SYLLABUS

Data Structures and algorithms

University year 2025- 2026

1. Information regarding the programme

1.1. Higher education institution	Babeş – Bolyai University
1.2. Faculty	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 (in English)
1.7. Form of education	Full time

2. Information regarding the discipline

2.1. Name of the discipline	**Data Structures and algorithms**					Discipline code	**MLE5022**
2.2. Course coordinator					Lect. PhD. Oneț-Marian Zsuzsanna		
2.3. Seminar coordinator					Lect. PhD. Oneț-Marian Zsuzsanna		
2.4. Year of study	1	2.5. Semester	2	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	**4**	of which: 3.2 course	**2**	3.3 seminar/laboratory/project	**1S + 1LP**
3.4. Total hours in the curriculum	56	of which: 3.5 course	28	3.6 seminar/laboratory/project	**28**
Time allotment for individual study (ID) and self-study activities (SA)					**hours**
Learning using manual, course support, bibliography, course notes (SA)					17
Additional documentation (in libraries, on electronic platforms, field documentation)					6
Preparation for seminars/labs, homework, papers, portfolios and essays					30
Tutorship					6
Evaluations					10
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	Fundamentals of programming
4.2. competencies	Medium programming skills

5. Conditions (if necessary)

5.1. for the course	Class room with projector
5.2. for the seminar /lab activities	

6.1. Specific competencies acquired

Professional/essential competencies	• development and maintenance of software systems • use of theoretical foundations of computer science as well as of formal models
Transversal competencies	• application of organized and efficient work rules, of responsible attitudes towards the didactic-scientific field, to bring creative value to own potential, with respect for professional ethics principles and norms • use of efficient methods and techniques to learn, inform, research and develop the abilities to bring value to knowledge, to adapt at the requirements of a dynamical society and to communicate efficiently in Romanian language and in an international language

6.2. Learning outcomes

Knowledge	The student knows: • The graduate has the necessary knowledge for using computers, developing software programs and applications, information processing.
Skills	The student is able to: • The graduate is able to identify complex problems and examine related issues to develop solving options and implement solutions. • The graduate is able to combine diverse information to formulate solutions and generate ideas for developing new products and applications.
Responsibility and autonomy:	The student has the ability to work independently to obtain: • The graduate has the ability to apply general rules to specific problems and produce relevant solutions.

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

7.1 General objective of the discipline	• Study of data structures (arrays, linked lists, heaps, hash tables, binary trees) that can be used to implement abstract data types
7.2 Specific objective of the discipline	• Study of the concept of abstract data type and the most frequently used abstract data types in application development. • Study of the data structures that can be used to implement these abstract data types. • Develop the ability to work with data stored in different data structures and to compare the complexities of their operations. • Develop the ability to choose the appropriate data structure in order to model and solve real world problems. • Acquire knowledge necessary to work with existing data structure libraries.

8. Content

8.1 Course	Teaching methods	Remarks
1. Introduction. Data structures. Abstract Data Types. • 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 • Conversation • Didactical demonstration	
3. 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 • ADT MultiMap: description, domain, interface and possible representations	• Exposure • Description • Conversation • Didactical demonstration	
4. Abstract Data Types II • ADT Stack: description, domain, interface and possible representations • ADT Queue: description, domain, interface and possible representations • ADT PriorityQueue: description, domain, interface and possible representations • ADT Deque: description, domain, interface and possible representations • ADT List : description, domain, interface and possible representations	• Exposure • Description • Conversation • Didactical demonstration	
5. 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	
6. **Linked Lists II** • Sorted linked lists: representation and operations • Circular linked lists: representation and operations • Linked lists on arrays: representation and operations	• Exposure • Description • Conversation • Didactical demonstration	
7. **Binary Heap** • Representations, specific operations. • HeapSort	• Exposure • Description • Conversation • Didactical demonstration	
8. **Hash Table** • Direct address tables • Hash tables: description, properties • Collision resolution through separate chaining	• Exposure • Description • Conversation • Didactical demonstration	
9. **Hash Table II**	• Exposure	

• Collision resolution through coalesced chaining • Collision resolution through open addressing	• Description • Conversation • Didactical demonstration	
10. **Hash Table III** • Perfect hashing • Linked hash tables • Containers represented over hash tables	• Exposure • Description • Conversation • Didactical demonstration	
11. **Trees. Binary Trees** • Concepts related to trees • Applications of trees • Description and properties of binary trees • Domain and interface of ADT Binary Tree • Possible representations of ADT Binary Tree • Binary tree traversals: recursive/non-recursive algorithms	• Exposure • Description • Conversation • Didactical demonstration	
12. Binary Search Trees • Description, properties • Representation • Operations: recursive and non-recursive algorithms • Containers represented over binary search trees	• Exposure • Description • Conversation • Didactical demonstration	
13. Balanced Binary Search Trees • AVL Trees	• Exposure • Description • Conversation • Didactical demonstration	
14. **Applications and data structure libraries in different programming languages**	• Examples • Exposure • Description • Conversation • Didactical demonstration	

Bibliography

1. T. Cormen, C. Leiserson, R. Rivest, C. Stein: Introduction to algorithms, Third Edition, The MIT Press, 2009
2. S. Skiena: The algorithms design manual, Second edition, Springer, 2008
3. N. Karumanchi: Data structures and algorithms made easy, CareerMonk Publications, 2016
4. M. A. Weiss: Data structures and algorithm analysis in Java, Third Edition, Pearson, 2012
5. R. Sedgewick: Algorithms, Addison-Wesley Publishing, 1984

8.2 Laboratory	Teaching methods	Remarks
		Laboratory is structured as 2 hour classes every second week. Laboratory problems assigned at a lab have to be presented in the next lab (exception is Lab1). Every assignment focuses on a given data structure. Students will receive a container (ADT) that has to be implemented using the given data structure.
Lab1. A1- Dynamic array - Example of a solved lab assignment (Demo)	• Exposure • Examples • Conversation	To be presented at Lab 3
Lab 2. Discussion about the Demo. Example of an extra operation	• Exposure • Examples • Conversation	During the lab students will get help with their first assignment.

Lab 3. A2 - Linked lists with dynamic allocation	• Exposure • Examples • Conversation	A1 has to be presented
Lab 4. A3 - Linked lists on array	• Exposure • Examples • Conversation	
Lab 5. A4 - Hash table	• Exposure • Examples • Conversation	
Lab 6. A5 - 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. S. Skiena: The algorithms design manual, Second edition, Springer, 2008 3. N. Karumanchi: Data structures and algorithms made easy, CareerMonk Publications, 2016 4. M. A. Weiss: Data structures and algorithm analysis in Java, Third Edition, Pearson, 2012 5. R. Sedgewick: Algorithms, Addison-Wesley Publishing, 1984		
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. Evaluating an arithmetic expression. Problems solved with binary heap.	• Exposure • Conversation • Examples • Debate	
6. Hash tables	• Exposure • Conversation • Examples • Debate	
7. Binary trees.	• Exposure • Conversation • Examples • Debate	
Bibliography 1. T. Cormen, C. Leiserson, R. Rivest, C. Stein: Introduction to algorithms, Third Edition, The MIT Press, 2009 2. S. Skiena: The algorithms design manual, Second edition, Springer, 2008 3. N. Karumanchi: Data structures and algorithms made easy, CareerMonk Publications, 2016 4. M. A. Weiss: Data structures and algorithm analysis in Java, Third Edition, Pearson, 2012 5. R. Sedgewick: Algorithms, Addison-Wesley Publishing, 1984		

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 content of this discipline is consistent with the content of the Data structures courses from other universities in Romania and abroad.
- The content of the discipline ensures the necessary fundamental knowledge needed for using abstract data types and data structures in application design.

10. Evaluation

Activity type	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Percentage of final grade
10.4 Course	• Correctness and completeness of the assimilated knowledge • Knowledge of applying the concepts	Written evalution (in the exam session): written exam	70%
10.5 Laborator	• 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%
10.6 Seminar	• Seminar activity	Active participation at the discussions during the seminar (asking and answering questions, volunteering to solve problems, etc.)	Maximum 0.5 points bonus, added to the final grade
10.7 Minimum standard of performance			

- 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 for the written exam, and minimum 5 as a final grade.

11. Labels ODD (Sustainable Development Goals)[1]

Not applicable.

Date: 15.04.2025	Signature of course coordinator Lect. PhD. Zsuzsanna ONEȚ-MARIAN	Signature of seminar coordinator Lect. PhD. Zsuzsanna ONEȚ-MARIAN

Date of approval: ...	Signature of the head of department Assoc.prof.phd. Adrian STERCA

[1] 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*.”.