UAB
Universitat Autònoma de Barcelona

Programación Orientada a los Objetos

2026/2027

Código: 104389
Créditos: 6

Titulación	Tipo	Curso
Matemática Computacional y Analítica de Datos	FB	1

Profesor/a de contacto

Nombre : Guillermo Torres

Correo electrónico : guillermo.torres@uab.cat

Equipo docente

Enric Sala Esteva

Idiomas de los grupos

Puede consultar esta información al final del documento.

Prerrequisitos

No hay prerequisitos oficiales, pero es necesario saber programar en Python.

Objetivos

El objetivo principal es que el estudiante adquiera conocimientos sobre los principios del Diseño Orientado a Objetos y los Patrones de Diseño, con el fin de elaborar diseños de software que se representen mediante diagramas UML y se implementen posteriormente en un lenguaje de programación orientado a objetos.

Esto incluye conceptos fundamentales como clases, objetos, métodos, atributos, herencia, composición y encapsulamiento. A partir de estos principios, se introducen los Patrones de Diseño de Software, que proporcionan soluciones reutilizables a problemas comunes en el desarrollo de software.

Los diagramas UML permiten representar de forma clara las estructuras y relaciones del software, facilitando la visualización y comparación de diferentes diseños, así como la planificación de su implementación.

Resultados de aprendizaje

- CM09 (Desarrollar soluciones algorítmicas eficientes a problemas computacionales de acuerdo con los requisitos establecidos para la programación orientada a objetos.) Desarrollar soluciones algorítmicas eficientes a problemas computacionales de acuerdo con los requisitos establecidos para la programación orientada a objetos.
- CM10 (Analizar la complejidad computacional de las soluciones algorítmicas para desarrollar e implementar aquella que garantice el mejor rendimiento de la programación orientada a objetos.) Analizar la complejidad computacional de las soluciones algorítmicas para desarrollar e implementar aquella que garantice el mejor rendimiento de la programación orientada a objetos.

- CM11 (Asegurar el funcionamiento correcto de una solución algorítmica de acuerdo con los requisitos del problema a resolver para la programación orientada a objetos.) Asegurar el funcionamiento correcto de una solución algorítmica de acuerdo con los requisitos del problema a resolver para la programación orientada a objetos.
- KM09 (Reconocer los métodos, sistemas y tecnologías propios de la programación orientada a objetos.) Reconocer los métodos, sistemas y tecnologías propios de la programación orientada a objetos.
- SM09 (Utilizar los sistemas operativos y programas de uso común en la programación orientada a objetos.) Utilizar los sistemas operativos y programas de uso común en la programación orientada a objetos.
- SM10 (Utilizar las estructuras algorítmicas y de representación de los datos apropiadas para la resolución de problemas en la programación orientada a objetos.) Utilizar las estructuras algorítmicas y de representación de los datos apropiadas para la resolución de problemas en la programación orientada a objetos.

Contenidos

1. Conceptos de orientación a objetos. Introducción a los principios fundamentales de la orientación a objetos, como el encapsulamiento, la herencia, la composición o el uso de interfaces, así como la representación gráfica mediante diagramas de clases UML.
2. Programación orientada a objetos en Python. Uso de un lenguaje de programación para aplicar los conceptos orientados a objetos, incluyendo características avanzadas del lenguaje.
3. Principios de diseño orientado a objetos. Análisis de principios para estructurar software mantenible y escalable, y asignar responsabilidades de forma coherente entre los componentes.
4. Patrones de diseño. Estudio de soluciones reutilizables para problemas comunes en el diseño de software.
5. Estilo de codificación y buenas prácticas. Aplicación de convenciones de estilo, documentación del código, uso de comentarios y técnicas para mejorar la trazabilidad y comprensión del software.
6. Desarrollo práctico incremental. Aplicación progresiva de los contenidos teóricos-prácticos mediante ejercicios y un proyecto principal dividido en fases.

Actividades formativas y Metodología

Título	Horas	ECTS	Resultados de aprendizaje
Clases de prácticas	13	0,52	CM09, CM10, SM09
Estudiar	50,5	2,02	
Clases de teoria	36	1,44	CM11, KM09, SM10
Programación para la resolución de problemas	50,5	2,02	

La asignatura sigue una metodología de aprendizaje basado en proyectos (ABP), en la que los estudiantes desarrollan progresivamente un proyecto de programación aplicando los principios de la Programación Orientada a Objetos (POO). El proyecto está estructurado en hitos (*milestones*), que permiten consolidar los contenidos trabajados a lo largo del curso, promoviendo un aprendizaje significativo, activo y orientado a la práctica.

Clases de teoría. Se introducen los conceptos clave del paradigma orientado a objetos y se proporcionan las bases necesarias para afrontar el desarrollo del proyecto. Estas clases incluyen también la resolución de pequeños problemas de diseño y programación orientada a objetos. Además, se ofrecen referencias bibliográficas (libros, artículos, recursos en línea) que complementan las sesiones presenciales. Las transparencias utilizadas en clase no deben considerarse como única fuente de estudio; se espera que los estudiantes consulten activamente los materiales recomendados, disponibles en el campus virtual de la asignatura.

Clases de prácticas. Al inicio del curso, los estudiantes se organizan en pequeños grupos de trabajo. Durante estas sesiones, 1) se revisa semanalmente el progreso en el proyecto y se resuelven dudas técnicas y/o conceptuales, y 2) se realiza una evaluación continua, tanto grupal como individual, según el baremo definido en el enunciado de la práctica. Estas sesiones no están diseñadas para desarrollar el proyecto completo desde cero, sino para hacer seguimiento y evaluar el trabajo que los grupos han avanzado previamente de manera autónoma.

Trabajo autónomo. El aprendizaje de la programación requiere dedicación y práctica constante. Por ello, se espera que los estudiantes trabajen de forma autónoma fuera del aula: estudiando los contenidos teóricos, resolviendo ejercicios, revisando ejemplos y avanzando de manera continua en el desarrollo del proyecto.

El campus virtual contiene todos los materiales de la asignatura, así como la planificación detallada del curso. Todos los contenidos escritos están redactados en inglés.

Nota: se reservarán 15 minutos de una clase dentro del calendario establecido por el centro o por la titulación para que el alumnado rellene las encuestas de evaluación de la actuación del profesorado y de evaluación de la asignatura o módulo.

Evaluación

Actividades de evaluación continuada

Título	Peso	Horas	ECTS	Resultados de aprendizaje
Teoría	50%	0	0	CM11, KM09, SM10
Práctica	50%	0	0	CM09, CM10, SM09

La evaluación de la asignatura se basa en un sistema de evaluación continuada que combina actividades individuales y grupales, con el objetivo de valorar tanto la comprensión conceptual como la aplicación práctica de los contenidos teórico-prácticos.

Cálculo de la nota final (FM)

Todas las notas están sobre 10 puntos. La nota final se calcula según el siguiente algoritmo:

$EX = \max(0.5 * (EX1 + EX2), EXR) + OP_{extra}$

$PR = 0.4 * M1 + 0.4 * M2 + 0.2 * PR_{eval}$

if (EX >= 4.0) and (PR >= 4.0):

FM = 0.5 * (EX + PR)

else:

FM = min(EX, PR)

Para aprobar el curso se requiere que FM >= 5 puntos.

Donde:

- EX es la nota de teoría que se obtiene desde:
- EX1 y EX2 corresponden al primer examen y segundo examen, dos pruebas escritas individuales realizadas durante el curso.
- EXR es el examen de recuperación que puede sustituir al promedio de EX1 y EX2 si mejora la nota.
- OP_{extra} son los puntos adicionales obtenidos mediante ejercicios opcionales de refuerzo.

- EX está sobre 10 puntos.
- PR es la nota de prácticas que se obtiene desde:
- M1 y M2 son los proyectos que se evalúan en las prácticas de manera grupal e individual a lo largo del curso.
- PR_{eval} es una evaluación individual de las prácticas.
- PR está sobre 10 puntos.

Evaluación Única

- Esta asignatura no prevé el sistema de evaluación única.

Evaluación de la práctica

- La práctica se compone de los proyectos M1 y M2, que se realizan en grupo y son evaluados *in situ* durante las sesiones prácticas.
- La entrega de los proyectos es obligatoria. La no entrega de un proyecto se calificará con un 0.
- Las entregas se realizarán a través del campus virtual. Las entregas fuera de plazo podrán obtener como máximo una puntuación de 5 puntos.
- En caso de dudas sobre la autoría o contribución individual dentro del grupo, la calificación pasará a ser individual, asignando la nota en función de la aportación de cada miembro.
- Las sesiones de prácticas son obligatorias y requieren asistencia presencial. Se tomará asistencia durante el curso. En caso de ausencia, el estudiante deberá justificarla adecuadamente. Si la ausencia no está debidamente justificada, la nota máxima que podrá obtenerse en la parte práctica de la asignatura será de 5 puntos.
- Se realizará una evaluación individual (PR_{eval}).
- Estas evaluaciones no tienen recuperación.

Evaluación de la teoría

- Se realizarán dos pruebas escritas (EX1 y EX2) a lo largo del curso.
- Existe un examen de recuperación (EXR) que permite mejorar la nota si es superior a la media de EX1 y EX2.
- Los ejercicios opcionales de refuerzo (OP_{extra}) podrán sumar puntos adicionales a la nota de teoría (EX) y no tienen recuperación. Se entregan a través del campus virtual, y no se aceptan entregas fuera de plazo. Los puntos obtenidos de OP_{extra} no pueden sumarse a un exámen o a la parte de prácticas para alcanzar el mínimo requerido.

Otros aspectos relevantes

- No se convalidan prácticas, ejercicios ni exámenes de cursos anteriores.
- En caso de no entregar un examen o una parte de la práctica, la nota de esa parte será cero.
- La nota final será \"no evaluable\" si el estudiante no ha presentado ninguna actividad evaluable.
- La calificación de Matrícula de Honor (MH) se concederá discrecionalmente a estudiantes con nota final superior a 9.0, valorando todas las actividades realizadas.
- Las fechas de entrega y evaluación se publicarán en el campus virtual y podrán modificarse en caso de incidencias, informándose siempre mediante dicho medio.
- Para los exámenes escritos se indicará fecha y lugar para la revisión por parte del estudiante; si no asiste, no se hará revisión posterior.
- Plagios, copias o cualquier irregularidad en actividades evaluables supondrán una calificación de cero en dicha actividad sin posibilidad de recuperación en el mismo curso académico, y en su caso, la suspensión de la asignatura.
- En ningún caso se podrá utilizar inteligencia artificial (IA) generativa para suplantar la actividad de aprendizaje del estudiante. Las tareas sospechosas de haber sido generadas por estas técnicas se evaluarán con un 0.

El Uso de IA:

- Las herramientas de inteligencia artificial podrán utilizarse como herramientas de apoyo al aprendizaje

(por ejemplo, para mejorar la redacción, el estilo, la claridad expositiva, la corrección lingüística o para obtener asistencia en aspectos técnicos). En ningún caso podrán sustituir y/o suplantar la actividad de aprendizaje del estudiante, ni su adquisición de los conocimientos específicos de la asignatura.
- No es aceptable utilizar herramientas de inteligencia artificial para generar contenidos del trabajo que estén sometidos a evaluación. Las tareas/actividades evaluables que se sospeche que han sido generadas por una IA en lugar del estudiante serán consideradas como una copia y se evaluarán con un 0.

Bibliografía

Básica

Las clases de teoría se basan en un conjunto de transparencias que sintetizan los conceptos clave del curso. Estos materiales están pensados como guía, por lo que se recomienda consultar las siguientes obras para una mejor comprensión y para ampliar conocimientos:

Patrones de diseño. E. Gama, R. Helm, R. Johnson, J. Vlissides. Pearson Educación 2003.
Es un clásico del patrones de diseño orientado a objeto, por los autores que propusieron este concepto. La versión original en inglés es de 1995 y por ello sus ejemplos y diagramas pueden parecer desactualizados, explica muy bien los fundamentos.

Design patterns explained simply. Alexander Shvets. Libro on-line, https://sourcemaking.com/design-patterns-ebook. Tambien del mismo autor otro libro online en https://refactoring.guru/design-patterns
Ofrece explicaciones claras e ilustradas con ejemplos en múltiples lenguajes, incluídos Python, C++ y Jave. Ideal como introducción práctica.

Complementaria

Head first Design Patterns. E. Freeman & E. Robson, 2021.
Un libro muy didáctico sobre patrones de diseño que utiliza ejemplos sencillos explicados con humor.

Python: master the art of design patterns. D. Phillips, C. Giridhar, S. Kasampalis. Packt Publishing, 2016.
Obra extensa (más de 700 páginas) centrada en la implementación detallada de patrones de diseño en Python. Muy útil para ver cómo se aplican realmente en este lenguaje.

Mastering object-oriented Python. Steven F.Lott. Packt publishing, 2014.
Aporta una visión avanzada del lenguaje Python, profundizando en aspectos clave de la programación orientada a objetos. Destacan capítulos como el dedicado al sistema de logging y otros temas específicos del lenguaje.

Software

- Python junto con el gestor de paquetes Pip o Anaconda para instalar librerías tales como NumPy y Matplotlib.
- IDE Spyder para programar y depurar código Python.

Grupos e idiomas de la asignatura

La información proporcionada es provisional hasta el 30 de noviembre. A partir de esta fecha, podrá consultar el idioma de cada grupo a través de este enlace. Para acceder a la información, será necesario introducir el CÓDIGO de la asignatura

Tipo de docencia	Grupo	Idioma	Semestre	Turno
(TE) Teoría	1	Español	segundo cuatrimestre	manaña-mixto
(PLAB) Prácticas de laboratorio	1	Catalán	segundo cuatrimestre	manaña-mixto
(PLAB) Prácticas de laboratorio	2	Catalán	segundo cuatrimestre	manaña-mixto