SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	**Babeș-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**				
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					10
Additional documentation (in libraries, on electronic platforms, field documentation)					6
Preparation for seminars/labs, homework, papers, portfolios and essays					12
Tutorship					6
Evaluations					10
Other activities:					

3.7 Total individual study hours	44
3.8 Total hours per semester	100
3.9 Number of ECTS credits	4

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. Specific competencies acquired

Professional competencies	C4.1. Definition of concepts and basic principles of computer science, and their mathematical models and theories. C4.3. Identification of adequate models and methods for solving real problems C4.5. Adoption of formal models in specific applications from different domains
Transversal competencies	CT1. Apply rules to: organized and efficient work, responsibilities of didactical and scientifical activities and creative capitalization of own potential, while respecting principles and rules for professional ethics 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.

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

7.1 General objective of the discipline	• Study of data structures that can be used to implement abstract data types (arrays, linked lists, heaps, hash tables, binary trees).
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/abstract data type libraries.

8. Content

8.1 Course	Teaching methods	Remarks
1. **Introduction. Data structures. Abstract Data Types** • Data abstractization and encapsulation • Pseudocode conventions • Complexities	- Exposure - Description - Examples - Didactical demonstration	

2. **Arrays. Iterators** • Dynamic array • Amortized 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** • Representation, 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**	- Exposure	

• Collision resolution through coalesced hashing • Collision resolution through open addressing	- Description - Conversation - Didactical demonstration	
10. Hash tables • Perfect hashing • Linked hash tables • Containers represented over hash tables	- Exposure - Description - Conversation - Didactical demonstration	
11. **Trees** • Concepts related to trees • Applications of trees **Binary Trees** • Description, properties • Domain and interface of ADT Binary Tree • Operations for ADT Binary Tree • 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 tables	- 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 (Python, C++, Java, C#)**	- 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.	- Exposure - Conversation - Examples - Debate	
2. Complexities	- Exposure - Examples - Debate - Conversation	
3. Sorted Multi Map – representation and implementation on a singly linked list.	- Exposure - Examples - Debate - Conversation	
4. Bucket sort, Lexicographic sort, radix sort. Merging two singly linked lists	- Exposure - Examples - Debate - Conversation	
5. Written test Hash tables – collision resolution through separate chaining	- Written test - Exposure - Examples - Debate - Conversation	Written test takes 50 minutes
6. Hash tables. Collision resolution through coalesced chaining.	- Exposure - Examples - Debate - Conversation	
7. Binary Trees	- Exposure - Examples - Debate - 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 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. 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: ADT Bag	- Exposure - Examples - Conversation	To be implemented and presented at Lab 2
Lab 3: Linked lists with dynamic allocation	- Exposure - Examples - Conversation	
Lab 4: • Linked lists on arrays • Binary heap and problems/functions using binary heap.	- 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. 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 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 (%)
10.4 Course	• Correctness and completeness of the assimilated knowledge • Knowledge of	Written evaluation (in the exam session): written exam	60%

	applying the course concepts		
10.5 Laboratory	• C++ implementation of the concepts and algorithms presented at the lectures • Lab assignment documentation • Respecting the deadlines for lab presentation	Correctness of the documentation (specifications, algorithms, complexities).	20%
10.6 Seminar	• Written test from seminar 5.	Written test	20%
10.6 Minimum performance standards			
• 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	Signature of course coordinator	Signature of seminar coordinator
03.05.2019	Lect. PhD. Oneț-Marian Zsuzsanna	Lect. PhD. Oneț-Marian Zsuzsanna

Date of approval

Signature of the head of department

Prof. PhD. Andreica Anca