unibz
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. They will practice to write small programs in different programming languages. Upon completion of the course, students shall also be able to judge strengths and weaknesses of different programming paradigms/languages, in particular in the context of specific application domains.

LECTURER	Johann Gamper, office POS 2.15, Faculty of Computer Science, Piazza Domenicani 3, johann.gamper@unibz.it, +39 0471 016140
SCIENTIFIC SECTOR OF THE LECTURER	INF/01
TEACHING LANGUAGE	English
OFFICE HOURS	Monday, 13:00-14:00, office POS 2.15, or email arrangement
TEACHING ASSISTANT	Theodoros Chondorgiannis
OFFICE HOURS	Wednesday, 14:00-16:00, office POS 2.12, or email arrangement theodoros.chondrogiannis@unibz.it
LIST OF TOPICS COVERED	• Overview of programming paradigms • Imperative paradigm • Functional paradigm

	• Logic paradigm • Concurrent Programming • Functional Programming
TEACHING FORMAT	Frontal lectures and labs (exercises). 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.

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 first part verifies the ability to solve problems by developing small computer programs in different programming languages. The second part verifies the understanding of key concepts of different programming paradigms and languages.
ASSESSMENT LANGUAGE	English
EVALUATION CRITERIA AND CRITERIA FOR AWARDING MARKS	The assessment of the course consists of a single written exam at the end that covers the whole course: 80% of the exam is to write small programs, at least one in each of the four programming languages covered in the course; 20% are questions about basic concepts. Criteria for the evaluation of the exam: correctness and clarity of programs/answers.

REQUIRED READINGS	Lecture notes available at 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	• Ruby • Prolog • Erlang • Haskell