esade

TEACHING GUIDE - 2025-2026

Programming with Data

UGRA_015721

Departments	Data, Analytics, Technology and Artificial Intelligence (DATA), Dept. of Operations, Innovation & Data Sciences
Teaching Languages	English
ECTS	6
Teacher responsible	Carlota Catot Bragós - carlota.catot@esade.edu

Course Goals

By the end of this course, students will be able to:

- Manipulate and analyze large datasets using Python libraries such as NumPy, applying vectorization and array operations to optimize performance.
- Apply object-oriented programming (OOP) principles, including classes, objects, methods, and attributes, to develop modular and reusable components within data analysis workflows.
- Design and implement solutions to complex data-driven problems by combining OOP techniques with data manipulation strategies, using features like inheritance, encapsulation, and polymorphism.
- Handle exceptions effectively to improve program robustness and apply basic design patterns to structure data processing tasks efficiently.
- Test and debug programs that integrate OOP and data analysis, using systematic techniques and tools appropriate for data-centric applications.

Previous knowledge

This course is intended for students who have completed an introductory programming course in Python and are motivated to deepen their understanding through object-oriented programming. To engage effectively with the course content, students should already be able to use basic Python syntax, including indentation, variables, data types, and comments. They should also be familiar with control flow structures such as if statements, for and while loops, and know how to define and call functions, with an understanding of arguments, return values, variable scope, and common built-in functions.

Prerequisits

This course will be managed through a dedicated eCampus website. Students will find there all the necessary materials, including session plans, assigned readings and pre-class work, class materials, and further references. Students should familiarize themselves with this environment before the start of the course
and check for updates regularly.

Recomended courses

Successful completion of the course "Computing Foundations" is recommended for this course.

Teaching methodology

This course carries a study load of 6 ECTS, equivalent to 150 hours of dedicated effort. The workload will be distributed as follows:

- Approximately 20% of the study hours will be allocated to synchronous class activities. The course comprises a total of 20 sessions, encompassing interactive lectures, in-class exercises, and quizzes, requiring approximately 40 hours of active participation.

- Approximately 80% of the study hours will be dedicated to projects, and exam preparations, involving autonomous work. This portion of the workload demands approximately 110 hours of self-directed study and completion of the assigned tasks.

Students are expected to engage actively in the synchronous sessions to grasp the course material effectively. Furthermore, they will need to devote significant time and effort to the assigned projects and exam preparations, fostering practical application of the learned concepts and strengthening their understanding of the subject. To provide a dynamic and engaging learning environment, the synchronous sessions will be thoughtfully structured, integrating theoretical lessons with in-class practical activities relevant to the course concepts. Although the schedule may vary occasionally, each session will typically have theory concept mixed with in-class exercises to help the student to learn in a more efficient way.

To ensure an interactive and effective learning experience, it is essential that each student brings a laptop to class. Students who don’t own a personal computer or who have issues working with it can request one on lease from CAU.

This approach aims to strike a balance between theoretical understanding and practical application, fostering active engagement and reinforcing the learned material. By integrating theory with hands-on exercises, students will have the opportunity to immediately apply the knowledge acquired, solidifying their comprehension and improving their problem-solving skills. Since all classes will be interactive, students must be prepared to be full participants. Otherwise, this will limit their learning and undermine the experience for the other students. Students' engagement and performance in class activities will be a fundamental part of this course. Additionally, upon completing a unit, students will be required to take a quiz assessment to evaluate their progress and understanding of the material. This approach aims to foster active participation, reinforce learning, and facilitate

regular assessment of students' comprehension.

As part of their independent work, students will embark on two projects to be completed in groups, each project will have 2 different parts and a part of the project statement will be presented after each unit as part of the learning of it. After each block of 2 units, there will be a session to present the project in a class presentation to see if the students have reached the expected knowledge. These projects serve as invaluable opportunities for students to reinforce their learnings by tackling in-depth, real-world problems that demand critical thinking and practical application of the concepts covered in class, and also learn to work in group. Each project is expected to require an approximate workload of approximately 20 hours.

Throughout these projects, students will have the chance to evaluate their knowledge and showcase their creativity, crafting original pieces of work that demonstrate their proficiency. Students are encouraged to discuss coding approaches to solve the assignments and work as a group to find the best solution as a group. This course will be managed through a dedicated moodle website. Students will find there all the necessary materials, including study guides for every session, class materials and further references. Students should familiarize themselves with this environment before the start of the course and check for updates regularly.

Description

Course contribution to program

The course "Programming with Data" explores the fundamental principles of data manipulation and analysis using the Python programming language. With an increased emphasis on leveraging libraries like NumPy, students will acquire essential skills to handle and analyze large datasets efficiently, design and implement scalable programs, and address complex data-driven challenges.

Throughout this course, participants will investigate both the principles of object-oriented programming (OOP) and the techniques of data manipulation. By gaining understanding of classes, objects, inheritance, polymorphism, encapsulation, and the utilization of NumPy for vectorization and array operations, students will be well-prepared to create Python-based solutions for real-world problems.

Content

#	Topic
1	Unit 1 - Presentation & Setup In this unit, students will be introduced to the course and the Visual Studio Code integrated development environment (IDE). Presentation & setup. - Introduction to the course - Overview of Visual Studio Code as an integrated development

#	Topic
1	environment (IDE). - Navigating the Visual Studio Code interface and using its features.
2	Unit 2 – Introduction to Python This unit offers a comprehensive review of Python essentials, including expressions, conditionals, loops, dictionaries, and input/output. These fundamental concepts are crucial for grasping upcoming object-oriented programming topics. Additionally, the unit introduces environment diagrams, enhancing students' understanding of program execution and scope. Introduction to Python. - Expressions - Names, assignments, and user-defined functions - Environment diagrams - Multiple environments - Conditional statements - Functions - Iterations - Data structures - Miscellaneous Python features - Input/output
3	Unit 3 - Object-oriented programming This unit offers an in-depth exploration of object-oriented programming (OOP) fundamentals, covering essential topics such as abstraction, inheritance, polymorphism, encapsulation, and error handling. Students will grasp the core concepts of OOP in Python, including classes, objects, inheritance types, and error management using exceptions. Intro to the OOP Paradigm - What is OOP? - OOP main concepts - Fundamentals of OOP in python - Classes, attributes, and objects in Python. - Relations between classes and objects. - OOP pillars: Inheritance, Abstraction, Polymorphism, Encapsulation - Error handling using exceptions. - Traceback understanding
4	Unit 4 - Testing and Debugging OOP Programs This unit focuses on testing and debugging OOP programs, introducing students to unit testing in Python and debugging techniques. Testing and debugging OOP programs. - Unit testing in Python. - TraceBacks - Debugging techniques for OOP code.
5	Unit 5 - Python Libraries and Modules In this unit, students will explore Python libraries and modules, gaining proficiency in their creation, application, and dependency management. The module provides an overview of essential libraries like NumPy, covering topics such as Series, DataFrames, nd-arrays, operations, indexing, slicing, and file I/O. Python Libraries and Modules - Introduction to Python libraries and modules. - Creating and using Python modules. - Exploring popular Python libraries. - Understanding third-party libraries and package managers. - Managing dependencies in Python projects. - Best practices for creating and using Python libraries and modules. Using library NumPy: - Numpy Vs Python list - nd-array - Operations - Indexing - Slicing - File input/output
6	Unit 6 – Course closure Concluding the course, this unit offers a concise recap of key Object-Oriented Programming paradigm elements. Students will reflect on takeaways and insights gained, preparing for the final exam through a Q&A session that addresses any remaining questions. Course closure - Recap of main aspects of Object-Oriented Programming paradigm. - Takeaways and lessons learned. - Final exam preparatory Q&A

Assessment

Tool	Assessment tool	Category	Weight %
In-class analysis and discussion of issues	Participation	Ordinary round	10.00%
Quizzes/tests	Quizzes	Ordinary round	10.00%
Written and/or oral exams	Mid-term exam	Ordinary round	20.00%
Group project	Projects	Ordinary round	20.00%
Attendance and punctuality	Attendance. In accordance with ESADE regulations,	Ordinary round	0.00%

Tool	Assessment tool	Category	Weight %
	attendance is mandatory for this course. Students who fail to attend 80% of the course will not be allowed to pass and will be required to sit the retake exam.		
Written and/or oral exams	Retake exam	Retake	75.00%
Quizzes/tests	Quizzes	Retake	5.00%
Group project	Projects	Retake	10.00%
In-class analysis and discussion of issues	Participation	Retake	10.00%
Written and/or oral exams	Final exam. A minimum grade of 4/10 is required to pass the course	Ordinary round	40.00%

PROGRAMS

BAI25-Bachelor in Business and Artificial Intelligence (Undergraduates: Business)

BAI25 Year 1 (Mandatory)

DBAI21-Double Degree in Business Administration and Artificial Intelligence for Business (Undergraduates: Business)

DBAI21 Year 2 (Optative)

DBAI23-Double Degree in Business Administration and Artificial Intelligence for Business (Undergraduates: Business)

DBAI23 Year 1 (Optative)

DBAI25-Double Degree in Business Administration and Business and Artificial Intelligence (Undergraduates: Business)

DBAI25 Year 1 (Basic)