COURSE UNIT (MODULE) DESCRIPTION

Course unit (module) title	Code
INTRODUCTION TO PROGRAMMING	

Academic staff	Core academic unit(s)
Coordinating: assoc. prof. Vytautas Rudžionis **Other:**	Kaunas Faculty Institute of Social Sciences and Applied Informatics Muitinės g. 8, LT–44280 Kaunas

Study cycle	Type of the course unit
First	Mandatory

Mode of delivery	Semester or period when it is delivered	Language of instruction
In class	1 semester	English

Requisites	
Prerequisites:	**Co-requisites (if relevant):**

Number of ECTS credits allocated	Student's workload (total)	Contact hours	Individual work
5	133	48	85

Purpose of the course unit		
After completing the course, students will be able to write medium-complexity programs, understand the structure of programming languages, be able to formulate program requirements, know programming concepts, and understand secure programming principles.		
Learning outcomes of the course unit	**Teaching and learning methods**	**Assessment methods**
Will know the essential components of programming languages and their relationships with computer and information systems architecture, will learn to transform algorithms into program code.	Lectures, tutorials, laboratory work, problem solving, active learning methods (algorithm analysis, program prototype writing, system prototype design)	Laboratory work, laboratory work defenses, independent systems analysis, problem solving, test work, exam
Will know program quality evaluation criteria, will be able to identify potentially dangerous program areas from a security perspective.	Lectures, tutorials, laboratory work, problem solving, active learning methods (algorithm analysis, program prototype writing, system prototype design)	Laboratory work, laboratory work defenses, independent systems analysis, problem solving, test work, exam
Will be able to write medium-complexity programs, evaluate program compliance with specifications, and know secure programming principles.	Lectures, tutorials, laboratory work, problem solving, active learning methods (algorithm analysis, program prototype writing, system prototype design)	Laboratory work, laboratory work defenses, independent systems analysis, problem solving, test work, exam

Content	Contact hours							Individual work: time and assignments	
	Lectures	Tutorials	Seminars	Workshops	Laboratory work	Internship	**Contact hours, total**	**Individual work**	**Tasks for individual work**
1. Program structure: computer system structure, operating system, program interaction with the operating system, machine codes, assembler, high-level and low-level programming languages, program development tools	2			2			**4**	**14**	Literature studies; completion of laboratory assignments
2. Program development stages: analysis of program development tools, program project, direct compilation and virtual compilation, program code analysis, error detection	4			8			**12**	**20**	Program writting
3. Basic programming language elements: variables; data types; expressions; operations;	4			8			**12**	**12**	Program writting
4. Basic elements of programming languages: program control, branching operators; conditional operators; loop operators; selection operators	4			6			**8**	**20**	Program writting and code analysis
5. Working with files. Basic data structures and their management. Principles of secure programming. Usability.	2			8			**10**	**17**	Program code analysis, principles of software system development, preparation for an exam.
Exam		2					**2**	**2**	Preparation for an exam
Total	**16**	**2**		**30**			**48**	**85**	

Assessment strategy	Weight %	Deadline	Assessment criteria
Control work (K1)	20	On a predefined date	The student is given a task to write a program within 1 hour. It is graded on a 10-point scale according to the following criteria: implementation of functional requirements; accuracy of algorithm implementation; accuracy of program code.
Control work (K2)	20	On a predefined date	The student is given a task to write a program within 1 hour. It is graded on a 10-point scale according to the following criteria: implementation of functional requirements;

			accuracy of algorithm implementation; accuracy of program code.
Individual program preparation (S)	20	On a predefined date	Students are given an assignment to create a decision algorithm, define functional requirements, and write a program prototype in the programming language of their choice. They are graded on a 10-point scale according to the following criteria: accuracy of algorithm implementation; accuracy of program code; efficiency of program code; implementation of functional requirements; ability to modify the code; program reliability; program functionality The use of code generation tools is prohibited.
Exam	40	On a predefined date	The test consists of 10 closed-ended questions (of varying difficulty, ranging from understanding algorithms to knowledge of theoretical foundations), each worth one point. The scoring is as follows: each question is worth one point. Exam scores are weighted with a coefficient of 0.5 in the final grade.
Final mark: 0.2*K1+0.2*K2+0.20*S+0.4*E Extern exam assessment strategy: Not applicable Using of AI tools not permited if not stated otherwise by lecturer			

REGARDING TAKING THE COURSE AS AN EXTERNAL STUDENT

Check ☒					**If permitted, specify the conditions**
Not permitted	☐	**Permitted**	☒		After completing all the work independently and submitting it to the instructor no later than 5 business days before the scheduled exam date

REGARDING THE USE OF GENERATIVE ARTIFICIAL INTELLIGENCE (GAI) TOOLS (SUCH AS "CHATGPT" OR OTHERS) WHILE STUDYING THE COURSE:

Check ☒				**If permitted, specify the conditions**
Not permitted	☒	**Permitted**	☐	

Check ☒				**If permitted, specify the conditions**
Not permitted	☒	**Permitted**	☐	

REGARDING PROGRESS IN ACHIEVING LEARNING OUTCOMES

A student who (1) consistently fails to demonstrate, throughout the semester during practical sessions (seminars, exercises, etc.) and (2) who has not fulfilled all interim assessment requirements and assignments within the timeframe specified in the course description, shall not be permitted to participate in the examination session.

Author (-s)	**Publishing year**	**Title**	**Issue of a periodical or volume of a publication**	**Publishing house or web link**
Required reading				
Halterman R.	2015	Fundamentals of C++ Programming		https://tfetimes.com/wp-content/uploads/2015/04/progcpp.pdf
Ulla Kirch-Prinz, Peter Prinz	2022	A Complete Guide to Programming in C++		JONES AND BARTLETT PUBLISHERS, https://www.idpoisson.fr/volkov/C++.pdf

K. Backmann	2012	Structured Programming with C++		https://tfetimes.com/wp-content/uploads/2015/04/structured-programming-with-c-plus-plus.pdf
Recommended reading				
Group of authors	2021	C++ Notes for Professionals book		https://books.goalkicker.com/CPlusPlusBook
Stanley B. Lippman Josée Lajoie Barbara E. Moo	2013	C++ Primer		Addison-Wesley, https://zhjwpku.com/assets/pdf/books/C++.Primer.5th.Edition_2013.pdf
Clifford A. Shaffer	2012	Data Structures and Algorithm Analysis		Virginia Tech Press, https://tfetimes.com/wp-content/uploads/2015/04/C-3e20120102.pdf

NOTE: *Including Open Educational Resources in the reading list is recommended*