SYLLABUS / FIȘA DISCIPLINEI

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

1.1. Institution / Instituția de învățământ superior	Universitatea de Vest din Timişoara
1.2. Faculty / Facultatea	Matematică și Informatică
1.3. Department / Departamentul	Computer Science (Informatică)
1.4. Study program field	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

2.1. Title of the course			Advanced Data Structures				
2.2. Teacher in charge of the course			Mircea Marin				
2.3. Teacher in charge of the seminar			Mircea Marin				
2.4. Study year	2	2.5. Semester	1	2.6. Examination type: E(xam)/C(olloquim)	C	2.7. Course type: M(andatory)/ E(lective)/ F(acultative)	DO

3. Estimated study time (number of hours per semester)

3.1. Attendance hours per week	3	out of which: 3.2 lecture	2	3.3. seminar	1
3.4. Attendance hours per semester / Total ore din planul de învățământ	42	out of which: 3.5 lecture	28	3.6. seminar	14

Distribution of the allocated amount of time / Distribuția fondului de timp*	**hours/ ore**
Individual study	35
Supplementary documentation at library or using electronic repositories	15
Preparing for laboratories, homework, reports etc.	20
Exams	6
Tutoring	8

3.7. Total number of hours of individual study	84
3.8. Total number of hours per semester	126

3.9. Number of credits (ECTS)	5

4. Prerequisites (if it is the case)

4.1. curriculum	Previous courses of Data Structures and Programming II
4.2. skills	Programming in C++

5. Requirements (if it is the case)

5.1. for the lecture	Beamer and whiteboard
5.2. for the seminar, laboratory	Lab with computers which have preinstalled C++ and a corresponding IDE (Code::Blocks or Eclipse)

6. Acquired skills

Professional skills	• To acquire knowledge about the characteristic properties of some advanced data structures, and the algorithms that make use of them • Good understanding of the differences between the alternative data structures that can be used for the same purpose, and of the complexity of the algorithms that make use of them.
Transversal skills	The capacity to model and solve efficiently several problems of practical interest using the data structures and algorithms presented in this course

7. Objectives of the course

7.1. General objective	Familiarity with the design and implementation of efficient algorithms by choosing the most suitable data structures and algorithms, and the ability to analyse the computational complexity (in time and space) of the implemented code.
7.2. Specfic objectives	Knowledge objectives: (1) to describe the operations specific to some classical problems and advanced data structures; (2) to make an analysis of the the computational complexity of the specific operations Ability objectives: (1) to identify the most suitable data structure to solve a concrete problem; (2) to make use of it in a concrete implementation in a high level programming language, like C++; (3) to argue convincingly that the data structures and algorithms used in the implementation are suitable

8. Content / Conținuturi*

8.1. Lecture / Curs	Teaching strategies	Remarks, details
C1. Dynamic sets and related operations. Data structures for fast, key-based, search. Binary search trees. Limitations and motivation for better data	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 12 of [1].

structures.		
C2. Red-Black trees. Properties. Main operations and their efficient implementation.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 13 of [1].
C3. B-trees. Definition, basic operations, and the study of their computational complexity.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 18 of [1].
C4. Data structures for the fast search of the minimum and maximum element. Binary heaps. Definition, basic operations, and their computational complexity. The heapsort algorithm.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 6 of [1].
C5. Amortized analysis. Comparison with worst-case time analysis. Methods used for amortized analysis. Some case studies.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: ● Chapter 17 of [1] ● Slides from the website of this course.
C6. Fibonacci heaps. Structure of Fibonacci heaps, main operations, and the study of their computational complexity.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 19 of [1]
C7. Data structures for disjoint sets. Operations and alternative representations. Analysis of runtime complexity.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples,	References: Chapter 21 of [1].

	pseudocode, and proofs.	
C8. Data structures for dictionaries. Hash tables. Hash functions, and the problem of key collisions. Alternative implementations.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 11 of [1].
C9. String matching algorithms. The problem. The naïve method, the Rabin-Karp algorithm, string matching based on finite automata.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 32 of [1].
C10. String matching algorithms (2). The Knuth- Morris-Pratt algorithm.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 32 of [1].
C11. Computational geometry. Specific problems, data structures, and some related algorithms.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: ● Chapter 33 of [1] ● Slides from the website of this course.
C12. Advanced design and analysis techniques. Greedy strategies. Applications in data compression and task scheduling.	Interactive lesson based on beamer presentation, with motivating problem, illustrated examples, pseudocode, and proofs.	References: Chapter 16 from [1].
C13. Recap.	Survey of the main data structures and algorithms presented in course. Drawing some final conclusions	References: Slides from the website of this course.
C14. Colloquium.	Written exam with questions related to the	

	topics presented in the courses.	

Recommended bibliography

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, lab	**Teaching/learning strategies**	**Remarks, details**
L1. Implementation of some concrete operations on red-black trees. Analysis of their runtime complexity.	Assisted practical labworks in C++, using a popular IDE (Code::Blocks or Eclipse). Learning by collaboration, dialog, and code testing.	The labwork is accessible from the website of the course. It contains the statements problems to be solved and related program templates that should be used to solve them. The teacher provides additional hints, and answers the questions the students may have.
L2. Implementation of some operations on B-Trees and binary heaps. Uses of these data strucures to solve some practical problems.	Idem.	Idem. The students should hand over the deliverables required by the previous labworks. The teacher will evaluate them, and may ask additional questions to evaluate the students.
L3. Implementation of some operations on Fibonacci heaps. Uses of Fibonacci heaps to solve some practical problems.	Idem.	Idem. The students should hand over the deliverables required by the previous labworks. The teacher will evaluate them, and may ask additional questions to evaluate the students.
L4. Implementation of some operations on a concrete representation of disjoint sets. Uses of disjoint sets to solve some practical problems.	Idem.	Idem. The students should hand over the deliverables required by the previous labworks. The teacher will evaluate them, and may ask additional questions to evaluate the students.
L5. Solving some concrete problems, using the string matching algorithms presented	Idem.	Idem. The students should hand over the deliverables required by the

in lectures 9 and 10.		previous labworks. The teacher will evaluate them, and may ask additional questions to evaluate the students.
L6. Modeling and implementing some interesting problems from computational geometry.	Idem.	Idem. The students should hand over the deliverables required by the previous labworks. The teacher will evaluate them, and may ask additional questions to evaluate the students.
L7. Final evaluation.	Idem.	Idem. The students should hand over the deliverables required by the previous labworks. The teacher will evaluate them, and may ask additional questions to evaluate the students.

Recommended bibliography

[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 related material published on the website of the course: http://web.info.uvt.ro/~mmarin/lectures/ADS/

9. Correlations between the content of the course and the requirements of the IT field

This course extends the knowledge acquired by the students in the course of Data Structures, and prepares them to know how to model and solve efficiently several problems by choosing the most suitable data structures and algorithms for the problem at hand.

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
10.4. Lecture / Curs	- Describe the representation of a data structure, and the meanings of its components - Indicate the operations specific to the data structures discussed in this lecture, and the pseudocode of their implementation - Identify the best data structure or algorithm to solve a concrete problem and to solve it by writing the pseudocode of an algorithm - Establish the order of complexity of an algorithm or operation - Illustrate diagrammatically the	Colloquium	40%

	outcome of a particular operation on some simple examples.		
10.5. Seminar/ lab	The ability to implement certain operations on data structures, in C++	Lab works	20%
	The ability to solve some concrete problems using the data structures and algorithms presented in the courses	Lab works	20%
	Give correct answers to questions about properties of data structures, the complexity of their related operations.	Questions during the seminar/lab	20%
10.6. Minimal knowledge for passing			
The minimal knowledge and abilities required to get a passing grade (which is 5) include: - to write down the pseudocode for a simple operation on a data structure described in this lecture - to indicate the computational complexity of an operation or algorithm - to implement correctly a simple algorithm or operation in C++			

Date/ Data completării

1.10.2016

Signature (lecture) /
Semnătura titularului de curs

Signature (seminar)
Semnătura titularului de seminar

Signature (director of the department)
Semnătura directorului de departament
Conf.dr. Victoria Iordan