ID1020 Algorithms and Data Structures 7.5 credits

Algoritmer och datastrukturer

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

Course syllabus for ID1020 valid from Spring 2019

Grading scale

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

Education cycle

First cycle

Main field of study

Technology

Specific prerequisites

A basic programming course such as ID1018: Programming I

Language of instruction

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

Intended learning outcomes

The aim of the course is to provide a solid knowledge of how one designs and analyses the most important classes of algorithms. The course intends to give the students skills that give them the opportunity to design computer programs that uses time and memory in an efficient way independently. The students should learn to identify problems that uses too much resources to be solved efficiently or that indeed is impossible to solve with common computing models. At the end of the course the students should be able to develop their own algorithms to problems and be able to compare different solutions and their efficiency.

After successfully having completed the course the students should be able to:

1. develop and implement algorithms and data structures and analyse them regarding correctness.
2. define the terms P, NP, NP-completeness and computability.
3. analyse how efficient algorithms and data structures are based on different measures on efficiency such as time and memory complexity.
4. write programs that implement larger units that use algorithms and data structures with the help of good programming principles such as the specification of API:s and the use of tests that utilise algorithms or data structures.
5. solve problems through the use of data structures such as linear lists, stacks, queues, hash tables, binary trees, heaps, tournament trees, binary search trees and graphs and write programs for the solutions.
6. solve problems by using algorithm design methods such as greedy algorithms, decomposition, dynamic programming backtracking and branch and bound and write program for the solutions.
7. given a specific problem either design an appropriate data structure or create an algorithm that solves the problem or identifies the problem as one that can not be solved efficiently

Course contents

Basic analysis

- Asymptotic analysis of bounds for the complexity in the worst, best and average cases.
- To identify differences between behaviours at best, worst and average cases.
- Ordo -, omega - and the theta-notation.
- The most common complexity classes.
- Empirical estimates of complexity.
- Time - and memory complexity.
- The use of recurrance relations to analyse algorithms.

Strategies for algorithms and data abstraction

- Brute-force algorithms.
- Greedy algorithms.

- Decomposition.
- Backtracking.
- Branch-and-bound.
- Heuristics.

Basic algorithms

- Simple numerical algorithms.
- Sequential and binary search algorithms.
- Quadratic sorting algorithms (selection, deposit).
- Sorting in O(N log N) (Quicksort, heapsort, mergesort).
- Hash tables including such that avoid collisions.
- Binary search trees.
- Representations of graphs.
- Depth first - and breadth first search.
- Shortest path -algorithms (the algorithms of Dijkstras and Floyds).
- Transitive closure (Floyds algorithm).
- Minimum spanning trees (the algorithms of Prims and Kruskals).
- Topological sort.

Basic computability

- Finite state machines.
- Context free grammars.
- Solvable and inconsistent problems.
- Computable and not computable functions.
- The Halting-problem.
- Consequences of undecidability.

P versus NP.

- Definition of the classes P and NP.
- NP-completeness (Cooks theorem).
- The most common NP-complete problems.
- Reduction techniques.

Course literature

Algorithms, 4th Edition, 2011. Robert Sedgewick and Kevin Wayne. Addison-Wesley. ISBN-10: 032157351X.

Examination

- ARBA - Course work, 4.5 credits, grading scale: P, F
- TENA - Written exam, 3.0 credits, grading scale: A, B, C, D, E, FX, 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.

The expected learning outcomes that have been defined above are examined both through continuous examination and through a written examination. The entire work of the students is individually.

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.