



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	Formal Languages and Automata						
2.2 Course Activities Holder	Becheru Alexandru						
2.3. Practical Activities Holder	Becheru Alexandru						
2.4. Study Year	3	2.5. Semester	2	2.6. Evaluation type	V	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					14
Additional documentation in library, on specialized electronic platforms and field					-
Training seminars / labs, homework, portfolios and essays					14
Tutorial					-
Examinations					2
Other activities.....					14
Total hours per individual activities					44
3.8. Total hours per semester					100
3.9. Number of credits					4

4. PRECONDITIONS (where appropriate)

4.1. of curriculum	• Special Mathematics, Computer Programming and Programming Languages
4.2. of competence	• Not applicable

5. CONDITIONS (where appropriate)

5.1. of the course	• Course support in electronic format and access to up-to-date documentation are provided. • The teaching process
--------------------	---

	has the following structure: o 70% theoretical presentation, based on the course support o 30% interactive activity (discussions with the students)
5.2. of seminar/ laboratory/project	• In the seminar, the emphasis is placed on understanding and using the methods

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

Knowledge	The student/graduate: 1. Identifies and describes basic concepts, principles and methods from the theory of formal languages and automata 2. Explains and interprets theoretical and practical results in the field of formal languages and automata 3. Describes, identifies, summarizes elementary concepts and methods regarding formal languages and the way they are applied in programming languages
Skills (Abilities)	The student/graduate selects and explains concepts specific to algorithm design, object-oriented programming, logic and functional programming. The student/graduate specifies requirements, analyzes, elaborates, develops and tests software systems in general-purpose (C etc.) and/or object-oriented (C++, Java etc.) programming languages, applying the elements specific to software engineering. The student/graduate designs, implements, develops and/or manages applications that include various types of databases. The student understands the main concepts underlying translation systems and source code compilation, and contributes to the development and improvement of tools specific to programming languages.
Responsibility and autonomy	The student/graduate behaves honorably, responsibly and ethically, in the spirit of the law, to ensure the reputation of the profession.

7. CONTENT

7.1. Course	The method of implementation	Teaching methods	Allocated time (hours)
1. 1. The representation of languages	Online sincron (sapt 1)	70% theoretical presentation, based on the course support; 30% interactive activity, discussions with the students	2
2. 2. Regular sets and right-linear languages	Online sincron (sapt 2, 3)	70% theoretical presentation, based on the course support; 30% interactive activity, discussions with the students	4
3. 3. Finite automata	Online sincron (sapt 4, 5)	70% theoretical presentation, based on the course support; 30% interactive activity, discussions with the students	4

4. 4. Properties of finite automata and regular sets		70% theoretical presentation, based on the course support; 30% interactive activity, discussions with the students	6
5. 5. Context-free languages	face to face	70% theoretical presentation, based on the course support; 30% interactive activity, discussions with the students	4
6. 6. Push-down automata	face to face	70% theoretical presentation, based on the course support	4
7. 7. Properties of context-free languages	face to face	70% theoretical presentation, based on the course support; 30% interactive activity, discussions with the students	4
Bibliography:			
1. 1. J.E. Hopcroft, R. Motwani, and J. D. Ullman - Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, 2001			
2. Marius Brezovan - Limbaje formale și aplicații în știința calculatoarelor, Editura SITECH, 2022			
2.			

7.2. Seminar/laboratory/project	The method of implementation	Teaching methods	Allocated time (hours)
1. 1. The representation of languages	Online sincron	Student–teacher interaction. Solving problems using mathematical proofs, as well as implementing the solutions	2
2. 2. Regular expressions and regular sets	Online sincron	Student–teacher interaction. Solving problems using mathematical proofs, as well as implementing the solutions	2
3. 3. Finite automata	Online sincron	Student–teacher interaction. Solving problems using mathematical proofs, as well as implementing the solutions	2
4. 4. Properties of regular sets	face to face	Student–teacher interaction. Solving problems using mathematical proofs, as	2

		well as implementing the solutions	
5. 5. Context-free languages	face to face	Student–teacher interaction. Solving problems using mathematical proofs, as well as implementing the solutions	2
6. 6. Push-down automata	face to face	Student–teacher interaction. Solving problems using mathematical proofs, as well as implementing the solutions	2
7. 7. Properties of context-free languages	face to face	Student–teacher interaction. Solving problems using mathematical proofs, as well as implementing the solutions	2
8. Laboratory			0
9. - Creating graphical interfaces	face to face	Practical demonstrations and code writing. Writing programs in object-oriented languages, interpreting the results and discussions with the students	2
10. - Implementing algorithms for regular expressions and regular sets	face to face	Practical demonstrations and code writing. Writing programs in object-oriented languages, interpreting the results and discussions with the students	2
11. - Implementing algorithms for deterministic finite automata	face to face	Practical demonstrations and code writing. Writing programs in object-oriented languages, interpreting	4
12. - Implementing algorithms for non-deterministic finite automata	face to face	Practical demonstrations and code writing. Writing programs in object-oriented languages, interpreting the results and discussions with the students	2
13. - Implementing algorithms for push-	face to face	Practical demonstrations and code writing.	4

down automata		Writing programs in object-oriented languages, interpreting the results and discussions with the students	
Bibliography:			
1. 1. J.E. Hopcroft, R. Motwani, and J. D. Ullman - Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, 2001 2. Marius Brezovan - Limbaje formale și aplicații în știința calculatoarelor, Editura SITECH, 2022 3. Susan H. Rodger and Thomas W. Finley - JFLAP: An Interactive Formal Languages and Automata Package, Jones & Bartlett Publishers, 2006			
2.			

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

The content of the course and of the applications is in line with similar disciplines at universities in the country and abroad. Also, the acquired competences are in accordance with the requirements of representative employers in the field (discussions having been held with representatives of local companies such as NetRom, Quest, Caphyon, CS Romania).

9. EVALUATION

Activity Type	9.1. Evaluation criteria	9.2. Evaluation methods	9.3. Final mark weight
9.4. Course	- Understanding the differences between the various categories of formal languages - Understanding the methods specific to each category of formal languages, regarding the specification grammars and the automata recognizing them - The ability to make connections between the taught notions - The ability to apply the methods specific to formal languages to various programming languages	Written exam	50%
9.5. Seminar/laboratory/project	Seminar - Understanding the way of representing formal languages -	Homework	30%

	The ability to determine, for each category of formal languages, the generating grammars, as well as their equivalence - The ability to determine, for each category of formal languages, the recognizing automata, as well as their equivalence. Laboratory - Understanding the algorithms specific to formal languages and automata - The ability to implement certain specific algorithms using an object-oriented language		
9.6. Minimum standard of performance			

Date of Completion	Signature of the Course Activities Holder,	Signature of the Practical Activities Holder,
23.08.2025	Becheru Alexandru	Becheru Alexandru

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

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