esade

TEACHING GUIDE - 2024-2025

Programming with Data

UGRA_015721

Departments	Dept. of Operations, Innovation & Data Sciences
Teaching Languages	English
ECTS	6
Teacher responsible	Allué Vall Roger - roger.allue@esade.edu

Course Goals

After successfully completing this course, students will have achieved the following learning objectives and competencies:

Proficiency in Data Manipulation and Analysis: Students will gain a thorough understanding of fundamental data manipulation and analysis techniques using Python libraries such as pandas and numpy. They will learn to efficiently handle, process, and analyze large datasets, employing vectorization and array operations to optimize performance.

Integration of Object-Oriented Programming (OOP) Principles: Students will develop a foundational understanding of object-oriented programming principles and demonstrate their ability to integrate these principles within their data analysis workflows. They will comprehend the concepts of classes, objects, methods, and attributes, enabling them to build modular and reusable components within their data-centric applications.

Design and Implementation of Data-Driven Solutions: Students will be able to design and implement solutions for complex data-driven problems using both OOP and data manipulation techniques. By employing libraries such as pandas and numpy, along with inheritance, encapsulation, and polymorphism, they will create flexible, scalable, and maintainable code that effectively addresses real-world data challenges.

Exception Handling and Design Patterns Overview: Students will acquire the knowledge and skills to handle exceptions gracefully, minimizing disruptions and enhancing the reliability of their programs. Additionally, they will explore common design patterns, focusing on their application in data processing and analysis to create elegant and efficient solutions for recurring programming problems.

Testing and Debugging Techniques: Students will demonstrate their ability to effectively test and debug their programs, incorporating both OOP and data manipulation aspects. They will employ systematic testing methodologies, ensuring robustness and stability in their code, and utilize tools and techniques specific to data analysis workflows.

Previous knowledge

This course welcomes students with a strong motivation and interest in learning the principles and concepts of object-oriented programming with Python. Prior exposure to an introductory programming course in Python will serve as a valuable starting point for effectively engaging with the OOP concepts covered in this course.

Prerequisits

· **Basic Syntax and Structure**: Understanding of Python's syntax, including the use of indentation to define blocks of code, and familiarity with basic constructs such as variables, data types, and comments.

· **Control Flow**: Proficiency in using control flow statements like if-else conditions, for and while loops, and understanding how to control the execution flow of a program.Functions:

· **Ability to define and call functions**, including understanding function arguments, return values, scope, and the use of built-in functions.

Recomended courses

Successful completion of the course **"Computing Foundations (2235.YR.015720.1)"** 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 30% 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 70% of the study hours will be dedicated to various assignments, 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 matter. 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 involve cycles of approximately 15 minutes of theory followed by 15 minutes of exercises.

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, every two weeks or 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 a series of assignments to be completed individually,
approximately scheduled every two or three weeks. These assignments 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. Each assignment is expected to require an approximate workload of 15 to 20 hours. Throughout these assignments, students will have the chance to evaluate their knowledge and showcase their creativity, crafting original pieces of work that demonstrate their proficiency. While students are encouraged to discuss coding approaches to solve the assignments, the final work products they submit must be their own, unless explicitly stated otherwise in the assignment instructions. 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 such as pandas and 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 pandas and 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 PyCharm integrated development environment (IDE). They will learn how to set up a Python project in PyCharm and navigate its interface. Presentation & setup. - Introduction to the course - Overview of PyCharm as an integrated development environment (IDE). - Setting up a Python project in PyCharm. - Navigating the PyCharm 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? - Difference between object and procedural oriented programming - OOP main concepts - Object Oriented Programming Languages. Fundamentals of OOP in python - Classes, attributes, and objects in Python. - Relations between classes and objects. Inheritance - Single and multiple inheritance. - Interface inheritance and implementation inheritance. - Benefits of inheritance. - Code reuse techniques. Abstraction - Abstract Base Classes. - Protocols Polymorphism - Polymorphism in Python. - Dynamic and static

#	Topic
3	polymorphism. - Method overriding and method overloading. Encapsulation - Encapsulation and information hiding in Python. - Decorators and property decorators. Error Handling - Error handling using exceptions.
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 Pandas and 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 Pandas - Series and Dataframes - Loading DataFrames from files - Most important DataFrame operations 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	20.00%
Written and/or oral exams	Exams	Ordinary round	30.00%
Individual or team exercises	Assignments	Ordinary round	40.00%
Attendance and punctuality	Attendance. In accordance with ESADE regulations, 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.	Ordinary round	0.00%
Written and/or oral exams	If a retake exam is needed, it will replace the mid-term and final exams, carrying 60% weight, while other course components will contribute the remaining part of the grade..	Retake	60.00%
Quizzes/tests	Quizzes	Retake	10.00%
Individual or team exercises	Assignments	Retake	25.00%
In-class analysis and discussion of issues	Participation	Retake	5.00%

PROGRAMS

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)