COURSE DESCRIPTION CARD - SYLLABUS

Course name
Object-oriented programming [S2ETI1>ProgrObiekt]

Course

Field of study Education in Technology and Informatics	Year/Semester 1/1
Area of study (specialization) –	Profile of study general academic
Level of study second-cycle	Course offered in polish
Form of study full-time	Requirements compulsory

Number of hours

Lecture 15	Laboratory classes 30	Other (e.g. online) 0
Tutorials 0	Projects/seminars 0	

Number of credit points
4,00

Coordinators	Lecturers
dr hab. inż. Agnieszka Rybarczyk agnieszka.rybarczyk@put.poznan.pl	dr hab. inż. Agnieszka Rybarczyk agnieszka.rybarczyk@put.poznan.pl

Prerequisites
The student should have a basic knowledge about basic computer programming and be familiar with the basic terminology and basic methods used to solve simple programming tasks. Student should possess skills in solving basic problems and in implementing, modifying and testing computer programs on their own. He or she should have the ability in acquiring knowledge from specific sources. He or she should understand the necessity of constant extending of programming knowledge. In the scope of social competences the student must present such attitudes as honesty, responsibility, perseverance, cognitive curiosity, creativity, personal culture, respect for other people.

Course objective
1. To become familiar with the object-oriented programming methodology. Acquiring practical skills in designing and implementing, runing and testing object-oriented programs. 2. To provide knowledge to students about object-oriented programming on the basis of C++ programming language on the beginner and intermediate level. 3. Develop students’ skills in solving basic algorithmic problems and the skills concerning the division of complex problems into the elementary steps that can be programmed in a given language.

Course-related learning outcomes

Knowledge:
1. student has extended and deepened knowledge of mathematics, physics, chemistry needed in the technical area, useful for formulating and solving complex tasks in the field of technical and it education.
2. student has knowledge in the area of computer aided technical education.
3. student has an orderly, theoretically based general knowledge of algorithms, computer system architecture, operating systems, network technologies, programming languages, graphics, artificial intelligence, databases, decision support, learning systems and software engineering.

Skills:
1. can use the acquired mathematical knowledge to describe processes, design models and implement algorithms.
2. student has the ability to self-study and can determine the directions of further learning.
3. student has the ability in acquiring information from literature, databases and other sources (in his/her native language and english), integrate it, interpret and critically evaluate it, draw conclusions and formulate and fully justify opinions.

Social competences:
1. understands the necessity of learning new skills and rising her or his qualifications. can inspire and organize the learning of others.
2. can interact and work in a team, assuming different roles.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Learning outcomes presented above are verified as follows:
1. Evaluation of the quality of the programming project.
2. Evaluation of knowledge by test.

Programme content

The program of lectures includes the following issues.
Introduction to the programming languages and paradigms; paradigm definition, discussion and presentation of the object-oriented paradigm. Object-oriented programming rationale resulting from the analysis of software crisis sources. The idea of a new programming paradigm that supports the design of high quality programs. Searching for an optimal programming language and methodologies suitable for building universal, reusable software modules. Discussion of object-oriented approach. Brief presentation of the history of object-oriented languages. Definitions of basic object concepts: object, attributes (variables) of an object, object methods, object method calls, class interfaces, objects as class occurrences. Introduction to C++. The differences between C and C++. Comparison of solutions to simple problems in a functional and object oriented paradigm. Definitions: class components, static class components, access modifiers. Examples of defining classes include: definitions of class constructors and destructors, operators overloding, variables and class methods. Encapsulation as a mechanism for limiting relationships between software modules. Friend classes, methods and operators in C++. Types of operators dedicated for copying complex objects. Overloaded operators, streaming data input and output operators. Class inheritance and subtype relation between classes. Definition of new features of derived classes, ovverriding methods and variables. Inheritance: base and derived classes, single and multiple inheritance, virtual inheritance. Virtual functions, defining, calling, abstract classes. Exception handling. Function templates. Class templates. Container classes.
During the laboratories students will learn about programming environment and start writing simple and more advanced programs. Next, students in two-person teams will implement advanced task (programming project) which presentation will take place during the last laboratory.

Teaching methods

1. lecture: presentations with numerous examples of basic and advanced C++ programs.
2. laboratories: exercises, solving task, practical exercises, discussions, teamwork.

Bibliography

Basic

1. Programowanie zorientowane obiektowo, Bertrand Mayer, Helion, Warszawa, 2005
2. Metody obiektowe w teorii i praktyce, Ian Graham, WNT, Warszawa, 2004
3. Język C++, Bjarne Stroustrup, WNT, Warszawa, 1994
4. Thinking in C++, B. Eckel, Helion 2003.
5. Programowanie obiektowe, Peter Coad, Edward Yourdon, Read Me, 1994
6. Analiza obiektowa, Peter Coad, Edward Yourdon, Read Me, 1994
7. Nowoczesne projektowanie w C++, Andrei Alexandrescu, WNT, 2005
8. Symfonia C++, J. Grębosz, Oficyna Kallimach, Kraków, 2001.

Additional

1. Język C++, J. Kisilewicz, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 2005.
2. Wprowadzenie do programowania w języku C++, J. Kniat, WPP, Poznań, 1995
3. Pasja C++, J. Grębosz, Oficyna Kallimach, Kraków, 2001
4. Programowanie w języku C++, J. Kniat , Nakom, Poznań, 2002.
5. Szkoła Programowania Język C++, S. Prata, Robomatic, 2002

Breakdown of average student's workload

	Hours	ECTS
Total workload	47	4,00
Classes requiring direct contact with the teacher	32	0,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	15	0,00