UAB
Universitat Autònoma de Barcelona

Programación Orientada a los Objetos

Código: 104389
Créditos ECTS: 6

2024/2025

Titulación	Tipo	Curso
2503740 Matemática Computacional y Analítica de Datos	FB	1

Contacto

Nombre: Silvana Silva Pereira

Correo electrónico: silvana.silva@uab.cat

Idiomas de los grupos

Puede consultar esta información al final del documento.

Prerrequisitos

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

Objetivos y contextualización

Una de las metodologías más extendidas de diseño y programación es la orientada a objetos, según la cual un software se organiza en clases que contienen métodos (procedimientos) y atributos (datos). Aparte del concepto de clase, otros elementos fundamentales son la herencia, la composición y la separación de las partes pública y privada de las clases. Estos elementos son, pues, necesarios para la programación OO, pero no suficientes: hay que saber cómo utilizarlos para llegar a soluciones que sean fácilmente extensibles en el futuro, ya que una ley del software es que los cambios son inevitables. Existe una serie de principios de diseño o heurísticas que nos dicen cómo emplear estos elementos con el fin de conseguirlo. Sobre estos principios se construyen los patrones de diseño, que son soluciones a nivel de diseño OO a problemas recurrentes que aparecen a menudo al resolver problemas (aparentemente) independientes. En esta asignatura se introducen y se practican los conceptos OO así como los principios y patrones de diseño.

El aprendizaje gira en torno a un proyecto --- la práctica --- en la cual es necesario aplicar todos los temas explicados para diseñar e implementar una aplicación de software. El lenguaje de programación utilizado en la asignatura será Python.

Resultados de aprendizaje

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

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

Contenido

1. Conceptos de orientación a objetos: encapsulamiento, herencia, composición, interfaz. Diagramas de clase UML
2. Programación orientada a objetos en Python
3. Principios de diseño orientado a objetos
5. Patrones de diseño: creacionales, estructurales, de comportamiento
6. Estilo de codificación, *logging*, comentarios y documentación

Actividades formativas y Metodología

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

- Clases de teoría. El profesorado introduce los contenidos teóricos y proporciona referencias útiles (libros, artículos, páginas web) para que el estudiante pueda continuar su aprendizaje, así como también desarrollar el proyecto propuesto. Las transparencias utilizadas en estas clases, pues, no deben tomarse como única fuente de estudio. Hay que estudiar los libros recomendados, los artículos, capítulos de libros de acceso abierto y otros, disponibles en la página web de la asignatura. En estas clases también resolveremos pequeños problemas, de diseño y/o programación OO en Python.

- Sesiones de prácticas. Los estudiantes deberán organizarse en pequeños grupos a principio del curso. En estas sesiones los estudiantes vienen a 1) revisar el trabajo hecho del proyecto cada semana y resolver dudas, y 2) a ser evaluados de manera continua, de acuerdo con el baremo que figura en el enunciado de la práctica. Por lo tanto, la práctica *no* se hace sólo durante estas sesiones sino que requiere un trabajo previo.

Todos los materiales de la asignatura junto con su planificación detallada se encontrarán en el campus virtual. Todos estos materiales estan escritos 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
Exámenes	50%	0	0	CM09, CM10, CM11, KM09, SM09, SM10
Práctica	50%	0	0	CM09, CM10, CM11, KM09, SM09, SM10

Una parte de la evaluación es en grupo y otra individual. La nota de la primera (PR) corresponde a la práctica. La segunda (EX), al promedio de dos pruebas escritas durante el trimestre (EX1, EX2), o bien del examen de recuperación (EXR).

El algoritmo para el cálculo de la nota final (FM) es el siguiente (todas las notas son sobre 10):

EX = max(0.5 * (EX1 + EX2), EXR)
if (EX >= 4.0) and (PR >= 4.0):
FM = 0.5 * (EX + PR)
else :
FM = min(EX, PR)

La práctica

La realización de la práctica és esencial para asimilar los conceptos de teoria. La primera cosa pues que queremos asegurar es que la pràctica haya sido realmente hecha por el grupo que la presenta, y que cada miembro del grupo haya contribuido en la misma medida que los demás. La segunda es que la práctica se haga de manera progresiva durante el curso, en paralelo a la teoria. Por eso haremos los siguiente:

- La nota de la práctica se obtiene combinando la nota de les partes en las que ésta se dividide. Cada parte será evaluada in situ en una de les sesiones de prácticas que anunciaremos a principio de curs ("hitos"). Los puntos que se pueden obtener de cada parte s'especificaran en el enunciado. Estas avaluaciones no tienen recuperación después.

- Los dias de avaluació de la práctica (hitos), si de resultas de las respuestas a les preguntas que haga el profesor durante la evaluación, tenemos dudas sobre la autoria o la contribución de alguno de los miembros del grupo, entonces la nota de prácticapasará de ser de grupo a individual y en función de cuanta i qué parte de la práctica haya hecho cada miembro.

Los ejercicios

Propondremos algunos ejercicios que son opcionales. Estos ejercicios se pueden hacer indivualmente o en pequeño grup, el mismo o diferente que el de práctiques. Los ejercicios pueden sumar punts a la nota de teoria EX, hasta un máximo de 10. Decimos 'pueden' porque la obtenciób de estos puntos extra estará condicionada a una prueba de validación para comprobar que el estudiant ha realitzado efectivamente exercicio que ha entregado. Estas pruebas de validación las haremos sólo una vez, en los exámenes EX1 y EX2, y no tienen recuperación después.

Otros

No convalidamos prácticas, exercicios ni exámenes de cursos anteriores. Los estudiantes repetidores no reciben ningún tratamiento diferenciado.

Si no se entrega un examen o una parte de la práctica, la nota de esta parte es cero. La nota final será 'no evaluable' si no se ha entregado nada evaluable, ya sea examen, práctica o ejercicio.

Los hitos de práctiques son evaluaciones equivalentes a examenes escritos. Si un estudiant no se presenta, la nota és cero.

Las notas MH las concederemos de manera discrecional a los estudiantes con nota final superior a 9.0 y teniendo en cuenta todo tipo de trabajos hechos (exámenes, práctica y ejercicios).

Las fechas de evaluación continuada y entrega de trabajos se publicaran en el campus virtual de la assignatura y pueden estar sujectas a cambios en respuesta a posibles incidéncies. Siempre se informará mediante el campus virtual, que será el mecanismo de comunicación entre docentes y estudiantes.

Para las actividades de evaluación detipo examen escrito, se indicará un lugar, fecha y hora de revisión en la que el estudiantepodrá revisar la activitat con el professor. Si el estudiante no se presenta a esta revisión, no se revisará posteriorment esta actividad.

Esta assignatura no sigue el sistema de evaluación única.

Sin perjuicio de otras medidas disciplinarias que se estimen oportunas, y de acuerdo con la normativa académica vigente, se calificarán con un cero las irregularidades cometidas por el estudiante que puedan conducir a una variación de la calificación de un acto de evaluación. Por lo tanto, plagiar, copiar o dejar copiar la práctica o cualquier otra actividad de evaluación implicará suspender con un cero y no se podrá recuperar en el mismo curso académico. Si esta actividad tiene una nota mínima asociada, entonces la asignatura quedará suspendida.

Bibliografía

Básica

Las clases de teoría están basadas en un juego de transparencias, que son un resumen, una síntesis de conceptos que se encuentran más bien explicados en estos libros, que vale la pena leer o consultar para comprenderlos mejor 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 son ya antiguos. Pero explica muy bien los conceptos.

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
Ben explicado e ilustrado con ejemplos, que además encontramos programados en Java, Python, C ++ y más.

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.
Un libro muy extenso (700pp) sobre patrones de diseño en Python. Un aspecto muy positivo es que muestra cómo programarlos en Python en detalle, uno a uno.

Mastering object-oriented Python. Steven F.Lott. Packt publishing, 2014.

Python avanzado, o más bien, entrando en los detalles del lenguaje. "Grasp theintricacies of object-oriented programming in Python in order to efficiently build powerful real-world applications". De interés el capítulo de logging y algún otro.

Software

- Python en la versión Anaconda, que ya incluye las librerias numpy, matplotlib.
- Pycharm en la versión "community edition", más el plugin PlantUML.

Lista de idiomas

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