Fakultät für Informatik
Facoltà di Scienze e Tecnologie informatiche
Faculty of Computer Science

SYLLABUS
COURSE DESCRIPTION

COURSE TITLE	**Programming Paradigms**
COURSE CODE	75038 (BSc in Computer Science and Engineering DM 270) / 70138 (BSc in Applied Computer Science DM 509)
SCIENTIFIC SECTOR	INF/01
DEGREE	Bachelor in Computer Science and Engineering
SEMESTER	2nd Semester
YEAR	2nd
CREDITS	6

TOTAL LECTURING HOURS	36
TOTAL LAB HOURS	18
PREREQUISITES	Students should have a solid mathematical foundation, good programming skills in an imperative or object -oriented language and be familiar with basic data structures and algorithms. These prerequisites are covered in the following courses: Analysis, Introduction to Programming, Programming Project, and Data Structures and Algorithms
COURSE PAGE	https://ole.unibz.it/

SPECIFIC EDUCATIONAL OBJECTIVES	• Type of course: “caratterizzanti” for L-31 and “affini o integrativi” for L-08 • Scientific area: “discipline informatiche” for L-31 and “formazione interdisciplinare” for L-8 Students will learn the key concepts and structures of the most popular programming paradigms, such as imperative, object-oriented, logic-oriented, functional and concurrent programming. They will practice to write small programs in different languages. Upon completion of the course, students shall have acquired basic programming skills in these languages and be able to judge strengths and weaknesses of different programming paradigms, in particular in the context of specific application domains.

LECTURER	Johann Gamper, https://www.unibz.it/en/faculties/computer-science/academic-staff/person/748-johann-gamper
SCIENTIFIC SECTOR OF THE LECTURER	INF/01
TEACHING LANGUAGE	English
OFFICE HOURS	Faculty of Computer Science, Piazza Domenicani 3, office POS 2.15; gamper@inf.unibz.it; +39 0471 016140

TEACHING ASSISTANT	Marco Montali
OFFICE HOURS	TBA
LIST OF TOPICS COVERED	• Overview of programming paradigms • Imperative paradigm • Functional paradigm • Logic paradigm • Concurrent Programming • Functional Programming
TEACHING FORMAT	Frontal lectures, labs and homework. In the frontal lectures, the basic concepts are introduced and explained together with some examples. In the labs, the students gain practical experience in solving problems using different programming languages. They apply the concepts learned during the lectures by writing small and medium sized programs. We will also distribute assignments for homework. The homework is optional. If students hand in their homework, it will be graded and considered as bonus points to increase the exam grade.

LEARNING OUTCOMES	**Knowledge and understanding** • Know various programming paradigms and languages. **Applying knowledge and understanding** • Be able to develop small and medium size programs using different programming languages and paradigms. **Ability to make judgments** • Be able to evaluate strengths and weaknesses of different programming languages in specific application contexts. **Ability to learn** • Have developed learning capabilities to pursue further studies with a high degree of autonomy;

ASSESSMENT	The assessment of the course consists of a single written exam at the end that covers the whole course, plus the optional homework if handed in by the students. The written exam is structured in two parts: • 80% of the exam is to write small programs in each of the programming languages covered in the course; • 20% are questions about basic concepts. The first part verifies the ability to solve problems by developing small programs in different programming languages. The second part verifies the understanding of key concepts of different programming paradigms and languages. The homework consists in writing small programs that need to be handed in together with a small README file.
ASSESSMENT LANGUAGE	English

EVALUATION CRITERIA AND CRITERIA FOR AWARDING MARKS	There are no requirements for attending the final exam. The final course mark is computed from the mark of the written exam plus the mark of the homework if handed in by the student. The homework is only considered if it is marked higher than the written exam. In this case, the final course mark will be the average of the two marks; otherwise, the homework has no impact. The criteria for the evaluation of the exam and the homework are: correctness, completeness and clarity of the programs and answers.

REQUIRED READINGS	*Lecture notes available on the course page*
SUPPLEMENTARY READINGS	• Bruce A. Tate: Seven Languages in Seven Weeks Pragmatic Bookshelf, 2010 (recommended!) • Maurizio Gabrielli, Simone Martini: Programming Languages: Principles and Paradigms Springer, 2010 (optional) • Allen B. Tucker, Robert E. Noonan: Programming Languages: Principles and Paradigms (2nd ed.) McGraw Hill, 2007 (optional)
SOFTWARE USED	The following freely available software is used: • Ruby • Prolog • Erlang • Haskell