COM2001: Advanced Programming Techniques (Second Semester)

Guide to Examinable Material

Lecture	**Concepts, Knowledge and Skills**
1	• How to write evaluation traces for recursive functions
2	• How to work with asymptotic complexity notations • Definition of worst-case vs average-case complexity
3	• How to design divide-and-conquer algorithms • How to use recurrence trees
4	• How to solve arithmetic series • How to solve geometric series • How to analyse algorithms with the Master Theorem
5	• Definition of defined and undefined values • How to use proof by induction (standard) • How to use proof by induction (structural)
6	• How to use proof by induction (strong) • How to use proof by induction in imperative languages
7	• How to work with axiomatic specifications of abstract data types • How to implement abstract data types in Haskell
8	• How to show completeness of abstract data type specifications
9-10	• Definition of problems efficiently solvable by dynamic programming • How to use subproblem graphs • How to efficiently solve a problem with dynamic programming

Guidance:

- 'Definition' means be able to reproduce and apply the definition in a question.
- 'How to' means be able to apply a procedure in answering a question.

Highest marks on questions will be reserved for *creative* applications of results and definitions covered during the course, to solve previously unseen problems.