COURSE NAME	**Mathematical Theory of Computation**					
Code	PMM204	Year of study	1. and 2.			
Course teacher	Milica Klaričić Bakula	Credits (ECTS)	5,0			
Associate teachers		Type of instruction (number of hours)	P	S	V	T
			45		15	
Status of the course	Compulsory and elective	Percentage of application of e-learning	25			
COURSE DESCRIPTION						
Course objectives	The aim of this course is to introduce basic concepts and results in mathematical theory of computation and give students a deeper insight into connection between mathematics and computer science. Students will learn about the formal connections between finite automata, push-down automata, regular expressions, grammars and formal languages and they will adopt basic techniques for program verification.					
Course enrolment requirements and entry competences required for the course	Enrolment requirements: Taken course Mathematical Logic. Entry competences: sets and relations; functions; axiomatic set theory; mathematical proofs (in particular various induction proofs); first order theories, first order logic.					
Learning outcomes expected at the level of the course (4 to 10 learning outcomes)	Upon successful completion of this course students will be able to: - define complete partial orders and continuous functions on them, explain their role in mathematical theory of computation - define finite automata, regular expressions and related classes of languages and explain / prove connections between them - formulate language recognised by a given FA, construct FA recognizing a given language, grammar or regular expression and formulate regular expression describing the language recognized by a given FA - for a given language formulate its generating KF grammar and vice versa - using Pumping Lemma for KFL or RL prove that a given language is not KFL or RL - formulate language recognised by a given PDA (Push-Down Automata) and design a PDA recognizing a given KF language - explain the difference between syntax and semantics of a programming language and argument the importance of program verification - define operative, denotational and axiomatic semantics of IMP and prove their equivalence - verify simple IMP programs using denotational semantics or Hoare logic.					
Course content broken down in detail by weekly class schedule (syllabus)	- Introduction; alphabets; languages (2) - Partial orders; Complete partial orders; Fixed points; Fixed Point Theorem (4) - Deterministic finite automata (DFA) and their languages - Non-deterministic finite automata (NFA) and their languages; Equivalence of DFA and NFA (2) - Non-deterministic finite automata with empty transitions (1) - Regular languages; Pumping Lemma (2) - Class RL; RL= FAL (2) - Decision algorithms for RL (2) - Minimization of FA (2) - Context-free languages; Class KFL (2) - Pumping Lemma for KFL (2) - Right-linear languages. Class RLL (2) - RLL = RL (2) - Algebraic laws for regular expressions (2) - Push-down automata (2) - IMP language (1) - The operative semantics of IMP (2) - The denotational semantics of IMP (2) - The axiomatic semantics of IMP (2)					

	- Equivalence of OS and DS of IMP (1) - Completeness of the Hoare rules (4)
Format of instruction	Lectures and exercises/problem classes.
Student responsibilities	Attending classes, doing homework assignments. Working individually through exercises, in addition to group work during classes, is essential for understanding the material.
Screening student work *(name the proportion of ECTS credits for each activity so that the total number of ECTS credits is equal to the ECTS value of the course)*	Attending classes, doing homework assignments: 2 ECTS. Final written exam: 1.5 ECTS. Final oral exam: 1.5 ECTS.
Grading and evaluating student work in class and at the final exam	Final written and oral exam: equally evaluated in the final grade.
Required literature (available in the library and via other media)	1. M. Klaričić Bakula, A. Matković, Matematička teorija računarstva, PMF, Split, 2015.
Optional literature (at the time of submission of study programme proposal)	1. J. E. Hopcroft, R. Motwani, J. D. Ullman, Introduction to Automata Theory, Languages and Computation, Addison Wesley 2001. 2. J. Martin, Introduction to Languages and the Theory of Computation, McGraw Hill, 2010. 3. G. Winskel, The Formal Semantics of Programming Languages, MIT Press 1993. 4. K. R. Apt, E. R. Olderog, Verification of Sequential and Concurrent Programs, Springer 1991. 5. Moll, Arbib and Kfoury, Introduction to Formal Language Theory, Springer 1988.
Quality assurance methods that ensure the acquisition of exit competences	Summary feedback for the whole class after the exam. Anonymous student survey.
Other (as the proposer wishes to add)	