GRADO

GUÍA DE ESTUDIO DE LA ASIGNATURA PROGRAMACIÓN ORIENTADA A OBJETOS

2ª PARTE | PLAN DE TRABAJO Y ORIENTACIONES PARA SU DESARROLLO

1.- Plan de Trabajo

La asignatura de Programación Orientada a Objetos requiere un estudio metódico y continuado a lo largo del cuatrimestre, dado que han de asimilarse los conceptos teóricos y prácticos generales relacionados la programación orientada a objetos. Para facilitar su superación es conveniente planificar las etapas de estudio desde el principio, teniendo en cuenta los plazos de entrega y dedicando semanalmente el tiempo necesario, ya que es difícil asimilar la asignatura si se deja el trabajo para el final del cuatrimestre.

El objetivo de esta asignatura es que el estudiante adquiera los conocimientos y competencias reflejados en la guía general de la asignatura. Programación Orientada a Objetos tiene asignados 6 ETCS (créditos europeos), siendo estos créditos 4 de teoría y 2 de práctica. Un crédito equivale a 25 horas, lo que implica unas 150 horas de estudio y trabajo en total a lo largo de las 13 semanas disponibles para el curso.

Los créditos asignados están en consonancia con los contenidos, distribuidos en dos unidades didácticas de siete y seis temas respectivamente, tal y como se detalla en el programa de la asignatura. La organización de los estudios teóricos se ajusta al desarrollo de la práctica . Además, el libro base está orientado a la aplicación práctica de los conceptos durante todo el proceso de estudio. Por ello es importante abordar la parte teórica y práctica en paralelo siguiendo en la medida de lo posible el cronograma propuesto.

El cuadro siguiente muestra el cronograma que marca unas pautas adecuadas para que el alumno medio, que comienza a estudiar al principio del segundo semestre, alcance los objetivos al final del curso. Este cronograma incluye los contenidos de cada tema y las actividades a realizar, tanto de estudio teórico como práctico.

Contenidos	*Plan de actividades*
Unidad Didáctica I: 7 Semanas	
Tema 1: Objetos y clases. Semana 1	
1- Estudiar el capítulo 1 y el Apéndice B del libro base de la Unidad Didáctica I 2- Instalar el software BlueJ y realizar los ejercicios sugeridos en el libro base 3- Definir las clases necesarias en el problema de la práctica	
Definiciones de clases. Semana 2	
1- Estudiar el capítulo 2 y leer los apéndices B, C y D del libro base para la Unidad Didáctica I 2- Realizar los ejercicios en el entorno BlueJ sugeridos en el libro base	
Tema 3. Interacción de objetos. Semana 3	
1- Estudiar el capítulo 3 del libro base para la Unidad Didáctica I 2- Leer el apéndice B del libro base para la Unidad Didáctica I. 3- Realizar los ejercicios en el entorno BlueJ sugeridos en el libro base 4- Definir los campos y los métodos necesarios para la resolución de la práctica.	

Tema 4. Agrupación de objetos. Semana 4
1- Estudiar el capítulo 4 del libro base para la Unidad Didáctica I 2- Realizar los ejercicios en el entorno BlueJ sugeridos en el libro base 3- Definición de estructuras de almacenamiento e implementación de los métodos y constructores necesarios para la realización de la práctica
Tema 5. Comportamientos más sofisticados. Semana 5
1- Estudiar el capítulo 5 del libro base para la Unidad Didáctica I 2- Leer el Apéndice I del libro base para la Unidad Didáctica I 4- Realizar los ejercicios en el entorno BlueJ sugeridos en el libro base 3- Incluir la documentación en el código de la práctica y encapsular los campos
Tema 6. Diseño de clases. Semana 6
1- Estudiar el capítulo 6 del libro base para la Unidad Didáctica I 2- Leer el Apéndice E del libro base para la Unidad Didáctica I 3- Realizar los ejercicios en el entorno BlueJ sugeridos en el libro base 4- Revisar el acoplamiento y cohesión de la solución propuesta en la práctica
Tema 7. Objetos con buen comportamiento. Semana 7
1- Estudiar el capítulo 7 del libro base para la Unidad Didáctica I 2- Leer los Apéndices G y H del libro base para la Unidad Didáctica I 3- Definir la estrategia de prueba a utilizar en la práctica
Tema 8: Mejora de la estructura mediante la herencia. Semana 8
1- Estudiar el capítulo 8 del libro base para la “Unidad Didáctica II”. 2- Realizar los ejercicios correspondientes del libro base. 3- Realizar los ejercicios resueltos en examines de años anteriores en los que se utilice la herencia.
Tema 9: Más acerca de la herencia. Semana 9
1- Estudiar el capítulo 9 del libro base para la “Unidad Didáctica II”. 2- Realizar los ejercicios correspondientes del libro base. 3- Realizar los ejercicios resueltos en examines de años anteriores en los que se utilice la herencia.
Tema 10: Técnicas de abstracción adicionales. Semana 10
1- Estudiar el capítulo 10 del libro base para la “Unidad Didáctica II”. 2- Realizar los ejercicios correspondientes del libro base. 3- Realizar los ejercicios resueltos en examines de años anteriores en los que se utilice la herencia.
Tema 11: Construcción de interfaces gráficas de usuario. Semana 11
1- Estudiar el capítulo 11 del libro base para la “Unidad Didáctica II”. 2- Realizar los ejercicios correspondientes del libro base. 3- Realizar los ejercicios resueltos en examines de años anteriores en los que se utilice la herencia.
Tema 12: Tratamiento de errores. Semana 12
1- Estudiar el capítulo 12 del libro base para la “Unidad Didáctica II”. 2- Realizar los ejercicios correspondientes del libro base. 3- Incorporar el control de errores y excepciones en la práctica.

Tema 13: Diseño de aplicaciones. Semana 12
1- Estudiar el capítulo 13 del libro base para la “Unidad Didáctica II”. 2- Realizar los ejercicios correspondientes del libro base. 3- Realizar los ejercicios resueltos en examines de años anteriores en los que se utilice la herencia.

2.- Orientaciones para el estudio de los contenidos

El temario se estructura en dos Unidades Didácticas y ha sido planteado de tal forma que el alumno pueda introducirse en los contenidos de la asignatura de una manera gradual, adquiriendo los conocimientos necesarios para comprender y aplicar programación orientada a objetos.

2.1. Unidad didáctica I. Fundamentos de programación orientada a objetos

2.1.1. Introducción

Esta primera unidad didáctica introduce al alumno en los fundamentos de la programación orientada a objetos, con el objetivo de que éste adquiera la base necesaria para realizar programas sencillos con una metodología apropiada. Esta unidad tiene una duración aproximada de 7 semanas.

Los contenidos de esta unidad didáctica se pueden estructurar en cuatro partes que, a su vez, establecen el orden en el que deberían estudiarse los diferentes conceptos teóricos presentes en esta unidad.

El primer objetivo de la unidad didáctica será fijar los conceptos básicos de la programación orientada a objetos, para lo que utilizaremos el lenguaje Java como lenguaje de programación y el entorno de desarrollo BlueJ como herramienta de trabajo. Se comenzará por el estudio y comprensión de las características fundamentales de clase, atributo y método (constructores, métodos para la encapsulación de una clase y métodos para la visualización). Esta parte de la unidad didáctica se corresponde con los temas 1, 2 y 3 del programa detallado de la asignatura.

En una segunda parte se estudiará cómo utilizar las diferentes estructuras de control, que permiten controlar el flujo de ejecución de un programa, sobre estructuras de almacenamiento estáticas y dinámicas. El estudio de estos contenidos requiere haber estudiado la primera parte de esta unidad didáctica y se corresponde con el tema 4 del programa detallado.

Finalmente, en una tercera parte de la unidad didáctica se profundizará en el diseño sofisticado de clases, así como en las técnicas para la producción de clases correctas, comprensibles y mantenibles, así como en las técnicas y estrategias para

probar, depurar, inspeccionar y corregir errores. Esta última parte de la unidad didáctica se corresponde con los temas 5, 6 y 7 del programa detallado y para su correcta comprensión es necesario haber estudiado las dos partes anteriores.

2.1.2. Descripción detallada de los contenidos

A continuación hacemos una breve descripción de cada uno de los temas:

- **Tema 1**. Se presentan los conceptos más fundamentales de la orientación a objetos (objetos, clases y métodos). Ofrece una introducción sólida y práctica de estos conceptos sin entrar en los detalles de la sintaxis Java, aunque presenta un primer acercamiento al código.
- **Tema 2**. Se centra en las definiciones de clase y presenta la sintaxis Java para crear el comportamiento de los objetos. Se describe como definir campos y como se implementan los métodos. En paralelo, en este tema se comienzan a introducir algunas sentencias Java.
- **Tema 3**. Se amplían los conceptos presentados en los temas anteriores al introducir la interacción entre objetos. Se profundiza en la colaboración entre objetos para conseguir un objetivo común. También se discute como un objeto puede crear otros objetos.
- **Tema 4**. Se presentan las colecciones de objetos como estructuras de datos útiles en Java. Se desarrollan las técnicas para llevar a cabo recorrido sobre las colecciones haciendo uso de las sentencias de control de flujo de programa en Java.
- **Tema 5**. Se ocupa de las bibliotecas de clases y las interfaces. En concreto se presenta la biblioteca estándar de Java, donde se discuten algunas de sus clases más relevantes. Así mismo, este tema pretende mostrar al alumno el modo en el que toda esta documentación de referencia puede ser accedida y consultada.
- **Tema 6**. Se discuten cuestiones relacionadas con la división del problema a tratar con el fin de obtener un conjunto de clases adecuado que facilite su implementación. Se presentan técnicas orientadas a un diseño de clases de buena calidad que incluyen conceptos tales como el diseño dirigido por responsabilidades, acoplamiento, cohesión y refactorización.
- **Tema 7**. Se presentan las cuestiones relacionadas con la producción de clases correctas, comprensibles y mantenibles. Se cubren cuestiones tales como la escritura de código claro y legible de probar y de depurar, haciendo hincapié en cuestiones de estilo y en los comentarios. También se introducen estrategias de prueba y se discuten varios métodos de depuración.

2.1.3. Contextualización

En el contexto de la asignatura, los objetivos de esta unidad didáctica son:

1- Comprender el concepto del paradigma orientado a objetos (POO) y sus características.
2- Profundizar en la programación orientada a objetos como el paradigma más importante hoy en día destacando sus características, conceptos y mecanismos básicos tales como las clases, objetos, relaciones entre clases, etc.
3- Entender los aspectos centrales de la estructura de un programa desde la perspectiva de orientación a objetos.
4- Presentar Java como el lenguaje orientado a objetos más utilizado e introducir al estudiante en los aspectos principales del lenguaje y en cómo programar sirviéndose de éstos.
5- Conocer la existencia de algoritmos, estructuras de datos, instrucciones y operadores.
6- Entender las orientaciones definidas en la ingeniería del software y conocer la guía de estilo propia del lenguaje Java.
7- Saber detectar y solucionar los posibles errores que pudieran existir.

2.1.4. Resultados de aprendizaje asociados a los contenidos

Los resultados de aprendizaje para cada uno de los temas de esta unidad didáctica son los siguientes:

Tema 1:

1- Asimilar los conceptos generales asociados a la programación en lenguaje Java. Los objetos y las clases.
2- Ser capaz de instanciar objetos de una clase e invocar métodos mediante el interfaz BlueJ.
3- Ser capaz de hacer pequeñas modificaciones en un código Java.

Tema 2:

1- Entender desde el código fuente los componentes de una clase: campos, constructores y métodos.
2- Manejar los operadores de asignación.
3- Implementar métodos sencillos.

Tema 3:

1- Manejar operadores lógicos y matemáticos sobre tipos primitivos.
2- Ser capaz de implementar la creación de un objeto desde otra clase y llamar a sus métodos.
3- Ser capaz de realizar un diagrama de clases sencillo.

Tema 4:

1- Implementar en Java un recorrido sobre estructuras de almacenamiento del tipo colección.
2- Entender los conceptos de biblioteca, paquete y objeto anónimo.

Tema 5:

1- Documentar una clase correctamente utilizando JavaDoc como estándar.
2- Ser capaz de distinguir entre campos que deban ser privados o públicos en función del problema, así como detectar aquellas situaciones en las que una clase deba definirse como estática.

Tema 6:

1- Ser capaz de definir un esquema de clases dado un problema, evitando el acoplamiento y la duplicación de código.

Tema 7:

1- Ser capaz de detectar y entender los errores dentro de un programa Java.
2- Ser capaz de hacer uso de un depurador de errores y de planificar pruebas en un entorno Java.

2.1.5. Bibliografía básica

Los contenidos correspondientes a la Unidad Didáctica I se pueden encontrar en el libro:

Programación orienta a objetos con Java. Una introducción práctica usando BlueJ. David J. Barnes y Michael Kölling. Pearson / Prentice Hall. 5ª edición. (Los capítulos que hay que estudiar se detallan en esta guía de estudio).

2.2. Unidad didáctica II. Estructuras de las Aplicaciones

2.2.1. Introducción

En una primera parte se estudian los conceptos de composición y herencia, que nos permiten, respectivamente, incluir referencias a objetos dentro de otros objetos y reutilizar una clase existente extendiendo su funcionalidad. Esta etapa requiere haber estudiado los contenidos de la primera parte de la Unidad Didáctica I y se corresponde con los temas 8, 9 y 10 del programa detallado de la asignatura.

En la segunda parte de esta Unidad Didáctica II se estudiarán aspectos relativos al manejo de excepciones y la construcción de interfaces gráficas de usuario, cuyo fin es la detección y corrección de errores y el desarrollo de aplicaciones aplicando sus principales etapas de desarrollo. Además, se tratará la implementación de un proceso completo de desarrollo software y la generación de interfaces gráficas.

2.2.2. Descripción detallada de los contenidos

A continuación hacemos una breve descripción de cada uno de los temas:

- **Tema 8**. Este tema está previsto para una semana. También desde el punto de vista de la arquitectura del sistema, en este tema se introducen las herramientas necesarias para asegurar la posibilidad de reutilizar el código implementado ante nuevas especificaciones del problema, centrándose en el concepto de herencia.
- **Tema 9**. Este tema está previsto para una semana. Se introducen los principales conceptos de la herencia. La herencia es central para comprender y usar lenguajes orientados a objetos, y para poder progresar a partir de aquí, es necesario comprender este tema con cierto nivel de detalle. En este tema además, se hace uso de un ejemplo para explorar las cuestiones más importantes que nos restan ver sobre herencia y polimorfismo.
- **Tema 10**. Este tema está previsto para una semana. En este tema se examinan otras técnicas relacionadas con la herencia, que se pueden usar para perfeccionar las estructuras de clases y mejorar la mantenibilidad y la extensibilidad. Estas técnicas introducen un mejor método de representación de las abstracciones en los programas orientados a objetos.
- **Tema 11**. Este tema está previsto para una semana. Las interfaces gráficas de usuario (IGU) se construyen a partir de objetos que interactúan, pero tienen una estructura muy especializada y es por esto que evitamos introducirlas antes de aprender la estructura de los objetos en términos más generales. Sin embargo, en este tema el alumno está preparado para dar una mirada a la construcción de las IGU. El objetivo principal es poder desarrollar programas que tengan una mejor representación visual.
- **Tema 12**. Se presentan los principales aspectos del manejo de errores mediante el lanzamiento y captura de excepciones. En este tema se pretende ver cómo anticiparse y responder a las posibles situaciones de error que pueden surgir durante la ejecución de un programa. También se realiza una breve introducción sobre los procesos de entrada y salida de texto como una de las situaciones en la que pueden aparecer fácilmente errores durante el tratamiento de los archivos.
- **Tema 13**. En este tema se presentan los pasos iníciales de un sistema de software, conocidos generalmente como las etapas de análisis y diseño. El primer paso de diseño de un sistema corresponde a un nivel mayor de abstracción con respecto al diseño de clases que fueron tratados en el tema 6.

2.2.3. Contextualización

En el contexto de la asignatura, los objetivos de esta unidad didáctica son:

1- Entender los aspectos centrales de la estructura de un programa.
2- Comprender el concepto el paradigma orientado a objetos (POO) y destacar las características de cada uno.
3- Profundizar en la POO como el paradigma más importante hoy en día, y destacar sus características, conceptos y mecanismos básicos: clases, objetos, relaciones entre clases...
4- Presentar Java como el LPOO más utilizado e introducir al estudiante en los aspectos principales del lenguaje y en cómo programar sirviéndose de éstos.
5- Conocer la existencia de algoritmos, estructuras de datos, instrucciones y operadores.
6- Entender las orientaciones definidas en la ingeniería del software y conocer la guía de estilo propia del lenguaje Java.
7- Conocer en qué consisten los posibles errores que pueden existir.

2.2.4. Resultados de aprendizaje asociados a los contenidos

Los resultados de aprendizaje para cada uno de los temas de esta unidad didáctica son los siguientes:

Tema 8:

1- Ser capaz de definir las relaciones de herencia entre clases dado un problema.
2- Ser capaz de extender una clase de una librería para crear una clase nueva.

Tema 9:

1- Ser capaz de utilizar polimorfismo dentro de una clase.
2- Ser capaz de aplicar tipos estáticos y dinámicos.
3- Ser capaz de emplear correctamente la sobreescritura.

Tema 10:

1- Ser capaz de representar abstracciones en programas orientados a objetos.
2- Ser capaz de escribir la interfaz de una clase.

Tema 11:

1- Ser capaz de desarrollar una interfaz gráfica para que una aplicación tenga una apariencia más similar a las típicas aplicaciones que se usan hoy en día.

Tema 12:

1- Ser capaz de implementar una excepción sencilla en código Java.

Tema 13:

1- Ser capaz de decidir, en un proyecto real de software, cuales son las clases que se necesitan para implementar la solución de un problema.
2- Ser capaz de aplicar las etapas de análisis y diseño en el desarrollo de un sistema de software.

2.2.5. Bibliografía básica

Los contenidos correspondientes a la Unidad Didáctica II se pueden encontrar en el libro:

Programación orienta a objetos con Java. Una introducción práctica usando BlueJ. David J. Barnes y Michael Kölling. Pearson / Prentice Hall. 2007. (Los capítulos que hay que estudiar se detallan en esta guía de estudio).

3.- Orientaciones para la realización del plan de actividades

Un aspecto importante de la metodología es aplicar los conocimientos adquiridos por medio de los ejercicios propuestos en el libro base a realizar sobre un entorno Java, y la realización de forma progresiva de la práctica a medida que se vayan introduciendo los conceptos teóricos.
Aunque los ejercicios del libro no están resueltos, el alumno puede solicitar ayuda al equipo docente a través de los canales establecidos en el curso virtual, y detallados en la guía el curso.

3.1. Medios y Recursos

Los medios y recursos de los que dispone el alumno están escritos en la primera parte de la guía del curso (secciones 11 y 12).

3.2. Evaluación

La evaluación de la asignatura se llevará a cabo a partir de las siguientes pruebas:

- Realización de una práctica obligatoria a lo largo el cuatrimestre.
- Realización de un examen teórico/práctico.

3.2.1. Práctica de la Asignatura

El trabajo del curso incluye la realización de una práctica obligatoria de programación. El enunciado de la misma estará disponible tanto en el curso virtual de la asignatura, como en la página web en abierto, en el departamento de Lenguajes y Sistemas Informáticos http://www.lsi.uned.es/ensenanzas-oficiales.php

Las prácticas son corregidas por los Tutores de los Centros Asociados, quienes las reenvían después a la sede central. La nota asignada por el tutor podrá incrementar

hasta un máximo de 1 punto en la nota final de la asignatura, por supuesto, siempre que esté aprobada. Las notas de las prácticas no se guardan de un curso para otro.

Las prácticas, por tanto, se organizan en los centros asociados bajo la responsabilidad de cada tutor, por lo que los alumnos deben ponerse en contacto con ellos lo antes posible al comienzo el curso para conocer el calendario de entrega y sesiones presenciales e las prácticas y la forma de entrega (correo electrónico, disquete, CD, curso virtual, etc.)

El programa editor recomendado es BlueJ, conjuntamente con el compilador incluido en el JDK. Esto quiere decir que los tutores encargados de las clases de prácticas sólo están obligados a dar soporte para un entorno de esas características. Una versión para instalar y un pequeño manual de instalación se encuentran disponibles tanto en la página web de la asignatura como en el entorno virtual. Por tanto el uso de otros entornos no garantiza al alumno ningún tipo de soporte en la instalación, configuración o ejecución tanto del entorno como de los programas desarrollados.

Por otro lado, el uso de una versión u otra del JDK puede afectar a la compilación de los programas, ya que de una versión a otra se incluyen novedades, por lo que es importante especificar la versión del JDK que se usó para la realización de la práctica, o usar la especificada por el equipo docente en caso de que la haya.

Por último, los programas deben compilar independientemente del entorno de edición que haya sido usado, debiendo el alumno especificar claramente los pasos a seguir para el correcto funcionamiento de las aplicaciones generadas en las prácticas. De no cumplirse este requisito y en caso de no poder ejecutar una práctica, ésta se considerará suspensa. Cualquier copia en las prácticas dará lugar a un suspenso para todo el curso académico.

3.2.2. Examen

El examen constará de dos partes, una teórica formada por preguntas tipo test y que será eliminatoria, y una segunda parte práctica formada por un problema de programación con varios apartados y en los que el alumno demostrará el nivel de los conocimientos adquiridos. Se incluirán también preguntas sobre la práctica obligatoria.

Para que el examen de un alumno sea calificado deberá haber asistido a dos sesiones presenciales de prácticas en su centro asociado y haber entregado y aprobado la práctica obligatoria.

Para que un alumno pueda aprobar la asignatura deberá haber superado un mínimo de preguntas establecido en la parte teórica (tipo test) del examen.

En la evaluación de la asignatura se tendrán en cuenta especialmente los aspectos relativos al diseño más que a los detalles propios de la implementación.