I044	**Functional Programming**	L	S	E	ECTS
		2	0	2	6

Course objectives. Introduction to the functional programming concepts through evaluation of the mathematical functions and their application. Mastering the techniques of Haskell programming language.

Course prerequisites. High school informatics.

Syllabus.
1. Introduction. Functions and functional programming. Introduction to Haskell.
2. Basic data types and classes. Lists and tuples. Function as a type. Polymorphic and overloaded types.
3. Branching. Guards. Patterns. Lambda expressions. Sections. Generators.
4. Recursion. Multiple arguments and multiple recursion. Mutual recursion**.**
5. Higher order functions and composition. List processing.
6. Parsing strings. Regular expressions.
7. Interactive programs. User input.
8. Declaration of types, classes and data. Recursive types. Instances.
9. Lazy evaluation and reducible expressions. Modular programming.

EXPECTED LEARNING OUTCOMES

No.	**LEARNING OUTCOMES**
1.	The ability to separate a given problem into a set of subproblems and their representation in form of mathematical functions.
2.	Integration of concepts of functional programming into procedural and object oriented programming in modern programming languages.
3.	Improving the knowledge in the field of functional programming via literature, scientific papers and attending conferences and discussions.
4.	Capability to unambigously present conclusions and findings to the experts and laymen based on the knowledge and experience.
5.	Gaining the ability to continue the education in the field of functional programming.

COUPLING OF THE EXPECTED LEARNING OUTCOMES, TEACHING PROCESS ORGANIZATION AND THE EVALUATION OF THE TEACHING OUTCOMES

TEACHING PROCESS ORGANIZATION	**ECTS**	**EXPECTED LEARNING OUTCOMES ****	**STUDENT ACTIVITY ***	**EVALUATION METHOD**	**SCORE**	
					min	**max**

Lecture attendance	1	1-5	Class atendance, discussion, solving the problems individually and in a team	Lists with signatures, observing the student activity during the lectures	0	0
Homework	1	1, 3-5	Solving the problems individually	Grading	20	30
Knowledge evaluation (repeated exam)	2	1, 3, 4	Preparation for the written exam	Grading	30	50
Student project	2	1-5	Solving the real life problem individually	Grading, discussion	0	20
TOTAL	6				50	100

Teaching process and grading. Lectures will be illustrated using examples written in Haskell programming language. Excersises will be held both in auditorium and computer laboratory with the focus on programming in Haskell. Lecture and excersise attendance is mandatory. The exam consists of the written part which will be held after all the lectures and excersises. Positive scores on repeated exams, which students write during the semester, can be used as a substitute for the written exam. Students may improve their grades by writing given pieces of homework and doing the final project.

Can the course be taught in English: Yes.

Basic literature:

1. G. Hutton, Programming in Haskell, Cambridge University Press, New York, 2016.

Recommended literature:

1. R. Bird, Pearls of Functional Algorithm Design, Cambridge University Press, New York, 2010.
2. On the web site http://www.ps.uni-saarland.de/alice/ one may find functional programing language Alice along with several articles dealing with programming in Alice.
3. J. D. Ullman, Elements of ML Programming, ML97 Edition
4. On the web site http://rextester.com/l/haskell_online_compiler one may excute Haskell scripts.