COURSE UNIT DESCRIPTION

Course unit title	Course unit code
Functional programming	

Lecturer(s)	Department where the course unit is delivered
Coordinator: doc. dr. Linas Laibinis **Other lecturers:**	Department of Computer Science Faculty of Mathematics and Informatics Vilnius University

Cycle	Type of the course unit
1st (BA)	Compulsory

Mode of delivery	Semester or period when the course unit is delivered	Language of instruction
Face-to-face	7th semester	English, Lithuanian

Prerequisites
Prerequisites: Informatics fundamentals, Data Structures and Algorithms

Number of credits Allocated	Student‘s workload	Contact hours	Individual work
5	138	54	84

Purpose of the course unit: programme competences to be developed

Purpose of the course unit:
to introduce the key concepts and principles of the functional programming paradigm, to solve problems and write programs in a functional style (e.g., using polymorphism and higher-order functions), to teach students the Haskell programming language.

Generic competences:

- Ability to apply the knowledge in practice. *(GK2)*.

Subject competences:

- Analysis and applications of continuous and discrete mathematical structures (SK4).
- Programming *(SK6)*.

Learning outcomes of the course unit: students will be able to	Teaching and learning methods	Assessment methods
• Understand the essential concepts of functional programming (closure, functional composition, recursion and induction, higher-order functions, pattern matching, polymorphism, etc.)	Lectures, problem-oriented teaching, case studies, literary reading, individual work, tutorials, laboratory work.	Laboratory works and results presentation, written exam (open, semi-open and close-ended questions and tasks).
• Build inductive user-defined data types and write efficient functional programs for them		
• Apply functional programming techniques to solve various problems from real world		
• Understand how imperative and functional programming styles can support each other and be used in combination in the current languages (such as Python, Java, Scala)		

Course content: breakdown of the topics	Contact hours							Individual work: time and assignments	
	Lectures	Tutorials	Seminars	Practice	Laboratory work	Practical training	**Contact hours**	**Individual work**	**Assignments**
Introduction to functional programming, its history and background, overview of the current functional languages, key functional programming concepts, introduction to Haskell.	4				2		**6**	**6**	Individual reading. Laboratory works. Self-control tasks.
Defining functions: guards, pattern matching, and recursion. The notions of closure and functional rewriting. Functional composition and currying. Higher-order functions.	6				3		**9**	**18**	
Lists, strings and tuples. Higher-order functions on lists: map, filter, list comprehension.	6				3		**9**	**18**	
Types and polymorphism. Computation as rewriting, lazy evaluation and infinite data structures. Conditional polymorphism and type classes.	4				2		**6**	**14**	
Non-linear data structures. User-defined datatypes. Functors and monads.	6				3		**9**	**15**	
Functional programming features in the current imperative languages (Python, Java, C#). Synergy of functional and object-oriented programming in Scala.	4				2		**6**	**8**	
Applications of functional programming in real world: parallel programming, the MapReduce framework in cloud, data analytics.	2				1		**3**	**5**	
Tutorials during the semester		4					**4**		
Final exam (written)							**2**		
Total	**32**	**4**			**16**		**54**	**84**	

Assessment strategy	Weight %	Deadline	Assessment criteria
Laboratory works	40	During the semester	The students are given a number of exercises (tasks) to be solved individually and/or in small groups for every practical session. Solutions must be presented until the next practical session. Separate exercises may be associated with different maximal number of points, depending on their difficulty. At least 50% of total points are required to take the exam.
Exam (written)	60	Exam session	During the given time, the students solve a number of theoretical and practical tasks.

Author	Publishing year	Title	Number or volume	Publisher or URL
Required reading				
L. Laibinis	2016	Functional programming (electronic course material, available online in the VU Virtual Learning Environment)		https://moodle.esec.vu.lt/course/view.php?id=26467
M. Lipovača	2011	Learn You a Haskell for Greater Good! (Available online)		No Starch Press
Recommended reading				
S. Thompson	2011	Haskell: The Craft of Functional Programming		Addison-Wesley
B. Sullivan et al.	2008	Real World Haskell (Available online)		O'Reilly