COURSE DESCRIPTION CARD - SYLLABUS

Course name
Introduction to programming [S1MNT1>WdP]

Course

Field of study
Mathematics of Modern Technologies

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
15

Laboratory classes
30

Other
0

Tutorials
0

Projects/seminars
0

Number of credit points

4,00

Coordinators

dr inż. Nadiia Bashova
nadiia.bashova@put.poznan.pl

Lecturers

Prerequisites

Basic knowledge of high school. Computer literacy

Course objective

The aim of the course is to familiarize students with the basics of computer programming and to teach the ba- sics of programming in Python and MATLAB. In particular, this includes providing students with basic informa- tion about computer arithmetic, structured programming, problem algorithmization and their programming (also in the form of functions), teaching students to be fluent in an integrated programming environment

Course-related learning outcomes

Knowledge:
• thestudenthasknowledgeabouttheuseofmathematicaltools[K_W01(P6S_WG),K_W05(P6S_WG), K_W07(P6S_WG)];
• the student knows the basics of computational and programming techniques [K_W01(P6S_WG), K_W05(P6S_WG), K_W07(P6S_WG)].

Skills:
• the student is able to construct an algorithm for solving a simple engineering task, implement and test it in a chosen programming environment [K_U04(P6S_UW), K_U05(P6S_UW), K_U11(P6S_UW)];
• the student is able to operate the devices in accordance with general requirements and knows how to apply the principles of health and safety at work in a computer laboratory [K_U04(P6S_UW), K_U05(P6S_UW), K_U11(P6S_UW)].

Social competences:
• the student is aware of the level of his knowledge [K_K01(P6S_KK), K_K02(P6S_KK)];
• the student is aware of deepening and broadening the knowledge of programming [K_K01(P6S_KK), K_K02(P6S_KK)].

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Lectures: passing the lecture in the form of a written test of a problematic and practical nature;
Laboratory classes: two tests during the semester. Bonus activity during classes.

Programme content

Update: 01.06.2023r.
Lectures& Laboratory classes:
• computer arithmetics;
- machine representation of numbers;
- encoding of integers and floating point numbers;
- convert decimal to binary systems and vice versa;
- coding negative integers;
- character encoding in the computer;
• algorithms;
- definition of algorithms;
- correctness of algorithms;
- pseudocode as one of the methods of writing algorithms;
- block diagrams as one of the methods of writing algorithms;
- blocks used in the recording of algorithms;
- examples of known algorithms;
• computing complexity;
- definition of computational complexity;
- cases of computational complexity;
- notation of capital O;
- determining computational complexity;
• operators, loops, and conditional instructions;
- arithmetic and logical operators;
- assignment operators;
- declaration of variables;
- conditional statements: if, switch;
- loops: for, while, do while;
- the keywords break, continue and return;
• arrays;
- array structure - matrices and vectors;
- array declaration;
- referencing array elements;
- iterating over array elements;
- operations on arrays;
• functions;
- function motivation in programming;
- examples of built-in functions;
- create functions;
- calling up functions;
- anonymous functions;
• comparison of basic instructions in matlab and python.

Course topics

The lecture program covers the following topics:
• computer arithmetics;
- machine representation of numbers;
- encoding of integers and floating point numbers;
- convert decimal to binary systems and vice versa;
- coding negative integers;
- character encoding in the computer;
• algorithms;
- definition of algorithms;
- correctness of algorithms;
- pseudocode as one of the methods of writing algorithms;
- block diagrams as one of the methods of writing algorithms;
- blocks used in the recording of algorithms;
- examples of known algorithms;
• computing complexity;
- definition of computational complexity;
- cases of computational complexity;
- notation of capital O;
- determining computational complexity;
• operators, loops, and conditional instructions;
- arithmetic and logical operators;
- assignment operators;
- declaration of variables;
- conditional statements: if, switch;
- loops: for, while, do while;
- the keywords break, continue and return;
• arrays;
- array structure - matrices and vectors;
- array declaration;
- referencing array elements;
- iterating over array elements;
- operations on arrays;
• functions;
- function motivation in programming;
- examples of built-in functions;
- create functions;
- calling up functions;
- anonymous functions;
• python data structures:
- dictionaries;
- tuples;
- sets;
• comparison of basic instructions in matlab and python.
The laboratory program covers the following topics:
• computer arithmetics;
- machine representation of numbers;
- encoding of integers and floating point numbers;
- convert decimal to binary systems and vice versa;
- coding negative integers;
- character encoding in the computer;
• algorithms;
- definition of algorithms;
- correctness of algorithms;
- pseudocode as one of the methods of writing algorithms;
- block diagrams as one of the methods of writing algorithms;
- blocks used in the recording of algorithms;
- examples of known algorithms;
• computing complexity;

- definition of computational complexity;
- cases of computational complexity;
- notation of capital O;
- determining computational complexity;
• operators, loops, and conditional instructions;
- arithmetic and logical operators;
- assignment operators;
- declaration of variables;
- conditional statements: if, switch;
- loops: for, while, do while;
- the keywords break, continue and return;
• arrays;
- array structure - matrices and vectors;
- array declaration;
- referencing array elements;
- iterating over array elements;
- operations on arrays;
• functions;
- function motivation in programming;
- examples of built-in functions;
- create functions;
- calling up functions;
- anonymous functions;
• python data structures:
- dictionaries;
- tuples;
- sets;
• comparison of basic instructions in matlab and python.

Teaching methods

Lectures: multimedia presentation supplemented with examples;
Laboratory classes: practical exercises and writing programs in Python and MATLAB.

Bibliography

Basic:
• Cormen T.H., Leiserson Ch.E., Rivest R.L. Wprowadzenie do algorytmów, WNT, 1994;
• Brzózka J., Dorobczyński L. MATLAB : środowisko obliczeń naukowo-technicznych, MIKOM, 2008;
• Summerfield M. Python 3: kompletne wprowadzenie do programowania„ Helion, 2010.

Additional:
• Mrozek B., Mrozek Z. MATLAB i Simulink Poradnik użytkownika. Wydanie II, Helion, Wrocław, 2004;
• Lutz M. Python. Wprowadzenie. Wydanie IV, Helion 2010.

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