# Data and Program Structures

Programme course

6 credits

Data- och programstrukturer

TDDA69

Valid from: 2017 Spring semester

**Determined by**
Board of Studies for Computer Science
and Media Technology

**Date determined**
2017-01-25

# Main field of study
Computer Science and Engineering, Computer Science

# Course level
First cycle

# Advancement level
G2X

# Course offered for
-
- Computer Science and Engineering, M Sc in Engineering
- Information Technology, M Sc in Engineering
- Computer Science and Software Engineering, M Sc in Engineering

# Entry requirements
Note: Admission requirements for non-programme students usually also include admission requirements for the programme and threshold requirements for progression within the programme, or corresponding.

# Prerequisites
Functional programming, preferably Common Lisp or Scheme, and also to some extent object oriented programming.

# Intended learning outcomes
Provide knowledge of various programming paradigms and evaluation strategies. The aim of the course is that the students should gain knowledge of different programming paradigms and strategies for evaluation, and how these can be implemented. After the course the student will be able to:

- describe aspects of evaluation and execution in different language models
- explain and demonstrate how design choices affect the expressiveness and efficacy of a programming language
- analyze and value programming languages based on their evaluation and compilation strategies
- implement programming languages in the form of an interpreter and a compiler

## Course content

The following topics are adressed during lectures and lessons:

- The Scheme programming language, used as a tool in the literature and during the labs, and as a model language for implementing other languages.
- The substition and environmental model.
- Programming paradigms and models of evaluation, e.g. functional, imperative, object oriented; lazy evaluation, streams, non-deterministic evaluation, logic programming.
- Aspects of evaluation, parameter passing, exection environments and models of recursion.
- Study and implmenentation of interpreters and compilers, where Scheme is the model language.

## Teaching and working methods

The theoretical part of the course is treated during the lectures. The seminars are used for preparing for the laboratory assignments and for exercising the solving of applied problems. The laboratory work provides practical experience.
The course runs over the entire spring semester.

## Examination

| | | | |
|---|---|---|---|
| LABA | Laboratory Work | 4.5 credits | U, G |
| TENA | Written examination | 1.5 credits | U, 3, 4, 5 |

## Grades

Alternative-grade scale, LiU, U, 3, 4, 5

## Other information

*Supplementary courses:*
Logic Programming, Programming Theory.

## Department

Institutionen för datavetenskap

## Director of Studies or equivalent

Peter Dalenius

## Examiner

Cyrille Berger

## Course website and other links

http://www.ida.liu.se/~TDDA69

## Education components

Preliminary scheduled hours: 64 h
Recommended self-study hours: 96 h

## Course literature

Abelson H. & Sussman, G.J. (1996) Structure and Interpretation of Computer Programs. MIT Press. Laborationsmaterial.

# Common rules

Regulations (apply to LiU in its entirety)

The university is a government agency whose operations are regulated by legislation and ordinances, which include the Higher Education Act and the Higher Education Ordinance. In addition to legislation and ordinances, operations are subject to several policy documents. The Linköping University rule book collects currently valid decisions of a regulatory nature taken by the university board, the vice-chancellor and faculty/department boards.

LiU's rule book for education at first-cycle and second-cycle levels is available at http://styrdokument.liu.se/Regelsamling/Innehall/Utbildning_pa_grund-_och_avancerad_niva.