SYLLABUS

FUNDAMENTAL ALGORITHMS
2024-2025

1. Program General Data

1.1. University	**„1 Decembrie 1918" of Alba Iulia**
1.2. Faculty	**Faculty of Informatics and Engineering**
1.3. Department	**Informatics, Mathematics and Electronics Department**
1.4. Area	**Computer Science**
1.5. Level	**Undergraduate**
1.6. Specialization	**Computer Science** **ESCO-08: 2511/ Systems Analyst, 2512/ Software developers Analyst 251201** **Computer System Programmer 251204** **Computer System Engineer 251203**

2. Subject General Data

2.1. **Subject**	*Fundamental algorithms*			2.2. **Code**		CSE202	
2.3. **Course holder/ Lecturer/** Instructor's Name	Domșa Ovidiu						
2.4. Teaching Assistant's Name	Domșa Ovidiu						
2.5. Year	**II**	2.6. Semester	**I**	2.7. Evaluation form (E – final exam/C-examination /VP)	**E**	2.8. Status (**C**– Compulsory, **Op** – optional, **F -** Facultative)	**C**

3. Course Structure (Weekly number of hours)

3.1. Weekly number of hours	**4**	3.2. course	***2***	3.3. seminar, laboratory	***2***
3.4. Total number of hours according to the curricula	**56**	3.5. course	***28***	3.6. seminar, laboratory	***28***

Time distribution:	Hours
Individual study using the lecture notes	***20***
Documentation (library)	***20***
Homework, Essays, Portfolios	***20***
Tutoring	**-**
Evaluation (exams)	***9***
Other activities…….	**-**

3.7 Total number of hours for individual study	**69**
3.8 Total number of hours according to the curricula	**56**
3.9 Total number of hours per semester	**125**
3.10 Credits	**5**

4. Prerequisites

4.1. Curricula prerequisites	*Imperative and procedural programming* *Algorithms and data structures* *Graph algorithms*
4.2. according to the general competencies	

5. Conditions

5.1. Conditions to support teaching	*Room equipped with video projector/board.*
5.2. Conditions for supporting seminar/laboratory activities	*Laboratory – computers. Software: Code Blocks, Internet acces.*

6. Competenţe specifice acumulate (cele alese de titular din grila de competente)

Professional competences	- Development of skills required to solve complex problems using the algorithms studied. - Identify the addressed problems with the studied techniques and algorithms. -The student will be able to translate in algorithmic language (pseudocode, programming language) the solution of complex problems. - Thoroughly study of data structures and algorithms concepts and the methods used for handling them (hash tables, trees, graphs).
Transversal competences	**Cognitive skills:** acquisition of basic and specific knowledge about the concept of fundamental algorithm; the ability to identify the applicability of the studied algorithms in real problems; understanding the need of using fundamental algorithms when addressing problems from an algorithmic perspective; acquiring basic knowledge on the concept of algorithms complexity. **Affective skills**: develop the capacity of analysis and understanding of a highly complex real problems and effectively address it from an algorithmic perspective. Team spirit: encouraging students to work in design, analysis and programming teams. Awarness of the importance of the knowledge and thoroughly study of fundamental algorithms.

7. Course objectives

6.1 General course objectives	*- Develop algorithmic thinking and skills for developing complex algorithms.* *- Learning basic tools for developing fundamental algorithms.* *- Knowledge of types of fundamental algorithms and their development methods.* *- Use of an advanced programming language for implementing the studied algorithms.*
6.2 Specific course objectives	

8. Course contents

Lectures	**Didactic methods used**	**Observaţii**
General principles for algorithm development.	***Lecture, discussions, examples***	
Complexity of algorithms. Asymptotic analysis of worst case scenario.	***Lecture, discussions, examples***	
Numerical algorithms. Optimization of numerical	***Lecture, discussions, examples***	

algorithms. Primality. Bell numbers. Stirling numbers. Catalan numbers. Numbers with special properties.		
Sorting: HeapSort, QuickSort, RadixSort, Median-Algorithms, Lower Bounds.	***Lecture, discussions, examples***	
Analysis of sorting and searching algorithms complexity.	***Lecture, discussions, examples***	
Parallel sorting: enumeration sort, odd-even transposition sort.	***Lecture, discussions, examples***	
Parallel sorting: bitonic sort, quicksort on a hypercube.	***Lecture, discussions, examples***	
Binary search trees.	***Lecture, discussions, examples***	
AVL trees. Red-black trees. B-trees.	***Lecture, discussions, examples***	
Hash tables. Collision resolution. Hash functions.	***Lecture, discussions, examples***	
Graph algorithms: Transitive Closure, Shortest Path Problems, Minimum Spanning Trees.	***Lecture, discussions, examples***	
Branch&Bound algorithms. Exemples of problems solved with the Branch&Bound method.	***Lecture, discussions, examples***	
NP-complete algorithms.	***Lecture, discussions, examples***	
Analysis, evaluation, and feed-back.	***Lecture, discussions, examples***	
References 1. Cormen T.H., Leiserson E.C., Rivest R.R., Introduction in algorithms, MIT Press, 2001. 2. Dahl O.J., Dijkstra E.W., Hoare C.A.R., Structured Programing, Academic Press, 1972. 3. Donald E. Knuth, The Art of Computer Programming, Volumes 1–3, Addison-Wesley Professional Volume 1: Fundamental Algorithms (3rd edition), 1997. Addison-Wesley Professional, Volume 2: Seminumerical Algorithms (3rd Edition), 1997. Addison-Wesley Professional, Volume 3: Sorting and Searching (2nd Edition), 1998. Addison-Wesley Professional.		
Seminars-laboratories	**Didactic methods used**	
General principles for algorithms development.	***laboratory works***	
Complexity of algorithms.	***laboratory works***	
Numerical algorithms. Goldbach conjecture. Bell numbers, Catalan numbers, Entringer numbers, Stirling. Combinatorial calculus. Modular exponentiation. Large numbers operations.	***laboratory works***	
Sorting: *HeapSort, QuickSort, RadixSort, BrickSort BucketSort, CountSort*.	***laboratory works***	
Analysis of sorting and searching algorithms complexity.	***laboratory works***	
Graph algorithms: graphs representations, graphs traversal, shortest paths.	***laboratory works***	
Graph algorithms: cycles, Eulerian graph, Hamiltonian graph, connectivity, strong connectivity, coupling, flow.	***laboratory works***	
Binary search trees.	***laboratory works***	
Red-black trees. B-trees.	***laboratory works***	
Evaluation of arithmetic expressions. Polish notation for arithmetic expressions.	***laboratory works***	
Practical applications. Examples of practical problems solved with efficient methods.	***laboratory works***	
References		

1. Cormen T.H., Leiserson E.C., Rivest R.R., Introduction in algorithms, MIT Press, 2001.
2. Dahl O.J., Dijkstra E.W., Hoare C.A.R., Structured Programing, Academic Press, 1972.
3. Donald E. Knuth, The Art of Computer Programming, Volumes 1–3, Addison-Wesley Professional Volume 1: Fundamental Algorithms (3rd edition), 1997. Addison-Wesley Professional, Volume 2: Seminumerical Algorithms (3rd Edition), 1997. Addison-Wesley Professional, Volume 3: Sorting and Searching (2nd Edition), 1998. Addison-Wesley Professional.

9. Corroborating Course content expectations to the epistemic community representatives, professional associations and employers representative for the curricula

- *Not applicable.*

10. Assessment

Activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Percentage from the final mark
10.4 Course	*Final evaluation*	*Written exam*	60%
	-	-	-
10.5 Seminar/laboratory	*Continuous assessment*	*Portfolio of laboratory practical works*	40%
	-		-
10.6 Minimum performance standard:			

Completion date Instructor's signature Teaching assistant's signature

……..……. …….

Date of approval within the department Head of departament's signature

…….. …….