Programming and Data Structures

Programme course

8 credits

Programmering och datastrukturer

TDDC76

Valid from: 2017 Spring semester

Determined by
Board of Studies for Electrical Engineering,
Physics and Mathematics

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

- Biomedical Engineering, M Sc in Engineering
- Applied Physics and Electrical Engineering - International, M Sc in Engineering
- Applied Physics and Electrical Engineering, M Sc in Engineering
- Physics and Nanotechnology
- Mathematics
- Engineering Biology, M Sc in Engineering
- Chemical Biology

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

A basic course in programming is required for this course, especially basic skills in problem solving and construction of small programs. Basic knowledge of computers and familiarity with unix based computer systems.

Intended learning outcomes

This course offer a deeper study in programming, especially imperative and object-oriented programming in the programming language C++, and also knowledge about frequently used data structures and algorithms. Upon completion of this course the

student should be able to:

- Design imperative and object-oriented programs and implement them in the programming language C++.
- utilize object-oriented program development methodology.
- Describe and use frequently used data structures and algorithms.

Course content

Lectures, lessons and programming exercises deal with the programming language C++ and its environment, object-oriented analysis and design, programming in C++, data structures and algorithms.

- Imperative programming in C++ (variables, constants, declarations, expressions, statements, functions, fundamental data types and data structures).
- Classes (declaration, data members, member functions, nested types, access specification for class members, constructors, destructors, derived classes, polymorphism, dynamic type control and type conversion).
- Object-oriented program design (object-oriented analysis, design and coding).
- The C++ standard library (input and output, character and string handling, containers).
- Data structures: lists, stacks, queues, priority queues, trees, especially search trees, hash tables, heaps, especially binary heap.
- Algorithms: searching and sorting.

Teaching and working methods

The course is arranged as a series of lectures, lessons and programming exercises, and a small programming project. Self-study hours must be set aside for literature study, programming exercises, project work and home examination.

Examination

UPG4	Optional reflection paper	U, G	0 credits
UPG3	Computer-based exercises	U, G	1.5 credits
LAB2	Laboratory work	U, G, VG	3.5 credits
PRA1	Project work	U, G	3 credits

The programming exercises give the student opportunity to practice and test her/his knowledge and skills in procedural and object-oriented programming.
The project work give the student opportunity to practice and test her/his knowledge and skills in object-oriented analysis, design and programming within a project group.

The computer based exercises tests the students knowledge about data structures and

algorithms and is also an opportunity for learning.
The final grade (U,3,4,5) for this course is based on the laboratory work ("pass" or "pass with distinction") in combination with the reflection paper ("pass" only). The laboratory work form the base with "pass" translated to grade 3 and "pass with distinction" to grade 4. The reflection paper may raise the grade one step to either 4 or 5.

Grades

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

Other information

Supplementary courses: This course is well suited for supplemantary courses where good knowledge in procedural and object-oriented programming, especially in C++, is required. Examples of such courses are courses covering compilers and interpreters, databases, concurrent programming, real-time programming, operating systems, design patterns, software engineering projects, and programming of parallel computers.

Department

Institutionen för datavetenskap

Director of Studies or equivalent

Ahmed Rezine

Examiner

Eric Elfving

Course website and other links

http://www.ida.liu.se/~TDDC76/

Education components

Preliminär schemalagd tid: 98 h
Rekommenderad självstudietid: 115 h

Course literature

Kompletterande litteratur

Böcker

Bjarne Stroustrup, (2013) *The C++ Programming Language, 4/E* Addison-WesleyStanley B. Lippman, Josée Lajoie, Barbara E. Moo, (2012) *C++ Primer, 5/E* Addison-Wesley

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.