SYLLABUS

Advanced Programming Techniques

University year 2025-2026

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	Departament of Computer Science
1.4. Field of study	Computer Science
1.5. Study cycle	Bachelor
1.6. Study programme/Qualification	Artificial Intelligence
1.7. Form of education	Full time

2. Information regarding the discipline

2.1. Name of the discipline	**Advanced Programming Techniques**					Discipline code	**MLE5258**
2.2. Course coordinator					Assoc. Prof. PhD Bocicor Maria Iuliana		
2.3. Seminar coordinator					Assoc. Prof. PhD Bocicor Maria Iuliana		
2.4. Year of study	2	2.5. Semester	3	2.6. Type of evaluation	E	2.7. Discipline regime	Compulsory

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

3.1. Hours per week	**6**	of which: 3.2 course	**2**	3.3 seminar/laboratory/project	**2 sem 2 lab**
3.4. Total hours in the curriculum	84	of which: 3.5 course	28	3.6 seminar/laboratory/project	**28 + 28**
Time allotment for individual study (ID) and self-study activities (SA)					**hours**
Learning using manual, course support, bibliography, course notes (SA)					10
Additional documentation (in libraries, on electronic platforms, field documentation)					5
Preparation for seminars/labs, homework, papers, portfolios and essays					14
Tutorship					5
Evaluations					7
Other activities:					
3.7. Total individual study hours	**41**				
3.8. Total hours per semester	**125**				
3.9. Number of ECTS credits	**5**				

4. Prerequisites (if necessary)

4.1. curriculum	Fundamentals of Programming, Object Oriented Programming, Data Structures and Algorithms
4.2. competencies	Average programming skills in a high level programming language

5. Conditions (if necessary)

5.1. for the course	• Classroom with projector
5.2. for the seminar /lab activities	• Laboratory with computers; Java, C# and programming languages, IntelliJ IDEA/Eclipse, Visual Studio IDE • Classroom with projector

6.1. Specific competencies acquired [1]

[1] One can choose either competences or learning outcomes, or both. If only one option is chosen, the row related to the other option will be deleted, and the kept one will be numbered 6.

Professional/essential competencies	• Analysis of software specifications. • Definition of software architecture. • Development and analysis of algorithms for solving problems.
Transversal competencies	• Application of rigorous and efficient work rules, manifestation of responsible attitudes towards the didactic-scientific field, to bring optimal and creative values to own potential in specific situations, with respect to professional ethics principles and norms. • Showing initiative.

6.2. Learning outcomes

Knowledge	• The student has the necessary knowledge for the development of software programs and applications in the Java and C# programming languages, and for the information processing. • The student has the ability to develop, design and create new applications using best practices of the field.
Skills	• The student has the ability to apply general rules to specific problems and produce relevant solutions. • The student is able to combine diverse information to formulate solutions and develop ideas for new products and applications.
Responsibility and autonomy:	• The student has the ability to work independently to write medium scale Java/C# programs using GUIs and to use classes written by other programmers when constructing their systems. • The student has the necessary skills to develop GUI applications using architectural templates suitable for specific user interaction applications.

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

7.1 General objective of the discipline	• To prepare an object-oriented design of small/medium scale problems and to learn the Java programming language, as well as to create graphical user interfaces.
7.2 Specific objective of the discipline	• To use object-oriented concepts in program analysis and design. • To use and implement solutions in the Java programming language. • To create GUI for the given requirements. • To apply design patterns in various contexts. • To use classes written by other programmers when constructing their • systems.

8. Content

8.1 Course	Teaching methods	Remarks
1. Introduction in Java	• Interactive exposure	

• Platform • Language syntax • Data types. Arrays • Examples	• Explanation • Conversation • Examples • Didactical demonstration	
2. Classes, inheritance • Classes • Object construction • Methods • Inheritance, polymorphism • Abstract classes, interfaces		
3. Generic types, collections in Java • Generic methods • Type erasure • Generic classes and subtyping • Wildcards • Java Collections Framework		
4. Exceptions, Java I/O, JUnit • Exceptions • Java I/O, streams, serialization • JUnit		
5. JDBC, Functional programming • JDBC API • Java 8 features: Lambda expressions, Java 8 Streams		
6. Graphical User Interfaces • JavaFX applications, scenes, layouts, UI controls • Events		
7. Graphical User Interfaces • Processing events • Model-View-Controller FXML		
8. Java Reflection, Concurrency • Java Reflection API • Concurrency: processes, threads, multithreaded programming in Java		
9. Concurrency • Threads in Java • Thread synchronization • Concurrent applications in Java		
10. Design Patterns • Creational patterns • Structural patterns • Behavioural patterns		
Design Patterns (cont.), Introduction in C# and .NET		
11. C# and .NET • The .NET Architecture • The C# programming language • Classes in C# • Generics • Delegates • Events • Lambda expressions • LINQ		
12. Revision • Revision of the most important topics covered by the course		

• Examination guide		
Bibliography 1. James Gosling, Bill Joy, Guy Steele, Gilad Bracha, Alex Buckley. 2. Eckel, B. Thinking in Java, 4th edition, Prentice Hall, 2006. 3. Eckel, B. Thinking in Patterns with Java, 2004. MindView, Inc. 4. E. Gamma, R. Helm, R. Johnson, J. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Longman Publishing, 1995. 5. The Java Tutorials: https://docs.oracle.com/javase/tutorial/ 6. Joseph Albahari and Ben Albahari, C# 4.0 in a Nutshell, Fourth Edition, O'Reilley, 2010.		
8.2 Seminar	Teaching methods	Remarks
1. Simple problems in Java. Classes.	• Interactive exposure • Explanation • Conversation • Examples • Didactical demonstration	The seminar is structured as a 2 hour class, every 2 weeks.
2. Layered architecture, generics, inheritance.		
3. Inheritance, interfaces, collections.		
4. Serialization, files.		
5. JDBC, Junit.		
6. Java 8 streams.		
7. JavaFX - Graphical User Interfaces.		
8. JavaFX - Graphical User Interfaces. FXML.		
9. Concurrency, threads.		
10. Design Patterns.		
11. C# - layered architecture, generics, inheritance.		
12. C# - LINQ.		
13. C# - Graphical User Interfaces.		
14. Examination example.		
8.3 Laboratory		
1. Setting up JDK, JRE and JVM, as well as an IDE of choice. Simple problems in Java.	• Explanation • Conversation	The laboratory is structured as a 2 hour class, every 2 weeks.
2. Layered architecture, inheritance, generics.		
3. Files, serialization, exceptions.		
4. Junit, JDBC.		
5. Java 8 features, Java 8 streams.		
6. JavaFX - Graphical User Interfaces.		
7. Laboratory test.		
8. JavaFX - Graphical User Interfaces (FXML).		
9. Concurrency, threads.		
10. C# - layered architecture, generics, inheritance.		
11. Laboratory test.		
12. C# - LINQ.		
Bibliography 1. James Gosling, Bill Joy, Guy Steele, Gilad Bracha, Alex Buckley. 2. Eckel, B. Thinking in Java, 4th edition, Prentice Hall, 2006. 3. Eckel, B. Thinking in Patterns with Java, 2004. MindView, Inc. 4. E. Gamma, R. Helm, R. Johnson, J. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Longman Publishing, 1995.		

5. The Java Tutorials: https://docs.oracle.com/javase/tutorial/
6. Joseph Albahari and Ben Albahari, C# 4.0 in a Nutshell, Fourth Edition, O'Reilley, 2010.

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 follows the ACM Curricula Recommendations for Computer Science studies.
- The content of the course is considered by the software companies as important for average software development skills.

10. Evaluation

Activity type	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Percentage of final grade
10.4 Course	The correctness and completeness of the accumulated knowledge and the capacity to design and implement correct Java programs.	Written examination (examination session)	30%
10.5 Seminar/laboratory	Ability to use course concepts in solving real problems.	Practical examination (examination session)	30%
	Correctness of delivered laboratory assignments and laboratory tests.	Laboratory assignments. Laboratory test. Observation during the semester.	40%

10.6 Minimum standard of performance

- Each student has to prove that they acquired an acceptable level of knowledge and understanding of the core concepts taught in the class, that they are capable of using knowledge in a coherent form, that they have the ability to establish certain connections and to use the knowledge in solving different problems in Java.
- For participating at the examination attendance is compulsory for seminar and for laboratory activities, as follows: minimum 5 attendances for seminar and minimum 6 attendances for laboratory activities.
- Successfully passing of the examination is conditioned by a minimum grade of 5 for each of the following: practical examination, written examination and final grade.

11. Labels ODD (Sustainable Development Goals)[2]

Not applicable.

[2] Keep only the labels that, according to the *Procedure for applying ODD labels in the academic process*, suit the discipline and delete the others, including the general one for *Sustainable Development* – if not applicable. If no label describes the discipline, delete them all and write „*Not applicable*.".

Date:
15.04.2025

Signature of course coordinator

Assoc. Prof. PhD. Bocicor Maria Iuliana

Signature of seminar coordinator

Assoc. Prof. PhD. Bocicor Maria Iuliana

Date of approval:
...

Signature of the head of department

Assoc.prof.phd. Adrian STERCA