Study programme(s): Computer Science

Level: bachelor

Course title: Theoretical Computer Science

Lecturer: Miloš Stojaković

Status: obligatory

ECTS: 6

Requirements: Discrete Structures 1

Learning objectives

Students should learn and understand the basic concepts and methods of computer science, all the way from its historical context, laying a solid foundation for an algorithmic approach to problem solving.

Learning outcomes

Minimum: At the end of the course, it is expected that a student understands basic notions of complexity theory, using it to distinguish between different classes of problems.

Desirable: At the end of the course, it is expected that a successful student masters the concept of hardness, being able to classify and tackle some standard algorithmic problems based on their complexity.

Syllabus

Alphabets, words, languages, measuring the information content of words, representation of algorithmic tasks, decidability. Finite automata, regular and context-free grammars.

Turing machines and computability. Complexity theory, space and time complexity. NP-hardness, polynomial reductions, NP-completeness.

Design of polynomial algorithms, examples. Algorithms for hard problems, examples.

Literature

- M. Sipser, *Introduction to the Theory of Computation.* Thomson Learning, 2012.
- J. Hromkovič, *Theoretical Computer Science: Introduction to Automata, Computability, Complexity, Algorithmics, Randomization, Communication, and Cryptography,* Springer, 2011.
- J.E. Hopcroft, R. Motwani, J.D. Ullman, *Introduction to Automata Theory, Languages, and Computations,* Prentice Hall, 2006.

Weekly teaching load

Lectures:	Exercises:	Practical Exercises:	Student research:	Other:
3	2	0	0	**0**

Teaching methodology

Blackboard lectures, blackboard exercises.

Grading method (maximal number of points 100)

Pre-exam obligations	**points**	**Final exam**	**points**
Colloquia	**50**	*Oral exam*	**50**