SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	**Babeş-Bolyai University of Cluj-Napoca**
1.2 Faculty	**Faculty of Mathematics and Computer Science**
1.3 Department	**Doctoral School in Mathematics and Computer Science**
1.4 Field of study	**Computer Science**
1.5 Study cycle	**Doctoral studies**
1.6 Study programme	**TRAINING PROGRAM BASED ON ADVANCED ACADEMIC STUDIES**

2. Information regarding the discipline

2.1 Name of the discipline	**Programming paradigms: basic and advanced concepts**						
2.2 Course coordinator	**Prof. dr. Bazil Pârv**						
2.3 Seminar coordinator	**Prof. dr. Bazil Pârv**						
2.4. Year of study	**1**	2.5 Semester	**1**	2.6. Type of evaluation	**C**	2.7 Type of discipline	**Compulsory**

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

3.1 Hours per week	**3**	Of which: 3.2 course	**2**	3.3 seminar/laboratory	**1 sem**
3.4 Total hours in the curriculum	**36**	Of which: 3.5 course	**24**	3.6 seminar/laboratory	**12**
Time allotment:					hours
Learning using manual, course support, bibliography, course notes					**65**
Additional documentation (in libraries, on electronic platforms, field documentation)					**62**
Preparation for seminars/labs, homework, papers, portfolios and essays					**60**
Tutorship					**17**
Evaluations					**10**
Other activities:					**-**

3.7 Total individual study hours	**214**
3.8 Total hours per semester	**250**
3.9 Number of ECTS credits	**10**

4. Prerequisites (if necessary)

4.1. curriculum	● Fundamentals of Programming ● Object-Oriented Programming ● Functional and Logic Programming
4.2. competencies	● Average programming skills

5. Conditions (if necessary)

5.1. for the course	● Videoprojector, Internet access
5.2. for the seminar /lab activities	● Computers, Internet access, UML tool

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. *Programming paradigms*. Definitions. Main programming paradigms. Programming styles. Evolution of programming languages	● Interactive exposure ● Explanation ● Conversation ● Didactical demonstration	
2. *Basic concepts 1*. Values and types. Variables and storage	● Interactive exposure ● Explanation ● Conversation ● Didactical demonstration	
3. *Basic concepts 2*. Bindings and scope. Control flow	● Interactive exposure ● Explanation ● Conversation ● Didactical demonstration	
4. *Advanced concepts 1*. Type systems. Composite types	● Interactive exposure ● Explanation ● Conversation ● Didactical demonstration	
5. *Advanced concepts 2*. Subroutines and control abstraction (procedural abstraction)	● Interactive exposure ● Explanation ● Conversation ● Didactical demonstration	
6. Advanced concepts 3. Data abstraction and object orientation. Generic abstraction	● Interactive exposure ● Explanation ● Conversation ● Didactical demonstration	

7. *Advanced concepts 4*. Errors and events. Concurrency	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
8. *Paradigms 1*. Imperative programming	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
9. *Paradigms 2*. Object-oriented programming	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
10. *Paradigms 3*. Concurrent programming	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
11. *Paradigms 4*. Functional programming	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
12. *Paradigms 5*. Logic programming	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
13. *Paradigms 6*. Event-driven programming	• Interactive exposure • Conversation	
14. *Paradigms 7*. Scripting	• Interactive exposure • 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
7. WEGNER, PETER: Concepts and paradigms of OOP, OOPSLA '89 Keynote talk

8.2 Seminar / laboratory	Teaching methods	Remarks
1. Establishing the programming language PL for paper Paper title: Programming language analysis – PL Requirements for paper Calendar/schedule of seminars/activities Weeks 1-4	Conversation, debate, case studies, presentations	Seminar is organized as a total of 14 hours – 2 hours every other week
2. Paper presentations phase 1 Weeks 5-9	Presentation, discussion	
3. Paper presentation phase 2 Weeks 10-14	Presentation, discussion	
4. Paper final version Week 14		

Bibliography

Students will search and use programming paradigms documentation

- on the department server (win/labor/Romana/master/PP)
- on the web, using main CS databases

The ELISA project http://jklunder.home.xs4all.nl

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

- This course follows the IEEE and ACM Curriculla Recommendations for Software Engineering studies;
- Courses with similar content are taught in the major universities in Romania offering similar study programs;
- Course content is considered very important by the software companies for improving 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	• Knowledge of the basic concepts of programming • Ability to apply different paradigms to different problem domains	• Written exam grade	40%
10.5 Seminar/lab activities	• Ability to study and analyse literature regarding a concrete programming language	• Paper grade • Seminar/lab attendance • Default	40% 10% 10%
Paper evaluation detailed		• Scheduling and presentation • Presentation phase 1 • Presentation phase 2 • Compliance to general requirements • Structure • Content	2 x 5% 10% 10% 10% 20% 40%
10.6 Minimum performance standards			
• At least grade 5 (from a scale of 1 to 10) at written exam, and paper work.			

Date	Signature of course coordinator	Signature of seminar coordinator
30.06.2021	Prof. dr. Bazil Pârv	Prof. dr. Bazil Pârv

Date of approval	Signature of the head of doctoral school
07.07.2021	Prof. dr. Gabriela Czibula