Compilers and Interpreters

Programme course

4 credits

Kompilatorer och interpretatorer

TDDD55

Valid from: 2017 Spring semester

Determined by
Board of Studies for Computer Science
and Media Technology

Date determined
2017-01-25

Main field of study
Information Technology, Computer Science and Engineering

Course level
First cycle

Advancement level
G2F

Course offered for
- Computer Engineering, B Sc in Engineering
- Programming
- Computer Science and Engineering, M Sc in Engineering
- Industrial Engineering and Management - International, M Sc in Engineering
- Industrial Engineering and Management, M Sc in Engineering
- Information Technology, M Sc in Engineering

Specific information
Overlapping course contents: TDDB44

Entry requirements
Note: Admission requirements for non-programme students usually also include admission requirements for the programme and threshold requirements for progression within the programme, or corresponding.

Prerequisites
Participants are expected to have knowledge of: a procedural programming language such as Pascal, internal data structures such as arrays and lists, and theory and implementation of abstract data types.

Intended learning outcomes

The aim of this course is to give a basic introduction to the theoretical and practical issues underlying the design and implementation of compilers. After the completion of the course you should be able to:

- explain and apply fundamental principles and techniques of compiler design
- explain and use methods for lexical analysis, top-down and bottom-up parsing
- explain and use methods for basic semantic analysis and syntax-directed translation
- construct and implement a top-down parser for a given context-free grammar,
- use a generator to build a lexical analyzer, apply a parser generator, and
- implement simple intermediate code generation from abstract syntax trees

Course content

Different types of translators such as compilers and preprocessors. Grammars and formal languages. Lexical and syntax analysis. Intermediate code generation. The course also gives a brief introduction to: Memory management and run-time organization. Code generation and code optimization. Basic usage of compiler construction tools. Criteria for language design. The labs contain exercises in hand-implementation and usage of tools for construction of smaller parts of a compiler, such as lexer, parser, generation of intermediate code.

Teaching and working methods

The theory is presented during the lectures. The laboratory assignments consists of building compiler components such as syntactical and lexical analysers.

Examination

LAB1	Laboratory work	2 credits	U, G
TEN1	Written examination	2 credits	U, 3, 4, 5

The questions in the written exam check how well the student has fulfilled the learning goals of the course. For passing the exam, deficits in fulfilling certain partial goals can be balanced by a better fulfilling of other partial goals.

Grades

Four-grade scale, LiU, U, 3, 4, 5

Department
Institutionen för datavetenskap

Director of Studies or equivalent
Ahmed Rezine

Examiner
Martin Sjölund

Education components
Preliminary scheduled hours: 32 h
Recommended self-study hours: 75 h

Course literature

Additional literature

Books

Aho, Lam, Sethi, Ullman, (2006) *Compilers - Principles, techniques, and tools*
Second edition Addison-Wesley

Compendia

Common rules

Regulations (apply to LiU in its entirety)

The university is a government agency whose operations are regulated by legislation and ordinances, which include the Higher Education Act and the Higher Education Ordinance. In addition to legislation and ordinances, operations are subject to several policy documents. The Linköping University rule book collects currently valid decisions of a regulatory nature taken by the university board, the vice-chancellor and faculty/department boards.

LiU's rule book for education at first-cycle and second-cycle levels is available at http://styrdokument.liu.se/Regelsamling/Innehall/Utbildning_pa_grund-_och_avancerad_niva.