GÖTEBORG UNIVERSITY
Faculty Board of the IT University

DIT229, Programming Languages, 7.5 higher education credits

First Cycle

The syllabus is the binding document.

1. Confirmation

The syllabus was confirmed by the Faculty Board of IT University on 2007-10-10 to be valid from the spring semester, 2008.

Field of education: Sciences.

Responsible department: Computer Science and Engineering.

2. Position in the educational system

The course is a part of the Computer Science Bachelor´s programme and an elective course at Göteborg University.

Kommentar [MSOffice1]: Vilka kurser är det? Räkna upp dem med koder. Viktigt ifall kursen ska vara en fristående kurs.

3. Entrance qualifications

The requirement for the course is to have successfully completed the first year at the Computer Science Bachelor´s programme or equivalent.

4. Course content

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. By experimenting with language extensions the students will get an insight into good and bad programming language designs.

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

5. Learning outcomes

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 who have passed the course should be able to:

- define the lexical structure of programming languages by using regular expressions, explain the functioning of finite automata, and implement lexical analysers by using standard tools;
- define the syntax of programming languages by using context-free grammars, explain the principles of LL and LR parsing, and implement parsers by using standard tools;
- define and implement abstract syntax;
- master the technique of syntax-directed translation and its efficient implementation in their chosen programming language;
- formulate typing rules and implement type checkers;
- formulate operational semantic rules and implement interpreters;
- write simple code generators;
- be familiar with the basic implementation issues of both imperative and functional languages;
- design and implement special-purpose programming languages.

6. Required reading

See separate literature list.

7. Assessment

The course is graded by group exercises and written individual exams.

A student who has failed a test twice has the right to change examiner, unless weighty argument can be adduced. The application should be sent to the department and has to be in writing.

8. Grading scale

The course is graded with the following marks: High Pass (VG), Pass (G) or Fail (U). On request, grades are mapped onto an ECTS grading scale using a fixed mapping, common for GU.

9. Course evaluation

The course is evaluated through meetings both during and after the course between teachers and student representatives. Further, an anonymous questionnaire can be 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.

10. Additional information

The course is given in English.