UNIVERSITY OF GOTHENBURG

THE IT FACULTY BOARD

DIT260, Advanced Functional Programming, 7,5 higher education credits

Advanced Functional Programming, 7.5 högskolepoäng

Second Cycle

---

1. Confirmation

The course syllabus was confirmed by The IT Faculty Board on 2009-09-19 and was last revised by The IT Faculty Board on 2011-11-27 to be valid from 2013-01-21.

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

2. Position in the educational system

The course is a part of the Computer Science Master's programme and an elective course at the University of Gothenburg. The course is also offered as an elective coure in the Software Engineering Master's programme. The level for the course in relation to degree requirements is Master´s degree, code A1F. The course has course/courses at second cycle level as entry requirements.

Main field of studies	*Specialization*
Computer Science-Algorithms and Logic	A1F, Second cycle, has second-cycle course/s as entry requirements

3. Entry requirements

To be eligible for the course, students should have successfully completed two years of an education aimed at a bachelor degree within Computer Science or equivalent. The courses DIT142 Functional programming, MMGD10 Introductory Discrete Mathematics and at least one of the courses DIT230 Programming language technology or DIT331 programming paradigms or equivalent are required.

Notions: Abstract syntax tree, semantics, interpreter, compiler.
Algorithms, complexity, divide-and-conquer. Induction proofs and simple logic, equality reasoning.

4. Course content

The aim of the course is to explore the powerful mechanisms that functional programming languages offer to solve real problems and structure larger programs. The focus lies on library design and the concept of embedded languages. In the course the student will study libraries for parsing, pretty printing, graphics, pseudo-parallel programming and interaction. The course will also present some recent research which can make the contents of the course vary to some degree. The programming language used in the course is Haskell.

Subcourses

Examination *(Examination)*, 3,0 hp grading scale: Fail (U), Pass (G), Pass with Distinction (VG)
Laboratory *(Laboratory)*, 4,5 hp grading scale: Fail (U), Pass (G), Pass with Distinction (VG)

5. Learning outcomes

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

Knowledge and understanding:

- define embedded domain specific languages (EDSLs)
- explain and exemplify (abstract) syntax and semantics
- reason about correctness of functional programs

Skills and abilities

- transform programs on the basis of such reasoning
- design embedded domain specific languages (EDSLs)
- implement EDSLs in Haskell (as combinator libraries)
- demonstrate ability to read, understand and extend Haskell programs which use advanced type system features: type classes, (generalized) algebraic datatypes, functors and monads and monad transformers
- use specification based development techniques
- formulate ant test properties about the program

Judgement and approach

- explain and discuss the above topics.

6. Literature

See separate literature list.

7. Assessment

The course is examined by 2-3 programming labs (U-VG) done in pairs during the course and an individual exam (U-VG) given in an examination hall at the end.

A student who has failed the examination twice has the right to request of the department a change of examiner. The request is to be in writing and submitted as soon as possible. the department is to grant such a request without delay.

In cases where a course has been discontinued or major changes have been made a student should be guaranteed at least three examination occasions (including the ordinary examination occasion) during a time of at least one year from the last time the course was given.

8. Grading scale

The grading scale comprises Fail (U), Pass (G), Pass with Distinction (VG).
Advanced Functional Programming (7.5 hecs): the final grade of the full course is based 60% on the lab result and 40% on the exam result.

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

Language of instruction: English.
It is recommended, but not required, to read the following courses beforehand: DIT600 Algorithms and DIT201 Logic in computer science or DIT321 Finite Automata Theory.