UNIVERSITÀ
DEGLI STUDI DI MILANO-BICOCCA

SYLLABUS DEL CORSO

Algoritmi e Programmazione

2526-2-E3501Q067

Aims

The aim of this course is to enable students to design object-oriented software systems, integrating different types of problem-solving strategies.
In line with the educational objectives of the degree program, the course is intended to provide students with fundamental knowledge of the design and implementation of software systems following the object-oriented programming paradigm. Students will also acquire the skills needed to model simple domains using the UML language and implement them in the Java programming language, in accordance with the object-oriented paradigm.

By the end of the course, the student will be able to:

- Knowledge and understanding
  Acquire fundamental knowledge related to the design and implementation of software systems following the object-oriented programming paradigm.
- Applying knowledge and understanding
  Apply such knowledge to model simple domains using UML and implement them in the Java programming language, consistently with object-oriented principles.
- Making judgements
  Integrate various problem-solving strategies and critically assess design and implementation choices within an object-oriented software development context.
- Communication skills
  Effectively communicate design solutions using appropriate technical terminology related to object-oriented programming and modelling tools.
- Learning skills
  Develop the ability to independently learn and explore additional techniques and tools for the design and development of object-oriented software systems.

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

A hybrid teaching approach is used, that combines lecture-based teaching (DE) and interactive teaching (DI). DE involves detailed presentation and explanation of theoretical content. DI includes active student participation through both questions ans simulations during exercise classes in presence and the problem solving activity of exercises and problems, to be accomplished during the practical laboratories under the supervision of a tutor.
Lessons (32 hours) are conducted in person and are delivered in Italian.
Exercise classes (12 hours) are conducted in person and are delivered in Italian.
Laboratories (12 hours) are conducted at distance and are in Italian.
In case of logistical needs, some hours may be delivered remotely, either in online or offline mode.

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, 3rd edition, Pearson

Semester

Second semester

Assessment method

Examination type

Written and Oral examination; the oral examination is not mandatory and is upon request by the student, and is possible if and only if the written examination is sufficient, equal or grater than 26/30. The mark range is 18-30 cum laude. The oral examination is about both theoretical questions and practical exercises and can increase the result of written examination by at most 4 points (but can lower the grade if it is not well done, too).

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 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 students during the course. (6th exam session).

Office hours

by appointment.

Sustainable Development Goals