



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	Competitive Programming						
2.2 Course Activities Holder	Prof. Marian Cristian Mihăescu, PhD.						
2.3. Practical Activities Holder	Prof. Marian Cristian Mihăescu, PhD.						
2.4. Study Year	2	2.5. Semester	2	2.6. Evaluation type	V	2.7. Discipline regime	DFA

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

3.1. Number of hours per week	2	in which: 3.2 course	1	3.3. sem./lab./pr.	1
3.4. Total hours of curriculum	28	in which: 3.5 course	14	3.6. sem./lab./pr.	14
3.7. Time distribution - hours/week					hours
Study after manual, course support, bibliography and notes					10
Additional documentation in library, on specialized electronic platforms and field					12
Training seminars / labs, homework, portfolios and essays					
Tutorial					
Examinations					
Other activities.....					
Total hours per individual activities					22
3.8. Total hours per semester					50
3.9. Number of credits					2

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, Data Structures and Algorithms.
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.
5.2. of seminar/ laboratory/project	The laboratory sessions use workstations equipped with C/C++ compilers for implementing, running, debugging, and testing solutions. Web browsers are also used to access online platforms for conducting programming competition simulations.

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 to competitive programming The specifics of competitive programming, online judges, solution evaluation, types of contests and problems, reference materials, and evaluation methods.	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. Data Structures Implemented in Custom Libraries	face to face		2

Binary Indexed Trees (Fenwick Trees), Segment Trees, Disjoint Set Union (Union-Find) Data Structure		- 80% theoretical presentations - 20% practical and interactive examples	
3. Data Structures Implemented in Standard Libraries	face to face		2
Heaps, Hash Tables, Balanced Binary Search Trees			
4. Graph Algorithms – Fundamental Algorithms	face to face		2
Shortest paths, Minimum Spanning Trees (MST), Strongly Connected Components (SCC)			
5. Graph Algorithms – Advanced Algorithms	face to face		2
Maximum Flow Ford–Fulkerson Algorithm, Maximum Matching in Bipartite Graphs			
6. Dynamic Programming	face to face		2
Classic Dynamic Programming Problems, Fundamental Dynamic Programming Techniques			
7. String Algorithms	face to face		2
Knuth–Morris–Pratt (KMP) Algorithm, Boyer–Moore Algorithm, Rabin–Karp Algorithm			
Bibliography:			
1. Laaksonen, A. (2017). Competitive programmer's handbook. Preprint.			
2. Halim, S., Halim, F., & Effendy, S. (2018). Competitive programming 4: The new lower bound of programming contests in the 2020s. Lulu. com.			
3. Steven S. Skiena (Author), Miguel A. Revilla (2013). Programming Challenges: The Programming Contest Training Manual			

7.2. Seminary	The method of implementation	Teaching methods	Allocated time (hours)
1. Introduction	face to face	Seminar activities are carried out on computers equipped with C/C++ programming language compilers and Internet access. During each seminar session, solutions are presented and discussed using the blackboard and video projector.	2
Development Environments and Tools, Programming Languages, Online Programming Platforms, Solution Evaluation Methods			
2. Sorting and Searching	face to face		2
Problems involving sorting algorithms and binary search.			

3. Data Structures Problems involving sets, maps, and priority queues.	face to face		2
4. Data Structures Problems involving Binary Indexed Trees (Fenwick Trees), Segment Trees, and Range Minimum Queries (RMQ).	face to face		2
5. Graph Algorithms Problems involving graph traversal, shortest paths, Minimum Spanning Trees (MST), Strongly Connected Components (SCC), and Maximum Flow.	face to face		2
6. Dynamic Programming Problems involving one-dimensional and two-dimensional dynamic programming, as well as interval dynamic programming.	face to face		2
7. String Algorithms Problems involving the Knuth–Morris–Pratt (KMP), Boyer–Moore, and Rabin–Karp algorithms.	face to face		2
Bibliography:			
1. Codeforces, https://codeforces.com/			
2. Code Submission Evaluation System, https://cses.fi/			
3. CodeChef, https://www.codechef.com/			
4. CS Academy, https://csacademy.com/			
5. pbInfo, https://www.pbinfo.ro/			

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	Solving the problems provided at the end of each course.	The problems are evaluated using the Accepted qualifier on the online evaluation platform.	30%

9.5. Laboratory	Solving problems in competition simulations. Solving problems in official competitions. Upsolving problems from competition simulations or online competitions.	The problems are evaluated using the Accepted qualifier on the online evaluation platform.	70% (20%, 40%, 10%)
9.6. Minimum standard of performance			
Solving at least 25 problems during the semester.			

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