

SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	Babeş-Bolyai University, Cluj-Napoca
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)	Data Structures and Algorithms Structuri de Date și Algoritmi						
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 Type of discipline	Compulsory
2.8 Code of the discipline	MLE5022						

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	1 sem + 1 lab
3.4 Total hours in the curriculum	56	Of which: 3.5 course	28	3.6 seminar/laboratory	28
Time allotment:					hours
Learning using manual, course support, bibliography, course notes					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	<ul style="list-style-type: none"> • Fundamentals of programming
-----------------	---

4.2. competencies	<ul style="list-style-type: none"> • Medium programming skills
-------------------	---

5. Conditions (if necessary)

5.1. for the course	<ul style="list-style-type: none"> • Class room with projector
5.2. for the seminar /lab activities	

6. Specific competencies acquired

Professional competencies	<p>C4.1. Definition of concepts and basic principles of computer science, and their mathematical models and theories.</p> <p>C4.3. Identification of adequate models and methods for solving real problems</p> <p>C4.5. Adoption of formal models in specific applications from different domains</p>
Transversal competencies	<p>CT1. Apply rules to: organized and efficient work, responsibilities of didactical and scientific activities and creative capitalization of own potential, while respecting principles and rules for professional ethics</p> <p>CT3. Use efficient methods and techniques for learning, gaining knowledge, researching and develop capabilities for capitalization of knowledge, accommodation to society requirements and communication in English.</p>

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

7.1 General objective of the discipline	<ul style="list-style-type: none"> • 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	<ul style="list-style-type: none"> • 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/abstract data type libraries.

8. Content

8.1 Course	Teaching methods	Remarks
<p>1. Introduction. Data structures. Abstract Data Types</p> <ul style="list-style-type: none"> • Data abstractization and encapsulation • Pseudocode conventions 	<ul style="list-style-type: none"> - Exposure - Description - Examples - Didactical demonstration 	

<ul style="list-style-type: none"> • Complexities 		
2. Arrays. Iterators <ul style="list-style-type: none"> • Dynamic array • Amortized analysis • Interface of an iterator 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
3. Abstract Data Types <ul style="list-style-type: none"> • 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 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
4. Abstract Data Types II <ul style="list-style-type: none"> • 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 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
5. Linked Lists <ul style="list-style-type: none"> • Singly linked list: representation and operations • Doubly linked list: representation and operations • Iterator for linked lists 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration - Case study 	
6. Linked Lists II <ul style="list-style-type: none"> • Sorted linked lists: representation and operations • Circular linked lists: representation and operations • Linked lists on arrays: representation and operations 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
7. Binary Heap <ul style="list-style-type: none"> • Representation, specific operations 	<ul style="list-style-type: none"> - Exposure - Description 	

<ul style="list-style-type: none"> • HeapSort 	<ul style="list-style-type: none"> - Conversation - Didactical demonstration 	
8. Hash Table <ul style="list-style-type: none"> • Direct address tables • Hash tables: description, properties • Collision resolution through separate chaining 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
9. Hash Table <ul style="list-style-type: none"> • Collision resolution through coalesced chaining • Collision resolution through open addressing 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
10. Hash tables <ul style="list-style-type: none"> • Perfect hashing • Linked hash tables • Containers represented over hash tables 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
11. Trees <ul style="list-style-type: none"> • Concepts related to trees • Applications of trees Binary Trees <ul style="list-style-type: none"> • Description, properties • Domain and interface of ADT Binary Tree • Tree traversals: recursive/non recursive algorithms. 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
12. Binary Search Trees <ul style="list-style-type: none"> • Description, properties • Representation • Operations: recursive and non-recursive algorithms • Containers represented over binary search tables 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
13. Balanced Binary Search Trees <ul style="list-style-type: none"> • AVL Trees 	<ul style="list-style-type: none"> - Exposure - Description - Conversation - Didactical demonstration 	
14. Applications and data structure libraries in different programming languages (Python, C++, Java, C#)	<ul style="list-style-type: none"> - 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 Seminar	Teaching methods	Remarks
		Seminar is structured as 2 hour classes every second week.
1. ADT Bag with generic elements. Representations and implementations on an array. Iterator for ADT Bag.	<ul style="list-style-type: none"> - Exposure - Conversation - Examples - Debate 	
2. Complexities	<ul style="list-style-type: none"> - Exposure - Examples - Debate - Conversation 	
3. Sorted Multi Map – representation and implementation on a singly linked list.	<ul style="list-style-type: none"> - Exposure - Examples - Debate - Conversation 	
4. Bucket sort, Lexicographic sort, radix sort. Merging two singly linked lists	<ul style="list-style-type: none"> - Exposure - Examples - Debate - Conversation 	
5. Hash tables – collision resolution through separate chaining	<ul style="list-style-type: none"> - Exposure - Examples - Debate - Conversation 	
6. Hash tables. Collision resolution through coalesced chaining.	<ul style="list-style-type: none"> - Exposure - Examples - Debate - Conversation 	
7. Binary Trees	<ul style="list-style-type: none"> - Exposure - Examples - Debate - Conversation 	
Bibliography		
<ol style="list-style-type: none"> 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 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 laboratory focuses on a given data structure. Students will receive a container (ADT) that has to be implemented using the given data structure.
Lab 1: Dynamic Array	- Exposure - Examples - Conversation	To be presented at Lab 3
Lab 2: Discussions about solving lab 1 assignment	- Exposure - Examples - Conversation	During the lab students will get help with their first assignment.
Lab 3: Linked lists with dynamic allocation	- Exposure - Examples - Conversation	Assignment from Lab 1 is presented
Lab 4: Linked lists on arrays	- Exposure - Examples - Conversation	Assignment from Lab 3 is presented
Lab 5: Hash Table	- Exposure - Examples - Conversation	Assignment from Lab 4 is presented
Lab 6: Binary Search Tree	- Exposure - Examples - Conversation	Assignment from Lab 5 is presented
Lab 7: Presentation of problem from Lab 6.	- Exposure - Examples - Conversation	Assignment from Lab 6 is presented
Bibliography		
<ol style="list-style-type: none"> 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

<ul style="list-style-type: none"> • The content of this discipline is consistent with the content of the Data structures and algorithms 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

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.1 Course	<ul style="list-style-type: none"> • Correctness and completeness of the assimilated knowledge • Knowledge of applying the course concepts 	Written evaluation (in the exam session): written exam	70%
10.2 Laboratory	<ul style="list-style-type: none"> • 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%
10.3 Seminar	<ul style="list-style-type: none"> • Seminar activity 	Active participation at the discussions during the seminar (asking and answering questions, volunteering to solve a problem, etc.)	Maximum 0.5 points, added to the final grade
10.4 Minimum performance standards			
<ul style="list-style-type: none"> • 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 5 seminar attendances and 6 laboratory attendances. • For successfully passing the examination, a student must have at least 5 for the laboratory and the written exam, and minimum 5 as a final grade. 			

Date

29.04.2021

Signature of course coordinator

Lect. PhD. Oneț-Marian Zsuzsanna

Signature of seminar coordinator

Lect. PhD. Oneț-Marian Zsuzsanna

Date of approval

Signature of the head of department

Prof. PhD. Diosan Laura