COURSE UNIT DESCRIPTION

Course unit title	Course unit code
Functional Programming	

Lecturer(s)	Unit
Coordinator: Viačeslav Pozdniakov **Other lecturers:**	Department of Software Engineering Faculty of Mathematics and Informatics Vilnius University

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

Mode of delivery	Semester or period when the course unit is delivered	Language of instruction
Face-to-face	5 and 7 semester	Lithuanian or English

Prerequisites
Prerequisites: Procedural Programming, Object Oriented Programming

Number of credits allocated	Student's workload	Contact hours	Individual work
5	130	66	64

Purpose of the course unit: programme competences to be developed

Purpose of the course unit – provide functional programming basics, introduce modern functional programming languages.

Generic competences:

- Communication and collaboration *(GK1)*.
- Life-long learning *(GK2)*.
- Social responsibility *(GK3)*.

Specific competences:

- Knowledge and skills of underlying conceptual basis *(SK4)*.
- Software development knowledge and skills *(SK5)*.
- Technological and methodological knowledge and skills, professional competence *(SK6)*.

Learning outcomes of the course unit: students will be able to	Teaching and learning methods	Assessment methods
Understand principles of functional programming and recognize them as well.	Lectures, discussions, laboratory works, self-dependent reading.	Written exam, laboratory works.
Write stateless (without any variables) programs.		
Get introduced to Category theory.		
Investigate features of any other functional programming languages.		
Apply functional programming design templates.		

Course content: breakdown of the topics	Contact hours							Individual work: time and assignments	
	Lectures	Tutorials	Seminars	Practice	Laboratory work (LW)	*Tutorial during LW*	**Contact hours**	**Individual work**	**Assignments**
Differences between procedure and function, recursion	2				2	*2*	**4**	**2**	Self-dependent reading. Laboratory work 1.
Lists, tuples, tail recursion	2				2		**4**	**2**	
Higher order functions. Lazy evaluation	2				2		**4**	**4**	
Haskell types, constructors, classes, inheritance	4				4		**8**	**6**	
Input/output in Haskell programs	2				2	*2*	**4**	**4**	Self-dependent reading. Laboratory work 2
Monads (IO, MonadPlus, etc.)	4				4		**8**	**4**	
Monad transformers	4				4		**8**	**9**	
Testing	2				2	*4*	**4**	**8**	Self-dependent reading. Laboratory work 3
Parser combinators	2				2		**4**	**2**	
Concurrent programming	3				3		**6**	**2**	
Software transactional memory	2				2		**4**	**2**	
Functional pearls	3				3		**6**	**2**	
Preparation for exam, exam itself		1					**2**	**17**	1 h for tutorial 1 h for exam 1 h for preparation
Total	**32**	**1**			**32**	*8*	**66**	**64**	

Assessment strategy	Weight %	Deadline	Assessment criteria
Exam	50%	January	All correctly answered exam tasks give 5 points. A student can take part in the examination only if he/she gets at least 1 point for laboratory works.
Laboratory work 1	20%	Week 6	Correctly written program gives 2 points. One week penalty after deadline – 0.2 points.
Laboratory work 2	20%	Week 12	
Laboratory work 3	10%	Week 16	Correctly written program gives 1 points. One week penalty after deadline – 0.1 points.

Author	Publishing year	Title	Number or volume	Publisher or URL
Required reading				
Bryan O'Sullivan, John Goerzen, and Don Stewart	2009	Real World Haskell		O'Reilly
Recommended reading				
Benjamin C. Pierce	1991	Basic Category Theory for Computer Scientists (Foundations of Computing)		The MIT Press
Will Kurt	(2017)	Get Programming with Haskell		Manning Publications
Christopher Allen, Julie Moronuki	(2017)	Haskell Programming from first principles		Gumroad
Miran Lipovača	2011	Learn You a Haskell for Great		http://learnyouahaskell.c

		Good!		om