Study programme(s): Information Technologies

Level: Bachelor

Course title: Compiler Construction 1

Lecturer: Mirjana Ivanović

Status: elective

ECTS: 7

Requirements: Object-Oriented programming 1, Data Structures and Algorithms 1

Learning objectives

The main objective of the course is to learn students about essential work of different phases of compilers and make them skilled to participate at bigger project and implement compilers for simple procedural and object-oriented languages.

Learning outcomes

Minimum: Successful students should be able to implement compiler for subset of procedural programming language based on given grammar for language specification.

Desirable: At the end of the course it is expected that successful students are able to develop adequate software for translation from input text to output text/form based on given specification (grammar rules).

Syllabus

Theoretical instruction

Techniques for programming languages specifications. Syntax diagrams, Backus and Extended Backus normal forms for programming language grammar specification. Context-free grammars, LL, LR and attributed grammars. Essential principles, tasks and phases of compilers: lexical analysis, syntax analysis using recursive descent technique, semantic analysis (type checking) and symbol table maintenance, code generation (using Virtual machine). Description of complete implementation of compiler for simple procedural (including some basic object-oriented concepts) programming language. Complier generators.

Practical instruction

Practical part is oriented to the gradual developement and extension of existing parts of code for a real compiler. During laboratory classes students have task to fully implement parts of compiler adding their own code. They gradualy develop compiler following theoretical classes.

Literature

Recomended

1. Hanspeter Mössenböck, Compiler Construction Slides, Institut für Systemsoftware, Johannes Kepler Universität Linz, Austria
2. V. Aho, J. D. Ullman: "Principles of Compiler Design", Addison-Wesley, 1977.
3. V. Aho, R. Sethi, J. D. Ullman "Compilers, Principles, Techniques and Tools, Addison-Wesley, 1985.

Weekly teaching load 5

Lectures:	Exercises:	Practical Exercises:	Student research:	Other:
2	1	2	/	/

Teaching methodology

Theoretical classes are based on the classical teaching model involving a projector and .ppt presentations. Basic principles and compiler functions are presented by illustrative examples. Students are expected to pass 3 theoretical tests.

At theoretical exercises, existing parts of code for compiler implementation will be explained in more details. During laboratory classes students will work on implementation of particular parts of their own versions of the compiler. All together 5 tasks-seminars are foreseen for complete implementation of the compiler and grades depend on number of completed tasks. At the oral exam students demonstrate understanding of principles of compilers.

Grading method (maximal number of points 100)

Pre-exam oblications	**points**	**Final exam**	**points**
Tests and practical tasks	60	Oral exam	40