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** **Principiile Programării Orientate spre Performanță**				
2.2 Course coordinator			**Boris Vleju, PhD**				
2.3 Seminar coordinator			**Boris Vleju, PhD**				
2.4. Year of study	**3**	2.5 Semester	**5**	2.6. Type of evaluation	**C**	2.7 Type of discipline	**Optional**
2.8 Code of the discipline			**MLE5109**				

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

3.1 Hours per week	**5**	Of which: 3.2 course	**2**	3.3 seminar/laboratory	**3**
3.4 Total hours in the curriculum	**70**	Of which: 3.5 course	**28**	3.6 seminar/laboratory	**42**
Time allotment:					hours
Learning using manual, course support, bibliography, course notes					5
Additional documentation (in libraries, on electronic platforms, field documentation)					6
Preparation for seminars/labs, homework, papers, portfolios and essays					10
Tutorship					4
Evaluations					5
Other activities:					0
3.7 Total individual study hours	**30**				
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	• Basic notions and 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	C1.1 Adequate description of programming paradigms and specific language mechanisms, as well as identifying the difference between semantic and syntactic aspects. C1.2 Explanation of some existing software applications, on levels of abstraction (architecture, packages, classes, methods) using basic knowledge adequately. C1.3 Development of appropriate source codes and unit testing of components in Java, based on given design specifications. C1.5 Development of program units and corresponding documentation. C4.3 Identify appropriate models and methods for solving real problems.
Transversal competencies	CT1 Apply rules to: organized and efficient work, responsabilities of didactical and scientifical activities and creative capitalization of own potential, while respecting principles and rules for professional ethics. CT2 Efficient fulfillment of organized activities in an interdisciplinary group and development of empathic abilities of interpersonal communication, relationship and collaboration with various groups. CT3 Use efficient methods and techniques for learning, knowledge gaining, and research and develop capabilities for capitalization of knowledge, accomodation to society requirements and communication in English.

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 • To get an inside knowledge of how these principles are applied în software systems for financial markets
7.2 Specific objective of the discipline	• To attain an enhanced level of object oriented design principles • To get a good grasp of Concurrent Programming in Java • To identify when, where, why and how the performance of a program can be enhanced • To be a solid base for preparing to become a Java certified programmer

8. Content

8.1 Course	Teaching methods	Remarks
1. Overview and requirements	Exposure: description, explanation,	

	examples, discussion of case studies	
2. Object Oriented and SOLID principles recap	Exposure: description, explanation, examples, discussion of case studies	
3. Collections and third party collection libraries	Exposure: description, explanation, examples, discussion of case studies	
4. Collections and third party collection libraries	Exposure: description, explanation, examples, discussion of case studies	
5. Functional programming (lambdas) in imperative languages	Exposure: description, explanation, examples, discussion of case studies	
6. Java memory model - GC, JIT	Exposure: description, explanation, examples, discussion of case studies	
7. Communication protocols	Exposure: description, explanation, examples, discussion of case studies	
8. Software systems for financial markets	Discussion of case studies, invited lecture	
9. Concurrency in small and large systems	Exposure: description, explanation, examples, discussion of case studies	
10. Concurrency in small and large systems	Exposure: description, explanation, examples, discussion of case studies	

11. Concurrency in small and large systems	Exposure: description, explanation, examples, discussion of case studies	
12. Concurrency in small and large systems	Exposure: description, explanation, examples, discussion of case studies	
13. Design patterns pitfalls	Exposure: description, explanation, examples, discussion of case studies	
14. Exam	Evaluation	

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 11 Programmer I Study Guide*, John Wiley & Sons, 2020

3. Joshua Bloch, *Effective Java (3rd Edition)*, Addison-Wesley Professional, 2018

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*, Addison-Wesley Professional, 2006

6. Vaskaran Sarcar, *Java Design Patterns (A Tour with 23 Gang of Four Design Patterns in Java)*, Apress, 2016

7. Robert C. Martin, *Clean Code (A Handbook of Agile Software Craftsmanship)*, Pearson Education, 2009

8.2 Seminar / laboratory	Teaching methods	Remarks
1. Application and process performance: setup & continuous integration	Explanation, examples, dialog, case-studies	
2. High performance collections – profiling in practice	Explanation, examples, dialog, case-studies	
3. Lambdas performance profiling	Explanation, examples, dialog, case-studies	
4. Concurrency in practice: communication system	Explanation, examples, dialog, case-studies	
5. Concurrency in practice: communication system	Explanation,	

	examples, dialog, case-studies	
6. Producer-consumer	Explanation, examples, dialog, case-studies	
7. Recap & Preparation for 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 11 Programmer I Study Guide*, John Wiley & Sons, 2020

3. Joshua Bloch, *Effective Java (3rd Edition)*, Addison-Wesley Professional, 2018

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*, Addison-Wesley Professional, 2006

6. Vaskaran Sarcar, *Java Design Patterns (A Tour with 23 Gang of Four Design Patterns in Java)*, Apress, 2016

7. Robert C. Martin, *Clean Code (A Handbook of Agile Software Craftsmanship)*, Pearson Education, 2009

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 during the course	Written exam or quizzes during the course (+ extra points accumulated from course activity and coding puzzles)	40%
10.5 Seminar/lab activities	Implementing course concepts	Lab assignments	60%
10.6 Minimum performance standards			
• At least grade 5 (1 to 10 scale) at all activities seminar/lab, written exam.			

- The final grade must be at least 5.

Date	Signature of course coordinator	Signature of seminar coordinator
06.02.2023	Boris Vleju, Phd	Boris Vleju, Phd

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