Appendix No. 1.5 to the Resolution No. 61/2025
of the Rector of the University of Rzeszów

SYLLABUS

REGARDING THE QUALIFICATION CYCLE FROM 2026 TO 2029
ACADEMIC YEAR 2027/2028

1. BASIC COURSE/MODULE INFORMATION

Course/Module title	Programming
Course/Module code *	
Faculty (name of the unit offering the field of study)	Faculty of Exact and Technical Sciences
Name of the unit running the course	Institute of Mathematics
Field of study	Mathematics
Qualification level	First-cycle studies (Bachelor's)
Profile	General academic
Study mode	Full-time
Year and semester of studies	Year 2, Semester 3
Course type	Specialisation course
Language of instruction	English
Coordinator	Paweł Pasteczka, PhD
Course instructor	Paweł Pasteczka, PhD

* - as agreed at the faculty

1.1.Learning format – number of hours and ECTS credits

Semester (no.)	Lectures	Classes	Laboratories	Seminars	Practical classes	Internships	others	**ECTS credits**
3	15		45					5

1.2. Course delivery methods
☒ traditional classroom-based instruction
☐ no distance learning / or optional use of distance learning methods and techniques

1.3. Course/Module assessment (exam, pass with a grade, pass without a grade)
Lecture – pass without a grade
Laboratory – pass with a grade

2. PREREQUISITES

Knowledge of mathematics and computer science at the high school level.

3. OBJECTIVES, LEARNING OUTCOMES, COURSE CONTENT, AND INSTRUCTIONAL METHODS

3.1. Course/Module objectives

O1	Familiarizing students with the high-level programming language Python – as a language with wide application and an extensive standard library.
O2	Acquiring the ability to program simple scripts.
O3	Acquiring the ability to use Python libraries.

3.2. COURSE/MODULE LEARNING OUTCOMES (TO BE COMPLETED BY THE COORDINATOR)

Learning Outcome	The description of the learning outcome defined for the course/module	Relation to the degree programme outcomes
LO_01	The student knows the syntax of the Python language and the programming environment in this language.	K_W05, K_W07
LO_02	The student is able to use Python to write uncomplicated programs. The student is able to use the Python standard library and apply it in their own programs.	K_U14, K_U22, K_K04, K_K05, K_K07

3.3. Course content (to be completed by the coordinator)

A. Lectures

Content outline
1. Program structure, basic types and operators
2. Control flow statements in Python
3. Variable, reference, object
4. Functions in Python
5. Classes in Python, part I
6. Classes in Python, part II
7. Classes in Python, part III
8. Modules, packages, assertions
9. Built-in tools — the Python standard library
10. File management
11. Common tasks in Python
12. Math and NumPy libraries in Python
13. Design Patterns and Version Control System (Git)
14. SageMath

15. Agile Programming

B. Classes, laboratories, seminars, practical classes

Content outline
1. Program structure, basic types and operators
2. Control flow statements in Python
3. Variable, reference, object
4. Functions in Python
5. Classes in Python, part I
6. Classes in Python, part II
7. Classes in Python, part III
8. Modules, packages, assertions
9. Built-in tools — the Python standard library
10. File management
11. Selected non-mathematical libraries
12. Math and NumPy libraries in Python
13. Single project (one project for one or two meetings)
14. SageMath

3.4. Methods of Instruction

Laboratory classes: problem solving, discussion, group work.
Lectures: problem-based lecture / multimedia presentation.

4. Assessment techniques and criteria

4.1 Methods of evaluating learning outcomes

Learning outcome	Methods of assessment of learning outcomes (e.g. test, oral exam, written exam, project, report, observation during classes)	Learning format (lectures, classes,...)
LO-01	project	lectures
LO-02	project	lectures, lab

4.2 Course assessment criteria

Lecture credit based on a project. Depending on the number of students attending the course, the single project can be either assigned to every student or assigned individually based on students' interests. It is possible to assign group projects; each student's contribution must be specified.

Students may be asked to describe their project individually or as a whole group during the online meetings.

5. Total student workload needed to achieve the intended learning outcomes – number of hours and ECTS credits

Activity	Number of hours
Course hours	60
Other contact hours involving the teacher (consultation hours, examinations)	5
Non-contact hours - student's own work (preparation for classes or examinations, projects, etc.)	65
Total number of hours	130
Total number of ECTS credits	5

* One ECTS point corresponds to 25-30 hours of total student workload

6. Internships related to the course/module

Number of hours	*Not applicable*
Internship regulations and procedures	*Not applicable*

7. Instructional materials

Compulsory literature: 1. Ramalho, Luciano. Fluent Python: Clear, concise, and effective programming. " O'Reilly Media, Inc.", 2015. 2. Matthes, Eric. Python crash course: A hands-on, project-based introduction to programming. no starch press, 2023. 3. Lutz, Mark. Learning python: Powerful object-oriented programming. " O'Reilly Media, Inc.", 2025. 4. Slatkin, Brett. Effective python: 90 specific ways to write better python. Addison-Wesley Professional, 2019.
Complementary literature: 1. NumPy User Guide (numpy.org) 2. SciPy documentation (scipy.org) 3. John D. Hunter et al., Matplotlib documentation (matplotlib.org) 4. Scott Chacon & Ben Straub, Pro Git, 2nd ed. (git-scm.com/book)

5. The Scrum Guide (scrumguides.org)
6. SageMath Documentation (doc.sagemath.org)

Approved by the Head of the Department or an authorised person