SYLLABUS

Rust Programming

University year 2026-2027

1. Information regarding the programme

1.1. Higher education institution	Babeş Bolyai University
1.2. Faculty	Faculty of Mathematics and Computer Science
1.3. Department	Department of Computer Science
1.4. Field of study	Computer Science
1.5. Study cycle	Bachelor
1.6. Study programme/Qualification	Computer Science
1.7. Form of education	Full time

2. Information regarding the discipline

2.1. Name of the discipline	**Rust Programming**					Discipline code	**MLE5270**
2.2. Course coordinator						Assoc. Prof. Mihai Florin Gabriel Crăciun	
2.3. Seminar coordinator						Drd. Tudor Jinga	
2.4. Year of study	3	2.5. Semester	6	2.6. Type of evaluation	C	2.7. Discipline regime	Elective

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

3.1. Hours per week	**7**	of which: 3.2 course	**2**	3.3 seminar/laboratory/project	**1 Lab +2 Pr**
3.4. Total hours in the curriculum	60	of which: 3.5 course	24	3.6 seminar/laboratory/project	**36**
Time allotment for individual study (ID) and self-study activities (SA)					**hours**
Learning using manual, course support, bibliography, course notes (SA)					5
Additional documentation (in libraries, on electronic platforms, field documentation)					6
Preparation for seminars/labs, homework, papers, portfolios and essays					10
Tutorship					4
Evaluations					5
Other activities					
3.7. Total individual study hours	**30**				
3.8. Total hours per semester	**90**				
3.9. Number of ECTS credits	**5**				

4. Prerequisites (if necessary)

4.1. curriculum	Fundamentals of Programming
4.2. competencies	Basic low-level programming skills

5. Conditions (if necessary)

5.1. for the course	Projector
5.2. for the seminar /lab activities	Internet access and support for personal laptops usage

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	• reasoning about the design and beahviour low level programs
Transversal competencies	• application of organized and efficient work rules, of responsible attitudes towards the didactic-scientific field, to bring creative value to own potential, with respect for professional ethics principles and norms • efficient development of organized activities in an interdisciplinary group and the development of empathetic abilities for • interpersonal communications, to relate to and cooperate with various groups

6.2. Learning outcomes

Knowledge	The student knows how to reason about low-reason programs written with the help of an ownership driven, linear type system.
Skills	The student is able to fluently write programs in the Rust language.
Responsibility and autonomy:	The student has the ability to work independently to obtain solutions to diverse problems requiring safe low-level programming.

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

7.1 General objective of the discipline	• Gaining a deep understanding of the Rust programing language
7.2 Specific objective of the discipline	• Understanding ownership and algebraic types, thread and memory safety enforced at compile time. • Being able to debug and solve compile time errors related to borrowing and ownership of values. • Writing idomatic Rust code using patterns specific to the language.

8. Content

8.1 Course	Teaching methods	Remarks
1. Introduction to Rust. Types and values, control flow basics	Presentation, conversation, case studies.	
2. Tuples and Arrays, References, User-Defined Types		
3. Pattern Matching, Methods and Traits, Generics		
4. Closures, Standard Library Types and Traits		
5. Memory Management and Smart Pointers		
6. Borrowing and Lifetimes		
7. Iterators, Modules, Testing		
8. Error Handling, Unsafe Rust		
9. Idiomatic Rust Patterns		
10. Rust concurrency I		
11. Rust concurrency II		
12. Exam		
Bibliography 1. Comprehensive Rust, Google Inc. https://google.github.io/comprehensive-rust 2. Klabnik, Steve, and Carol Nichols. *The Rust programming language*. No Starch Press, 2023. 3. Blandy, Jim, Jason Orendorff, and Leonora FS Tindall. *Programming Rust*. " O'Reilly Media, Inc.", 2021. 4. Gjengset, Jon. *Rust for rustaceans: idiomatic programming for experienced developers*. No Starch Press, 2021.		
8.2 Seminar / laboratory	Teaching methods	Remarks
1. Introduction to Rust. Control flow structures. Structures and ownership	Presentation, conversation, case studies.	Labs will be held once every two week.
2. Traits		
3. Enums and errors		
4. Data structures and iterators		
5. Threads and Futures		
6. Final assignment check		
Bibliography 1. 100 Exercises to Learn Rust https://academy.jetbrains.com/course/27805 2. Comprehensive Rust, Google Inc. https://google.github.io/comprehensive-rust 3. Klabnik, Steve, and Carol Nichols. *The Rust programming language*. No Starch Press, 2023. 4. Blandy, Jim, Jason Orendorff, and Leonora FS Tindall. *Programming Rust*. " O'Reilly Media, Inc.", 2021. 5. Gjengset, Jon. *Rust for rustaceans: idiomatic programming for experienced developers*. No Starch Press, 2021.		

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 follows the IEEE and ACM curricular recommendations for studies in computer science.
- Software companies consider the course content useful in developing the students' modeling and programming skills.

10. Evaluation

Activity type	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Percentage of final grade
10.4 Course	Understanding the specifics of the Rust programming language	Written exam and quizzes throughout the semester	50%
10.5 Seminar/laboratory	Solving Rust exercises	Lab assignments based on the chapters of the 100 Rust Exercises RustRover course	50%
10.6 Minimum standard of performance			
• A grade of at least 5 for the course and laboratory activities, respectively. • Final grade at least 5			

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

Not applicable.

Date:
7.04.2026

Signature of course coordinator

Assoc. Prof. EMihai Florin Gabriel Crăciun

Signature of seminar coordinator

Drd. Tudor Jinga

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*.”.