



DISCIPLINE ACADEMIC SHEET
ACADEMIC YEAR 2025 – 2026

1. PROGRAMME DATA

1.1. Higher Education Institution	UNIVERSITY OF CRAIOVA
1.2. Faculty	Automation, Computers and Electronics
1.3. Department	Computers and Information Technology
1.4. Field of Study	Computers and Information Technology
1.5. Study level	L(License / Undergraduate)
1.6. Organizational form	IF
1.7. Study Program (name/code) /Qualification	Computers / L2060101010

2. DISCIPLINE DATA

2.1 Discipline Name	Data Structures and Algorithms						
2.2 Course Activities Holder	Prof. Marian Cristian Mihăescu, PhD.						
2.3. Practical Activities Holder	Lect. Paul Ștefan Popescu, PhD						
2.4. Study Year	2	2.5. Semester	1	2.6. Evaluation type	E	2.7. Discipline regime	DOB

3. ESTIMATED TOTAL TIME (hours per semester of teaching activities)

3.1. Number of hours per week	4	in which: 3.2 course	2	3.3. sem./lab./pr.	2
3.4. Total hours of curriculum	56	in which: 3.5 course	28	3.6. sem./lab./pr.	28
3.7. Time distribution - hours/week					hours
Study after manual, course support, bibliography and notes					15
Additional documentation in library, on specialized electronic platforms and field					15
Training seminars / labs, homework, portfolios and essays					20
Tutorial					19
Examinations					
Other activities.....					
Total hours per individual activities					69
3.8. Total hours per semester					125
3.9. Number of credits					5

4. PRECONDITIONS (where appropriate)

4.1. of curriculum	• The student must have fundamental knowledge of the following disciplines: Computer Programming and Programming Languages, Algorithm Design.
4.2. of competence	• Minimal C/C++ coding ability.

5. CONDITIONS (where appropriate)

5.1. of the course	- Teaching the course requires a room equipped with a video projector for displaying the course lecture notes and a blackboard for presenting live examples. Lecture notes and up-to-date documentation are available. The teaching process is structured as follows: 80% theoretical presentations using slides 20% practical and interactive examples presented on the blackboard
5.2. of seminar/ laboratory/project	Practical laboratory sessions use workstations equipped with C/C++ compilers and integrated development environments (IDEs) for compiling, building, running, and debugging source code.

6. COURSE OBJECTIVES – expected learning outcomes that are supported by completing and passing the course

Knowledge	Student/Graduate: - Working with fundamentals of mathematics, engineering and informatics. - Proper use in professional communication of the eigen concepts of calculability, complexity, programming paradigms and modeling of computer and communications systems. - Design of the hardware, software and communication components. - Building some hardware, software and communication components using design methods, languages, algorithms, data structures, protocols and technologies.
Skills (Abilities)	Student/Graduate: - Solving problems using computer science and engineering tools. - Applying solutions by means of engineering tools and methods. - Improving the performance of hardware, software and communication systems. - Application of methods and underlying principles for increasing the performance of hardware, software and communication systems.
Responsibility and autonomy	Student/Graduate: - Design, life cycle management, integration and integrity of hardware, software and communication systems. - Specifying relevant criteria regarding life cycle, quality, safety and the computer system interaction with the environment and the human operator.

7. CONTENT

7.1. Course	The method of implementation	Teaching methods	Allocated time (hours)
1. Introduction	face to face	Teaching is conducted using both the blackboard and slides. Lecture notes and up-to-date documentation are available. The teaching process is structured as follows:	2
2. Binary heaps, Binary search trees, AVL trees	face to face		2
3. Splay trees, Red-black trees	face to face		2
4. B- Trees. TRIE Trees	face to face		2
5. Optimal binary search trees, Hash tables.	face to face		2
6. Graphs Representations,	face to face		2

Search, Connectivity, Dijkstra's Algorithms		- 80% theoretical presentations - 20% practical and interactive examples	
7. Graphs. Weighted independent sets, Bellman-Ford 's Algorithm	face to face		2
8. Graphs. All pairs shortest path, Roy-Floyd Algorithms, Johnson's Algorithm	face to face		4
9. Graphs. Minimum cost spanning trees, Topological sort	face to face		4
10. Graphs. Strongly connected components, Kosaraju's Algorithms, Max Flow, Ford-Fulkerson's Algorithm	face to face		2
11. Substring search. KMP, Rabin-Karp, Boyes-Moore Algorithms	face to face		2
12. Analysis of algorithms			2
Bibliography:			
1. Roughgarden, Tim. Algorithms Illuminated. Soundlikeyourself Publishing, 2022.			
2. Cormen Thomas, Leiserson Charles, Rivest Ronald – Introduction to Algorithms, M.I.T. Press 19929. Schild, H., Manual complet C++, Ed. Teora, 2003.			
3. Robert Sedgewick and Kevin Wayne, Algorithms, 4th Edition, Addison-Wesley, 2011.			

7.2. Seminar/laboratory/project	The method of implementation	Teaching methods	Allocated time (hours)
1. Binary search trees, AVL trees	face to face	Practical laboratory sessions take place on workstations equipped with C/C++ compilers and integrated development environments (IDEs).	3
2. Binary heaps	face to face		3
3. TRIE Trees	face to face		3
4. B Trees	face to face		3
5. SSSP. Dijkstra and Belman-Ford Algorithms	face to face		3
6. APSP. Roy-Floyd Algorithm	face to face	Laboratory notes are provided to students and contain a detailed description of each practical session.	3
7. MST. Kruskal and Prim Algorithms	face to face		3
8. Max Flow. Ford-Fulkerson Algorithm	face to face		3
9. Substring search. KMP Algorithm	face to face	Practical activities consist of: - 70% coding, compiling, building, running, and debugging; - 30% code review, interpretation of results, and discussions with students.	4
Bibliography:			
1. Wu, Y., & Wang, J. (2005). Data Structure Practice. CRC Press.			

2. Halim, Steven, Felix Halim, and Suhendry Effendy. Competitive programming 4: The new lower bound of programming contests in the 2020s. Lulu. com, 2018.
3. Steven S. Skiena, Miguel A. Revila, Programming Chalanges – The Programming Contest Trening Manual, Springer-Verlag New York, 2003.

8. Correlating the content of the discipline with the expectations of representatives from the epistemic community, professional associations and representative employers in the field related to the program

Competencies are provided in accordance with the provisions of the National Register of Qualifications in Higher Education (RNCIS).
The course syllabus has been discussed with S.C. NetRom Software, Craiova.

9. EVALUATION

Activity Type	9.1. Evaluation criteria	9.2. Evaluation methods	9.3. Final mark weight
9.4. Course	Understanding fundamental theoretical concepts; Ability to analyze fundamental algorithmic problems and identify the data structures and algorithms required to solve them; Ability to perform comparative analyses of different options regarding the available data structures and algorithms.	Final oral exam	40%
	The quality and number of problems solved.	Problem solving	10%
9.5. Laboratory	Correct implementation of all laboratory activities; Compilation, building, execution, debugging, and code review of all written source code.	Continuous assessment of students' achievements throughout all laboratory activities.	50%
9.6. Minimum standard of performance			
Acceptable level of knowledge demonstrated in all practical laboratory sessions; Minimum grade of 5 in the final oral examination.			

Date of Completion	Signature of the Course Activities Holder,	Signature of the Practical Activities Holder,
23.08.2025	Prof. Marian Cristian Mihăescu, PhD.	Lect. Paul Ștefan Popescu, PhD

Date of Approval in the Department	Signature of the Department Director,
25.08.2025	Ș.l.dr.ing. Nicolae ENESCU

Date of Approval in the Faculty	Signature of the Dean,
	Prof.dr.ing. Dorin ȘENDRESCU