

## SYLLABUS

### 1. Information regarding the programme

1.1 Higher education institution	<b>Babeş Bolyai University</b>
1.2 Faculty	<b>Faculty of Mathematics and Computer Science</b>
1.3 Department	<b>Department of Computer Science</b>
1.4 Field of study	<b>Computer Science</b>
1.5 Study cycle	<b>Master</b>
1.6 Study programme / Qualification	<b>High Performance Computing and Big Data Analytics</b>

### 2. Information regarding the discipline

2.1 Name of the discipline		<b>Programming paradigms</b>					
2.2 Course coordinator		<b>Prof.PhD. Bazil Parv</b>					
2.3 Seminar coordinator		<b>Prof.PhD. Bazil Parv</b>					
2.4. Year of study	<b>1</b>	2.5 Semester	<b>1</b>	2.6. Type of evaluation	<b>E</b>	2.7 Type of discipline	<b>compulsory</b>

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

3.1 Hours per week	<b>3</b>	Of which: 3.2 course	<b>2</b>	3.3 seminar/laboratory	<b>1</b>
3.4 Total hours in the curriculum	<b>42</b>	Of which: 3.5 course	<b>28</b>	3.6 seminar/laboratory	<b>14</b>
Time allotment:					hours
Learning using manual, course support, bibliography, course notes					<b>30</b>
Additional documentation (in libraries, on electronic platforms, field documentation)					<b>30</b>
Preparation for seminars/labs, homework, papers, portfolios and essays					<b>70</b>
Tutorship					<b>14</b>
Evaluations					<b>14</b>
Other activities: .....					<b>-</b>
3.7 Total individual study hours					<b>158</b>
3.8 Total hours per semester					<b>200</b>
3.9 Number of ECTS credits					<b>7</b>

### 4. Prerequisites (if necessary)

4.1. curriculum	<ul style="list-style-type: none"> <li>• Fundamentals of Programming</li> <li>• Object-Oriented Programming</li> <li>• Functional and Logic Programming</li> </ul>
4.2. competencies	<ul style="list-style-type: none"> <li>• Average programming skills</li> </ul>

## 5. Conditions (if necessary)

5.1. for the course	<ul style="list-style-type: none"> <li>• Videoprojector</li> </ul>
5.2. for the seminar /lab activities	<ul style="list-style-type: none"> <li>• Computers</li> </ul>

## 6. Specific competencies acquired

<b>Professional competencies</b>	<ul style="list-style-type: none"> <li>• Understanding and working with basic concepts in computer programming;</li> <li>• Capability of analysis and synthesis;</li> <li>• Proficient use of tools and languages specific to software systems development</li> <li>• Knowing the specifics of main programming paradigms.</li> </ul>
<b>Transversal competencies</b>	<ul style="list-style-type: none"> <li>• Professional communication skills; concise and precise description, both oral and written, of professional results,</li> <li>• Independent work capabilities; able to fulfill different roles</li> <li>• Antepreneurial skills;</li> </ul>

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

7.1 General objective of the discipline	<ul style="list-style-type: none"> <li>• Know and understand fundamental concepts of programming.</li> <li>• Be able to apply different programming paradigms to different programming projects</li> </ul>
7.2 Specific objective of the discipline	<p>At the end of the course, students</p> <ul style="list-style-type: none"> <li>• know the main features of different programming paradigms: procedural, object-oriented, functional, logical, component-based, event-based</li> <li>• have a good understanding of the following terms: variable, object, data type, component, interface, polymorphism;</li> <li>• learn the similarities and differences between component-based programming and object-oriented programming in the frame of inheritance and composition issues;</li> <li>• understand the importance of component's scale, granularity, and architectural aspects;</li> </ul>

## 8. Content

8.1 Course	Teaching methods	Remarks
1. Programming paradigms. Definitions. Main programming paradigms. Programming styles	Exposure,description, explanation, debate and dialogue, discussion of case studies	
2. Software component definition. Basic terms:	explanation, debate	

software component, object, module, interface, software reuse. Standardization issues	and dialogue, discussion of case studies	
3. Components, interfaces, and re-entrance. Different interface types for components. The constituents of a contract	Exposure,description, explanation	
4. Components, interfaces, and re-entrance. The client-server relation in procedural-, object-, and component-based systems. Components and distributed systems	Exposure,description, explanation	
5. Polymorphism. The data type concept in a programming language context. Type extensibility and independent extensibility of software components	Exposure,description, explanation	
6. Polymorphism. Safety issues in component-based systems. Interfaces and contract evolution	Exposure,description, explanation	
7. Reuse mechanisms: inheritance and object composition. Kinds of inheritance. Using inheritance: advantages and pitfalls	Exposure,description, explanation, discussion of case studies	
8. Reuse mechanisms: inheritance and object composition. Interface inheritance. Delegation, composition, inheritance, and polymorphism	Exposure,description, explanation, discussion of case studies	
9. Architectural issues in component-based systems. Reusing components. Classifying components with respect to their reuse	Exposure,description, explanation, discussion of case studies	
10. Architectural issues in component-based systems. Design patterns. Frameworks. Software architecture in component-based systems	Exposure,description, explanation, discussion of case studies	
11. Programming styles in a component world. Connexion-oriented programming. Events and messages	Exposure,description, explanation, discussion of case studies	
12. Programming styles in a component world. Dispatch interfaces and metaprogramming. Scripting	Exposure,description, explanation, discussion of case studies	
13. Wiring models for software components. General features of a wiring model. OMG CORBA, OMA	Exposure,description, explanation, discussion of case studies	
14. Wiring models for software components. Sun Java: JavaBeans, Enterprise Java Beans. Microsoft: COM, ActiveX, COM+, .NET. Final review	Exposure,description, explanation, discussion of case studies	
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. Establish paper title	Conversation, debate, case studies	Seminar is organized as a total of 7 hours – 2 hours every other week
2. Establish 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	
<b>Bibliography</b> Students will search and use programming paradigms documentation on the web, using main CS databases The ELISA project <a href="http://jklunder.home.xs4all.nl">http://jklunder.home.xs4all.nl</a>		

### 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

<ul style="list-style-type: none"> <li>• This course follows the IEEE and ACM Curricula Recommendations for Software Engineering studies;</li> <li>• Courses with similar content are taught in the major universities in Romania offering similar study programs;</li> <li>• Course content is considered very important by the software companies for improving average software development skills</li> </ul>
--

### 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 concepts of programming; - apply different programming paradigms to different problem domains	Written exam	40%
10.5 Seminar/lab activities	- be able to study and review literature regarding	-Paper work -Project work	20% 20%

	programming paradigms - be able to solve a problem using different programming paradigms	-Seminar/lab attendance -Default	10% 10%
<b>10.6 Minimum performance standards</b>			
<ul style="list-style-type: none"> <li>At least grade 5 (from a scale of 1 to 10) at written exam, paper and project work.</li> </ul>			

Date

Signature of course coordinator

Signature of seminar coordinator

October 1, 2012

Prof.PhD. Bazil PARV

Prof.PhD. Bazil PARV

Date of approval

Signature of the head of department

.....

.....