COURSE UNIT DESCRIPTION

Course unit title	Course unit code
Data structures and algorithms	

Annotation
This course consists of fundamental and selected topics, addressing them aims to teach designing applied solutions for various problems of computer science, analysing and evaluating them, with the aim to pick the best-suited one for the given situation. The course contains both classical and modern problems of computer science, aiming to give starting knowledge from various fields of computer science, along with fundamentals, thus encouraging and facilitating further studies of those fields. After the course, a student should be able to pick or design appropriate data structures for the given situation, and to apply or to create an algorithm for solving common problems, and to use given environment to implement it all in the source code.

Lecturer(s)	Department where the course unit is delivered
Coordinator: Irmantas Radavičius **Other lecturers:**	Faculty of Mathematics and Informatics Vilnius University

Cycle	Type of the course unit
First	Compulsory

Mode of delivery	Semester or period when the course unit is delivered	Language of instruction
Mixed	1st semester	English, Lithuanian

Prerequisites
Prerequisites: Programming fundamentals

Number of credits allocated	Student‘s workload	Contact hours	Individual work
5	134	66	68

Purpose of the course unit: program competences to be developed
The purpose of the course unit is to present key programming concepts and to develop an ability to formalize and to algorithmize domain specific problems and create programs to solve them. **Generic competences:** ability to analyse and systemise the information (GK1) **Specific competences:** analysis and applications of continuous and discrete mathematical structures (SK4), development of algorithms and their complexity evaluation (SK5), programming (SK6), mathematical and computer modelling (SK10)

Learning outcomes of the course unit: students will be able to	Teaching and learning methods	Assessment methods
Students will be able to: • define and explain main concepts, related to programming languages, data and control structures and algorithms • apply fundamental algorithms for information processing (data organization, sorting, search, etc) tasks • formalize various problems from various domains and use elements of imperative programming create programs to solve them	Lectures Assignments Individual work	Homework Assignments Exam (written)

- find and use various (not explicitly presented during lectures) problem solving methods and algorithms for specific tasks		

Course content: breakdown of the topics	**Contact hours**						**Individual work: time and assignments**	
	Lectures	Tutorials	Seminars	Practice	Laboratory work	**Contact hours**	**Individual work**	**Individual work**
Data structures. Algorithms. Algorithm analysis. Asymptotic complexity. Empiric time measurement.	2				2	**4**	**4**	Individual reading Problem solving Homework: 1) computations 2) lists 3) sorting 4) search 5) graphs
Abstract data types. Lists, their types and implementations.	2				2	**4**	**4**	
Methods of computation. Iterative and recursive computing.	2				2	**4**	**4**	
Search problem and its complexity. Linear and binary search. Search trees.	2				2	**4**	**4**	
Balances search trees. AVL trees, 2-3 trees, 2-3-4 trees, B-trees.	2				2	**4**	**4**	
Sorting problem and its complexity. Features of sorting algorithms. Comparison-based sorting. Bubble sort and its variations. Selection, insertion sorting. Comb and Shell sort.	2				2	**4**	**4**	
Divide and conquer strategy. Tree sort, heap sort, merge sort, quick sort algorithms and their variations.	2				2	**4**	**4**	
Stacks, queues. Hash tables, collisions and collision management.	2				2	**4**	**4**	
Data-specific sorting algorithms. Counting sort, radix sort algorithms and variations.	2				2	**4**	**4**	
Graphs. Assignment problem and Hungarian algorithm.	2				2	**4**	**4**	
Distance. Euclidian and Manhattan distance. Shortest paths in Graphs, Dijkstra and A* algorithms.	2				2	**4**	**4**	
Edit distance. Text matching, Hamming, Levenstein distance and algorithms for their computation.	2				2	**4**	**4**	
Graph matching problem. Graph isomorphism. Graph edit distance and algorithms for its computation.	2				2	**4**	**4**	
Streams in graphs. Ford-Fulkerson algorithm.	2				2	**4**	**4**	
Algorithms for data encoding and data compression. Caesar's cipher. Huffman algorithm.	2				2	**4**	**4**	
Error correcting codes. Hamming codes.	2				2	**4**	**4**	
Exam						**2**	**4**	
Total:	32				32	**66**	**68**	

Assessment strategy	Weight, perc.	Deadline	Assessment criteria
Homework	30	Each homework has initial deadline, has to be improved upon during the semester and the final version has to be submitted until the end of the semester. Deadlines for initial submission: first for the 3rd, second for the 6th, third for the 9th, fourth for the 12th, and fifth for the 15th week of the semester.	All five homework assignments are evaluated equally (0.6 points each). Each has specific requirements for the initial and for the final submission, which have to be met on time to get the maximal grade. Failing to meet them or submitting late might give a partial grade.
Assignments	30	During each week of the semester, during classes, various assignments are given which have to be solved during the current week	All the topics and assignments to study them are treated equally, that is, proper submission (the answers are correct and/or programs are submitted that meet the requirements provided) gives 3/N points, for that topic, where N is the number of weeks in the semester.
Exam (written)	40	June	During the exam, participants of the course solve problems of various types and complexity. Exam consists of five parts: algorithm analysis, data structures; sorting algorithms; graphs algorithms; text-processing algorithms. To pass the exam, one has to get at least 30% of possible points (1.2 or more), and to take it one has to have at least 50% (3 points or more) of the points during the semester.
Extern		The student can repeat the course externally, if before they have participated fully and they accept the previously collected number of points. In this case, the points get accounted for and the student only repeats the exam. The student who is taking the course unit externally must inform the lecturer in the beginning of the semester and get the written consent with the above-mentioned number of points confirmed. If the student has not collected the minimal number of points required to pass, or the number of points collected does not suit the student, the subject cannot be repeated externally.	

Author	Publishing year	Title	Number or volume	Publisher or URL
Required reading				
V.Tumasonis	2004	Informatika		https://klevas.mif.vu.lt/~vladas/Informatika/
A.Juozapavičius	2007	Duomenų struktūros ir efektyvūs algoritmai		TEV, Vilnius
S.Ragaišis	2018	DSA kurso informacija		https://klevas.mif.vu.lt/~ragaisis/ADS2018/index.html
Recommended reading				
M.A.Weiss	1992	Data structures and algorithm analysis		The Benjamin/Cummings Publishing Company, Inc.