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

2. Data about the subject

2.1 Subject name	***Fundamental Algorithms***			Subject code	**16**
2.2 Course responsible / lecturer	Prof. dr. eng. Potolea Rodica - Rodica.Potolea@cs.utcluj.ro				
2.3 Teachers in charge of seminars / Laboratory / project	Prof. dr. eng. Potolea Rodica - Rodica.Potolea@cs.utcluj.ro As. drd. eng. Negru Vlad-Andrei - Vlad.Negru@cs.utcluj.ro As. drd. eng. Petrache Remus-Mircea - Remus.Petrache@cs.utcluj.ro				
2.4 Year of study	II	2.5 Semester	3	2.6 Type of assessment (E - exam, C - colloquium, V – verification)	E
2.7 Subject category	Formative category: DF – fundamental, DS – speciality, DC – complementary				DF
	Optionality: DOB – mandatory, DOP – optional (alternative), DFA – optional (free choice)				DOB

3. Estimated total time

3.1 Number of hours per week	5	of which:	Course	2	Seminars	1	Laboratory	2	Project	-
3.2 Number of hours per semester	70	of which:	Course	28	Seminars	14	Laboratory	28	Project	-
3.3 Individual study:										
(a) Manual, lecture material and notes, bibliography										21
(b) Supplementary study in the library, online and in the field										26
(c) Preparation for seminars/laboratory works, homework, reports, portfolios, essays										16
(d) Tutoring										8
(e) Exams and tests										9
(f) Other activities:										-

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	Imperative programming languages (C) Data Structures and Algorithms
4.2 Competence	Acquire the abilities of designing, implementing, testing and evaluating programs to solve specific problems

5. Requirements (where appropriate)

5.1. For the course	Whiteboard, projector, computer
5.2. For the applications	Computers/Network of computers, C ++

6. Specific competence

6.1 Professional competences	**C3.** Problems solving using specific Computer Science and Computer Engineering tools (5 credit points) • **C3.1-** Identifying classes of problems and solving methods that are specific to computing systems • **C3.2 -** Using interdisciplinary knowledge, solution patterns and tools, making experiments and interpreting their results • **C3.3 -** Applying solution patterns using specific engineering tools and mehods • **C3.4 -** Evaluating, comparatively and experimentally, the available alternative solutions for performance optimization • **C3.5 -** Developing and implementing informatic solutions for concrete problems **C4.** Improving performances of hardware, software and communication systems • **C4.1 -** Identifying and describing the defining performance elements of hardware, software and communication systems • **C4.2 -** Explaining the interaction of the factors that determine the performances of hardware, software and communication systems • **C4.3 -** Applying fundamental methods and principles for increasing performance of hardware, software and communication systems • **C4.4 -** Choosing criteria and methods for performance evaluation of hardware, software and communication systems • **C4.5 -** Developing performance based professional solutions for hardware, software and communication systems
6.2 Cross competences	N/A

7. Expected Learning Outcomes

Knowledge	Describes, identifies and summarizes elementary concepts and methods regarding the design and analysis of algorithms, namely: • Mathematical foundations and complexity – analyzing the efficiency of algorithms and the difficulty of problems • Fundamental algorithms – sorting, selection and basic problems • Data structures – organizing data for efficient operations (trees, heaps, hashing, sets) • Design and graph techniques – strategies (PD, greedy, damped analysis) applied to graph problems
Skills	Implement, test and debug algorithms on various data structures (lists, trees, graphs); analyze and compare the complexity and efficiency of implementations; proposes incrementally more efficient (ideally optimal) solutions; analyzes and compares different solutions for the same problem; estimates theoretically (through analysis) and experimentally (through evaluation of measurements) the performance of algorithms.
Responsibilities and autonomy	Demonstrates autonomy in the application of knowledge and methods specific to the fields of Computers and Information Technology, assumes responsibility for the quality of the proposed technical solutions and for compliance with professional, legal and ethical standards.

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

8.1 General objective	• Acquiring modern study of algorithms: design and analysis

8.2 Specific objectives	• Learn to identify and design efficient solutions to problems • Learn methods to evaluate efficiency • Learn the basic polynomial algorithms • Learn basic computational complexity • Algorithms description with focus on control structures • Learning the correct implementation following the pseudocode • Efficient implementation of key polynomial algorithms • Estimation of algorithms' efficiency: space and processing time

9. Contents

9.1 Lectures	Hours	Teaching methods	Notes
Mathematical Foundations: Asymptotical notation, Recurrence	2	Whiteboard, projector, computer; Lectures, discussions, Q&A sessions (Teams + Moodle).	
Complexity Classes	2		
Sorting and Order Statistics	2		
Sorting and Order Statistics (continued)	2		
Advanced Data Structures : Hash Tables, Trees	2		
Advanced Data Structures: Heaps, Disjoint Sets	2		
Design and Analysis Advanced Techniques: Dynamic Programming	2		
Design and Analysis Advanced Techniques: Greedy Algorithms	2		
Design and Analysis Advanced Techniques: Amotized Analysis	2		
Graphs: Search in a Graph, Minimal Spanning Tree	2		
Graphs: Shortest path	2		
Graphs: Max Flow	2		
Graphs: Bipartite Graphs	2		
Learn the basic Complexity sets and representative problems	2		
Bibliography: 1. T. Cormen, C. Rleiserson, R. Rivest, C. Stein, *Introduction to Algorithms, Second Edition*, The MIT Press, 2001			
9.2 Applications - Seminars / Laboratory / Project	Hours	Teaching methods	Notes
Efficient implementation and comparison of sorting algorithms	2	Hands on work on specific algorithms; weekly assessment, feedback, and assistance.	
Efficient implementation and comparison of sorting algorithms (continued)	2		
Efficient implementation and comparison of lists algorithms	2		
Efficient implementation and comparison of lists algorithms (continued)	2		
Efficient implementation and comparison of trees algorithms	2		
Efficient implementation and comparison of trees algorithms (continued)	2		
Implementation of augmented data structures	2		
Implementation of augmented data structures (continued)	2		
Efficient implementation of graphs algorithms	2		
Efficient implementation of graphs algorithms (continued)	2		
Efficient implementation of graphs algorithms (continued)	2		
Efficient implementation of graphs algorithms (continued)	2		
Approximation algorithms	2		
Final Evaluation	2		
Bibliography 1. T. Cormen, C. Rleiserson, R. Rivest, C. Stein, *Introduction to Algorithms, Second Edition*, The MIT Press, 2001			

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

The topic is fundamental in the field of Computer and Information Technology, its content is beyond dispute, familiarizing students with the principles of algorithms design and analysis. The content is similar (including the textbook) with all representative computer science departments in the world, is a core course in the ACM curricula and was rated by the Romanian governmental agencies (CNEAA and ARACIS).

11. Evaluation

Activity type	Assessment criteria	Assessment methods	Weight in the final grade
Course	Theoretical analysis and problem solving skills	Final Exam (FE) (oral / written / Moodle)	50%
		2-3 Course Quizzes (written / Moodle)	20%
Seminar	Hands on Problem solving skills	Implementation / hands on	=
Laboratory			
Minimum standard of performance: Final Grade calculus (FG): 20% Quiz (written/Moodle; during courses; min 2 max 3 Quizzes, equal weights, averaged) + 30% laboratory (evaluation of each assignment, equal weights, averaged) + 50% Final Exam (FE) Conditions for participating in the final exam: Laboratory ≥ 5 Conditions for promotion: Final Exam ≥ 5, Final Grade ≥ 5 FE format: Quiz (Moodle) for FE ≤ 7; Oral problem solving for 7<FE ≤ 10 (subscription-based; conditions apply); Re-Examination format: Quiz (Moodle) max grade 5; for better grade Oral Examination			

Date of filling in: 28.04.2026	**Responsible**	**Title, First name Last name**	**Signature**
	Course	Prof.dr.eng. Rodica POTOLEA	
	Applications	Prof.dr.eng. Rodica POTOLEA	
		As.dr.eng. Vlad-Andrei NEGRU	
		As.dr.eng. Remus-Mircea PETRACHE	

Date of approval in the department	Head of department, Prof.dr.eng. Rodica Potolea
Date of approval in the Faculty Council	Dean, Prof.dr.eng. Vlad Mureșan