Università degli Studi di L'Aquila - Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

Course catalogue

Page compiled on 23/09/2019

Programme of Module "Laboratorio di Programmazione I"

- Code: F1I056
- Type of course unit: Compulsory (Bachelor Degree in Computer Science curriculum General)
- Level of course unit: Undergraduate Degrees
- Semester: 1

Number of ects credits: (Bachelor Degree in Computer Science) 6 (workload 150 hours)

Teachers: Monica Nesi (monica.nesi@univaq.it)

1	**Course objectives**	The goal of this course is to introduce the basic notions of imperative and object-oriented programming. On successful completion of this module, the student should be able to solve simple problems, implement the related algorithms in a structured programming language, and use a computer to run simple programs.
2	**Course content and learning outcomes (dublin descriptors)**	Topics of the module include: • Algorithms, programs and programming languages. Flow charts, structure of a program. • Basic data types. Constants, variables, arithmetic expressions, boolean expressions. Assignment. input/Output primitives. Control structures: sequence, conditional, iteration and loop. • Structured data types: array, string. • Methods. Block structure and scoping rules. Parameter passing. Side-effects of methods. Recursion and recursive methods. • Classes and objects: basic notions, object creation and their manipulation. Static methods. Array of objects. • Inheritance and hierarchies. Polymorphism and late binding. Exceptions and their handling. On successful completion of this module, the student should : • have profound knowledge of the basic concepts of imperative and object-oriented programming, have knowledge and understanding of algorithms for solving simple problems, have knowledge and understanding of the core of the programming language used for coding the algorithms, have knowledge of a simple environment to run programs; • understand and apply definitions as given by the syntax and the semantics of the programming language; • analyse and discuss different algorithms for solving a problem and different implementations of the same algorithm in the given language; • explain and illustrate the fundamental notions of basic and structured data types, control structures, methods definition and invocation, parameter passing, recursive methods, classes and objects, inheritance and exceptions; • demonstrate skill in problem-solving, demonstrate ability to use a (subset of a) programming language to code algorithms and test the solutions on computer, demonstrate capacity of abstraction and modularity.
3	**Course prerequisites**	Basic notions of mathematics (in particular, sets and functions). No knowledge on programming and specific programming languages is required.
4	**Teaching methodsand language**	Lectures and exercises **Language**: Italian **Reference textbooks** • Cay Horstmann, ***Concetti di informatica e fondamenti di JAVA***. Apogeo. 2007.

		• Marco Bertacca e Andrea Guidi, ***Programmare in Java***. McGraw-Hill. 2007.
5	**Assessment methods**	Pre-Assessment: There is no formal pre-assessment, but the pre-requisites of the course are clearly stated on the module website. Fulfillment of such pre-requisites is verified by formative assessment. Additional lectures or individual homework might be provided by the teacher in case significant problems are detected. Formative Assessment: The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the course that they will be involved (in turns) in: - Questioning and discussion by means of open oral questions to the class or single students. - Assignment of written exercises to be solved during a lecture and a student is then selected for oral presentation of her/his solution to the class. Summative Assessment: Written test followed by an optional oral exam. In order to help the students to split the workload, an optional mid-term written test will also be provided, which is meant to cover the first part of the course. The written test includes exercises on (i) evaluation of Java code and understanding of its syntax and semantics, (ii) defining static methods (iterative and recursive) using data structures and objects and possibly raising exceptions and (iii) defining classes, subclasses and instance methods using inheritance, information hiding, code reuse, late binding. The written test aims at verifying (1) the level of knowledge of the constructs of the subset of the programming language introduced during the course and (2) the skills in problem solving and in implementing the proposed algorithms with correct code. This in order to verify the ability of analyzing simple problems, of applying the techniques learnt during the course, of implementing the proposed solutions correctly, and of evaluating alternative solutions. Criteria of evaluation will be: (i) the level of knowledge of the course topics and the ability of reasoning on them; (ii) the proper use of technical terminology and of the mathematical concepts underlying the course topics, e.g. functions and logic; and (iii) the clarity and completeness of solutions. The written test (3 hours) consists in: (a) Three or four exercises to cover point (i) and (ii), 50% of total marks; (b) Three or four exercises to cover point (iii), 50% of total marks. All parts can result in negative marks if the answer is omitted or seriously flawed. The oral examination is optional, but it becomes compulsory in certain cases, e.g. when the answers to exercises on defining methods are omitted or seriously flawed. The oral test (max 1 hour) consists of one question for each serious mistake in the written test (the answer compensates the negative marks obtained therein) and one question for at most 3 extra points that the student intends to add to the written test marks. Possibly, for a complete evaluation of the competence acquired, additional subjects are proposed by the teacher.