COURSE DESCRIPTION CARD - SYLLABUS

Course name
Information technology [S1MiKC1E>INF]

Course

Field of study
Microelectronics and Digital Communication

Year/Semester
1/2

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
English

Form of study
full-time

Requirements
compulsory

Number of hours

Lecture
30

Laboratory classes
30

Other
0

Tutorials
0

Projects/seminars
0

Number of credit points

4,00

Coordinators

dr inż. Michał Sybis
michal.sybis@put.poznan.pl

Lecturers

Prerequisites

Basic knowledge of mathematical logic and combinatorics. The ability to formulate simple algorithms. Capable of obtaining information from literature and other sources in Polish or English; able to integrate acquired information, interpret it, and draw conclusions. Aware of the limitations of their own knowledge and skills, understanding the necessity of further learning and self-improvement.

Course objective

The course introduces fundamental concepts of procedural programming in C++. Students learn key language constructs such as data types, operators, conditional statements, and loops, as well as how to use functions to structure their code. Special attention is given to recursion mechanisms, their applications, and comparisons with the iterative approach. The course also covers topics related to input and output handling, memory management, and modular code organization. The goal of the course is to develop the ability to independently write correct, readable, and efficient programs.

Course-related learning outcomes

Knowledge:
1. Student has basic theoretical and practical knowledge of programming in C and C++, with a particular

focus on the principles of constructing correct programs, using functions, loops, conditional statements, and the fundamentals of recursion.
2. Student has general knowledge of designing simple programs and utilizing basic library functions in everyday programming practice.

Skills:
1. The student is able to analyze a problem from an algorithmic perspective, considering criteria such as program execution speed, scalability of the applied solutions, and the adequacy of the chosen methods.
2. The student can correctly break down a problem into smaller parts, select appropriate algorithmic approaches, identify key dependencies, and effectively organize code using functions and modules

Social competences:
1. Student understanding the need for broader dissemination of knowledge in the field of modern information technology.
2. Student is aware of the possibilities and limitations of contemporary computer science while being open to its applications in new areas of everyday life, economy, technology, and science.
2. Student has the ability to formulate independent opinions on currently used and available technologies and solutions in the design of modern IT systems.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Knowledge acquired in the lecture is verified by an exam.
Skills acquired in laboratory classes are verified on the basis of fulfilling tasks assigned in class or project. In both didactic forms, a passing threshold of 50% of the possible points is adopted. The following grading scale is used: < 50% 2.0; 50%-59% 3.0; 60%-69% 3.5; 70%-79% 4.0; 80%-89% 4.5; 90%-100% 5.0.

Programme content

As part of the course, students will become familiar with the fundamentals of programming in C++. The course structure in C++ covers the basic elements of a program and the organization of source code. Topics include fundamental data types, binary number representation, operators, and expressions, including arithmetic, logical, comparison, and assignment operators. The course also covers bitwise operations, control statements, arrays, functions, and recursion. Additionally, students will learn debugging and code profiling techniques, enabling efficient diagnosis and optimization of programs. Laboratory classes provide a practical approach to the topics discussed in lectures, allowing students to apply their acquired knowledge in real implementations. During the exercises, students will become familiar with the basics of programming in C++, learning how to create, compile, and run programs

Course topics

The scope of the lecture and laboratory classes includes:
1. Structure of a program in C++.
2. Basic data types, data conversion.
3. Operators and expressions, bitwise operations.
4. Number systems.
5. Control statements.
6. Arrays, multidimensional arrays.
7. Functions, argument passing, function overloading, recursion.
8. Selected sorting and searching methods.
9. Debugging and code profiling.

Teaching methods

Lecture: Multimedia presentation illustrated with examples provided on the board.
Laboratories: Practical exercises - implementation of tasks assigned by the instructor.

Bibliography

Basic:

1. Jerzy Grębosz, Symfonia C++ : programowanie w języku C++ orientowane obiektowo. T. 1/2/3, 2000
2. Jerzy Grębosz, Pasja C++ : szablony, pojemniki i obsługa sytuacji wyjątkowych w języku C++. T. 1/2, 2004
3. Jerzy Grębosz, Opus Magnum C++11 : programowanie w języku C++. T. 1/2/3, 2018
4. Bruce Eckel, Thinking in C++. Edycja polska

Additional:
1. Stephen Prata, Język C++.
2. T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Wprowadzenie do algorytmów, WNT, Warszawa, 2004

Breakdown of average student's workload

	Hours	ECTS
Total workload	120	4,00
Classes requiring direct contact with the teacher	60	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	60	2,00