COURSE DESCRIPTION

Data Structures

Academic year 2026-2027

1. Programme-related data

1.1. Higher Education Institution	Babeș – Bolyai University
1.2. Faculty	Mathematics and Computer Science
1.3. Department	Department of Computer Science
1.4. Field	Mathematics
1.5. Level of study	Bachelor
1.6. Degree programme / Qualification	Mathematics and Computer Science
1.7. Form of education	Full time

2. Course-related data

2.1. Course title	**Data Structures**			Course code	**MLE5105**
2.2. Course coordinator	Lect. PhD. Hotea Diana – Lucia				
2.3. Seminar coordinator	Lect. PhD. Hotea Diana – Lucia				
2.4. Year of study	1	2.5. Semester	2	2.6. Type of assessment	Colloquium exam
2.7. Course status	Compulsory			2.8. Course type	Specialisation subject

3. Total estimated time (hours per semester of teaching activities)

3.1. Number of hours per week	**4**	of which: 3.2. course	**2**	3.3. seminar/ laboratory/ project	**1S+1L**
3.4. Total of hours in the curriculum	56	of which: 3.5. course	28	3.6. seminar/ laboratory	**28**
Time allocation for individual study (IS) and self-taught activities (ST)					**hours**
Learning from textbooks, course materials, bibliography, and notes (IS)					30
Additional research in the library, on subject-specific electronic platforms, and on-site					4
Preparing seminars/ laboratories/ projects, assignments, reports, portfolios, and essays					50
Tutoring (professional guidance)					5
Examinations					5
Other activities					
3.7. Total hours of individual study (IS) and self-taught activities (ST)				**94**	
3.8. Total hours per semester				**150**	
3.9. Number of credits				**6**	

4. Prerequisites (where applicable)

4.1. curriculum-related	Algorithms and Programming
4.2 skills-related	Medium programming skills

5. Specific conditions (where applicable)

5.1. course-related	Classroom with projector
5.2. seminar/laboratory-related	

6.1. Competencies resulting from the completion of the degree programme *(as referred to in the curriculum)*[1]

[1] The professional and/or transversal skills targeted by the subject for which the course description is prepared will be copied from the curriculum of the degree programme. For each competency, the complete entry, including the competency code, will be copied with the exact wording that appears in the curriculum, without any changes.

Professional competencies	
Competency code	**Competency**
PC4	develop open source software
PC5	synthesize information
PC16	create software
Transversal competencies	
Competency code	**Competency**
TC2	Use digital devices and applications

6.2. Learning outcomes relevant to the degree programme (as referred to in the curriculum)[2]

Learning outcomes targeted by the subject		
Competency code	**Knowledge and comprehension**	**Specific academic skills**
PC4	6. The student/graduates identifies, explains, and argues fundamental concepts of data structures, algorithms, and programming paradigms, as well as computer architecture.	6. The student/graduates designs, develops, and demonstrates complex software solutions using efficient algorithms and diverse programming paradigms.
PC5	9. The student/graduate defines the concepts from basic computer science and/or applied mathematics disciplines.	9. The student/graduate identifies and applies suitable techniques to solve exercises and problems from the major disciplines of mathematics.
PC16	16. The student/graduates names, provides examples, concludes, specifies, recognizes, and critically argues methods for designing and managing complex IT projects using modern strategies.	16. The student/graduates initiates, prepares, executes, and proposes methods for developing complex IT projects.
TC2	6. The student/graduates identifies, explains, and argues fundamental concepts of data structures, algorithms, and programming paradigms, as well as computer architecture.	6.The student/graduates designs, develops, and demonstrates complex software solutions using efficient algorithms and diverse programming paradigms.

7. Subject-specific learning outcomes

Knowledge and comprehension
1. The student knows the concept of abstract data type and the most commonly used abstract data types used in application development.
2. The student understands the data structures with which these abstract data types can be implemented.

If no competency is copied from either of the two categories, the row corresponding to that category is deleted from the table.

[2] The learning outcomes relevant for the degree programme and targeted by the subject for which the course description is prepared will be listed. The entries, copied without any changes from the Curriculum by subject type (Core Subject/Specialisation Subject/Complementary Subject), are listed under the corresponding competency.

Specific academic skills
1. The student is able to design and implement applications based on the use of abstract data types.
2. The student is able to process data stored in various data structures: arrays, linked lists, hash tables, trees.
3. The student is able to design and implement algorithms that process these data structures.

8. Contents

8.1. Course	Teaching and learning methods	Remarks[3]
1. Data structures. Abstract Data Types. Algorithm analysis • Abstract Data Types and Data Structures • Pseudocode conventions • Complexities	• Exposure • Description • Examples • Didactical demonstration	
2. Arrays. Iterators • Dynamic array • Amortized complexity analysis • Interface of an iterator	• Exposure • Description • Examples • Didactical demonstration	
3. Linked Lists • Singly linked list: representation and operations • Doubly linked list: representation and operations • Iterator for linked lists	• Exposure • Description • Conversation • Didactical demonstration • Case study	
4. Abstract Data Types • ADT Set: description, domain, interface and possible representations • ADT Map: description, domain, interface and possible representations • ADT Matrix: description, domain, interface and possible representations	• Exposure • Description • Conversation • Didactical demonstration • Case study	
5. Linked Lists II • Sorted linked lists: representation and operations • Linked lists on arrays: representation and operations	• Exposure • Description • Conversation • Didactical demonstration	
6. Abstract Data Types II • ADT List: description, domain, interface and possible representations • ADT Stack: description, domain, interface and possible representations • ADT Queue: description, domain, interface and possible representations	• Exposure • Description • Conversation • Didactical demonstration • Case study	
7. Hash Table • Direct address tables	• Exposure • Description	

[3] For example, organisational aspects, recommendations for students, specific aspects relating to the course/seminar, such as inviting experts in the field, etc.

• Hash tables: description, properties • Collision resolution through separate chaining	• Conversation • Didactical demonstration	
8. Hash Table II • Collision resolution through coalesced chaining • Collision resolution through open addressing	• Exposure • Description • Conversation • Didactical demonstration	
9. **Trees. Binary Trees** • Concepts related to trees • Applications of trees • Possible representations • Tree traversals • Description and properties of binary trees • Domain and interface of ADT Binary Tree	• Exposure • Description • Conversation • Didactical demonstration	
10. Binary Trees II • Possible representations of ADT Binary Tree • Binary tree traversals: recursive/non- recursive algorithms	• Exposure • Description • Conversation • Didactical demonstration	
11. Binary Heap • Definition, representations, specific Operations • HeapSort	• Exposure • Description • Conversation • Didactical demonstration • Case study	
12. ADT Priority Queue • Description, domain and interface • Possible representations • Implementation on heap	• Exposure • Description • Conversation • Didactical demonstration	
13. Applications of the studied DS	• Conversation • Debate	
14. Evaluation		

Bibliography
1.T. Cormen, C. Leiserson, R. Rivest, C. Stein: Introduction to algorithms, Third Edition, The MIT Press, 2009
2. Clifford A. Shaffer, A Practical Introduction to Data Structures and Algorithm Analysis, Third Edition, 2010
3. N. Karumanchi: Data structures and algorithms made easy, CareerMonk Publications, 2016
4. Narasimha Karumanchi, Data Structures and Algorithms Made Easy: Data Structures and Algorithmic Puzzles,Fifth Edition, 2016
5. M. A. Weiss: Data structures and algorithm analysis in Java, Third Edition, Pearson, 2012

8.2. Laboratory	**Teaching and learning methods**	**Remarks**
		The laboratory is structured as 2 hour classes every second week. Laboratory problems assigned at a lab have to be presented in the next lab (excepting the first lab assignemnt). Every laboratory focuses on a given data structure. Students will receive a container (ADT) that has to be implemented using the given data structure.

Lab1. Discussion about solving lab problems	• Exposure • Examples • Conversation	
Lab 2. Dynamic array	• Exposure • Examples • Conversation	To be presented at Lab 3
Lab 3. Linked lists with dynamic allocation	• Exposure • Examples • Conversation	
Lab 4. Linked lists on array	• Exposure • Examples • Conversation	
Lab 5. Hash Table	• Exposure • Examples • Conversation	
Lab 6. Binary Search Tree	• Exposure • Examples • Conversation	
Lab 7. Presentation of problem from Lab 6	• Exposure • Examples • Conversation	

Bibliography

1. T. Cormen, C. Leiserson, R. Rivest, C. Stein: Introduction to algorithms, Third Edition, The MIT Press, 2009
2. Clifford A. Shaffer, A Practical Introduction to Data Structures and Algorithm Analysis, Third Edition, 2010
3. N. Karumanchi: Data structures and algorithms made easy, CareerMonk Publications, 2016
4. Narasimha Karumanchi, Data Structures and Algorithms Made Easy: Data Structures and Algorithmic Puzzles, Fifth Edition, 2016
5. M. A. Weiss: Data structures and algorithm analysis in Java, Third Edition, Pearson, 2012

8.3 Seminar	Teaching methods	Remarks
		Seminar is structured as 2 hour classes every second week.
1. ADT Bag with generic elements. Representations and implementation on an array. Iterator for ADT Bag	• Exposure • Conversation • Examples • Debate	
2. Complexities	• Exposure • Conversation • Examples • Debate	

3. Bucket sort, Lexicographic sort, radix sort. Merging two sorted singly linked lists.	• Exposure • Conversation • Examples • Debate	
4. Sorted MultiMap – representation and implementation on a singly linked list	• Exposure • Conversation • Examples • Debate	
5. Hash tables. Collision resolution through coalesced chaining	• Exposure • Conversation • Examples • Debate	
6. Binary trees.	• Exposure • Conversation • Examples • Debate	
7. Problems solved with heaps	• Exposure • Conversation • Examples • Debate	

Bibliography

1. T. Cormen, C. Leiserson, R. Rivest, C. Stein: Introduction to algorithms, Third Edition, The MIT Press, 2009
2. Clifford A. Shaffer, A Practical Introduction to Data Structures and Algorithm Analysis, Third Edition, 2010
3. N. Karumanchi: Data structures and algorithms made easy, CareerMonk Publications, 2016
4. Narasimha Karumanchi, Data Structures and Algorithms Made Easy: Data Structures and Algorithmic Puzzles, Fifth Edition, 2016
5. M. A. Weiss: Data structures and algorithm analysis in Java, Third Edition, Pearson, 2012

9. Evaluation

Type of activity	9.1 Evaluation criteria[4]	9.2 Evaluation methods[5]	9.3 Percentage in the final grade
9.4. Course	• Correctness and completeness of the assimilated knowledge • Knowledge of applying the concepts	Written evalution: written exam	70%
9.5. Lab activities	• C++ implementation of the concepts and algorithms presented at the lectures • Lab assignment documentation • Respecting the deadlines for lab presentation	Correctness of the implementation and documentation (representation, specifications, algorithms, complexities).	30%

[4] The evaluation criteria must directly reflect the learning outcomes targeted at the level of the degree programme respectively at the level of the subject. More specifically, the learning outcomes set out in the expected learning outcomes are assessed.

[5] Both final evaluation methods and ongoing evaluation strategies should be established.

9.6 Seminar Activity	• Activity during the seminar	Evaluation of seminar activity – at most 0.5 bonus points awarded for activity during the seminars.	
9.7 Minimum standard for passing			
• Knowledge of the basic concepts. Each student has to prove that he/she has acquired an acceptable level of knowledge and understanding of the domain, that he/she is capable of expressing the acquired knowledge in a coherent form, that he/she has the ability of using this knowledge for problem solving. • For participating at the written exam, a student must have at least 6 lab attendances and 5 seminar attendances. • For successfully passing the examination, a student must have at least 5 as a final grade.			

10. SDG labels (Sustainable Development Goals)[6]

	○	Sustainable Development Generic Label						
								
○	○	○	○	○	○	○	○	X
								No label applies
○	○	○	○	○	○	○	○	○

Date of entry:
22.05.2026

Signature of course coordinator

Lect. PhD. Diana – Lucia HOTEA

Signature of seminar coordinator

Lect. PhD. Diana – Lucia HOTEA

[6] Select a single label which, according to the *Implementation of SDG labels in the academic process*, best matches the subject. If the subject addresses sustainable development in a generic manner (i.e. by presenting/introducing the general framework of sustainable development, etc.), then the Sustainable Development generic label may be applied. If none of the labels describe the subject, select the last option: *"No label applies."*

Date of approval in the department:
...

Signature of the head of department

Assoc.prof.PhD. Adrian STERCA