COURSE DESCRIPTION CARD - SYLLABUS

Course name
Program languages [S1IBio1>JP_1]

Course

Field of study
Biomedical Engineering

Year/Semester
1/2

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
15

Laboratory classes
30

Other (e.g. online)
0

Tutorials
0

Projects/seminars
0

Number of credit points

4,00

Coordinators

dr hab. inż. Maciej Tabaszewski
maciej.tabaszewski@put.poznan.pl

Lecturers

dr hab. inż. Maciej Tabaszewski
maciej.tabaszewski@put.poznan.pl

Prerequisites

Basic knowledge of logic and computer science

Course objective

The lecture is intended to present the basics of theoretical knowledge regarding programming languages, characteristics of selected languages (Python, C, C++), and procedural and object-oriented programming. The structures of programs in these languages are also discussed with examples. The laboratories emphasize the practical side of program writing skills, by solving short, easy to quickly understand problems in the field of statistics, time series modeling, signal analysis, basic numerical methods, etc.

Course-related learning outcomes

Knowledge:
The student recognizes and knows the features of procedural, object-oriented and visual programming
The student knows the basic structures of selected programming languages
The student knows the concepts of classes, structures, objects, inheritance, polymorphism, encapsulation

Skills:
The student can create dedicated software

Social competences:
The student understands the role of computerization in the modern economy. Is able to participate creatively

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Lecture: Exam based on a test, 20-25 closed questions, passing threshold - 50% of the maximum points
Laboratory: Passing on the basis of tests before the exercises and on the basis of two practical tests - 50% of the maximum points

Programme content

Basic principles and structures in high-level programming. Issues related to structured and object-oriented programming. Creating programs in Python, C, C++. Skills training based on simple programs solving specific issues.

Course topics

Lecture:
General rules for constructing programs. Compilers and interpreters. Programming in high-level languages, overview and division of languages. Characteristics of selected languages. Visual programming languages. Functional programming. Object-oriented programming. Recursion.
Basics of programming in Python, C/C++
Python language - variables, data types, operators, built-in functions, input and output functions, references, loops, conditional statements, strings, lists, tuples, dictionaries, sets, functions (defined, ways of passing arguments), list expressions, lambda functions, writing and reading from text and binary files. Modules. Creating your own classes, overloading operators.
Exception handling. Functions with a variable number of arguments. Creating charts - matplotlib library, basic calculations using the numpy module
C/C++ language
libraries, compilation process, linking, preprocessor, variables, declarations and definitions of variables, arrays, pointers, pointer arithmetic, operators, loops, conditional statement, selection statement, structures, creating functions, input and output functions, writing and reading from a text file and binary, functions with a variable number of arguments, dynamic memory allocation.
C++
References, object-oriented programming. Concepts of classes, objects, encapsulation, inheritance, polymorphism, abstraction. Operator overloading, streams, exceptions, namespaces, STL library

Lab:
Creating programs related to the following topics: simple economic models, time series modeling, statistics (including linear regression and correlation analysis), digital signal analysis, selected simple numerical methods.
Basics of programming in Python - built-in functions, input-output operations, simple calculations, simulations, basics of code control, strings and lists, defining and calling functions (exercises 1-4).
Dictionaries and sets, writing to and reading from a file, sorting, filtering, searching for elements (exercise 5,6).
Importing data from MATLAB and any binary files with a known structure (exercise 7).
Using matplotlib to graphically illustrate the results and the random module. (exercise 8)
Basics of object-oriented programming, creating classes and operator overloading (exercises 9,10).
Using numpy for matrix calculations (exercise 11).
Practical skills test (exercise 12)
Basics of programming in C/C++ - program compilation, variables and variable types, input/output operations, creating functions, code control instructions, arrays and pointers, basics of object-oriented programming including operator overloading, creating function templates (exercises 13,14)
Practical test of skills (exercise 15).

Teaching methods

Lecture: multimedia presentation with theory and examples.
Laboratory exercises: practical exercises, writing short programs, together and independently

Bibliography

Basic:
M. Dawson, Python dla każdego, podstawy programowania, Wydanie III, Helion
A. Bell, Python, uczymy się programowania, Helion
M. Lutz, Python, wprowadzenie, Helion
J. Liberty, S.Rao, B.L. Jones, C++ dla każdego, Wydanie II, Helion
B. Stroustrup, Programowanie, Teoria i praktyka z wykorzystaniem C++,Helion

Additional:
W. McKinney, Python w analizie danych, Wydanie II, Helion,
R. Sedegewick, Algorytmy w C++, Wydawnictwo RM, Warszawa

Breakdown of average student's workload

	Hours	ECTS
Total workload	100	4,00
Classes requiring direct contact with the teacher	47	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	53	2,00