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	**Software Engineering**

2. Information regarding the discipline

2.1 Name of the discipline	**Programming paradigms**						
2.2 Course coordinator	**Prof.PhD. Bazil Parv**						
2.3 Seminar coordinator	**Prof.PhD. Bazil Parv**						
2.4. Year of study	**1**	2.5 Semester	**1**	2.6. Type of evaluation	**E**	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**
3.4 Total hours in the curriculum	**42**	Of which: 3.5 course	**28**	3.6 seminar/laboratory	**14**

Time allotment:	Hours
Learning using manual, course support, bibliography, course notes	**30**
Additional documentation (in libraries, on electronic platforms, field documentation)	**30**
Preparation for seminars/labs, homework, papers, portfolios and essays	**70**
Tutorship	**14**
Evaluations	**14**
Other activities:	-

3.7 Total individual study hours	**158**
3.8 Total hours per semester	**200**
3.9 Number of ECTS credits	**8**

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 • know the main features of different programming paradigms: procedural, object-oriented, functional, logical, component-based, event-based • have a good understanding of the following terms: variable, object, data type, component, interface, polymorphism; • learn the similarities and differences between component-based programming and object-oriented programming in the frame of inheritance and composition issues; • understand the importance of component's scale, granularity, and architectural aspects;

8. Content

8.1 Course	Teaching methods	Remarks
1. *Programming paradigms*. Definitions. Main programming paradigms. Programming styles	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
2. *Software component definition*. Basic terms: software component, object, module, interface, software reuse. Standardization issues	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
3. *Components, interfaces, and re-entrance*. Different interface types for components. The constituents of a contract	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
4. *Components, interfaces, and re-entrance*. The client-server relation in procedural-, object-, and component-based systems.	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
5. *Polymorphism*. The data type concept in a programming language context. Type extensibility and independent extensibility of software components	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
6. *Polymorphism*. Safety issues in component-based systems. Interfaces and contract evolution	• Interactive exposure • Explanation • Conversation	

	• Didactical demonstration	
7. *Reuse mechanisms: inheritance and object composition*. Kinds of inheritance. Using inheritance: advantages and pitfalls	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
8. *Reuse mechanisms: inheritance and object composition*. Interface inheritance. Delegation, composition, inheritance, and polymorphism	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
9. *Architectural issues in component-based systems*. Reusing components. Classifying components with respect to their reuse	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
10. *Architectural issues in component-based systems*. Design patterns. Frameworks. Software architecture in component-based systems	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
11. *Programming styles in a component world*. Connexion-oriented programming. Events and messages	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
12. *Programming styles in a component world*. Dispatch interfaces and metaprogramming. Scripting	• Interactive exposure • Explanation • Conversation • Didactical demonstration	
13. *Wiring models for software components*. General features of a wiring model. OMG CORBA, OMA	• Interactive exposure • Conversation	
14. *Wiring models for software components*. Sun Java: JavaBeans, Enterprise Java Beans. Microsoft: COM, ActiveX, COM+, .NET. Final review	• Interactive exposure • Conversation	

Bibliography

1. D'SOUZA, DESMOND FRANCIS - WILLS, ALAN CAMERON: Objects, Components, and Frameworks with UML : The Catalysis Approach, Addison-Wesley, 1999.
2. SZYPERSKI, CLEMENS: Component Software. Beyond Object-Oriented Programming, Addison-Wesley (1st ed. 1998, 2nd ed. 2002).
3. STROUSTRUP, BJARNE The C++ Programming Language Special Edition, Addison-Wesley, 2000 chapter 2
4. VAN ROY, PETER; HARIDI, SEIF Concepts, Techniques and Models of Computer Programming, MIT Press, 2004
5. WEGNER, PETER; Concepts and paradigms of OOP, OOPSLA ’89 Keynote talk

8.2 Seminar / laboratory	Teaching methods	Remarks
1. Establishing the paper title	Conversation, debate, case studies, presentations	Seminar is organized as a total of 14 hours – 2 hours every other week
2. Establish the project title	Conversation, debate, case studies, examples	
3. Paper presentations & project progress reports	Exposure, debate, case studies, examples	
4. Paper presentation & project progress reports	Exposure, debate, case studies, examples	
5. Paper presentations & project progress reports	Exposure, debate, case studies, examples	
6. Paper presentations & project progress reports	Exposure, debate, case studies, examples	
7. Project presentation	Exposure, live demos	

Bibliography

Students will serch 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	• knowing the basic concepts of programming • applying different paradigms to different problem domains	Written exam	40%
10.5 Seminar/lab activities	• be able to study and review literature regarding programming paradigms • be able to solve a problem using different paradigms	• Paper work • Project work • Seminar/lab attendance • Default	20% 20% 10% 10%
10.6 Minimum performance standards			
• At least grade 5 (from a scale of 1 to 10) at written exam, paper and project work.			

Date	Signature of course coordinator	Signature of seminar coordinator
April 30, 2014	Prof.PhD. Bazil PARV	Prof.PhD. Bazil PARV
Date of approval		Signature of the head of department
....		Prof.PhD. Bazil PARV