PLAN DOCENTE DE LA ASIGNATURA

Curso académico: 2026/2027

Identificación y características de la asignatura					
Código	501275				
Denominación (español)	Desarrollo de Programas				
Denominación (inglés)	Program Development				
Titulaciones	Grado en Ingeniería Informática en Ingeniería de Computadores Grado en Ingeniería Informática en Ingeniería del Software Doble Grado en ADE / Ingeniería Informática en Ingeniería de Software Doble Grado en ADE / Ingeniería Informática en Ingeniería de Computadores				
Centro	Escuela Politécnica				
Módulo	Común a la rama de Informática				
Materia	Programación				
Carácter	Obligatoria	ECTS	6	Semestre	3

Profesorado		
Nombre	Despacho	Correo-e
José Mª Conejero Manzano	9 (Pabellón Telecomunicaciones)	chemacm@unex.es
María Encarnación Sosa Sánchez	41 (Pabellón Informática)	esosa@unex.es

Área de conocimiento	Lenguajes y Sistemas Informáticos
Departamento	Ingeniería de Sistemas Informáticos y Telemáticos
Profesor/a coordinador/a (si hay más de uno)	María Encarnación Sosa Sánchez

Competencias

1. Competencias básicas establecidas para Grado en el Anexo I 3.2 del RD 861/2010:
CB1: Que los estudiantes hayan demostrado poseer y comprender conocimientos en un área de estudio que parte de la base de la educación secundaria general, y se suele encontrar a un nivel que, si bien se apoya en libros de texto avanzados, incluye también algunos aspectos que implican conocimientos procedentes de la vanguardia de su campo de estudio.
CB2: Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio.
CB3: Que los estudiantes tengan la capacidad de reunir e interpretar datos relevantes (normalmente dentro de su área de estudio) para emitir juicios que incluyan una reflexión sobre temas relevantes de índole social, científica o ética.
CB4: Que los estudiantes puedan transmitir información, ideas, problemas y soluciones a un público tanto especializado como no especializado.

CB5: Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía.

2. Competencias generales:

CG01 - Capacidad para concebir, redactar, organizar, planificar, desarrollar y firmar proyectos en el ámbito de la ingeniería en informática que tengan por objeto, de acuerdo con los conocimientos adquiridos según lo establecido en el apartado 5 del anexo II de la resolución antes mencionada para la tecnología específica de Ingeniería del Software y de Ingeniería de Computadores, la concepción, el desarrollo o la explotación de sistemas, servicios y aplicaciones informáticas.

CG02 - Capacidad para dirigir las actividades objeto de los proyectos del ámbito de la Informática de acuerdo con los conocimientos adquiridos según lo establecido en el apartado 5 del anexo II de la resolución antes mencionada para la tecnología específica de Ingeniería del Software y de Ingeniería de Computadores.

CG03 - Capacidad para diseñar, desarrollar, evaluar y asegurar la accesibilidad, ergonomía, usabilidad y seguridad de los sistemas, servicios y aplicaciones informáticas, así como de la información que gestionan.

CG04 - 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 del anexo II de la resolución antes mencionada para la tecnología específica de Ingeniería del Software y de Ingeniería de Computadores.

CG05 - 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 del anexo II de la resolución antes mencionada para la tecnología específica de Ingeniería del Software y de Ingeniería de Computadores.

CG08 - 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.

CG09 - 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/a Técnico/a en Informática.

CG10 - Conocimientos para la realización de mediciones, cálculos, valoraciones, tasaciones, peritaciones, estudios, informes, planificación de tareas y otros trabajos análogos de informática, de acuerdo con los conocimientos adquiridos según lo establecido en el apartado 5 del anexo II de la resolución antes mencionada para la tecnología específica de Ingeniería del Software y de Ingeniería de Computadores.

3. Competencias específicas:

CI07: Conocimiento, diseño y utilización de forma eficiente de los tipos y estructuras de datos más adecuados a la resolución de un problema.

CI08: Capacidad para analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.

4. Competencias transversales:

CT03: Capacidad para resolver problemas.

CT07: Capacidad de análisis y síntesis.

Contenidos

Descripción general del contenido:

Fundamentos teóricos de la programación orientada a objetos y de los lenguajes de programación. Uso de lenguajes orientados a objetos para el desarrollo de sistemas software. Estudio de estructuras de datos, sus aplicaciones y propiedades. Aplicación de técnicas de verificación y validación para garantizar la corrección y calidad de los programas.

Denominación del tema 1: **Clases y Objetos.**

Este tema introduce los conceptos básicos de la programación orientada a objetos, incluyendo la definición de clases, creación de objetos, campos, métodos, constructores e instancias. También se estudian aspectos fundamentales como el ámbito y la vida útil de las variables, junto con nociones básicas del lenguaje como tipos primitivos y operadores.

Contenidos del tema 1:

1. Clases y Objetos
2. Campos y Estado
3. Métodos y Parámetros
4. Constructores
5. Instancias y Variables
6. Ámbito y vida útil
7. Métodos de acceso y mutadores
8. Anexos:
   - 8.1. Tipos de datos primitivos
   - 8.2. Operadores
   - 8.3. Sobrecarga de operadores

Descripción de las actividades prácticas del tema 1:

- Explicación de la metodología que se seguirá en las sesiones de laboratorio.
- Ejercicios de toma de contacto con el IDE (*Entorno de Desarrollo Integrado*) que usaremos en la asignatura.

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 2: **Interacción de objetos.**

En este tema se profundiza en la creación e interacción entre objetos, abordando la modularización, la abstracción, los diagramas de clases y objetos, y la sobrecarga de métodos y constructores. Además, se estudian conceptos como la ocultación de información, el uso de bibliotecas de clases, la gestión de memoria y el paso de parámetros.

Contenidos del tema 2:

1. Modularización y abstracción
2. Tipos de datos: Primitivos y Objetos
3. Diagrama de clase
4. Creación de objetos
5. Diagrama de objetos
6. Sobrecarga de constructores
7. Sobrecarga de métodos
8. Interacción de objetos
9. Ocultando información
10. Valores nulos
    - 10.1. Valores por defecto
11. Gestión interna de la memoria
    - 11.1. Tipos de datos primitivos
    - 11.2. Tipos de datos objeto
    - 11.3. Liberación de memoria
    - 11.4. Paso de parámetros: tipos primitivos vs objetos.

12. Bibliotecas de Clases
13. Documentación de clases
14. Anexos
    14.1. Depurar programas

Descripción de las actividades prácticas del tema 2:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 3: **Buen comportamiento de los objetos.**

Este tema se centra en la prevención y detección de errores durante el desarrollo de programas. Se introducen las pruebas de software, la automatización de pruebas unitarias y el uso de técnicas de depuración para mejorar la fiabilidad del código.

Contenidos del tema 3:

1. Prevención y detección de errores
2. Pruebas (testing)
    2.1. Automatización de pruebas unitarias
3. Depuración (debugging)

Descripción de las actividades prácticas del tema 3:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 4: **Agrupando y ordenando objetos.**

Se estudia el uso de colecciones de tipo **ArrayList** para almacenar, organizar y procesar conjuntos de objetos. También se trabajan técnicas de iteración, ordenación y procesamiento funcional de colecciones, junto con estructuras como objetos anónimos y arrays bidimensionales.

Contenidos del tema 4:

1. Colecciones
2. Iteración
3. Orden en colecciones de elementos
4. Procesamiento funcional de colecciones
5. Anexos
    5.1. Objetos Anónimos
    5.2. Arrays bidimensionales

Descripción de las actividades prácticas del tema 4:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 5: **Herencia.**

Este tema introduce la herencia como mecanismo para reutilizar y extender código mediante jerarquías de clases. Se abordan conceptos como el polimorfismo, las clases y métodos abstractos, la extensión de jerarquías y las limitaciones relacionadas con la herencia múltiple.

Contenidos del tema 5:

1. Herencia
2. Polimorfismo
3. Clases y métodos abstractos
4. Extendiendo jerarquías de clases
5. Herencia múltiple

Descripción de las actividades prácticas del tema 5:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 6: **Comportamientos más avanzados.**

Se presentan elementos avanzados del lenguaje que permiten mejorar la organización y expresividad de los programas. Entre ellos se incluyen colecciones, clases envoltorio, variables y métodos de clase, constantes y tipos enumerados.

Contenidos del tema 6:

1. Colecciones de elementos
2. Clases wrapper (envoltorio).

3. Variables y métodos de clase.
4. Constantes
5. Tipos enumerados

Descripción de las actividades prácticas del tema 6:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 7: **Manejo de errores.**

Este tema aborda la programación defensiva y las técnicas para anticipar, detectar e informar errores durante la ejecución de un programa. Se estudia el lanzamiento y manejo de excepciones como mecanismo fundamental para construir aplicaciones más robustas.

Contenidos del tema 7:

1. Programación Defensiva.
   1.1. Anticipar posibles errores
2. Lanzamiento y manejo de Excepciones.
3. Informar de errores.

Descripción de las actividades prácticas del tema 7:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 8: **Manejo de ficheros (Entrada/Salida).**

Se estudian los mecanismos de entrada y salida de datos mediante ficheros. El tema incluye la lectura y escritura de archivos, así como la serialización de objetos para almacenar y recuperar información de forma persistente.

Contenidos del tema 8:

1. Entrada y salida de datos usando ficheros
2. Escritura en ficheros
3. Lectura de ficheros
4. Serialización de objetos

Descripción de las actividades prácticas del tema 8:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 9: **Mejorando la calidad del código.**

Este tema se centra en criterios de calidad del diseño software, como el acoplamiento, la cohesión y el diseño dirigido por responsabilidad. También se introduce la refactorización como técnica para mejorar la estructura interna del código sin modificar su comportamiento externo.

Contenidos del tema 9:

1. Acoplamiento
2. Cohesión
3. Diseño dirigido por responsabilidad
4. Acoplamiento implícito
5. Refactorización
6. Formas de Cohesión

Descripción de las actividades prácticas del tema 9:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 10: **Principios y Patrones de Diseño.**

Se estudian las características de un buen diseño software y los principales principios que ayudan a construir sistemas mantenibles y extensibles. El tema incluye los principios SOLID y una introducción a los patrones de diseño, su clasificación y ejemplos representativos de algunos patrones de diseño.

Contenidos del tema 10:

1. Características de un Buen Diseño
2. Principios de Diseño
3. Principios SOLID
4. Patrones de Diseño
   4.1. Clasificación de los Patrones

4.2. Ejemplos de Patrones

Descripción de las actividades prácticas del tema 10:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Denominación del tema 11: **Antipatrones de Diseño.**

Este tema presenta prácticas de diseño inadecuadas que pueden afectar negativamente a la calidad, mantenimiento y evolución del software. Se estudia la clasificación de los antipatrones y se analizan ejemplos frecuentes para aprender a identificarlos y evitarlos.

Contenidos del tema 11:

1. Clasificación de los Antipatrones
2. Ejemplos de AntiPatrones

Descripción de las actividades prácticas del tema 11:

Resolución de problemas que impliquen entender y aplicar los contenidos tratados.

Actividades formativas

Horas de trabajo del alumno/a por tema		Horas Gran grupo	Actividades prácticas				Actividad de seguimiento	No presencial
Tema	**Total**	**GG**	**CH**	**L**	**O**	**S**	**TP**	**EP**
1	8	3		2				3
2	9	3		2				4
3	7	1		2				4
4	14	4		4				6
5	22	4		4			1	14
6	8	2		2				4
7	5	1		2				2
8	5	1		2				2
9	8	2		2				4
10	29	6		6			1	16
11	2	1		0				1
Evaluación	33	2		2				28
TOTAL	150	30		30			2	88

GG: Grupo Grande (85 estudiantes).
CH: Actividades de prácticas clínicas hospitalarias (7 estudiantes)
L: Actividades de laboratorio o prácticas de campo (15 estudiantes)
O: Actividades en sala de ordenadores o laboratorio de idiomas (20 estudiantes)
S: Actividades de seminario o de problemas en clase (40 estudiantes).
TP: Tutorías Programadas (seguimiento docente, tipo tutorías ECTS).
EP: Estudio personal, trabajos individuales o en grupo, y lectura de bibliografía.

Metodologías docentes[6]

- En Clases teórico-prácticas en el aula:
  - clases expositivas para el desarrollo de los contenidos fundamentales de las materias
  - actividades breves, individuales o en grupo que permitan aplicar los conceptos expuestos y resolver problemas, facilitando la participación activa de los estudiantes
- En sesiones de laboratorio. Actividades prácticas, sesiones de laboratorio guiadas, seminarios de resolución de problemas, etc. en grupos bajo la dirección de un profesor. Se podrán incluir actividades previas y posteriores a las sesiones de laboratorio y seminario que ayuden a conseguir los objetivos propuestos. Se fomentarán especialmente las actividades encaminadas al desarrollo de proyectos, supuestos prácticos, informes, etc.

- En tutorías programadas individuales o en grupos pequeños se realizará un seguimiento más individualizado del estudiante, con actividades de formación y orientación. Principalmente, se utilizarán para el seguimiento de los trabajos planteados, debate sobre alternativas y evaluación de los objetivos alcanzados.
- Realización de actividades, trabajos y estudio por parte del estudiante, de manera autónoma, individualmente o en grupo. Las actividades que el estudiante desarrollará de manera no presencial estarán orientadas principalmente a la adquisición de conocimientos básicos en el ámbito de la Informática y al desarrollo de los proyectos y trabajos solicitados, bien individualmente o en grupo.

Más concretamente, el desarrollo de la asignatura se realizará en sesiones presenciales de dos tipos: sesiones de teoría y sesiones de práctica.

- Con respecto a las **sesiones de teoría**, en estas se comenzarán resolviendo las dudas sobre la parte de los materiales del tema a tratar que los estudiantes deben haber trabajado antes de cada sesión (siendo la dedicación aconsejada a este trabajo previo de aproximadamente media hora). A continuación, se procederá a trabajar el resto del material alternando explicaciones teóricas y distintos tipos de actividades que fomenten la participación activa de los estudiantes con el objetivo de facilitar el aprendizaje de los conceptos del tema.
- En cuanto a las **sesiones prácticas**, el estudiante parte de un guión previo en el que tiene detallados los conceptos que se van a trabajar en la sesión, una serie de ejercicios previos aconsejados, así como una serie de ejercicios que deberá realizar durante la sesión. En este guión, se detallarán también los ejercicios propuestos no presenciales que el estudiante debe realizar después de cada sesión para afianzar los conceptos trabajados durante la sesión.

Resultados de aprendizaje[6]

Al completar la materia, el estudiante:

- Puede utilizar de manera eficaz un entorno de programación que incluya herramientas de edición, compilación, depuración y documentación de programas.
- Justifica la utilización de distintos paradigmas de programación y plataformas de desarrollo de software en un determinado contexto.
- Busca, analiza, sintetiza y critica nueva información para aprender nuevos lenguajes, algoritmos, técnicas, paradigmas y metodologías de programación aplicables a distintas áreas, teniendo como objetivo la actualización continua de los conocimientos y competencias.
- Analiza, planifica, diseña y desarrolla soluciones algorítmicas y programas robustos y correctos a problemas planteados, argumentando las decisiones tomadas, evaluando el resultado final y documentando el código y el proceso.

Conoce y aplica en actividades de nivel medio las competencias transversales fundamentales de la profesión

Sistemas de evaluación[6]

El sistema de evaluación se rige por la Normativa de Evaluación de la Universidad de Extremadura (DOE 06/10/2020).
Así, cada estudiante podrá ser calificado en la asignatura atendiendo a dos modalidades diferentes:

- Evaluación **continua**: sistema de evaluación constituido por diversas actividades distribuidas a lo largo del semestre de docencia de una asignatura. Esta modalidad incluye una prueba final, entendida como el conjunto de actividades de evaluación que tienen condicionada su celebración a la fecha oficial de examen para cada convocatoria.
- Evaluación **global**: sistema de evaluación constituido exclusivamente por una prueba final, que englobe todos los contenidos de la asignatura y que se realizará en la fecha oficial de cada convocatoria.

El estudiante podrá elegir, a través del Campus Virtual, la modalidad con la que quiere ser evaluado para cada convocatoria. En caso de ausencia de solicitud expresa por parte del estudiante, la modalidad asignada será la de evaluación **continua**.

En ambas modalidades es posible obtener la máxima calificación. La principal diferencia entre ambas modalidades radica en la distinta ponderación de los **bloques de calificación** que componen la asignatura.
Estos bloques se han realizado basándose en la metodología de la asignatura y son los siguientes:

- Bloque 1. Evaluación de proyectos de programación y sus informes técnicos
- Bloque 2. Evaluación de conceptos
- Bloque 3. Cuadernos de laboratorio

Bloque 1: proyecto de programación e informe técnico
La realización de proyectos de programación es la parte más importante de la asignatura y permitirá evaluar la mayoría de las competencias técnicas y transversales desarrolladas por el estudiante.
A lo largo del semestre, los estudiantes desarrollarán proyectos de programación guiados y divididos en partes. Estos proyectos podrán desarrollarse desde cero o partir de proyectos previos semi implementados. Durante el desarrollo de los proyectos los grupos tendrán un seguimiento por parte del equipo docente. Durante las sesiones prácticas y teóricas se impartirán todos los conocimientos y conceptos relacionados con la asignatura. La comprensión de estos conceptos es clave para que los estudiantes puedan aplicarlos en los proyectos de programación.
Los proyectos deberán entregarse funcionando correctamente y documentado en las fechas indicadas que serán publicadas en tiempo y forma a través del Campus Virtual. Los proyectos y sus informes técnicos deben ajustarse a los criterios especificados por el equipo docente de la asignatura.
Los proyectos se realizarán en grupos de 3 estudiantes.

Bloque 2. Evaluación de conceptos básicos
Se realizará una evaluación individual sobre los conceptos básicos (teóricos y prácticos) estudiados durante el semestre. Esta evaluación podrá realizarse de forma virtual y podría constar de pruebas tipo test, resolución de problemas, preguntas cortas, resolución de ejercicios o resolución de problemas prácticos relacionados con el proyecto o desarrollo escrito.

Bloque 3: Evaluación de cuadernos de laboratorio
Se propondrán a lo largo del semestre cuestionarios y/o ejercicios que serán realizados de forma individual por los estudiantes. Dichos cuestionarios podrán realizarse de forma virtual y estarán relacionados con los conceptos teóricos y prácticos que se estudian en la asignatura, así como con el desarrollo de los proyectos de programación. Asimismo, se podrán proponer ejercicios o lecciones a realizar en

sesiones de laboratorio, así como posibles ejercicios prácticos entregables en las clases de teoría.

Ponderación y normativa específica de los bloques en cada Modalidad de evaluación:

Evaluación continua.

Evaluación del Bloque 1:

- Este bloque es **recuperable** (puede ser entregado en todas las convocatorias).
- Su calificación es un 45% de la calificación final de la asignatura.
- Para superar este bloque, el estudiante por la modalidad de evaluación continua debe:
    - Entregar **todos los proyectos de programación e informes técnicos**.
    - Obtener una calificación mínima de 5 en la segunda entrega.
    - La calificación del Bloque 1 se calculará de forma ponderada (15% y 25% respectivamente) con las dos entregas del proyecto. Esta calificación debe ser como mínimo de 5 para superar la asignatura.
    - **o** En caso de superarlo, la calificación de este bloque se mantiene durante las siguientes convocatorias del mismo curso académico.

Evaluación del Bloque 2:

- Este bloque es **recuperable** (puede repetirse en todas las convocatorias).
- La calificación de este bloque supone un 45% de la calificación final de la asignatura.
- Para superar este bloque, el estudiante debe obtener una calificación mínima de 5.
- En caso de superarlo, la calificación de este bloque se mantiene durante las siguientes convocatorias del mismo curso académico.

Evaluación del Bloque 3:

- Este bloque **no es recuperable**, de modo que una vez finalizada la convocatoria oficial de enero, no se propondrán más actividades en laboratorio.
    - Sin embargo, si un estudiante ha obtenido puntos en este bloque, la nota del mismo será tenida en cuenta en las siguientes convocatorias del mismo curso académico.
- La puntuación de este bloque supondrá el 10% de la calificación final de la asignatura.

Cálculo de la calificación final del estudiante por Evaluación Continua:

Si el estudiante supera cada bloque de acuerdo a las especificaciones anteriores, la calificación final se calculará según la siguiente fórmula:

Calificación final = Bloque1 * 0,45 + Bloque2 * 0,45 + Bloque3 * 0,10

Evaluación global.

Evaluación del Bloque 1:

- Este bloque es **recuperable** (puede ser entregado en todas las convocatorias).

- Su calificación es un 45% de la calificación final de la asignatura.
- Para superar este bloque, el estudiante por la modalidad de evaluación global debe:
    - Entregar **todos los proyectos de programación e informes técnicos** en la convocatoria oficial de la asignatura
    - Obtener una calificación mínima de 5.
    - En caso de superarlo, la calificación de este bloque se mantiene durante las siguientes convocatorias del mismo curso académico.

Evaluación del Bloque 2:

- Este bloque es **recuperable** (puede repetirse en todas las convocatorias).
- La calificación de este bloque supone un 55% de la calificación final de la asignatura.
- Para superar este bloque, el estudiante debe superar un examen teórico (sobre **todos los contenidos** de la asignatura, incluido el Bloque 1) y obtener una calificación mínima de 5.
- En caso de superarlo, la calificación de este bloque se mantiene durante las siguientes convocatorias del mismo curso académico.

Evaluación del Bloque 3:

- Los estudiantes en modalidad de evaluación global **no tienen posibilidad de realizar este bloque**.

Cálculo de la calificación final del estudiante por Evaluación Global:

Si el estudiante supera cada bloque de acuerdo a las especificaciones anteriores, la calificación final se calculará según la siguiente fórmula:

Calificación final = Bloque1 * 0,45 + Bloque2 * 0,55

En la siguiente tabla puede verse un resumen final del sistema de evaluación para ambas modalidades de evaluación:

Bloque	Ponderación ev. **continua**	Ponderación ev. **global**	Calificación mínima	Recuperable
1. Proyecto de programación e informe técnico	45%	45%	5	Sí
2. Evaluación conceptos básicos	45%	55%	5	Sí
3. Evaluación de cuadernos de laboratorio	10%	-	-	No

Si no se cumplen los requisitos mínimos para los bloques 1 y 2, y la calificación final obtenida es menor que 3, esa será la calificación en esa convocatoria. En otro caso, la calificación en esa convocatoria será "Suspenso – 3".

Si el estudiante entrega o se presenta a alguna de las actividades que tienen una calificación mínima, se entenderá que se ha presentado a esa convocatoria. En otro caso se considerará "No presentado".

La copia o el plagio demostrados en cualquier actividad supone una nota final de SUSPENSO (0) en la convocatoria y una nota de 0 en los bloques no recuperables para todos los implicados, además de las actuaciones legales indicadas según la normativa vigente. Es responsabilidad del estudiante o grupo de estudiantes la custodia y protección de su proyecto (se utilizará un software específico de detección de copias en programas).

Bibliografía (básica y complementaria)

Bibliografía básica:

- David J. Barnes, Michael Kolling. Programación Orientada a objetos con Java usando BlueJ. 6a Edición. Pearson. 2017
- Alexander Shvets. Sumérgete en los Patrones de Diseño. Refactoring Guru. 2020

Bibliografía complementaria:

- Roberto Rodríguez, Encarna Sosa, Álvaro Prieto. Programación Orientada a Objetos. Editado por Librería Álvaro. 2004
- Bertrand Meyer. Construcción de Software Orientada a Objetos. 2ª Edición. Prentice Hall. 1999
- Bruce Eckel. Piensa en Java. 4ª Edición. Pearson. 2007
- Erich Gamma, Richard Helm, Ralph Johnson, John M. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. 1994
- Eric Freeman, Elizabeth Freeman, Kathy Sierra, Bert Bates. Head First Design Patterns. O'Reilly. 2004
- Tony Bevis. Java Design Patterns Essentials. 2nd Edition. Ability First. 2012
- Robert C. Martin. Código Limpio: Manual de estilo para el desarrollo ágil de software. 2012
- Joshua Bloch. Effective Java. 3rd ed. 2017.

Otros recursos y materiales docentes complementarios

Medios materiales utilizados:

- Teoría: aula virtual, aula física, cañón de vídeo, pizarra y dispositivos portátiles de los estudiantes.
- Práctica: aula virtual, laboratorio de ordenadores con todas las herramientas software de la asignatura correctamente instaladas, cañón de vídeo y pizarra.

Todo el material y recursos utilizados en la asignatura están disponibles en el aula virtual de la misma:

- Transparencias para cada tema de teoría.
- Guiones de las sesiones de laboratorio.
- Videotutoriales
- Planificación del curso, etc.

Además, en la **biblioteca** del centro existen ejemplares de los **libros aconsejados en la bibliografía**. Los manuales y enlaces digitales podrán ser consultados y/o descargados durante las sesiones prácticas, en las cuales se dispone de acceso a internet.