

CS-250

**Algorithms**

Svensson Ola Nils Anders

<b>Cursus</b>	<b>Sem.</b>	<b>Type</b>
Communication systems minor	H	Opt.
Communication systems	BA3	Obl.
Computational science and Engineering	MA1, MA3	Opt.
Computer science minor	H	Opt.
Computer science	BA3	Obl.
Cyber security minor	H	Opt.
Data science minor	H	Opt.
HES - IN	H	Obl.
HES -SC	H	Obl.
Mathematics	BA5	Opt.

Language of teaching	English
Credits	6
Session	Winter
Semester	Fall
Exam	Written
Workload	180h
Weeks	14
<b>Hours</b>	<b>6 weekly</b>
Courses	4 weekly
Exercises	2 weekly
<b>Number of positions</b>	

**Summary**

The students learn the theory and practice of basic concepts and techniques in algorithms. The course covers mathematical induction, techniques for analyzing algorithms, elementary data structures, major algorithmic paradigms such as dynamic programming, sorting and searching, and graph algorithms.

**Content****Mathematical Induction**

- Mathematical background, Euler's formula for trees, Schwartz-Zippel lemma.

**Analysis of Algorithms**

- O-notation, time and space complexity, recurrence relations, probabilistic analysis.

**Data structures**

- Arrays, linked lists, trees, heaps, hashing, graphs.

**Design of algorithms by induction**

- Evaluating polynomials, divide-and-conquer algorithms, dynamic programming.

**Greedy Algorithms**

- Spanning tree and shortest path algorithms

**Sorting and searching**

- Merge sort, bucket sort, quicksort, heapsort, binary search.

**Graphs algorithms and data structures**

- Graphs traversals, shortest paths, spanning trees, transitive closure, decompositions, matching, network flows.

**Complexity**

- Polynomial reductions, NP-completeness.

**Keywords**

algorithms, data structures, efficiency, problem solving

## Learning Prerequisites

### Recommended courses

Advanced ICC I

## Learning Outcomes

By the end of the course, the student must be able to:

- Illustrate the execution of algorithms on example inputs
- Describe basic data structures such as arrays, lists, stacks, queues, binary search trees, heaps, and hash tables
- Analyze algorithm efficiency
- Compare alternative algorithms and data structures with respect to efficiency
- Choose which algorithm or data structure to use in different scenarios
- Use algorithms and data structures taught in the course on concrete problem instances
- Design new algorithms and data structures based on known methods
- Prove the correctness of an algorithm

## Teaching methods

Ex cathedra lecture, exercises in classroom

## Assessment methods

- Programming exercises (15%)
- Theoretical homework (15%)
- Final exam (70%)

## Resources

### Bibliography

Thomas Cormen, Charles Leiserson, Ronald Rivest, Clifford Stein: *Introduction to algorithms*, Third Edition, MIT Press, 2009.

## Ressources en bibliothèque

- [Introduction to algorithms / Cormen](#)