UNIVERSITÀ
DEGLI STUDI DI MILANO-BICOCCA

COURSE SYLLABUS

Algorithms and Programming

2223-2-E3501Q067

Aims

To design and implement software systems integrating different problem solving methods. This course aims at introducing the basic knowledge of software systems from the object-oriented paradigm perspective. Moreover, the course will provide competencies to model simple domains through the UML language and code them into the Java Programming Language, according to the object-oriented paradigm.

Contents

The course teaches object oriented programming and software engineering principles. The student will be able to model problems according to the object oriented paradigm and translate it into Java programs.

Detailed program

- Introduction to basic principles of object oriented programming (information hiding, inheritance, polymorphism) and UML language (Unified Modeling Language).
- Hints on software cycle of life.
- Java as programming language and platform.
- Object Oriented Programming in Java: classes and objects, attributes and methods.
- Advanced Object Oriented Programming in Java: inheritance and polymorphism.
- Exceptions, ArrayList, Geenrics and Collection Framework.

Prerequisites

Structured Programming (Laboratory of Mathematics and Informatics course)

Teaching form

- Lectures: 4 CFU
- Exercise classes: 1 CFU
- Laboratory: 1 CFU

Textbook and teaching resource

All the information about the course as well as the lessons slides and practical exercises will be available through the learning platform of the University, at the elearning.unimib.it link.

The suggested texdtbook will be:

- W. Savitch: "Programmazione di base e avanzata con Java", a cura di Daniela Micucci, 2nd edition, Pearson

Semester

Second semester

Assessment method

Examination type

Written and Oral examination; the oral examination is not mandatory, but necessary to obtain a "cum laude" merit. The mark range is 18-30/30. The oral examination is about both theoretical questions and practical exercises and can increase the result of written examination by at most 4 points.

The written examination is divided into two parts: the first one is devoted to evaluate theoretical skills about object oriented programming, by means of a collection of close-ended questions; the second one concerns the design and implementation of a simple software system, with the aim to demonstrate the student's capability to solve correctly a practical problem, on the basis of object-oriented programming principles considered during the lectures, without generating any kind of error (i.e. compile time, runtime, logical errors).

The arithmetic mean (possibly weighted) of the two marks defines the final mark proposed to the student: in case it is sufficient, the student can accept it as is or modify it by means of an oral examination (possibly decreasing the final mark). Oral examination is possible if and only if written examination is sufficient. The teacher has the faculty to establish mandatory oral examinations for those students whose written examinations, although sufficient, present some criticalities: for example, in case of not sufficient theoretical questions whereas pratical exercises are good, or viceversa.

Five exam sessions are stated: June, July, September, January and February; moreover two partial examinations are proposed to studets during the course. Partial examinations can be attended if and only if the student has accomplished the tasks proposed during practical laboratories.

Office hours

Thursday, between 11 a.m. and 12 a.m., or by appointment.

Sustainable Development Goals