UNIVERSITY OF MITROVICA "ISA BOLETINI"

Course Curriculum Model (Syllabus)

Fakulty:	Faculty of Mechanical and Computer Engineering	
Department:	Computer Science and Engineering	
Level:	VI	
Code of the course:	201-CSE	
Course:	COMPUTING II	
Course Status:	Obligatory	(mandatory)
Semester:	III	(autumn)
Number of hours per week:	3+2	
ECTS:	6	
Time / location:		
Year of studies:	II	
Lecturer:	Prof. Ass. Dr. Muzfer Shala	
Assistent:	MSc. Agon Bajgora	
Contact details:	Lecturer	Assistant
Email:	muzafer.shala@umib.net	agon.bajgora@umib.net
Telefon:	+38349441338	+38345188550

Content	On this course about C# Programming student will continue with the most complicated concepts of the basics of programming, where more specifically are reviewed: data structures and implementation with C# and in the .NET platform. Further we look into linear data structures and their implementation in .NET Framework , list structures, linked list , dynamic self-stretching array, stack, tail, deck, trees, binary trees, balanced trees, black and red trees, B-trees, graphs, oriented, disoriented and weighed graphs. The study of the classic algorithms for crawling tree-like structure data will be done: breadth first search (BFS), depth-first search (DFS) and some algorithms on graphs. Then we review the data structure dictionary, associative array and multitude and their implementations in .NET. The special attention will be payed to the hashtables and their implementations, we study some algorithms for hashing, resolving collisions and some specifics with hashtables in .NET and C#. Will be done the comparison of the basic data structures in programming and analyze their effectiveness in terms of basic operations such as searching, inserting, deleting, etc. We look into important concepts of object-oriented programming such as abstraction, encapsulation, inheritance and polymorphism. To the students are given the recommendations for writing quality programming code and for effective programming problem solving with focus on testing all possible scenarios, coming from the problem.
Purpose	Intermediate problem solving and computer programming concepts, including algorithmic strategies, recursion, and effective design and use of data structures and application programming interfaces (APIs)

Accessibility	Upon successful completion of this course, students will: 1. Identify and explain a programming development lifecycle, including planning, analysis, design, development, and maintenance. 2. Demonstrate a basic understanding of object-oriented programming by using structures and classes in software projects. 3. Use object-oriented programming techniques to develop executable programs that include elements such as inheritance and polymorphism. 4. Document and format code in a consistent manner. 5. Apply basic searching and sorting algorithms in software design. 6. Apply single- and multi-dimensional arrays in software. 7. Use a symbolic debugger to find and fix runtime and logical errors in software. 8. Demonstrate a basic understanding of programming methodologies, including object oriented, structured, and procedural programming. 9. Describe the phases of program translation from source code to executable code	
Program	**weeks**	**Lecture**
	First week:	Course Introduction
	Second week:	Linear Data Structures
	***Third week*:**	Trees
	Fourth week:	Graphs
	Fifth week:	Dictionaries, Hash-Tables and Sets
	***Sixth week*:**	Data Structures and Algorithm Complexity
	Seventh week:	Object-Oriented Programming Principles
	Eighth week:	Object-Oriented Programming Principles
	***Ninth week*:**	High-Quality Programming Code
	***Tenth week*:**	Lambda Expressions and LINQ
	***Eleventh week*:**	Methodology of Problem Solving
	Twelfth week:	Sample Programming Exam – Topic #1
	***Thirteenth week*:**	Sample Programming Exam – Topic #2
	Fourteenth week:	Sample Programming Exam – Topic #3
	Fifteenth week :	Projects review

Literature	**Principal literature:** Fundamentals of Computer Programming with C#: Object-Oriented Programming, Data Structures, by Svetlin Nakov, Vesselin Kolev, Nakov's Team **Recommended Literature:**
Teaching methodology	• Lecture • Flipped classroom • Discussion • Group work • Exercises • Homework • Real world projects

Contribution to student workload (which should correspond to student learning outcomes - 1 ECTS credit = 25 hours)

Activity	Hours	Days/weeks	Total
Lectures	3	15	45
Exercise sessions (with TA)	1	15	15
Practical work	1	15	15
Office hours	0	0	0
Fieldwork	0	0	0
Midterms, seminars	0	0	0
Homework	2	15	30
Self-study	1	15	15
Final exam preparation	1	15	15
Time spent in exams	2	2	4
Projects, presentations, etc	1	11	11
Total			150

Evaluation

Teaching methodology:
(according to the Statute and Regulation for studies of UMIB)

Tests / Colloquia	
Practical test during exercises	30%
Projects	20%
Work in the classroom	20%
Final exam	30%

29.09.2022, Mitrovica **Course provider:**

(Name Surname)

_____

(Signature)