Syllabus

Course Description

Course Title	Object Oriented and Functional Programming
Course Code	76277
Course Title Additional	
Scientific-Disciplinary Sector	INFO-01/A
Language	English
Degree Course	Bachelor in Computer Science
Other Degree Courses (Loaned)	
Lecturers	dr. Julien Louis Michel Corman, JulienLouisMichel.Corman@unibz.it https://www.unibz.it/en/faculties/engineering/academic-staff/person/37010
Teaching Assistant	
Semester	Second semester
Course Year/s	1
CP	6
Teaching Hours	40
Lab Hours	20
Individual Study Hours	90
Planned Office Hours	18
Contents Summary	• Object-oriented design: objects, classes, interfaces, inheritance and polymorphism • Abstract data types (set, list, associative array, queue, ...) and related data structures (linked list, hash table, ...), • Composite types, type inference, generics • Basic input/output, serialization, streams, error handling, custom exceptions, debugging • Introduction to functional programming: function composition, recursion, currying, closures, function types • Functional principles applied to object-oriented programming: immutability, pure functions, lambda abstractions

Course Topics	This course belongs to the type "Attività formativa caratterizzante" and the subject area is "Scientifico-Tecnologico". The course is designed to help students develop generic, object-oriented and functional programming skills. After completing the course, students should be able to implement algorithms to solve simple programming problems and select appropriate data structures, write readable, concise, modular, and documented code.
Keywords	object-oriented programming, functional programming
Recommended Prerequisites	The course requires knowledge of the basics of imperative and object-oriented programming, particularly in Java, as taught in the "Introduction to Programming" course.
Propaedeutic Courses	
Teaching Format	The course includes frontal lectures and lab exercises.
Mandatory Attendance	Attendance to lectures and labs is optional. However, non-attending students should contact the lecturer at the start of the course to discuss the modality of their independent study. The evaluation process is slightly different for attending and non-attending students. It is described in the fields "Assessment" and "Evaluation criteria and criteria for awarding marks" below.
Specific Educational Objectives and Learning Outcomes	Knowledge and Understanding - D1.2: Know in details the fundamental principles of programming - D1.3: Have a solid knowledge of the most important data structures and programming techniques Applying knowledge and understanding - D2.2: Be able to develop small and medium size programs using different programming languages and paradigms. - D2.3: Be able to solve problems using programming methodologies. Ability to make judgments - D3.1: Be able to collect and interpret useful data and to judge information systems and their applicability. - D3.2: Be able to work autonomously according to the own level of knowledge and understanding.

	- D3.3: Be able to take the responsibility for development of projects or IT consulting. Communication skills - D4.1: Be able to use one of the three languages English, Italian and German, and be able to use technical terms and communication appropriately. - D4.4: Be able to work in teams for the realization of IT systems. Learning skills - D5.1: Have developed learning capabilities to pursue further studies with a high degree of autonomy. - D5.2: Have acquired learning capabilities that enable to carry out project activities in companies, public institutions or in distributed development communities. - D5.3: Be able to follow the fast technological evolution and to learn cutting edge IT technologies and innovative aspects of last generation information systems.
Specific Educational Objectives and Learning Outcomes (additional info.)	
Assessment	The assessment is based on optional individual assignments and a final exam. The assignments focus on topics taught during lectures and are designed to motivate students to study throughout the semester while consolidating the theoretical concepts covered in class. The exam evaluates whether students have acquired the expected notions and programming skills.
Evaluation Criteria	The final mark is based on the marks obtained in the assignments and the mark of the exam. Each assignment with a mark superior to the exam mark counts for 5% of the final mark. The other assignments are ignored. Here is an example. Alice gets 24/30 at the final exam. She also did 3 of the 5 assignments, and got the following marks: - Assignment 1: 28/30 - Assignment 2: 18/30 - Assignment 5: 30/30 Assignment 2 is ignored for Alice, because her mark at this

	assignment (10/30) is inferior to her final exam mark (24/30). The other two assignments count for 5% each. So her final mark will be 25/30 and is computed as follows: - 90% for the final exam, - 5% for Assignment 1, and - 5% for Assignment 5. After submitting and assignment, students must attend the following lab (online or in presence). During this lab, they may be asked to explain their solutions.
Required Readings	Lecture material on the course's website.
Supplementary Readings	- Miran Lipovaca. Learn you a Haskell for great good. No Starch Press, 2012. - Graham Hutton. Programming in Haskell. Cambridge University Press. 2nd edition, 2016. - Kathy Sierra, Bert Bates, and Trisha Gee. Head First Java: A Brain-Friendly Guide. O'Reilly Media, 3rd edition, 2022. - Herbert Schildt. Java: The Complete Reference. McGraw Hill. 11th edition, 2018. - Joshua Bloch. Effective Java. Addison-Wesley Professional, 3rd edition, 2017.
Further Information	Recommended software: - the Glasgow Haskell Compiler (https://www.haskell.org/ghcup) and an IDE for Haskell, e.g. Visual Studio Code (https://github.com/microsoft/vscode), - Java 21 and an IDE for Java, e.g. IntelliJ IDEA (https://www.jetbrains.com/idea/) or Visual Studio Code (https://github.com/microsoft/vscode).
Sustainable Development Goals (SDGs)	Quality education