SYLLABUS

1. Data about the program of study

1.1 Institution	The Technical University of Cluj-Napoca
1.2 Faculty	Faculty of Automation and Computer Science
1.3 Department	Computer Science
1.4 Field of study	Computer Science and Information Technology
1.5 Cycle of study	Bachelor of Science
1.6 Program of study / Qualification	Computer science / Engineer
1.7 Form of education	Full time
1.8 Subject code	12.00

2. Data about the subject

2.1 Subject name	***Data Structures and Algorithms***					
2.2 Course responsible / lecturer	Lect. dr. eng. Marius Joldoş - Marius.Joldos@cs.utcluj.ro					
2.3 Teachers in charge of seminars / laboratory / project	Lect. dr. eng. Marius Joldoș Lect. dr. eng. Ciprian Pocol - Ciprian.Pocol@cs.utcluj.ro					
2.4 Year of study	I	2.5 Semester	2	2.6 Type of assessment (E - exam, C - colloquium, V - verification)		E
2.7 Subject category	*DF – fundamentală, DD – în domeniu, DS – de specialitate, DC – complementară*					DD
	DI – Impusă, DOp – opțională, DFac – facultativă					DI

3. Estimated total time

3.1 Number of hours per week	5	of which	Course	3	Seminars		Laboratory	2	Project	
3.2 Number of hours per semester	70	of which	Course	42	Seminars		Laboratory	28	Project	

3.3 Individual study:	
(a) Manual, lecture material and notes, bibliography	30
(b) Supplementary study in the library, online and in the field	20
(c) Preparation for seminars/laboratory works, homework, reports, portfolios, essays	20
(d) Tutoring	5
(e) Exams and tests	5
(f) Other activities:	0

3.4 Total hours of individual study (suma (3.3(a)...3.3(f)))	80
3.5 Total hours per semester (3.2+3.4)	150
3.6 Number of credit points	6

4. Pre-requisites (where appropriate)

4.1 Curriculum	Computer Programming course
4.2 Competence	Programming in C

5. Requirements (where appropriate)

5.1. For the course	
5.2. For the applications	

6. Specific competence

6.1 Professional competences	**C1 –** Operating with basic Mathematical, Engineering and Computer Science concepts • **C1.1 –** Recognizing and describing concepts that are specific to the fields of calculability, complexity, programming paradigms, and modeling computational and communication systems • **C1.2 –** Using specific theories and tools (algorithms, schemes, models, protocols, etc.) for explaining the structure and the functioning of hardware, software and communication systems • **C1.3 –** Building models for various components of computing systems • **C1.4 –** Formal evaluation of the functional and non-functional characteristics of computing systems • **C1.5 –** Providing a theoretical background for the characteristics of the designed systems
6.2 Cross competences	N/A

7. Discipline objective (as results from the *key competences gained*)

7.1 General objective	To acquaint the students with a wide range of fundamental algorithms and data structures. To learn how to use general methods for development of algorithms, as well as mathematical tools for analyzing the correctness and efficiency of algorithms.
7.2 Specific objectives	• To choose the appropriate data structure for modelling a given problem. • To compare and contrast the cost and benefits of dynamic and static structure implementations. • To compare iterative and recursive solutions for elementary problems. • To determine when a recursive solution is appropriate for a problem. • To determine the time and space complexity of simple algorithms and recursively defined algorithms. • To design and implement algorithms using development techniques such as: greedy, divide-and-conquer, backtracking, dynamic programming, branch and bound. • To write C programs that use data structures such as: arrays, linked lists, stacks, queues, trees, hash tables, and graphs. • To implement in C the most common sorting algorithms.

8. Contents

8.1 Lectures	Hours	Teaching methods	Notes
About the course (objectives, outline, recommended reading). Problem solving. Notions of Algorithmics (growth of functions, efficiency, programming model). Dynamic singly-linked lists	3	Lectures, demos and discussions	Uses a video-projector
Operations on dynamic lists. Doubly-linked and circular lists	3		
Trees – definitions, traversals. ADT Tree. Implementations. Binary Search Trees.	3		
Sets ADTs and Implementations. Dictionary ADT. Hash Tables. Mapping ADT. Priority Queue ADT.	3		
Advanced Set Representation Methods. AVL trees. 2-3 Trees. B+ trees. Union-Find Set ADT.	3		
Graphs. Definitions. Representations. ADT's. Traversals for graphs and applications	3		
Graphs. Topological sort, strongly connected components, articulation points . Applications.	3		
Algorithm Design Techniques I. Backtracking. Search Tree Strategies (branch and bound)	3		
Algorithm Design Techniques II. Brute Force Algorithms. Greedy Algorithms.	3		

Algorithm Design Techniques III. Divide-and-Conquer.	3		
Algorithm Design Techniques IV. Dynamic Programming.	3		
Algorithm Design Techniques IV. Search Tree Strategies (branch and bound). Local Search.	3		
Sorting Algorithms	3		
Review	3		

Bibliography
1. Aho, Hopcroft, Ullman. Data Structures and Algorithms, Addison-Wesley, 427 pages, 1987.
2. Cormen, Leiserson, Rivest, Stein: Introduction to Algorithms, 4th edition. MIT Press / McGraw Hill, 1291 pages, 2022.

8.2 Applications – Seminars/Laboratory/Project	Hours	Teaching methods	Notes
Review of C Programming. Laboratory requirements, IDE and homework presentation.	2	Tutoring, discussions, and assisted program development	PCs equipped with MinGW C development kit and Code-blocks IDE
Singly-linked Lists, Stacks and Queues.(Array-based and Dynamic Allocation Implementations)	2		
Arbitrary Trees. Binary Trees	2		
Binary Search Trees	2		
Hash Tables.	2		
Laboratory Test 1	2		
Graph Representations and Traversals (BFS) and applications	2		
Graph Representations and Traversals (DFS) and applications	2		
Algorithm Design I Backtracking and Branch and Bound	2		
Algorithm Design II. Divide & Conquer	2		
Algorithm Design I. Greedy	2		
Algorithm Design III. Dynamic Programming and Heuristics.	2		
Laboratory Test 2	2		

Bibliography
1. Moodle course Web Site available at https://moodle.cs.utcluj.ro/

Se vor preciza, după caz: tematica seminariilor, lucrările de laborator, tematica și etapele proiectului.

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The contents of the course is in accordance with the ACM Computer Science Curricula recommendations. The contents was discussed with the Computer Science departments of the similar technical universities in Bucharest and Timișoara and was evaluated by CNEAA and Aracis

10. Evaluation

Activity type	Assessment criteria	Assessment methods	Weight in the final grade
Course	The understanding of the concepts taught and the ability to solve problems	Three in-class tests (T) + Final Written exam (W)	60% W + 10% T
Seminar	-	-	-
Laboratory	Quality of the assigned applications and in-class tests	Analysis and evaluation of the solved assignments and two in-class tests	30%
Project	-	-	-

Minimum standard of performance: : evaluation grade ≥ 5 Grade calculus: 40% laboratory + 60% exams and tests Conditions for participating in the final exam: Laboratory ≥ 5
Conditions for promotion: final written exam grade ≥ 5 and final written exam problems grade ≥ 5

Date of filling in: 06.06.2024	**Teachers**	**Title First name Last name**	**Signature**
	Course	Lect.dr.eng. Marius Joldos	
	Applications	Lect.dr.eng. Ciprian Pocol	

Date of approval in the department 20.02.2024	Head of department, Prof.dr.eng. Rodica Potolea
Date of approval in the Faculty Council 22.02.2024	Dean, Prof.dr.eng. Mihaela Dînșoreanu