SYLLABUS

Advanced methods of programming software applications

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	Mathematics
1.5. Study cycle	Bachelor
1.6. Study programme/Qualification	Mathematics Computer Science
1.7. Form of education	Full time

2. Information regarding the discipline

2.1. Name of the discipline	**Advanced methods of programming software applications**					Discipline code	**MLE5235**
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	C	2.7. Discipline regime	Compulsory

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

3.1. Hours per week	**4**	of which: 3.2 course	**2**	3.3 seminar/laboratory/project	**1 sem 1 lab**
3.4. Total hours in the curriculum	56	of which: 3.5 course	28	3.6 seminar/laboratory/project	**14 + 14**
Time allotment for individual study (ID) and self-study activities (SA)					**hours**
Learning using manual, course support, bibliography, course notes (SA)					20
Additional documentation (in libraries, on electronic platforms, field documentation)					10
Preparation for seminars/labs, homework, papers, portfolios and essays					28
Tutorship					4
Evaluations					7
Other activities:					
3.7. Total individual study hours	**69**				
3.8. Total hours per semester	**125**				
3.9. Number of ECTS credits	**5**				

4. Prerequisites (if necessary)

4.1. curriculum	Algorithms and Programming, Object Oriented Programming basics, Data Structures
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. 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	• Development and analysis of algorithms for solving problems. • Programming in high level languages (Java, C#).
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. • Use of efficient information resources and techniques to learn and develop the professional abilities in Romanian language and in an international language.

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 • Platform • Language syntax • Data types. Arrays • Examples	• Interactive exposure • 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		
11. Design Patterns (cont.), Introduction in C# and .NET		
12. C# and .NET • The .NET Architecture • The C# programming language • Classes in C# • Generics • Delegates • Events • Lambda expressions • LINQ		
13. 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. Layered architecture. 2. Inheritance, interfaces, packages, iterators. 3. Generics, collections, exceptions. 4. Serialization, files, JDBC. 5. Graphical User Interfaces with JavaFX. 6. Concurrency, threads.	• Interactive exposure • Explanation • Conversation • Examples • Didactical demonstration	The seminar is structured as a 2 hour class, every 2 weeks.

7. Design patterns.		
8.3 Laboratory		
1. Setting up JDK, JRE and JVM, as well as an IDE of choice. Simple problems in Java. 2. Layered architecture, generics, exceptions. 3. Files, serialization, JUnit. 4. JDBC, functional programming (Java 8 streams). 5. Laboratory test. 6. Graphical User Interfaces. 7. Practical examination.	• Explanation • Conversation	The laboratory is structured as a 2 hour class, every 2 weeks.

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 (C)	30%
10.5 Seminar/laboratory	Ability to use course concepts in solving real problems.	Practical examination (C)	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.

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

[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*.”.