Course syllabus

Kompilatorer
Compilers

EDAN65, 7,5 credits, A (Second Cycle)

Valid for: 2023/24
Faculty: Faculty of Engineering, LTH
Decided by: PLED C/D
Date of Decision: 2023-04-18

General Information

Elective for: C4-pv, D4-is, D4-pv, E4, E4-pv, F4, F4-pv, Pi4
Language of instruction: The course will be given in English on demand

Aim

The fundamental theories and methods used in Compiler Construction are central to the discipline of Computer Science, and widely applicable in both research and software construction.

The course aims to give knowledge of the principal structure of a compiler and about the basic theories and methods used to implement the different parts of the compiler. It should give an understanding for how the theories and the methods can be used in related areas like translation between different source languages, analysis of program properties, interpretation, and analysis of other artificial languages. The course also includes how advanced object-oriented techniques, such as design patterns and aspect-oriented programming can be applied to compiler construction.

Learning outcomes

Knowledge and understanding
For a passing grade the student must

- understand formalisms for syntactic descriptions: regular expressions, context-free grammars, and abstract grammars.
- understand formalisms for program analysis: attribute grammars
- be able to describe compiler phases, runtime systems, and different kinds of internal representations.

Competences and skills
For a passing grade the student must

- be able to implement parsers for programming languages with parser-generators
- be able to implement semantic analysis and code generation using attribute grammars and imperative programming
- be able to implement runtime systems by interpretation

Contents

The architecture of a compiler. The definition of programming languages: regular expressions, context-free grammars, abstract grammars, attribute grammars. Methods: scanning, parsing, static-semantic analysis, code generation. Runtime systems: memory allocation, automatic memory management (garbage collection). Tools: Scanner-generators, parser-generators, abstract syntax-tree generators, generatorer för attributevaluering. Design patterns (visitor, interpreter). Application areas for compiler construction.

Examination details

Grading scale: TH - (U,3,4,5) - (Fail, Three, Four, Five)
Assessment: Written examination. To qualify for the written examination, students must have completed their programming assignments. The final grade for the entire course is based on the result of the written examination.

The examiner, in consultation with Disability Support Services, may deviate from the regular form of examination in order to provide a permanently disabled student with a form of examination equivalent to that of a student without a disability.

Parts
Code: 0114. **Name:** Written Examination.
Credits: 4,5. **Grading scale:** TH. **Assessment:** Written examination. The final grade of the entire course is based on the result of this exam. To qualify for the written exam, students must have completed their laboratory work. **Contents:** Written examination
Code: 0214. **Name:** Laboratory Work.
Credits: 3. **Grading scale:** UG. **Assessment:** Completed laboratory exercises. **Contents:** Laboratory work

Admission

Admission requirements:

- EDAA01 Programming - Second Course or EDAA30 Programming in Java - Second Course

Assumed prior knowledge: Object-oriented Programming in Java. Basic data structures.
The number of participants is limited to: No
The course overlaps following course/s: EDA200, EDA180

Reading list

- Appel, A W: Modern Compiler Implementation in Java. Cambridge University Press, 2002, ISBN: 052182060X. Recommended textbook.

Contact and other information

Course coordinator: Professor Görel Hedin, Gorel.Hedin@cs.lth.se
Course homepage: http://cs.lth.se/edan65