Course Organization

Course Learning Outcomes

1. Implement basic data structures such as arrays, linked lists, binary trees and hash tables in a language of choice.

2. Use recursion as an algorithm design technique in a piece of software.

3. Analyse and implement a range of algorithms, including searching and sorting algorithms, and express their complexity using suitable notation.

4. Use predicate logic, set theory and relational algebra to write assertions and use laws to prove them.

5. Demonstrate an understanding of inductively generated structures and proof by induction.

6. Deploy basic concepts of combinatorics in solving mathematical problems.

Course Outline

0	*Levelling up – A view at background knowledge and skills*
1	**Introduction to the course** **Warming up (Solving simple problems using algorithms and data structures)**
2	**Anaysing algorithms**
3	**Data Collections**
4	**Sequences, Induction, Recurrence**
5	**Searching and Sorting: Exploring Algorithmic Design Patterns**
6	**Graphs**
7	**Counting and Probability**
8	**Formal Reasoning**

The learning plan

There will be considerable focus on *problem-based learning*

- Well known pedagogy, more suitable for GAs? (that's the idea at least!)
- There *will* (alas!) be "direct instruction" as well
  - "problem-based learning" does not really work well all on its own
  - required background knowledge and scaffolding will be interleaved
  - some units will have relatively more direct instruction than others

The scheduled class times will be divided into two types of sessions

- Lecture + problem-solving (most days)
- "Get-a-tick" days (mostly Wednesdays, focussed almost entirely on problem-solving)

- Problem-solving: Both paper and computer based
  - → **please bring both pen/papers + laptops to ALL live sessions**

Assessments

Assessment	Due Date	Weight
Workshop participation	Continuous	5%
Class tests (5 x 1%)	Continuous	5%
Assessed Exercise 1	25 Nov	10%
Assessed Exercise 2	12 Dec	10%
Assessed Exercise 3	20 Dec	10%
Final Exam	Early Feb	60%

Feedback & Course changes

- You will get **feedback** in the following ways:
    - 1-1 and group feedback from course instructors and tutors
        - >50% contact hours dedicated to tutorial/lab style interaction
    - Peer feedback during tutorials
    - Individual coursework grade with detailed breakdown of marks, and comments where appropriate
    - Overall, general commentary on courseworks
    - Automated feedback in class tests
    - Grade feedback on exam
    - Overal, general feedback on exam
    - Office hours (by request)
- **Course changes**
    - Project-based assessment “added” (brought back)

Pair up!

I’d like you all to pair up for the problem-solving sessions.

Everyone should be trying out *most* problems on their own to begin with, specially the smaller ones, but:

it is highly encouraged that you: **think → pair → share**

We may also try some **pair programming** for some of the larger problems

Occasionally, I might ask 2 pairs to team up into groups of 4

Learning Resources

Moodle Books!

- Notes and narrative in moodle book chapters
  - Let’s have a quick look
- Live lectures and slides
  - Recording will be made available
- Links to other (online) texts, videos
- Not the least: Problem-solving!
  - **Some problem-solving chapters will have additional notes interleaved with the problems.**
- Unless sign-posted “deep dive” (or an obvious dad-joke), everything on moodle books is part of the learning outcomes and “examinable”