HOGESCHOOL ROTTERDAM / CMI

Algorithms

INFDEV01-6A

Number of study points:	4 ects
Course owners:	G. Costantini, F. Di Giacomo

Modulebeschrijving

Module name:	Algorithms
Module code:	INFDEV01-6A
Study points and hours of effort for full-time students:	This module gives 4 ects, in correspondance with 112 hours: • 2 x 8 hours frontal lecture • 3 x 8 hours self-study for the theory • the rest is self-study for the practicum
Examination:	Written examination and practical assignment (with oral check)
Course structure:	Lectures
Prerequisite knowledge:	Object oriented programming
Learning tools:	• Book: Algorithms (4rd edition); author R. Sedgewick • Presentations (in pdf): found on N@tschool • Assignments, to be done at home (pdf): found on N@tschool
Connected to competences:	• Realisation
Learning objectives:	At the end of the course, the student can: • to do • to do • to do • to do

Content:	• Performance analysis • Basic data structures • Sorting algorithms • Advanced data structures (trees, graphs) • Path algorithms • Dynamic programming
Course owners:	G. Costantini, F. Di Giacomo
Date:	1 november 2015

1 General description

In this course basic algorithms and data structures are presented.

1.1 Relationship with other teaching units

This course builds upon the development courses of the first year.

Advanced data structures presented in this course are used in RDBMS and informally presented during INFDEV03-5.

Knowledge acquired through the algorithms course is also useful for some of the projects. A word of warning though: projects and development courses are largely independent, so some things that a student learns during the development courses are not used in the projects, some things that a student learns during the development courses are indeed used in the projects, but some things done in the projects are learned within the context of the project and not within the development courses.

2 Course program

Lesson	Topic	Book paragraphs
1	Arrays; Complexity of algorithms (empirical analysis, O notation)	1.4
2	Sorting algorithms (insertion sort, mergesort)	2.2?
3	List, Queue, Stack, Bag; Hash tables	1.3, 3.4?
4	Binary search trees	3.2
5	Balanced search trees: 2-3 search trees	3.3
6	Graphs (undirected, directed, Dijkstra shortest path)	4.1, 4.2, 4.4
7	Dynamic programming; Floyd-Warshall	
8	Recap of the course	

3 Assessment

The course is tested with two exams: a series of practical assignments (accompanied by a brief oral check), and a written exam. The final grade is determined by the practical assignments. However, to receive the grade in the practical assignment you **must** have a sufficient (i.e. ≥ 5.5) grade in the written exam.

3.1 Theoretical examination

The general shape of a theoretical exam for the course is made up of ...? The exam lasts two lesson hours (100 minutes). No help is allowed during the exam.

3.2 Practical examination

The practical assignment must be done individually.
You must upload your projects on Github and (only at the end) on N@tschool.
The intermediate deadlines will be checked through the commits in Github.
There will be oral checks to verify the authorship of code.
The framework for the assignment comes only for .NET languages: allowed languages are C# and F#.
A detailed description of the practical assignment will be uploaded on N@tschool.

3.3 Herkansing

If one part of the assessment is not sufficient (exam and/or practical assignment), then you can repeat that part in the following block:

- In week 10 of the following block you can repeat the written exam.
- The deadline for the practical assignment is the end of week 9 of the following block.

Bijlage 1: Toetsmatrijs

Learning goals	Dublin descriptors
RDBMS	1, 2, 3, 4, 5
ORM	1, 2, 4, 5
OPT	1, 2, 4, 5
NON-REL	1, 5
TRANS-CONS	1, 4

Dublin-descriptors:

1. Knowledge and understanding
2. Applying knowledge and understanding
3. Making judgements
4. Communication
5. Learning skills