

Content

| Course Code | Course Name | Semester | Theory | Practice | Lab | Credit | ECTS |
|-------------|---------------------------------------|----------|--------|----------|-----|--------|------|
| MAT231 | Algorithms and Advanced Programming I | 3 | 1 | 1 | 1 | 3 | 3 |

| | |
|------------------------|--|
| Prerequisites | |
| Admission Requirements | |

| | |
|-------------------------|---|
| Language of Instruction | French |
| Course Type | Compulsory |
| Course Level | Bachelor Degree |
| Objective | The purpose of this course is to improve students programming capabilities by the study of some common algorithms, their implementations and their applications to sample computational problems. |
| Content | Programming review (with Python): variables and state, conditionals, loops, functions Basic data structures: list, multi dimensional array, tree Common algorithms: search, sort, aggregate functions Recursion: numeric computation, tree traversal (inorder/preorder/postorder) Algorithm Analysis: time/space complexity classes |
| References | The Art of Computer Programming - Donald Knuth Python - How to Program - Deitel Data Structures and Algorithms Using Python - Rance D. Necaise Data Structures and Algorithms with Object-Oriented Design Patterns in Python - Bruno R.Preiss |

Theory Topics

| Week | Weekly Contents |
|------|--|
| 1 | Programming review: value, expression, variable, data type, assignment, program state, enumerating loops |
| 2 | Programming review: conditionals, execution branching, conditional loops, nested loops and conditionals |
| 3 | Programming review: functions, parameters, return value, code flow, stack frames, variable scope |
| 4 | Sequences, patterns, multidimensional patterns from loop indices, data dependence |
| 5 | Implementing aggregate functions: min, max, sum, count, avg, std.dev, unique |
| 6 | Sorting values on a list: naive approach, insertion sort, bubble sort, merge sort |
| 7 | Midterm I |
| 8 | Recursion: depth bounding, flow of function calls, examples: factorial, fibonacci, quick sort |
| 9 | Trees: depth first, breadth first traversal, in-order/pre-order/post-order traversal |
| 10 | Stack, Queue, relation of stack with recursion, recursion removal |
| 11 | Midterm II |
| 12 | Numerical algorithms: random number generation, root finding, linear regression |
| 13 | Search: simple search, binary search, searching recursively |
| 14 | Time/space complexity, Complexity classes, comparison of algorithms |