UNIVERSITY OF GOTHENBURG

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DIT231 Programming Language Technology, 7.5 credits

Programspråksteknik, 7,5 högskolepoäng

Second Cycle

Confirmation

This course syllabus was confirmed by Department of Computer Science and Engineering on 2012-11-12 and was last revised on 2019-02-07 to be valid from 2019-09-02, autumn semester of 2019.

Field of education: Science 100%
Department: Department of Computer Science and Engineering

Position in the educational system

The course is a part of the Computer Science Master's Programme and a single-subject course at the University of Gothenburg.

The course can be part of the following programmes: 1) Computer Science, Master's Programme (N2COS), 2) Applied Data Science Master's Programme (N2ADS), 3) Computer Science, Bachelor's Programme (N1COS) and 4) Software Engineering Master's Programme (N2SOM)

Main field of studies	*Specialization*
Software Engineering	A1N, Second cycle, has only first-cycle course/s as entry requirements
Computer Science-Algorithms and Logic	A1N, Second cycle, has only first-cycle course/s as entry requirements

Entry requirements

The requirements for the course is to have successfully completed courses corresponding to 60 hec in the subject of Computer Science, including

• 7.5 hec in programming (for example DIT143 Functional programming, DIT953 Objektorienterad programmering and design, or equivalent);
• 7.5 hec in data structures (for example DIT961 Data structures, DIT725 Logic,

algorithms and data structures or equivalent).

English B level or English proficiency equivalent to IELTS 6.5 no part under 5.5 or TOEFL 575 p, TWE score 4.5 is also required.

Learning outcomes

After completing the course the student is expected to be able to:

Knowledge and understanding

- explain the functioning of finite automata;
- explain the principles of LL and LR parsing;
- explain the concept of compiler correctness;

Competence and skills

- define and implement abstract syntax;
- define the lexical structure of programming languages by using regular expressions, and implement lexical analyzers by using standard tools;
- define the syntax of programming languages by using context-free grammars and implement parsers by using standard tools;
- write simple code generators;
- apply the technique of syntax-directed translation and its efficient implementation in their chosen programming language;
- formulate typing rules and implement type checkers for functional and imperative languages;
- formulate operational semantic rules and implement interpreters for functional and imperative languages;

Judgement and approach

- judge about programming language design issues as for example with the respect to efficiency and usability.

Course content

The aim of the course is to give understanding of how programming languages are designed, documented, and implemented. The course covers the basic techniques and tools needed to write interpreters, and gives a summary introduction to compilation as well.

The students will learn about grammars when writing the syntax analysis and about type systems when implementing the type checker. When implementing the interpreter and compiler the students will learn about practical implementation concerns as well as the theory of formal semantics.

Sub-courses

1. **Written exam** *(Tentamen),* 6 credits
   Grading scale: Pass with Distinction (VG), Pass (G) and Fail (U)
2. **Laboration** *(Labborationer),* 1.5 credits
   Grading scale: Pass (G) and Fail (U)

Form of teaching

The teaching consists of lectures, exercises, and laborations, as well as individual supervision in connection to the laborations.

Language of instruction: English

Assessment

The course is examined through an individual written exam in an examination hall at the end of the course and laboratory assignments carried out individually or in pairs.

If a student, who has failed the same examined component twice, wishes to change examiner before the next examination, a written application shall be sent to the department responsible for the course and shall be granted unless there are special reasons to the contrary (Chapter 6, Section 22 of Higher Education Ordinance).

In cases where a course has been discontinued or has undergone major changes, the student shall normally be guaranteed at least three examination occasions (including the ordinary examination) during a period of at least one year from the last time the course was given.

Grades

The grading scale comprises: Pass with Distinction (VG), Pass (G) and Fail (U).
To be awarded the grade Pass with Distinction (VG), the student must pass the sub-course Laboration and get the grade Pass with Distinction on the sub-course Written exam.

Course evaluation

The course is evaluated through meetings both during and after the course between teachers and student representatives. Further, an anonymous questionnaire is used to ensure written information. The outcome of the evaluations serves to improve the course by indicating which parts could be added, improved, changed or removed.

Additional information

The course replaces the course DIT230 Programming Languges. The course cannot be included in a degree which contains DIT230. Neither can the course be included in a degree which is based on another degree in which the course DIT230 is included.

The course cannot be included in a degree which contains DIT229. Neither can the course be included in a degree which is based on another degree in which the course DIT229 is included.

Course literature to be announced the latest 8 weeks prior to the start of the course.

The course is a joint course together with Chalmers.