Object-oriented programming - course description

General information

Course name	Object-oriented programming
Course ID	11.3-WE-AutP-O-OP-Er
Faculty	Faculty of Computer Science, Electrical Engineering and Automatics
Field of study	Automatic Control and Robotics
Education profile	academic
Level of studies	Erasmus programme
Beginning semester	winter term 2017/2018

Course information

Semester	2
ECTS credits to win	5
Course type	obligatory
Teaching language	english
Author of syllabus	• dr hab. inż. Paweł Majdzik, prof. UZ

Classes forms

The class form	Hours per semester (full-time)	Hours per week (full-time)	Hours per semester (part-time)	Hours per week (part-time)	Form of assignment
Lecture	30	2	-	-	Credit with grade
Laboratory	30	2	-	-	Credit with grade

Aim of the course

To provide basic knowledge about object programming paradigms.

To provide basic knowledge about abstract data typing definition with member functions (encapsulation),

To provide basic knowledge about inheritance, polymorphism and virtual functions, templates of classes and functions.

To give basic skills in designing programs and utilizing tools (e.g. tools from Standard Template Library).

Prerequisites

Principles of programming, Algorithms and data structures

Scope

Introduction to object programming. Concept of abstract data typing. Class definition. Encapsulation – declaration and definition of class member methods. Private and public class members. Constructors and destructors. Default and copy constructors. Synthesized constructors. Destructors.

Operators overloading. User defined conversions: converting function, converting constructor. Functions overloading: friend functions and inline functions, constructor and operator conversion.

Inheritance rules. Inheritance and the composition of objects. Protected members. Multiple and multi-base inheritance. Problem of variable names in multi-base inheritance. Polymorphism. Polymorphism. Virtual functions. Pure virtual functions. Early and late binding. Time and memory costs connected with application of polymorphism. Abstract classes - defining and examples of abstract classes application in object-oriented programs.

Standard Template Library. Function templates. Specialized functions. Phases of function adjustment. Class templates. Definition of class templates. Class templates versus microdefinitions. Containers and algorithms, iterators, associative containers, function objects. Designing of object-oriented programming. Design pattern .

Adapter pattern, facade pattern, bridge pattern etc..

Teaching methods

Lectures, laboratory exercises.

Learning outcomes and methods of theirs verification

Outcome description	Outcome symbols	Methods of verification	The class form
Knows basic design templates and understands their meanings in flexible software design.		• activity during the classes • an exam - oral, descriptive, test and other • an ongoing monitoring during classes	• Lecture • Laboratory
Can define and implement basic integral class elements: constructors, operator functions, destructors		• an ongoing monitoring during classes • internship's documentation	• Lecture • Laboratory
Student is able to design and implement simple object programs		• a draft • activity during the classes	• Laboratory

Outcome description	Outcome symbols	Methods of verification	The class form
Understands basic concepts related to object programming: encapsulation, homogeneity		• an exam - oral, descriptive, test and other • an ongoing monitoring during classes	• Lecture • Laboratory
Can define and implement basic integral class elements: constructors, operator functions, destructors		• an exam - oral, descriptive, test and other • an ongoing monitoring during classes	• Lecture • Laboratory

Assignment conditions

Lecture – the passing condition is to obtain a positive mark from the examination.

Laboratory – the passing condition is to obtain positive marks from all laboratory exercises to be planned during the semester.

Calculation of the final grade: lecture 50% + laboratory 50%

Recommended reading

Eckel B.: Thinking in C++, Prentice Hall, US Ed edition, 2002 2.

Stroustrup B.: The C++ Programming Language, Addison – Wesley, 2004

Further reading

Lippman S.B.: Inside the C++ Object Model, Addison – Wesley, 1996

Notes

Modified by dr hab. inż. Wojciech Paszke, prof. UZ (last modification: 29-04-2020 11:51)