SYLLABUS / FIŞA DISCIPLINEI

1. Information on the study programme / Date despre programul de studii

1.1. Institution / Instituția de învățământ superior	Universitatea de Vest din Timișoara
1.2. Faculty / Facultatea	Matematică și Informatică
1.3. Department / Departamentul	Computer Science (Informatică)
1.4. Study program field/Domeniul de studii	Computer Science (Informatică)
1.5. Study cycle / Ciclul de studii	Bachelor / Licență
1.6.Study programme / Programul de studii / calificarea*	Computer Science / Informatică în limba engleză / Database administration / *Administrator baze de date - 252101; Computer network administration / Administrator de retea de calculatoare - 252301; Analyst / Analist - 251201; Research assistant in computer science / Asistent de cercetare în informatica - 214918; Teacher in secondary schools / Profesor în învatamântul gimnazial - 233002; Programmer / Programator - 251202; Software systems designers / Proiectant sisteme informatice - 251101*

2. Information on the course / Date despre disciplină

2.1. Title of the course / Denumirea disciplinei	Advanced Data Structures						
2.2. Teacher in charge of the course / Titularul activităților de curs	Conf. Dr. Mircea Marin						
2.3. Teacher in charge of the seminar / Titularul activităților de seminar	Conf. Dr. Mircea Marin						
2.4. Study year / Anul de studii	2	2.5. Semester / Semestrul	1	2.6. Examination type / Tipul de evaluare	C	2.7. Course type / Regimul disciplinei: M(andatory)/ E(lective)/ F(acultative)	DO

3. Estimated study time (number of hours per semester) / Timpul total estimat (ore pe semestru al activităților didactice)

3.1. Attendance hours per week / Număr de ore pe săptămână	3	Out of which / din care: 3.2 curs	2	3.3. seminar/laborator	1
3.4. Attendance hours per semester / Total ore din planul de învățământ	42	Out of which / din care: 3.5 curs	28	3.6. seminar/laborator	14

Distribution of the allocated amount of time / Distribuția fondului de timp*	**hours/ ore**
Studiu după manual, suport de curs, bibliografie și notițe	35
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate	15
Pregătire seminarii/laboratoare, teme, referate, portofolii și eseuri	20

Examinări	6
Tutorat	8

3.7. Total ore studiu individual	84
3.8. Total ore pe semestru	126
3.9. Număr de credite	5

4. Prerequisites (if it is the case) / Precondiții (acolo unde e cazul)

4.1. curriculum / de curriculum	Courses on Algorithms and Data Structures II, and Programming II
4.2. skills / de competențe	Programming in C++ and/or Java

5. Requirements (if it is the case) / Condiții (acolo unde e cazul)

5.1. for the lecture / de desfășurare a cursului	Tablet, Google Meet, Google Classroom
5.2. or the seminar, laboratory / de desfășurare a seminarului/laboratorului	Computers with preinstalled C++ and corresponding IDEs- e::Blocks or Eclipse

6. Acquired skills / Competențe specifice acumulate

Professional skills / Competențe profesionale	• The capacity to acquire knowledge about the characteristic features of some advanced data structures and the algorithms that use them • Capacity to understand the differences between different data structures that can be used for the same purpose, and of the complexity of the algorithms that use them.
Transversal skills / Competențe transversale	• The ability to model and solve in an efficient way many problems of practical interest using the data structurs and algorithms presented at this course.

7. Objectives of the course / Obiectivele disciplinei (reieșind din grila competențelor specifice acumulate)

7.1. General objective / Obiectivul general al disciplinei	Get acquainted with (1) the design and implementation of some efficient algorithms by choosing the most suitable data structures and algorithms, and (2) the analysis of time- and speace-complexity of the algoritms.

7.2. Specfic objectives / Obiectivele specifice	*Knowledge Objectives (KO)*:): (1) Indicate the operation specific to some well-established data problems and advanded data structures (2) Perform a correct analysis of the runtime complexity of the operations that are being performed *Ability Objectives (AO)*: (1) Identify the data structures that are best suited for solving a particular problem. (2) Use the adequate data structures in concrete applications implemented in a high-level programming language, like C++ or Java. (4) Argue convincingly that the data structures and algorithms used in the application are adequate. *Attitudinal Objectives (AtO)*: identify the advantages / disadvantages of the alternative approaches to solve a problem.

8. Conținuturi*

8.1. Lecture / Curs	**Teaching strategies / Metode de predare**	**Remarks, details / Observații**
C1.(2h) Dynamic sets and characteristic operations. Data structures for fast key-based search. Binary search trees – Limitations and motivation for the design of better data structures.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
C2.(2h) Red-black trees. Properties. Main operations and their efficient implementation.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
C3.(2h) B-trees: Definition, main operations, complexity study.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
C4.(2h) Amortized analysis. Case studies.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein:

		Introduction to Algorithms. MIT Press, 3rd edition, 2009.
C5.(2h) Data structures for the fast retrieval of minimum or maximum) element. Binomial heaps. Definition, main operations, and their runtime complexity.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
C6.(2h) Fibonacci heaps. Definition, main operations, and their runtime complexity.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
C7.(2h) Disjoint-set data structures. Main operations, alternative imple-mentations, and the analysis of their runtime complexity.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009
C8.(2h) Mid-term exam	Exam	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009
C9.(2h) String-matching algorithms. The naïve method, the Rabin-Karp algorithm, the method based on finite automata.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009
C10.(2h) Keyword trees and suffix trees. Applications.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.

C11.(2h) Data structures for dictionaries. Dispersion tables, hash functions. Implementation alternatives.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
C12.(2h) Computational geometry. Specific problems and data structures. Algorithms.	Lecture, conversation, illustration	References / Referinte 1. M. Marin– slides– classroom 2. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
C13.(2h) Survey of the content of the lecture.		References / Referinte 1. M. Marin– slides– classroom
C14.(2h) Colloquium.	Exam.	

Recommended bibliography / Bibliografie

1. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
2. D.C. Kozen: *The Design and Analysis of Algorithms*. Springer Verlag, Inc., 1991.
3. S. Skiena: *The Algorithm Design Manual*, second edition. 2008

8.2. Seminar/laborator	Metode de predare/ invățare	Remarks / Observații
L1.(2h) Correct implementation of some operations of red-black trees. Analysis of their run-time complexity.	Practical labworks in C++ or Java, using a suitable IDE (Eclipse or Code::Blocks). Learning by collaboration, dialog and code testing	The labworks will be posted on Google Classroom. Students may ask questions,a nd the professor may answer them on Google Classroom.
L2.(2h) Use B-trees to solve some concrete problems.	Practical labworks in C++ or Java, using a suitable IDE (Eclipse or Code::Blocks). Learning by collaboration, dialog and code testing	Idem. Students should submit their labworks on time. The labworks will be evaluated by the professor, who may ask questions before grading the labwork.
L3.(2h) Use Binomial heaps to solve some concrete problems.	Practical labworks in C++ or Java, using a suitable IDE (Eclipse or Code::Blocks). Learning by collaboration, dialog and code testing	Idem. Students should submit their labworks on time. The labworks will be evaluated by the professor, who may ask questions before grading the labwork.
L4.(2h) Implement an API for disjoint-sets in C++ or Java, and use it to solve some practical problems.	Practical labworks in C++ or Java, using a suitable IDE (Eclipse or Code::Blocks). Learning by collaboration, dialog and code testing	Idem. Students should submit their labworks on time. The labworks will be evaluated by the professor,

		who may ask questions before grading the labwork.
L5.(2h) Implement an API for string-matching in C++ or Java, and use it to solve some concrete problems.	Practical labworks in C++ or Java, using a suitable IDE (Eclipse or Code::Blocks). Learning by collaboration, dialog and code testing	Idem. Students should submit their labworks on time. The labworks will be evaluated by the professor, who may ask questions before grading the labwork.
L6.(2h) Model and implement some interesting problems from computational geometry.	Practical labworks in C++ or Java, using a suitable IDE (Eclipse or Code::Blocks). Learning by collaboration, dialog and code testing	Idem. Students should submit their labworks on time. The labworks will be evaluated by the professor, who may ask questions before grading the labwork.
L7.(2h) Exam	Evaluation of projects.	

Recommended bibliography / Bibliografie

1. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: *Introduction to Algorithms*, MIT Press, 3rd edition, 2009.
2. T.H. Cormen, C. Lee and E. Lin: *Instructor's Manual to Accompany* [1], MIT Press 2002.
3. Lecture notes and extra material posted on Google Classroom.

9. Correlations between the content of the course and the requirements of the IT field / Coroborarea conținuturilor disciplinei cu așteptările reprezentanților comunității epistemice, asociațiilor profesionale și angajatorilor reprezentativi din domeniul aferent programului

The content is consistent with the structure of similar courses from other universities.

10. Evaluation / Evaluare*

Activity / Tip de activitate	10.1. Evaluation criteria / Criterii de evaluare**	10.2. Evaluation methods / Metode de evaluare***	10.3. Weight in the averaged mark / Pondere din nota finală
10.4. Lecture / Curs	- Describe the representation of a data strcture and the meanings of its components. - Indicate the operations specific to the data structures discussed in this course, and the pseudocode of their implementation - Identify the most suitable data structure or algorithm to solve a concrete problem, and indicate the pseudocode of the solution - Indicate the runtime complexity of a particular algorithm or operation on a data structure - Illustrate in diagrammatic form the outcome of a particular	Midterm exam	30%
		Colloquim	30%

	operation on some simple examples.		
10.5. Seminar/laborator	Ability to use effectively some operations on data structures in applications implemented in C++	Practical labworks	10%
	Ability to solve some concrete problems using the data structures and algorithms presented in this course.	Practical labworks	10%
	Correct answers to questions about the properties of the data structures and the complexity of their operations.	Questions and Quizzes during the seminar / lab	20%
10.6. Minimal knowledge for passing / Standard minim de performanță			
The minimum requirements in order to get a minimum grade (which is 5) to pass this exam are: - To write pseudocode for a simple operation on a data structure described in this course - To indicate the runtime complexity of an operation or algorithm - To use correctly a particular operation or algorithm			

Date/Data completării
21.09.2020

Semnătura titularului de curs
Conf. Dr. Mircea Marin

Semnătura titularului de seminar
Conf. Dr. Mircea Marin

Signature (director of the department)
Semnătura directorului de departament