University of Gdańsk

Subject card

Subject name and code	Computer Programming, PG_00182290						
Field of study	Physics						
Date of commencement of studies	October 2025		Academic year of realisation of subject	2025/2026			
Education level	Bachelor's studies		Subject group	Obligatory subject group in the field of study			
Mode of study	full-time studies		Mode of delivery	at the university			
Year of study	1		Language of instruction	Polish polish			
Semester of study	1		ECTS credits	3.0			
Learning profile	academic		Assessment form	credit			
Conducting unit	Division of Surface Physicochemistry and Interfacial Phenomena -> Institute of Experimental Physics -> Faculty of Mathematics, Physics and Informatics -> Rector						
Name and surname of lecturer (lecturers)	Subject supervisor		dr inż. Krzysztof Dorywalski				
	Teachers						
Lesson types	Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
	Number of study hours	0.0	0.0	45.0	0.0	0.0	45
	E-learning hours included: 0.0						
Learning activity and number of study hours	Learning activity	Participation in didactic classes included in study plan		Participation in consultation hours	Self-study	SUM	
	Number of study hours	45		0.0	30.0	75	
Subject objectives	The aim of the course is to acquire knowledge and skills in programming using a selected high-level language.						

Learning outcomes

Course outcome	Subject outcome	Method of verification
[FIZL3_W12] knows the methods of numerical analysis, knows at least one package for symbolic calculations at an advanced level, knows application software packages for presentation of results and data analysis; knows one programming language fluently	Student: - has knowledge of structured and object-oriented programming in a selected high-level language (Python) - knows data types, control statements, arithmetic and logical operators in Python; - understands object-oriented programming concepts and can apply them in practical tasks; - knows and can use computational packages and tools for symbolic calculations; - knows and can use user software packages for data analysis and results presentation.	[SW1] oral statement/ conversation/discussion [SW2] presentation/project/paper/ report [SW5] implementation of a problem task
[FIZL3_U12] can write, compile, run, test and document a computer program written by himself	The student can formulate a simple numerical algorithm to solve a given problem. - The student can write and run a computer program in Python, processing numerical and text data. - The student can apply structured and object-oriented programming approaches to develop programs ready for testing and documentation. - The student can write a program that reads from and writes to alphanumeric files. -The student can test a program, identify and correct errors, and prepare documentation describing the program’s functionality.	[SU1] oral statement/conversation/ discussion [SU2] presentation/project/paper/ report [SU5] implementation of a problem task [SU6] demonstration of practical skills [SU8] observation of student's independent or team work
[FIZL3_K02] can precisely formulate problems aimed at deepening the understanding of a given topic	- The student is able to precisely formulate a computational problem for further analysis. - The student can implement the formulated problem in a selected programming language. - The student can analyze and interpret the results to deepen the understanding of the studied subject.	[SK1] oral statement/conversation/ discussion [SK2] presentation/project/paper/ report [SK5] implementation of a problem task [SK6] demonstration of practical skills [SK8] observation of student's independent or team work

Subject contents

1. Introduction to Python and the Spyder IDE.

2.Data types, arithmetic and logical operators, conditional and control statements.

3. Complex data types: lists, tuples, dictionaries.

4. Procedural programming: functions, modules, basic work with packages.

5. Reading from and writing to alphanumeric files.

6. Elements of object-oriented programming: classes, objects, methods.

7. Basics of computational and data visualization packages.

Prerequisites and co-requisites

Basic knowledge of mathematics and physics, ability to use a computer and think logically, and readiness for self-directed learning and developing programming skills.

Assessment methods and criteria

Subject passing criteria	Passing threshold	Percentage of the final grade
completing programming tasks during the semester	51.0%	80.0%
homework	0.0%	20.0%

<table>
<tr><td rowspan="3">Recommended reading</td><td>Basic literature</td><td>A.1. Used during classes:
• Instructions and materials provided by the instructor.

A.2. Recommended for self-study:
• Eric Matthes, Python Crash Course, 3rd Edition, No Starch Press, 2022.
• Al Sweigart, Automate the Boring Stuff with Python, No Starch Press, 2019.
• Mark Reed, Python Programming for Beginners, Independently Published, 2022.
• Python Documentation: https://docs.python.org/3/
• Online materials on computational and data visualization packages, e.g., NumPy and Matplotlib.</td></tr>
<tr><td>Supplementary literature</td><td>R. Johansson, Numerical Python: Scientific Computing and Data Science Applications with Numpy, SciPy and Matplotlib. apress 2024</td></tr>
<tr><td>eResources addresses</td><td></td></tr>
<tr><td>Example issues/ example questions/ tasks being completed</td><td colspan="2">Write a Python program that:
1. Reads experimental measurement data from a CSV file (e.g., time, temperature, voltage).
2. Performs basic data analysis: calculates mean values, standard deviations, minimum and maximum values for each column.
3. Creates plots showing relationships between the data (e.g., temperature vs. voltage).</td></tr>
<tr><td>Work placement</td><td colspan="2">Not applicable</td></tr>
</table>

Document generated electronically. Does not require a seal or signature.