DD1362 Programming Paradigms 6.0 credits

Programmeringsparadigm

This is a translation of the Swedish, legally binding, course syllabus.

If the course is discontinued, students may request to be examined during the following two academic years

Establishment

On 2020-10-13, the Head of the EECS School has decided to establish this official course syllabus to apply from spring semester 2021, registration number J-2020-1862.

Grading scale

A, B, C, D, E, FX, F

Education cycle

First cycle

Main field of study

Technology

Language of instruction

The language of instruction is specified in the course offering information in the course catalogue.

Intended learning outcomes

After passing the course, the student should be able to:

- apply and explain general concepts in programming, in particular flow of control, recursion, interpretation, paradigms and computing models
- apply and explain basic concepts in functional programming, in particular clean functions, reference transparency, higher order functions, lazy evaluation, immutability, types and classes
- apply and explain basic concepts in formal languages and syntax analysis, in particular automata, regular expression, grammars, lexical analysis and recursive descent
- write own client-server programs as well as use protocols and be able to explain how they are interpreted and written

in order to

- obtain a broader perspective on programming
- be able to assess which paradigm and which programming language that is appropriate to solve a certain assignment
- be able to use adequate programming style in a chosen programming paradigm
- be able to participate in discussions about programming paradigms, history of programming languages, language definition, properties of type systems, principles of language design, language translation, programming principles and programming concepts actively

Course contents

Functional programming: the function concept, higher order functions, currying, evaluation strategies, streams, pattern matching, overloading, polymorphism, interpretation, types and classes.

Formal languages and syntax analysis: automata, regular expressions, grammars, lexical analysis, recursive descent, classes of languages

Internet programming.

Language translation: interpretation, compilation and linking.

Specific prerequisites

Completed courses in programming equivalent to DD1337 and algorithms and data structures equivalent to DD1338.

Active participation in a course offering where the final examination is not yet reported in LADOK is considered equivalent to completion of the course.

Registering for a course is counted as active participation.

The term 'final examination' encompasses both the regular examination and the first re-examination.

Examination

- LAB1 - Laboratory work, 3.5 credits, grading scale: A, B, C, D, E, FX, F
- TEN1 - Examination, 2.5 credits, grading scale: P, F

Based on recommendation from KTH's coordinator for disabilities, the examiner will decide how to adapt an examination for students with documented disability.

The examiner may apply another examination format when re-examining individual students.

Ethical approach

- All members of a group are responsible for the group's work.
- In any assessment, every student shall honestly disclose any help received and sources used.
- In an oral assessment, every student shall be able to present and answer questions about the entire assignment and solution.