COURSE PRESENTATION FORM – ACADEMIC YEAR 2010/2011

COURSE NAME	**Data Structures and Algorithms**
COURSE CODE	70147 (BSc) / 70010 (BSc Old)
LECTURER	Roberto Sebastiani
TEACHING ASSISTANT	Gennaro Iaccarino (ITA) Roberto Sebastiani (EN) TBD (GE)
TEACHING LANGUAGE	English
CREDIT POINTS	8
LECTURE HOURS	48
EXERCISE HOURS	24
TIME SPAN	21.02.2011 - 11.06.2011
TIME TABLE	See Timetable Page
OFFICE HOURS LECTURER	During the lecture time span: by previous appointment only, Tuesdays 9.30-10.30 and 12.30 13.30, Wednesdays 9.30-10.30, Faculty of CS, POS Building, piazza Domenicani 3, office 2.10
OFFICE HOURS TEACHING ASSISTANT	TBD
PREREQUISITES	Basic programming skills and mathematical knowledge
OBJECTIVES	Acquire an in-depth understanding of a broad set of computer science techniques to solve algorithmic problems. These techniques are fundamental building blocks that re-occur in many areas of computer science. Become familiar with the design and analysis of divide and conquer algorithms. Learn how to be precise and mathematically rigorous when designing and verifying algorithms.
SYLLABUS	(the following topics will not necessarily be taught in this order) • algorithmic problems, recursion, sorting • algorithmic complexity, correctness of algorithms • divide-and-conquer, recurrences • heaps, heapsort, quicksort • pointers, abstract data types • binary search trees, red-black trees • hashing • dynamic programming

- DFS, DFS, topological sort
- minimum spanning trees, shortest path
- tractable and intractable problems, NP completeness

TEACHING FORMAT Frontal lectures and lab classes

ASSESSMENT Written exam (100%)

READING LIST Textbook:

- *Introduction to Algorithms*, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein (CLRS), 2nd or 3rd edition

There is a set of lecture notes for each week. It is recommended that you download and print the lecture notes immediately before the respective lectures.

SOFTWARE USED Java

LEARNING OUTCOME A toolbox with the most important techniques for solving algorithmic problems. Ability to use a systematic, precise, and mathematically rigorous approach to design and reason about algorithms. Basic understanding of algorithmic complexity.

COURSE PAGE http://disi.unitn.it/~rseba/DIDATTICA/dsa2011_BZ/