Subject code	Credits
INF4001	6

Course title in Lithuanian

FORMALIOSIOS KALBOS IR JŲ TAIKYMAI

Course title in English

FORMAL LANGUAGES AND APPLICATIONS

Short course annotation in Lithuanian (up to 500 characters)

Dalyko tikslas – supažindinti su pagrindinėmis formaliųjų kalbų sąvokomis, baigtinių bei steko tipo automatų savybėmis ir programavimo kalbų transliavimo metodais. Studentai išmoksta sudaryti deterministinius ir nedeterministinius baigtinius automatus, reguliariasias išraiškas, steko tipo automatus, nuo konteksto nepriklausančias gramatikas, transformuoti tarpusavyje baigtinius automatus ir reguliariasias išraiškas, ir transformuoti tarpusavyje steko tipo automatus ir nuo konteksto nepriklausančias gramatikas. Įsisavinami pagrindiniai leksinės ir sintaksinės analizės principai, leksinių skenerių ir sintaksinių analizatorių generatoriai JLex ir JCUP, pagrindiniai kodo generavimo ir optimizavimo principai. Įgyjama grupinio darbo patirtis sudarant paprastos nuo konteksto nepriklausančios kalbos kompiliatorių.

Short course annotation in English (up to 500 characters)

This course is an introduction to formal languages and compiler design. During the course, students will gain knowledge on finite automata, regular and context-free languages and grammars, pushdown automata, lexical analysis, parsing, compiler compilers, semantic analysis, code generation and optimization as well as gain practical experience in compiler construction. The course structure consists of lectures, laboratory works, and individual work.

Prerequisites for entering the course

Discrete Structures and Mathematical Logic (INF1006). Demonstrate familiarity with basic programming concepts.

Course aim

Develop understanding of the use and properties of the common classes of formal languages, grammars, and automata.

Content

No	Content (topics)
1.	Automata. Introduction to Formal Proofs; Additional forms of Proof; Inductive Proofs; Central Concepts of Automata Theory.
2.	Finite Automata. Deterministic Finite Automata; Non-deterministic Finite Automata; Finite Automata with epsilon-transitions.
3.	Regular Expressions and Languages. Regular Expressions; Finite Automata and Regular Expressions; Applications of Regular Expressions; Algebraic Laws for Regular Expressions.
4.	Properties of Regular Languages. Proving non-regularity; Closure properties; Decision properties; Equivalence and Minimization of Automata.
5.	Context-Free Grammars and Languages. Context-Free Grammars (CFGs); Parse Trees; Applications of Context-Free Grammars; Ambiguity in Grammars and Languages.
6.	Pushdown Automata. Definition of Pushdown Automata (PDA); The Language of a PDA; Equivalence of PDAs and CFGs; Deterministic Pushdown Automata.
7.	Properties of Context-Free Languages. Normal Forms for CFGs; The Pumping Lemma for Context-Free Languages (CFLs); Closure properties of Context-Free Languages; Decision properties of CFLs.
8.	Translation of languages and the structure of a compiler. Brief survey of programming paradigms; Programming language implementation issues; Survey of compiler and interpreter stages.
9.	Lexical analysis. Lexical analyser; Lex; symbol table.
10.	Syntax analysis. BNF and EBNF; Deterministic syntax analysis, FIRST and FOLLOW, LL and LR grammars.
11.	Deterministic top-down syntax analysis. Left recursion removal. Left factoring. Predictive syntax analysis. Recursive descent.
12.	Deterministic bottom-up syntax analysis. Simple precedence analysis, LR analysis; Yacc.
13.	Declaration, modularity, and storage management. Declaration models; Parameterization mechanisms; Type parameterization; Mechanisms for sharing and restricting visibility of declarations; Garbage collection.
14.	Code generation. Intermediate and object code; Intermediate representations; Implementation of code generators; Code generation by tree walking; Context-sensitive translation; Register use.
15.	Optimization. Machine-independent optimization; Data-flow analysis; Loop optimizations; Machine-dependent optimization.

Distribution of workload for students (contact and independent work hours)

Lectures	**45 hours**
Laboratory work	**30 hours**
Individual students work	**85 hours**
Total:	**160 hours**

Structure of cumulative score and value of its constituent parts

Final written exam (50%), mid-term written exam (17%), and assessments of laboratory (practical) work (33%).

Recommended reference materials

No.	Publication year	Authors of publication and title	Publishing house	Number of copies in		
				University library	*Self-study rooms*	*Other libraries*
Basic materials						
1.	2011	A.Deveikis. Formaliųjų kalbų praktikumas.	VMU			etalpykla.vdu.lt
2.	2010	T.Æ. Mogensen. Basics of Compiler Design.	DIKU, University of Copenhagen			http://www.diku.dk/~torbenm/Basics
3	2001	J.E.Hopcroft, R.Motwani and J.D.Ullman. Introduction to Automata Theory, Languages, and Computation 2nd Ed.	Addison Wesley	1		
Supplementary materials						
4	2004	.B.Yehezkael. Course Notes on Formal Languages and Compilers.	Jerusalem College of Technology	http://homedir.jct.ac.il/~rafi/		
5	1986	A.V.Aho, R.Sethi, and J.D.Ullman. Compilers: Principles, Techniques, and Tools.	Addison Wesley	http://cs.uccs.edu/~gsc/pub/phd/ftorres/doc/Compiler.pdf		

Course programme designed by

Doc. Dr. Algirdas Deveikis, Department of Applied Informatics