STUDY MODULE DESCRIPTION FORM

Name of the module/subject **Programming Languages**	Code **1010632211010630597**

Field of study **Mechanika i budowa maszyn**	Profile of study (general academic, practical) **(brak)**	Year /Semester **1 / 1**
Elective path/specialty **Gas technology and renewable energy**	Subject offered in: **English**	Course (compulsory, elective) **obligatory**

Cycle of study: **Second-cycle studies**	Form of study (full-time,part-time) **full-time**

No. of hours Lecture: **1** Classes: **-** Laboratory: **1** Project/seminars: **-**	No. of credits **2**

Status of the course in the study program (Basic, major, other) **(brak)**	(university-wide, from another field) **(brak)**

Education areas and fields of science and art	ECTS distribution (number and %**)**
technical sciences	**2 100%**
Technical sciences	**2 100%**

Responsible for subject / lecturer:

dr inż. Przemysław Grzymisławski
email: przemyslaw.grzymislawski@put.poznan.pl
tel. tel. 61 665 21 35
Wydział Maszyn Roboczych i Transportu
ul. Piotrowo 3A, 60-965 Poznań

Prerequisites in terms of knowledge, skills and social competencies:

1	**Knowledge**	The student possesses elementary knowledge of the fundamentals of computer science, i.e. the computer architecture, types of variables, the general knowledge of the language of low, medium and high levels used in programming computers and typical engineering applications in the field of computer simulation of physical systems.
2	**Skills**	The student is able to use the concepts in the description of programming languages. The student is able to deal with specific problems that arise during the writing of programs.
3	**Social competencies**	Students can cooperate in a group, taking the different roles. The student is able to define priorities, which are important in solving the tasks posed before her/him. The student demonstrates self-reliance in solving problems, acquiring and improving her/his knowledge and skills.

Assumptions and objectives of the course:

The aim of the course is to provide students with information concerning the selected programming languages (Python, C++), the definitions and concepts. Students acquire knowledge and skills in the creation of computer programs.

Study outcomes and reference to the educational results for a field of study

Knowledge:

1. Has an extended knowledge in the area of information technology concerning computer programming and software for engineering calculations and simulation of physical systems. - [K2A_W05] - [-]

Skills:

1. Is able to use a common numerical computations system for programming a simple simulation task with limited degrees of freedom. - [K2A_U02] - [-]

Social competencies:

1. Understands the need for lifelong learning; is able to inspire and organize the learning process of others. ? [K2A_K01] - [-]
2. Is able to set priorities for realization of undertaken tasks. - [K2A_K04] - [-]
3. Is able to think and act in an entrepreneurial manner. - [K2A_K05] - [-]

Assessment methods of study outcomes

Written exam of lectures, written and practical credit of laboratory

Course description

Construction of computer programs. Comparison of the structure of C++ and Python. Discussion of the declaration constants, variables and variable types. Arithmetic operators. Functions - value of functions and parameters, making arguments be passed by value and by reference. Expressions - attribution, data comparison, priorities and communication. Branching and loops. Arrays and structures. Standard libraries in Python. The basic concepts of numerical calculations: iteration, interpolation, approximation, extrapolation, numerical integration, solving ordinary differential equations. Square root algorithm, algorithms for finding zeros of functions - Newton's method, secants and bisection method, method using numerical integration of Richardson extrapolation, solving ordinary differential equations using Euler's method and the midpoint method. The procedures for these algorithms in Python.

Basic bibliography:

1. Python programming for absolute beginner. Michael Dawson.
2. Programming Python. Mark Lutz.

Additional bibliography:

1. Beginning Python. From Novice to Professional. Magnus Lie Hetland
2. Effective Python. 59 Ways to Write Better Python. Brett Slatkin.
3. www.python.org
4. www.codecademy.com

Result of average student's workload

Activity	Time (working hours)
1. Preparation for the lecture	3
2. Participation in the lecture	15
3. Fixing the lecture	10
4. Consultation for the lecture	5
5. Preparing to pass the lecture	5
6. Participation in the completion of the lecture	1
7. Preparation of laboratory classes	10
8. Participation in the laboratory classes	15
9. Consultation for the laboratory classes	5
10. Preparing to pass the laboratory classes	10
11. Participation in the completion of the laboratory	1

Student’s workload

Source of workload	hours	ECTS
Total workload	80	2
Contact hours	42	2
Practical activities	41	0