UNIVERSITÀ
DEGLI STUDI DI MILANO-BICOCCA

SYLLABUS DEL CORSO

Programmazione C++

2627-3-E3101Q133

Aims

At the end of the course, the student will be able to design and develop modular and maintainable programs. It will also be able to apply modern C++ programming techniques to develop high performance and graphic applications, and managing resources correctly. The acquired skills will enable the student to address and understand complex C++ applications.

Knowledge and understanding
Knowledge of the fundamental concepts of programming in the C++ language.
Knowledge of techniques and best practices for maintaining code correctness and integrity.

Applied knowledge and understanding
Ability to apply knowledge to create medium/large programs
Ability to correctly use the techniques and tools presented

Judgment and decision-making
Ability to design code to meet functional requirements and data and memory integrity

Communication skills
Ability to write code documentation correctly, document and clearly justify design choices

Learning skills
Ability to independently explore advanced aspects of C++ programming
Ability to extend the skills acquired to different programming problems

Contents

The course aims to give the student the necessary knowledge to face the development of C ++ applications in a correct way and to address the problems related to resource management . To this end, through the intensive use of case studies, the critical issues and difficulties inherent to the C++ language and the techniques best suited to address them will be shown. A cross-platform framework for the development of C ++ graphical applications will also be presented.

Detailed program

Introduction to C++.
Basic concepts of C++ programming
- data types, pointers, reference, scoping
- casting,
C ++ as an object-oriente programming language
- classes, constructors and destructors, overloading, friend methods
- inline, constness

Advanced C ++ programming concepts
- operator overloading
- virtual methods, abstracts, polymorphism
- inheritance
Generic programming
- template
- iterators

The Standard library (STL)
- The container classes
- The algorithms
- Functors
- Multithreading

Use of external libraries
- Static libraries
- Dynamic libraries
- The OpenMP library

The new C ++ 11, C ++ 14 standards

GUI applications
- QT Creator development environment
- Development of graphical interfaces
- Event management
- The Qt libraries, QTWidgets

Prerequisites

Basic programming language skills

Teaching form

Teaching delivered in Italian.

The teaching is structured as follows:

48 hours of lectures in delivery mode, in presence
20 hours of tutorials in delivery mode, in presence

Textbook and teaching resource

Bjarne Stroustrup, The C++ Programming Language - Special Edition, Addison Wensley.

Bruce Eckel, Thinking in C++ vol. 1 e vol. 2, Prentice Hall (available online)

Peter Van Weert, Marc Gregoire, C++ Standard Library Quick Reference, Apress

Lee Zhi Eng, Qt5 C++ GUI Programming Cookbook, Packt Publishing

Handouts

Semester

Third year, first semester

Assessment method

Verification of learning includes homework, a design test and an oral test.

There are two in-progress tests with theory questions pertaining to the course topics covered so far.
Each test contains 15 free-ytext response and multiple-choice questions.
The test is passed if 11 out of 15 answers are considered correct.
Each test, if passed, guarantees 2/30 points in the final grade and replaces the theory questions in the oral on those topics.

The project involves developing a solution to an assigned problem using the techniques and tools seen in the course. The problem is defined in such a way as to test the acquisition of practical and problem solving skills. The text of the project is published about three weeks prior to the exam call, and students have two weeks for submission. The maximum grade for the project is 26/30. Some criteria on project evaluation are:

- If as a result of testing by faculty during evaluation (e.g., calls to functions not tested by you), the code does not compile, the exam is NOT passed.
- Implementation of unsolicited code does not give additional points but if incorrect penalizes the final grade.
- Errors regarding memory management are considered GRAVE.
- Project grade does not depend on the amount of code written.

The oral exam consists of a discussion of the developed solution, theory questions on the concepts presented in class, and reasoning and deduction questions. The evaluation of the oral test, allows you to increase (or decrease) the base grade obtained earlier in the project and tests.

Office hours

By appointment

Sustainable Development Goals

QUALITY EDUCATION | INDUSTRY, INNOVATION AND INFRASTRUCTURE