University of Gdańsk

Subject card

Subject name and code	Programming Basics, PG_00155439						
Field of study	Mathematics						
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			
Semester of study	2		ECTS credits	5.0			
Learning profile	academic		Assessment form	credit			
Conducting unit	Institute of Mathematics -> Faculty of Mathematics, Physics and Informatics -> Rector						
Name and surname of lecturer (lecturers)	Subject supervisor		dr Iwona Krzyżanowska				
	Teachers						
Lesson types	Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
	Number of study hours	0.0	0.0	60.0	0.0	0.0	60
	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	60		5.0	60.0	125	
Subject objectives	Students acquire the ability to algorithmize simple problems and implement them in a high-level, general-purpose programming language, e.g. Python, based on the basic elements of this language. Developing the ability to control and verify one's own and other people's reasoning and source code. Preparation for in-depth intellectual, literary and technical exploration of IT problems for which simple tools are not sufficient.						

Learning outcomes

Course outcome	Subject outcome	Method of verification
[MATL3_K09] is ready to critically evaluate arguments, find gaps in reasoning and constructively criticize other people's reasoning	The student is ready to evaluate the arguments critically, find gaps in reasoning, and constructively criticize other people's reasoning in the subject content.	[SK5] implementation of a problem task [SK8] observation of student's independent or team work
[MATL3_W10] knows and understands the basics of computational and programming techniques supporting the work of a mathematician and understands their limitations	The student knows and understands the basic types of variables, expressions, basic control instructions, ways of defining functions, and basic input-output operations of the learned language, and knows and understands their limitations.	[SW5] implementation of a problem task
[MATL3_U11] can recognize problems, including practical issues, that can be solved algorithmically; is able to specify such a problem	The student can formulate an algorithm for a simple problem, e.g. mathematical, justify its correctness, implement this algorithm in the known language, select appropriate data types and control instructions, design the program structure, separating subroutines (functions).	[SU5] implementation of a problem task
[MATL3_U12] knows how to arrange and analyse an algorithm in accordance with the specification and save it in the selected programming language	The student can formulate an algorithm for a simple problem, e.g. mathematical, justify its correctness, implement this algorithm in the known language, select appropriate data types and control instructions, design the program structure, separating subroutines (functions).	[SU5] implementation of a problem task
[MATL3_U10] is able to use the known software package or programming language to solve selected issues from the acquired fields, in particular mathematical analysis, linear algebra and statistics	The student can formulate an algorithm for a simple problem, e.g. mathematical, justify its correctness, implement this algorithm in the known language, select appropriate data types and control instructions, design the program structure, separating subroutines (functions).	[SU5] implementation of a problem task
[MATL3_K03] is ready to work in a team; understands the need for systematic work on all projects that have a long-term character	The student is ready to work in a team; understands the need to work on all long-term projects systematically.	[SK5] implementation of a problem task [SK8] observation of student's independent or team work
[MATL3_U09] is able to plan how to solve a specific problem and prepare a correct record of this solution, providing strict and precise justification for the correctness of his reasoning	The student can formulate an algorithm for a simple problem, e.g. mathematical, justify its correctness, implement this algorithm in the known language, select appropriate data types and control instructions, design the program structure, separating subroutines (functions).	[SU5] implementation of a problem task
[MATL3_W12] knows and understands the principles of occupational health and safety	The student knows and understands the principles of occupational health and safety in a computer lab.	[SW1] oral statement/ conversation/discussion
[MATL3_K01] is ready to accept the limitations of his/her own knowledge and understands the need for further education	The student knows the limitations of his or her own knowledge and is ready for further education in the subject content.	[SK8] observation of student's independent or team work
[MATL3_K04] is ready to understand and appreciate the importance of intellectual honesty in one's own and other people's actions; is willing to act ethically	The student is ready to be guided by the principles of intellectual honesty and ethics in his/her actions and to recognize their importance in the conduct of others.	[SK8] observation of student's independent or team work
[MATL3_U13] can compile, run and test a self-written computer programme	The student is able to compile, run and test a computer program written independently in a high-level language, e.g. Python.	[SU5] implementation of a problem task [SU8] observation of student's independent or team work

<table>
<tr><td>Subject contents</td><td>• Preparing the environment.
• Basic types of variables (lists, dictionaries, tuples, sets).
• Creating simple scripts.
• Conditional if statement.
• For, while loops.
• Pseudo-random numbers.
• Working with the file, writing, reading and modifying.
• Testing and error handling.
• Writing your own functions.
• GUI graphical user interface, e.g. tkinter module, graphics.py.
• Introduction to the Class.</td></tr>
<tr><td>Prerequisites and co-requisites</td><td></td></tr>
<tr><td>Assessment methods and criteria</td><td>
<table>
<tr><th>Subject passing criteria</th><th>Passing threshold</th><th>Percentage of the final grade</th></tr>
<tr><td>Observation of student's attitude</td><td>51.0%</td><td>0.0%</td></tr>
<tr><td>tests and/or project</td><td>51.0%</td><td>100.0%</td></tr>
</table>
</td></tr>
<tr><td>Recommended reading</td><td>
<table>
<tr><td>Basic literature</td><td>A. Literature required to finally pass the course (pass the exam):

A.1. used during classes

• https://docs.python.org/3/

A.2. studied independently by the student

• https://docs.python.org/3/
• Lutz M., Python. Wprowadzenie.</td></tr>
<tr><td>Supplementary literature</td><td>• https://www.w3schools.com/python
• Heinold B., A Practical Introduction to Python Programming
• McKinney W., Python w analizie danych. Przetwarzanie danych za pomocą pakietów Pandas i NumPy oraz środowiska IPython</td></tr>
<tr><td>eResources addresses</td><td></td></tr>
</table>
</td></tr>
<tr><td>Example issues/ example questions/ tasks being completed</td><td>no aplicable</td></tr>
<tr><td>Work placement</td><td>Not applicable</td></tr>
</table>

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