SYLLABUS

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	**Master**
1.6 Study programme / Qualification	**High Performance Computing and Big Data Analytics**

2. Information regarding the discipline

2.1 Name of the discipline (en) (ro)	**Programming Paradigms** **Paradigme de Programare**						
2.2 Course coordinator	**Assoc. Prof. Eng. Florin Craciun**						
2.3 Seminar coordinator	**Assoc. Prof. Eng. Florin Craciun**						
2.4. Year of study	**1**	2.5 Semester	**1**	2.6. Type of evaluation	**E**	2.7 Type of discipline	**Optional**
2.8. Code of the discipline	**MME8028**						

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:					hours
Learning using manual, course support, bibliography, course notes					28
Additional documentation (in libraries, on electronic platforms, field documentation)					28
Preparation for seminars/labs, homework, papers, portfolios and essays					35
Tutorship					14
Evaluations					14
Other activities:					-

3.7 Total individual study hours	119
3.8 Total hours per semester	175
3.9 Number of ECTS credits	7

4. Prerequisites (if necessary)

4.1. curriculum	• Fundamentals of Programming • Object-Oriented Programming • Functional and Logic Programming
4.2. competencies	• Average software development skills

5. Conditions (if necessary)

5.1. for the course	projector
5.2. for the seminar /lab activities	projector

6. Specific competencies acquired

Professional competencies	• Understanding and working with basic concepts in computer programming; • Capability of analysis and synthesis; • Proficient use of tools and languages specific to software systems development; • Knowing the specifics of main programming paradigms
Transversal competencies	• Professional communication skills; concise and precise description, both oral and • written, of professional results; • Independent work capabilities; able to fulfill different roles; • Antepreneurial skills.

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

7.1 General objective of the discipline	• Know and understand fundamental concepts of programming. • Be able to apply different programming paradigms to different programming projects
7.2 Specific objective of the discipline	At the end of the course, students should • know the main features of different programming paradigms: procedural, object-oriented, concurrent, functional, logical, event-based, scripting • have a good understanding of the following concepts: value, type, variable, binding, procedural abstraction, data abstraction, object, class, component, interface, polymorphism; • learn the similarities and differences between different programming paradigms in terms of the concepts they implement

8. Content

8.1 Course	Teaching methods	Remarks
1. Basic concepts	• Interactive exposure	

	• Explanation • Conversation • Didactical demonstration	
2. Oz syntax, data structures	• Explanation • Conversation • Didactical demonstration	
3. Oz syntax, data structures	• Explanation • Conversation • Didactical demonstration	
4. Statements, Kernel Language, Abstract Machine	• Explanation • Conversation • Didactical demonstration	
5. Higher-Order Programming	• Explanation • Conversation • Didactical demonstration	
6. Lambda Calculus	• Explanation • Conversation • Didactical demonstration	
7. Tupled Recursion and Exceptions	• Explanation • Conversation • Didactical demonstration	
8. Types, ADT, Components	• Explanation • Conversation • Didactical demonstration	
9. Declarative Concurrency	• Explanation • Conversation • Didactical demonstration	
10. Declarative Concurrency	• Explanation • Conversation • Didactical	

	demonstration	
11. Declarative Concurrency	• Explanation • Conversation • Didactical demonstration	
12. Stateful Programming	• Explanation • Conversation • Didactical demonstration	
13. Relational Programming	• Explanation • Conversation • Didactical demonstration	
14. Constraint Programming	• Explanation • Conversation • Didactical demonstration	

Bibliography

1. SCOTT, MICHAEL L.: Programming Language Pragmatics, 4th ed, Morgan-Kaufmann, 2016
2. SEBESTA, ROBERT W.: Concepts of Programming Languages, 10th ed, Pearson Education, 2012
3. SZYPERSKI, CLEMENS: Component Software. Beyond Object-Oriented Programming, Addison Wesley (1st ed. 1998, 2nd ed. 2002 with GRUNTZ, DOMINIK and MURER, STEFAN).
4. STROUSTRUP, BJARNE: The C++ Programming Language Special Edition, Addison-Wesley, 2000 chapter 2
5. VAN ROY, PETER; HARIDI, SEIF: Concepts, Techniques and Models of Computer Programming, MIT Press, 2004
6. WATT, David A.: Programming Language Design Concepts, Wiley, 2004

8.2 Seminar / laboratory	Teaching methods	Remarks
1. Research papers allocation for the oral presentation	Use practical tools to implement group projects. Discuss research papers.	Seminar is organized as a total of 14 hours – 2 hours every second week Project is organized as a total of 14 hours – 2 hours every
2. First programming assignment	Use practical tools to implement group projects. Discuss research papers.	
3. Research papers presentations	Use practical tools to implement group	

	projects. Discuss research papers.	
4. Second programming assignment	Use practical tools to implement group projects. Discuss research papers.	
5. Research papers presentations.	Use practical tools to implement group projects. Discuss research papers.	
6. Third programming assignment	Use practical tools to implement group projects. Discuss research papers.	
7. Research papers presentations.	Use practical tools to implement group projects. Discuss research papers.	
Bibliography Research papers Mozart System		

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 Curriculla Recommendations for Software Engineering studies;
- The content of the course is considered by the software companies as important for average software development skills

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	- know the basic principle of the domain; - apply the course concepts - problem solving	Oral exam	40.00%
10.5 Seminar/lab activities	- be able to apply course concepts - be able to do a critical evaluation of research papers	-Paper presentation - 3 programming assignments	15.00% 3x15.00%
10.6 Minimum performance standards			
➢ At least grade 5 (from a scale of 1 to 10) at both oral exam and seminar work.			

Date Signature of course coordinator Signature of seminar coordinator

..... Assoc. Prof. Eng. Florin CRACIUN Assoc. Prof. Eng. Florin CRACIUN

Date of approval	Signature of the head of department
....	...........…............................