## 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
Continuous assessment with final exam.

Resources
Bibliography

Ressources en bibliothèque
• *Introduction to algorithms* / Cormen