plataformas hardware y software para el desarrollo y la ejecución de sistemas, servicios y aplicaciones informáticas, de acuerdo con los conocimientos adquiridos según lo establecido en el apartado 5 de la resolución BOE-A-2009-12977.

CG5 Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad, de acuerdo con los conocimientos adquiridos según lo establecido en el apartado 5 de la resolución BOE-A-2009-12977.

CG6 Capacidad para concebir y desarrollar sistemas o arquitecturas informáticas centralizadas o distribuidas integrando hardware, software y redes de acuerdo con los conocimientos adquiridos según lo establecido en el apartado 5 de la resolución BOE-A-2009-12977.

CG8 Conocimiento de las materias básicas y tecnologías, que capaciten para el aprendizaje y desarrollo de nuevos métodos y tecnologías, así como las que les doten de una gran versatilidad para adaptarse a nuevas situaciones.

CG9 Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad. Capacidad para saber comunicar y transmitir los conocimientos, habilidades y destrezas de la profesión de Ingeniero Técnico en Informática.

Competencias específicas:

CIB4 Conocimientos básicos sobre el uso y programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación en ingeniería.

CIB5 Conocimiento de la estructura, organización, funcionamiento e interconexión de los sistemas informáticos, los fundamentos de su programación, y su aplicación para la resolución de problemas propios de la ingeniería.

Resultados de aprendizaje

Los resultados de aprendizaje esperados, determinados a partir de las competencias específicas incluidas en la memoria verificada de la titulación como competencias específicas, son los siguientes:

- RA1. Crear soluciones algorítmicas a problemas y ser capaz de representarla como programa orientado a objetos.
- RA2. Describir la evolución de los lenguajes de programación, los diferentes paradigmas disponibles hoy día y sus principales características.

- RA3. Expresar el concepto de tipo de dato y tipo abstracto de dato y ser capaz de identificar las características principales de un sistema de tipos. Interpretar el diseño modular y los conceptos cohesión y acoplamiento.
- RA4. Explicar los fundamentos de la orientación a objetos y ser capaz de identificar las diferencias entre la representación basada en objetos y los modelos de flujo de datos.
- RA5. Aplicar los fundamentos del paradigma orientado a objetos mediante un lenguaje de programación orientado a objetos utilizando un entorno de desarrollo.
- RA6. Describir técnicas y metodologías de desarrollo: especificación de requisitos, análisis, diseño, prueba y depuración de aplicaciones orientadas a objetos. Emplear lenguajes de modelado.
- RA7. Aplicar técnicas de diseño de interfaces gráficas de usuario para realizar el acceso a las aplicaciones orientadas a objetos.
- RA8. Desarrollar aplicaciones informáticas robustas utilizando diferentes estructuras de datos, aplicando algoritmos de ordenación y búsqueda sobre los mismos y realizando su persistencia.

3. CONTENIDOS

1. Introducción a los lenguajes de programación: Historia de los lenguajes de programación, breve presentación de los distintos paradigmas. Comparación entre intérpretes y compiladores; fases de la traducción de lenguajes; aspectos independientes y dependientes de la máquina. El concepto de máquina virtual, jerarquía de máquinas virtuales, lenguajes intermediarios.
2. Fundamentos de la programación orientada a objetos: Evolución hacia la programación orientada a objetos, crisis del software, calidad del software. Abstracción, tipos de datos, tipos abstractos de datos, diseño modular, reutilización. Diseño orientado a objetos, encapsulación y ocultación de información, separación entre comportamiento e implementación, clases, subclases y herencia, jerarquía de clases, clases abstractas, polimorfismo.
3. Sintaxis del lenguaje de programación orientado a objetos: Revisión de tipos de datos, operadores, estructuras de control, control de acceso, clases, atributos, métodos.
4. Análisis y diseño Orientado a Objetos: Identificación y modelado de clases y relaciones, lenguajes y notaciones de modelado, pruebas y documentación de programas.
5. Tratamiento de errores. Mecanismos de tratamiento y recuperación de errores en tiempo de ejecución, excepciones, robustez de programas.
6. Desarrollo de interfaces gráficas de usuario: Manejo de API, uso de API gráficas sencillas; diseño de GUI, programación dirigida por eventos, arquitectura MVC.
7. Estructuras de datos y librería de I/O: Implementación de estructuras de datos en el paradigma orientado a objetos. Algoritmos básicos (ordenación y búsqueda) sobre estas estructuras. Tratamiento de ficheros y técnicas de persistencia de objetos.

Bloques de contenido (se pueden especificar los temas si se considera necesario)	Total de clases, créditos u horas
Introducción a los lenguajes de programación Historia de los lenguajes de programación, breve presentación de los distintos paradigmas. Comparación entre intérpretes y compiladores; fases de la traducción de lenguajes; aspectos independientes y dependientes de la máquina. El concepto de máquina virtual, jerarquía de máquinas virtuales, lenguajes intermediarios.	2 horas
Fundamentos de la programación orientada a objetos. Evolución hacia la programación orientada a objetos, crisis del software, calidad del software. Abstracción, tipos de datos, tipos abstractos de datos, diseño modular, reutilización. Diseño orientado a objetos, encapsulación y ocultación de información, separación entre comportamiento e implementación, clases, subclasses y herencia, jerarquía de clases, clases abstractas, polimorfismo.	12 horas
Sintaxis del lenguaje de programación orientado a objetos: Revisión de tipos de datos, operadores, estructuras de control, control de acceso, clases, atributos, métodos.	8 horas
Análisis y diseño Orientado a Objetos Identificación y modelado de clases y relaciones, lenguajes y notaciones de modelado, pruebas y documentación de programas.	12 horas
Tratamiento de errores Mecanismos de tratamiento y recuperación de errores en tiempo de ejecución, excepciones, robustez de programas.	4 horas
Desarrollo de interfaces gráficas de usuario. Manejo de API, uso de API gráficas sencillas; diseño de GUI, programación dirigida por eventos, arquitectura MVC.	8 horas

Estructuras de datos y librería I/O: Implementación de estructuras de datos en el paradigma orientado a objetos. Algoritmos básicos (ordenación y búsqueda) sobre estas estructuras. Técnicas de persistencia de objetos. Tratamiento de ficheros y técnicas de persistencia de objetos.	10 horas
PECs	4 horas

4. METODOLOGÍAS DE ENSEÑANZA-APRENDIZAJE.- ACTIVIDADES FORMATIVAS

4.1. Distribución de créditos (especificar en horas)

Número de horas presenciales:	28 horas de teoría + 28 horas de laboratorio + 4 horas de examen de evaluación
Número de horas del trabajo propio del estudiante:	90 horas
Total horas	150 horas

4.2. Estrategias metodológicas, materiales y recursos didácticos

La asignatura Desarrollo de Software se organiza como una asignatura cuatrimestral de 6 ECTS (150 horas).

En la primera sesión presencial el profesor servirá cumplida información sobre la asignatura. El foro de comunicación habitual es el aula en la que se imparte la asignatura. En el caso de los materiales de estudio, tanto para las sesiones presenciales como para el trabajo individual, se dispondrá de la plataforma institucional accesible desde Aula Virtual (www.uah.es).

Esta plataforma será también un medio de comunicación entre los participantes en el proceso educativo, mejorando la coordinación, gestionando la entrega de tareas y permitiendo la difusión de información que le permita al alumno realizar un seguimiento del grado de consecución de los distintos hitos planteados para la asignatura. Junto con el correo electrónico se dispondrá un foro con distintas líneas de discusión, generales y grupales.

Además se dispone de una bibliografía de referencia para preparar cada uno de los bloques temáticos. Si se desea ampliar todavía más los conocimientos los profesores facilitarán más libros, revistas o páginas web que puedan ser de interés.

El profesor impartirá en las clases presenciales los conocimientos necesarios e instruirá a los alumnos sobre la adecuada dinámica para la correcta realización de los trabajos, tanto supervisados como autónomos.

Estrategias metodológicas:

Las primeras sesiones (clases) estarán dedicadas a la presentación de la asignatura, a la difusión de los conceptos básicos e imprescindibles para el correcto entendimiento de la materia y al establecimiento de las bases y normas de la dinámica de trabajo de los participantes en el proceso de evaluación continua.

Junto con sesiones basadas en clases teórico-prácticas, y con el fin de mejorar el grado de adquisición de competencias que los participantes en el proceso formativo muestren, en la asignatura se hará uso de una combinación de:

- Trabajo individual del alumno centrado en Resolución de Problemas.

- Trabajo en grupo del alumno, cuyos resultados son supervisados por el profesor.
- Clases prácticas, supervisadas por el profesor.
- Desarrollo autónomo de ejercicios prácticos.
- Laboratorios, supervisados por el profesor.
- Desarrollo autónomo de ejercicios prácticos en laboratorio.
- Resolución de problemas.
- Presentación de resultados de las actividades.

Durante el transcurso de la asignatura el alumno puede ser requerido para la entrega de una o varias memorias o para la participación en distintas actividades complementarias que relacionen los resultados obtenidos en otras actividades; o bien para completar una prueba escrita.

Naturalmente, se dispondrá de las tutorías de atención al alumno.

Materiales y recursos:

Todo el material docente generado al efecto por los profesores en el seno de la asignatura será distribuido desde la plataforma de apoyo a la docencia. Esta plataforma será también un medio para comunicar y evaluar el progreso de los alumnos ya que se permite el envío y evaluación de trabajos, también se servirá para la resolución de dudas mediante la utilización de una serie de foros.

Además se dispone de una bibliografía de referencia para preparar cada uno de los bloques temáticos. Si se desea ampliar todavía más los conocimientos los profesores podrán facilitar más libros, revistas o páginas web que puedan ser de interés.

5. EVALUACIÓN: Procedimientos, criterios de evaluación y de calificación

En la convocatoria ordinaria el método de evaluación por defecto es la “evaluación continua”, con características de evaluación formativa para servir de realimentación en el proceso de enseñanza-aprendizaje por parte del alumno.

Opcionalmente, y de manera justificada, el alumno podrá solicitar ante el director del centro la evaluación mediante prueba única, lo que deberá de ser solicitado por escrito y en los plazos reglamentados. Este método de evaluación consiste en un examen teórico y una práctica de laboratorio.

En la convocatoria extraordinaria la evaluación se basará en una prueba única, consistente en un examen teórico y una práctica de laboratorio, en la que se determinará el grado de dominio de las competencias de la asignatura.

Procedimientos de Evaluación

El rendimiento de los alumnos se evaluará atendiendo a los conocimientos y destrezas adquiridas. Los métodos a emplear serán: la resolución de casos prácticos y trabajos planteados, defensa pública de determinados trabajos, realización de una práctica final, así como la realización de exámenes para comprobar los conocimientos teóricos de la asignatura.

Para superar la asignatura se realizarán una serie de pruebas que demuestren que el alumno ha adquirido los resultados de aprendizaje establecidos. Se establecerán mecanismos adecuados para garantizar la viabilidad de la superación de las prácticas de laboratorio y de la teoría de la asignatura a aquellos alumnos que deban optar por la evaluación final o por la convocatoria extraordinaria.

En líneas generales se establecen los siguientes criterios de evaluación continua de la asignatura:

- La parte teórica se valorará en un 50% y la de prácticas de laboratorio en otro 50%, para obtener el 100% de la nota de la asignatura.
- Para la parte teórica se realizarán dos PEI (Prueba de Evaluación Intermedia) representando cada una el 50% de la nota total de teoría. (PEI1 – PEI2)
- Para la parte de prácticas de laboratorio se realizarán tres PL (Prueba de Laboratorio), teniendo un peso del 20%, 30% y el 50% (PL1 – PL2 – PL3) respectivamente, aunque este peso puede variar.

Como se ha especificado anteriormente los alumnos que deban realizar la prueba única consistirá en un examen teórico y una práctica de laboratorio teniendo cada uno de ellos un 50% de peso.

Se establecen los siguientes criterios de evaluación específicos para la asignatura:

- CE1. El alumno conoce la historia y las características de los lenguajes de programación y es capaz de describir las diferencias entre los distintos paradigmas de programación.
- CE2. El alumno comprende la programación modular y conoce los criterios y principios del diseño modular.
- CE3. El alumno conoce los principios en los que se basa la orientación a objetos como abstracción, encapsulación, ocultamiento de la información, extensibilidad y reutilización.
- CE4. El alumno ha adquirido los conocimientos básicos del paradigma orientado a objetos tales como clase y objeto.
- CE5. El alumno ha adquirido los conocimientos avanzados del paradigma orientado a objetos tales como herencia, interface, clase abstracta y polimorfismo.
- CE6. El alumno sabe aplicar técnicas de análisis y diseño orientado a objetos expresarlo en un lenguaje de modelado como UML.
- CE7. El alumno sabe programar en un lenguaje orientado a objetos aplicando las técnicas del paradigma orientado a objetos y utilizar una herramienta de desarrollo.
- CE8. El alumno sabe desarrollar una aplicación orientada a objetos mediante una interfaz de usuario gráfica.
- CE9. El alumno es capaz de desarrollar una aplicación orientada a objetos robusta en la que se utilicen estructuras de datos complejas y persistencia.

En las siguientes tablas se indica el porcentaje en la calificación (entre 0 y 100) de cada prueba, y su relación con los criterios de evaluación, resultados de aprendizaje y competencias generales.

Convocatoria ordinaria:

Competencias	Resultado Aprendizaje	Criterio de Evaluación	Instrumento de Evaluación	Porcentaje
CG4, CG5, CG6, CG8, CG9. CIB4, CIB5	RA1, RA2, RA3, RA4, RA5	CE1, CE2, CE3, CE4, CE5	PEI1	25
	RA5, RA6, RA7, RA8	CE6, CE7, CE8, CE9	PEI2	25
	RA5	CE7	PL1	15
	RA5, RA6, RA7, RA8	CE6, CE7, CE8, CE9	PL2	35

Convocatoria final y extraordinaria: En esta convocatoria se realizará una prueba teórica (PEF) y otra práctica (PLF).

Competencias	Resultado Aprendizaje	Criterio de Evaluación	Instrumento de Evaluación	Porcentaje
CG4, CG5, CG6, CG8, CG9. CIB4, CIB5	RA1, RA2, RA3, RA4, RA5, RA6, RA7, RA8	CE1, CE2, CE3, CE4, CE5, CE6, CE7, CE8, CE9	PEF	50
	RA5, RA6, RA7, RA8	CE6, CE7, CE8, CE9	PLF	50

6. BIBLIOGRAFÍA

Bibliografía Básica

- **Conceptos y fundamentos de orientación a objetos**
 - Meyer, B. Construcción de Software Orientado a Objetos. 2ª Edición. Prentice Hall, 1998.
 - Muñoz, C., Niño, A., Vizcaíno A. Introducción a la programación con orientación a objetos. Prentice Hall, 2003.
 - Amescua, A. Análisis y diseño estructurado y orientado a objetos de sistemas informáticos. McGraw-Hill, 2003.
 - Stevens, P., Pooley, R. Utilización de UML en ingeniería del software con objetos y componentes. Prentice Hall, 2002.

- **Programación orientada a objetos en Java**
 - Arnow, D., Weiss, G. Introducción a la programación con Java. Un enfoque orientado a objetos. Addison Wesley, 2001.
 - Jiménez, A., Pérez, F. M. Aprende a programar con Java, 2ª edición. Paraninfo, 2016.
 - Eckel, B. Piensa en Java. Cuarta Edición. Prentice Hall, 2007.
 - Cadenhead, R. Programación Java 8. Anaya, 2014.
 - Deitel, P., Deitel, H. Cómo programar en Java. 9ª edición. Pearson, 2012. 10ª edición actualizada a Java 8, 2015.

Bibliografía Complementaria

- Perez, J. M. Problemas resueltos de programación en lenguaje Java. Thomson, 2003.
- Camacho, D. Programación, algoritmos y ejercicios resueltos en Java. Prentice Hall, 2003.
- Joyanes, L. Programación orientada a objetos. Segunda Edición. McGraw-Hill, 1998.
- Martin, R. UML para programadores Java. Prentice Hall, 2004.