# Course Outline

# COMP1927
# Computing 2

### Teaching Period 2 2012

## Contents


- Course Staff
- Course Details
- Course Aims
- Student Learning Outcomes
- Assumed Knowledge
- Teaching Rationale
- Assessment
- Academic Honesty and Plagiarism
- Preliminary Course Schedule
- Resources for Students
- Course Evaluation and Development
- Credit for the materials


## Course Staff

**Contact Details**

| Staff Name | Role | Email | Phone |
|---|---|---|---|
| Mei Cheng Whale | Course Admininstrator | meicheng at cse.unsw.edu.au | 55683 |
| Gabi Keller | Course Convenor | keller at cse.unsw.edu.au | 56032 |
| Claude Sammut | Lecturer | claude at cse.unsw.edu.au | 56932 |

## Course Details

**Course Code:** COMP1927
**Course Title:** Computing 2
**Units of Credit:** 6
**Course WebSite:** http://www.cse.unsw.edu.au/~cs1927/
**Handbook Entry:** http://www.handbook.unsw.edu.au/undergraduate/courses/2012/COMP1927.html

## Course Aims

In this course we will introduce fundamental data structures and algorithms that are essential to understanding and developing modern software. We will focus on their design and methods for testing and analysing them. Our analysis will enable us to compare alternative data structures and algorithms for particular problem cases. We will also cover additional aspects of the C programming language that were not covered in its predecessor course, COMP1927. However, this course is not simply a second C programming course. The focus is on the ideas and abstractions behind the data structures and algorithms that are used.

## Student Learning Outcomes

By the end of the course, you should:

- be familiar with fundamental data structures and algorithms
- be able to choose/develop an appropriate data structure for a given problem
- be able to choose/develop appropriate algorithms to manipulate this data structure
- be able to reason about the effectiveness of data structures and algorithms for solving a given problem
- be able to analyse the performance characteristics of algorithms
- be able to package a set of data structures and algorithms as an abstract data type

This course contributes to the development of the following graduate attributes:

| Graduate Capability | Acquired in |
|---|---|
| scholarship: understanding of their discipline in its interdisciplinary context | lecture, assignments |
| scholarship: capable of independent and collaborative enquiry | lab work, assignments |
| scholarship: rigorous in their analysis, critique, and reflection | lab work, assignments, tutorials |

| | |
|---|---|
| scholarship: able to apply their knowledge and skills to solving problems | lab work, tutorials, assignments |
| scholarship: ethical practitioners | lecture, tutorials |
| scholarship: capable of effective communication | lab work, tutorials, lectures |
| scholarship: information literate | ? |
| scholarship: digitally literate | lab work, assignments |
| leadership: enterprising, innovative and creative | assignments, tutorials |
| leadership: capable of initiating as well as embracing change | assignments |
| leadership: collaborative team workers | lab work, assignments |
| professionalism: capable of independent, self-directed practice | lab work, assignments |
| professionalism: capable of lifelong learning | lecture, lab work, assignment, tutorials |
| professionalism: capable of operating within an agreed Code of Practice | lecture, assignments |
| global citizens: capable of applying their discipline in local, national and international contexts | assignments |

# Assumed Knowledge

The official pre-requisite for this course is either COMP1917 or COMP1921.

Whether or not you satisfy the pre-requisite:

- we expect that you can program the C programming language: be familiar with arrays, strings, pointers, dynamic memory allocation, recursion, abstract data types, linked lists and trees
  - see also https://wiki.cse.unsw.edu.au/info/CoreCourses/Syllabus
- be able to design, implement, debug, test and document small programs (up to several hundred lines of code).

we expect that you are familiar with the Linux environment on CSE computers

# Teaching Rationale

The course consists of four main components:

- **Lectures:** Introduction of concepts, techniques and problem solving methods.
- **Tutorials:** Clarification and in depth discussion of the material introduced in the lecture.
- **Labs:** Learning how to apply the methods and techniques discussed in the lecture to small to middle sized problems.
- **Assignments:** Learning how to apply the methods and techniques in a wider context.

# Assessment

## Tutes

Tute attendance is compulsory. You will receive tute marks for attending and participating in the tute. The tute mark will be part of your final mark (see table for details). However, the purpose of this mark is not to assess you, but to provide a motivation to prepare for and actively participate in the tutorials, as we found that this significantly improves the learning outcome.

## Labs

You will receive lab marks for successfully completing the lab exercises. As with the tute mark, the lab mark will be part of your final mark, but it's main purpose is not assessment, but to make sure you constantly practice. There will be no late submissions.

## Assignments

The assignments are significant programming exercises and can either be done individually or in teams of two. There will be sufficient time to complete the assignments. You should start working on the assignment as soon as you can to allow for possible sick days and similar. Extensions will only be possible in exceptional circumstances, when you were not able to work on the assignment for a substantial part of the time allowed for the assignment. Late submissions are possible for up to three days after the deadlines, with penalties of 10% of the maximum mark deducted per day. Assignments are the primary vehicle for learning the material in this course. If you don't do them, or simply copy and submit someone else's work, you have wasted a valuable learning opportunity, and you take the risk of being caught for plagiarism. If you are caught, you will be awarded an overall mark of zero for the assignment, and possibly the course, and receive an entry in the school plagiarism register. This does not look good on your academic record. However, you should be aware that re-use of algorithms and data structures developed outside this course is allowed. You should not re-use code however, but if you feel it is justified, you should give credit to the original author or source. Assignment related questions may be part of the exam

### Exam

There will be a four-hour closed-book exam for this course. The exam will consist of a practical (programming) part and a theory part. The final exam contributes 60% of the final mark for this course. There will be no second-chances to pass the final exam. Attendance at the exam will be treated as an indication by you that you are feeling well enough to sit that exam. If you are feeling so unwell on the day of the exam that you cannot attend, go to the doctor and get a medical certificate. If you put in a request for supplementary assessment after already attending the original exam, your request will not be granted. Also, if your special consideration request does not provide documented evidence that you personally have been severely affected by some unforeseen situation, then your request will not be granted. The supplementary exam may be a written or an oral exam. Note that supplementary exams are typically more difficult than the original exam to reflect the fact that you have had extra preparation time.

### Summary

| Component | Percentage of overall mark |
|---|---|
| Tutorial mark | 10% |
| Lab mark | 10% |
| Assignment 1 | 10% |
| Assignment 2 | 10% |
| Exam | 60% |

## Academic Honesty and Plagiarism

**Plagiarism** is defined as "using the words or ideas of others and presenting them as your own". UNSW and CSE treat plagiarism as academic misconduct, which means that it carries penalties as severe as being excluded from further study at UNSW. There are several on-line sources to help you understand what plagiarism is and how it is dealt with at UNSW:

- Learning Centre: Plagiarism and Academic Integrity
- MyUNSW: Plagiarism and Academic Misconduct
- CSE: Addendum to UNSW Plagiarism Guidelines
- CSE: Yellow Form (whose terms you have agreed to)

Make sure that you read and understand these. Ignorance is not accepted as an excuse for plagiarism.

## Preliminary Course Schedule

| Week | Topic | Book Chapter |
|---|---|---|
| 1 | Introduction, Linked lists | Chapter 3 |
| 2 | Abstract Data Types | Chapter 4 |
| 3 | Recursion and Trees | Chapter 5 |
| 4 | Work Complexity and Trees | Chapter 5 |
| 5 | Sorting | Chapter 6.1 - 6.5, Chapter 7 & 8 |
| 6 | Symbol Tables, Binary Search Trees | Chapter 12 |
| 7 | Balanced Trees | Chapter 13 |
| 8 | Binary Search Tables | Chapter 13 |
| 9 | Hash Tables, Graph Intro | Chapter 14, Chapter 17 |
| 10 | Graph Search | Chapter 18 |
| 11 | DAGs, Minimum Spanning Trees, Shortest Path | Chapter 21 |
| 12 | Revision | |

## Resources for Students

A list of topics and references to the course textbooks will be given on the course web site. The course textbooks are:

- **Algorithms in C, Parts 1-4: Fundamentals, Data Structures, Sorting, Searching (3rd Edition)**
- **Algorithms in C, Part 5: Graph Algorithms (3rd Edition)**

both by Robert Sedgewick, Addison Wesley

These two books are available as a bundle from the UNSW bookshop. They are expensive, but have relevance well beyond this course, and will serve as a useful reference on the bookshelf of any serious programmer.

The following books might also serve as additional reference material on the C programming language:

- Brian W. Kernighan and Dennis M. Ritchie: **The C Programming Language**, 2nd edition, Prentice Hall, USA, 1988, ISBN 0-13-110370-9.
- Paul Davies, **The Indispensable Guide to C**, Addison Wesley, England, 1995, ISBN 0-201-62438-9.

## Course Evaluation and Development

This course is being continuously improved and we will conduct a survey at the end of session to obtain feedback on the quality of the various course components. Your participation in the survey will be greatly appreciated. Also, please feel free to contact your tutor or lecturer with any feedback or suggestions for improvement during the session.

## Credit for the materials

Slides, notes, and other material are partially from previous lecturers of this course, including Angela Margaret Finlayson, John Shepherd, Manuel Chakravarty, Richard Buckland, Albert Nymeyer and Maurice Pagnucco. Some slides and programs are adapted from the teaching material provided with Robert Sedgewick's books.