SYLLABUS

Computational Thinking

University year 2025-2026

1. Information regarding the programme

1.1. Higher education institution	Babeş-Bolyai University
1.2. Faculty	Faculty of Biology and Geology
1.3. Department	Department of Molecular Biology and Biotechnology
1.4. Field of study	Biology
1.5. Study cycle	Master
1.6. Study programme/Qualification	Bioinformatics
1.7. Form of education	Full-time

2. Information regarding the discipline

2.1. Name of the discipline	Computational Thinking					Discipline code	MME8181
2.2. Course coordinator					Conf. Dr. Camelia Şerban		
2.3. Seminar coordinator					Conf. Dr. Camelia Şerban		
2.4. Year of study	**1**	2.5 Semester	1	2.6. Type of evaluation	E	2.7 Type of discipline	Elective

3. Total estimated time (hours/semester of didactic activities)

3.1 Hours per week	4	Of which: 3.2 course	2	3.3 seminar/laboratory	2
3.4 Total hours in the curriculum	56	Of which: 3.5 course	28	3.6 seminar/laboratory	28
Time allotment for individual study (ID) and self-study activities (SA)					**hours**
Learning using manual, course support, bibliography, course notes (SA)					38
Additional documentation (in libraries, on electronic platforms, field documentation)					36
Preparation for seminars/labs, homework, papers, portfolios and essays					40
Tutorship					4
Evaluations					8
Other activities:					-
3.7. Total individual study hours	133				
3.8. Total hours per semester	175				
3.9. Number of ECTS credits	7				

4. Prerequisites (if necessary)

4.1. curriculum	-
4.2. competencies	-

5. Conditions (if necessary)

5.1. for the course	Video projector
5.2. for the seminar /lab activities	Computers, specific development environment

6.1. Specific competencies acquired [1]

[1] One can choose either competences or learning outcomes, or both. If only one option is chosen, the row related to the other option will be deleted, and the kept one will be numbered 6.

Professional/essential competencies	• C1.1 Description of programming paradigms and of language specific mechanisms, as well as identification of syntactic and semantic differences. • C1.3 Elaboration of adequate source code and testing of components in a given programming language, based on given specifications. • C1.4 Testing applications based on testing plans. • C1.5 Developing units of programs and corresponding documentation.
Transversal competencies	• CT1 Application of efficient and rigorous working rules, manifest responsible attitudes towards the scientific and didactic fields, respecting professional and ethical principles. • CT2 Use of efficient methods and techniques for learning, information, research and development of abilities for knowledge exploitation, for adapting to the needs of a dynamic society and for communication in a widely used foreign language.

6.2. Learning outcomes

Knowledge	The student has the necessary knowledge for using computers, developing software programs and applications, information processing. The student has knowledge related to programming, mathematics, engineering and technology and has the skills to use them to create complex information technology systems.
Skills	The student has the ability to develop, design and create new applications, systems or products using best practices of the field. The student has the necessary skills for computer program design and software systems analysis. The student has the ability to apply general rules to specific problems and produce relevant solutions. The student has the necessary skills to understand and use object-oriented programming concepts to develop software applications of medium complexity.
Responsibility and autonomy:	The student is able to identify complex problems and examine related issues to develop solving options and implement solutions. The student has the ability to choose and use programming paradigms (procedural, object-oriented, functional) to develop software applications appropriate for the specific domain of the application being developed.

7. 7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	To develop the foundations of Computational Thinking, concepts, methods and techniques
7.2 Specific objective of the discipline	To understand how Computational Thinking can be used by data scientists in order to organize structured and unstructured data for addressing business problems.

8. Content

8.1 Course	Teaching methods	Remarks
1. Introduction to Computational Thinking:	• Interactive exposure • Live coding • Explanation • Practical examples	
2. Functions		
3. Testing.		
4. Compound types: list, tuple, dictionary		

5. Searching & Sorting 6. Modular programming 7. User defined types 8. Lambda functions 9. Introduction to Data Science in Python: Pandas data-frames; Matplotlib plotting 10-11 Statistical Thinking in Python 12-14 Intermediate Python for Data Science	• Case-study discussions	

Bibliography

1. Kleinberg and Tardos – *Algorithm Design*. Pearson Educational, 2014
2. (http://www.cs.princeton.edu/~wayne/kleinberg-tardos/)
3. *The Python language reference*. (https://docs.python.org/3/reference/index.html)
4. *The Python standard library*. (https://docs.python.org/3/library/index.html)
5. *The Python tutorial*. (https://docs.python.org/3/tutorial/index.html)
6. Kent Beck - *Test Driven Development: By Example*. Addison-Wesley Longman, 2002.

8.2 Seminar / laboratory	Teaching methods	Remarks
1. Simple Applications 2. Simple Applications 3. Simple Applications 4. Modular Programming. User defined types 5. Lambda 6. Introduction to Python libraries for Data Science 7. Statistical Thinking	• Interactive exposure • Explanation • Conversation • Didactical demonstration	

Bibliography

1. Kleinberg and Tardos – *Algorithm Design*. Pearson Educational, 2014
2. (http://www.cs.princeton.edu/~wayne/kleinberg-tardos/)
3. *The Python language reference*. (https://docs.python.org/3/reference/index.html)
4. *The Python standard library*. (https://docs.python.org/3/library/index.html)
5. *The Python tutorial*. (https://docs.python.org/3/tutorial/index.html)
6. Kent Beck - *Test Driven Development: By Example*. Addison-Wesley Longman, 2002.

9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

The course respects the IEEE and ACM Curricula Recommendations for Computer Science studies.
The course exists in the studying program of all major universities in Romania and abroad.
The content of the course is considered the software companies as important for average programming skills

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	Know concepts and methods from the domain of Computationl Thinking	Project development for a specific domain	50%
10.5 Seminar/lab activities	Apply the concepts and methods learnt for solving problems to a from a specific domain	Project verification and presentation	50%
10.6 Minimum performance standards – minim 5 grade			

11. Labels ODD (Sustainable Development Goals)[2]

Not applicable.

Date:	Signature of course coordinator	Signature of seminar coordinator
15.04.2025	**Conf. Dr. Camelia Șerban**	**Conf. Dr. Camelia Șerban**

Date of approval:	Signature of the head of department
...	Assoc.prof.phd. Adrian STERCA

[2] Keep only the labels that, according to the *Procedure for applying ODD labels in the academic process*, suit the discipline and delete the others, including the general one for *Sustainable Development* – if not applicable. If no label describes the discipline, delete them all and write „*Not applicable.*".