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	**Bachelor**
1.6 Study programme / Qualification	**Computer Science**

2. Information regarding the discipline

2.1 Name of the discipline (en) (ro)			**Principles of Performance Oriented Coding**				
2.2 Course coordinator			**Lect. PhD. Radu D. Găceanu**				
2.3 Seminar coordinator							
2.4. Year of study	**3**	.5 Semester	**5**	.6. Type of evaluation	**C**	.7 Type of disciplin	**Optional**
2.8 Code of the discipline							

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	14
Additional documentation (in libraries, on electronic platforms, field documentation)	8
Preparation for seminars/labs, homework, papers, portfolios and essays	14
Tutorship	14
Evaluations	8
Other activities:	0

3.7 Total individual study hours	58
3.8 Total hours per semester	100
3.9 Number of ECTS credits	4

4. Prerequisites (if necessary)

4.1. curriculum	• Object Oriented Programming • Advanced Programming Methods
4.2. competencies	• Average Java programming skills

5. Conditions (if necessary)

5.1. for the course	• Projector
5.2. for the seminar /lab activities	• Laboratory with internet access and ability to use personal laptops

6. Specific competencies acquired

Professional competencies	C1.5 Development of program units and corresponding documentation
Transversal competencies	CT2 Efficient fulfillment of organized activities in an interdisciplinary group and development of empathic abilities of interpersonal communication, relationship and collaboration with various groups

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

7.1 General objective of the discipline	• To acquire a deeper insight of Java Core Technologies
7.2 Specific objective of the discipline	• To attain an enhanced level of object oriented design principles (in Java) • To get a good grasp of Concurrent Programming in Java • To be a solid base for preparing to become a Java certified programmer

8. Content

8.1 Course	Teaching methods	Remarks
1. Build systems and distributed version control systems	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
2. Advanced Java class design (1)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
3. Advanced Java class design (2)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
4. Design patterns pitfalls (1)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
5. Design patterns pitfalls (2)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
6. String processing and regular expressions in practice	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	

7. Collections (and third party collection libraries)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
8. Functional programming (lambdas) in imperative languages	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
9. Java I/O	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
10. Concurrency (in small and large systems) (1)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
11. Concurrency (in small and large systems) (2)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
12. Concurrency (in small and large systems) (3)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
13. Concurrency (in small and large systems) (4)	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	
14. Exam	Exposition: presentation, explanations, practical examples, demonstrations, case-study discussions	

Bibliography

1. Jeanne Boyarsky, Scott Selikoff, *OCA: Oracle Certified Associate Java SE 8 Programmer I Study Guide*, John Wiley & Sons, Dec 11, 2014

2. Jeanne Boyarsky, Scott Selikoff, *OCP: Oracle Certified Professional Java SE 8 Programmer II Study Guide,* John Wiley & Sons, Dec 14, 2015

3. Joshua Bloch, *Effective Java (2nd Edition)*, Createspace Independent Pub, Oct 2, 2014

4. Joshua Bloch, Neal Gafter, *Java puzzlers: traps, pitfalls, and corner cases,* Addison-Wesley, 2005

5. Tim Peierls, Brian Goetz, Joshua Bloch, Joseph Bowbeer, Doug Lea, David Holmes, *Java Concurrency in Practice,* Pearson Education, May 9, 2006

8.2 Seminar / laboratory	Teaching methods	Remarks
1. Build systems and distributed version control systems	Explanation, examples, dialog, case-studies	
2. Advanced Java class design (1)	Explanation, examples, dialog, case-studies	
3. Advanced Java class design (2)	Explanation, examples, dialog, case-studies	
4. Design patterns pitfalls (1)	Explanation, examples, dialog, case-studies	
5. Design patterns pitfalls (2)	Explanation, examples, dialog, case-studies	

6. String processing and regular expressions in practice	Explanation, examples, dialog, case-studies	
7. Collections (and third party collection libraries)	Explanation, examples, dialog, case-studies	
8. Functional programming (lambdas) in imperative languages	Explanation, examples, dialog, case-studies	
9. Java I/O	Explanation, examples, dialog, case-studies	
10. Concurrency (in small and large systems) (1)	Explanation, examples, dialog, case-studies	
11. Concurrency (in small and large systems) (2)	Explanation, examples, dialog, case-studies	
12. Concurrency (in small and large systems) (3)	Explanation, examples, dialog, case-studies	
13. Concurrency (in small and large systems) (4)	Explanation, examples, dialog, case-studies	
14. Exam	Explanation, examples, dialog, case-studies	

Bibliography

1. Jeanne Boyarsky, Scott Selikoff, *OCA: Oracle Certified Associate Java SE 8 Programmer I Study Guide*, John Wiley & Sons, Dec 11, 2014

2. Jeanne Boyarsky, Scott Selikoff, *OCP: Oracle Certified Professional Java SE 8 Programmer II Study Guide,* John Wiley & Sons, Dec 14, 2015

3. Joshua Bloch, *Effective Java (2nd Edition)*, Createspace Independent Pub, Oct 2, 2014

4. Joshua Bloch, Neal Gafter, *Java puzzlers: traps, pitfalls, and corner cases,* Addison-Wesley, 2005

5. Tim Peierls, Brian Goetz, Joshua Bloch, Joseph Bowbeer, Doug Lea, David Holmes, *Java Concurrency in Practice,* Pearson Education, May 9, 2006

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 Curricula Recommendations for Computer Science studies
- The course is very well appreciated by the software industry the content being set up in very close collaborations with various software companies

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)

10.4 Course	- understanding the concepts and language features presented at the course	written exam	30%
10.5 Seminar/lab activities	- implementing course concepts and algorithms	Lab assignments	70%

10.6 Minimum performance standards

At least grade 5 (1 to 10 scale) at all activities seminar/lab, written exam, practical exam (and the final grade at least 5).

Date	Signature of course coordinator	Signature of seminar coordinator
....	Lect. PhD. Radu D. Găceanu	....

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