COURSE DESCRIPTION CARD - SYLLABUS

Course name
Introduction to programming [S1Cybez1>WdP]

Course

Field of study
Cybersecurity

Year/Semester
1/1

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
Polish

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

5,00

Coordinators

prof. dr hab. inż. Jerzy Nawrocki
jerzy.nawrocki@put.poznan.pl

Lecturers

Bartosz Lewandowski
bartosz.lewandowski@put.poznan.pl

prof. dr hab. inż. Jerzy Nawrocki
jerzy.nawrocki@put.poznan.pl

dr hab. inż. Bartosz Walter prof. PP
bartosz.walter@put.poznan.pl

Prerequisites

The prerequisites comprise knowledge of mathematics and computer science at the basic level specified in the core curriculum for general secondary school and technical school.

Course objective

The course aims to provide students with advanced knowledge of various programming paradigms and the most important areas of software engineering and to support their development of programming skills.

Course-related learning outcomes

Knowledge:
The course provides knowledge about various programming paradigms, languages based on them , and the most important areas of software engineering (K1_W06).

Skills:
Upon completion of the course, the student will have basic programming skills, including using various paradigms and programming languages (K1_U10) and software quality assessment (K1_U09).

Social competences:
The course raises students' awareness of the importance of individual and team work skills (K1_K05).

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

1. Short quizzes conducted during the lecture by means of the eKursy platform (Moodle)
2. Ongoing assessment as part of laboratory exercises
3. Individual test
4. Team competition
5. Exam

Programme content

The course scope includes programming languages, C/C++ and Python, NASM assembly basics, regular expressions, rule-based programming, computational complexity, SQL and using it in C/C++ and Python, parallel processing in C/C++ and Python, JavaScript, microcontroller and Arduino programming, use cases, testing and code reviews, selected UML diagrams, and selected design patterns.

Course topics

Imperative programming
Assembly language
Functions and procedures
Real numbers and numerical methods
Object-oriented programming
Text processing and regular expressions
Rule-based programming
Computational Complexity
Databases and SQL
Parallel Processing
Event-Driven Processing and JavaScript
Embedded Systems Programming
Requirements Specification
Software Quality Assessment
Software Architecture and Design

Teaching methods

Lecture with handouts available on the eKursy platform
Repetition of the material from the previous lecture in the form of quizzes conducted by means of eKursy
Solving programming assignments during laboratory classes
Team competition as a practical introduction to teamwork

Bibliography

Basic:
Kernighan, Brian W., and Dennis M. Ritchie. "The C Programming Language", Prentice-Hall, 1988.
Lutz, Mark. "Learning Python", O'Reilly, 2013.

Additional:
Nawrocki, Jerzy R. "Programowanie komputerów IBM PC w języku asemblera metodą systematyczną", Wydawnictwo Politechniki Poznańskiej, 1991. (in Polish)

Breakdown of average student's workload

	Hours	ECTS
Total workload	130	5,00
Classes requiring direct contact with the teacher	60	2,50
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	70	2,50