Introductory information about
COMP20003 Algorithms and Data Structures
Second (Spring) Semester, 2016

Nir Lipovetzky
July 28, 2016

Staff

Lecturer

Dr Nir Lipovetzky
people.eng.unimelb.edu.au/nlipovetzky
Email: nir.lipovetzky@unimelb.edu.au
6.17 Doug McDonell Building

Head Tutor

Grady Fitzpatrick
Email: g.fitzpatrick@student.unimelb.edu.au

Students and Prerequisits

This subject is an offering in the BSc(Computing and Software Systems) degree and is open to other students who have the appropriate prerequisites.

As stated in the University Handbook, prerequisites include 25 points (two subjects) of university level mathematics and a passing grade in either Engineering Computation COMP20005 or Foundations of Algorithms COMP10002.

Note that the subject is particularly designed for students who are coming in from Engineering Computation COMP20005. Students who have taken Foundations of Algorithms COMP10002 may find that there is some overlap with material they have previously covered, during the first 3–4 weeks.

Subject Description

This subject covers algorithms and data structures that are basic in computer science. Lecture coverage includes theoretical complexity analysis for all algorithms covered and implementation details.

Algorithms covered focus on graph algorithms, but include selected searching and sorting algorithms as well.

An important part of the subject is developing facility with C programming. The programming aspect of the subject will be covered largely in the workshops, in which programming exercises will be undertaken in a supported atmosphere, and in individual assignments in which algorithms will be implemented.

Aims

The aims of this subject are for students to:

- become familiar with the library of algorithms that are basic tools in computer science;
- acquire the skill to be able to calculate worst case time and space complexity for any algorithm;
- be able to use their knowledge of algorithms and complexity analysis to critically analyze a problem and decide on the most appropriate algorithm for the task;
- develop the programming facility needed to implement basic algorithms in the C programming language.

Contacting us

General enquiries about the subject should be submitted to the *discussion forum* on the LMS. If you have a personal or confidential concern, please contact your lecturer directly. If you contact the staff directly and we feel that your message is better sent to the forum, we reserve the right to answer your message merely by asking you to post it to the forum.

Questions about the workshop content should, in the first instance, be directed to your tutor during the workshop time.

For questions of a general nature that arise outside of the workshop time, and where the topic may be of interest to other students in the subject, please use the appropriate LMS Discussion Forum.

The lecturer is available to answer questions at the end of every lecture. Other consultations will be held by appointment only.

Seeking assistance

There are a number of mechanisms available for assistance detailed on LMS. You should also feel free to approach your lecturer. He is the final arbiter of all that happens in the subject, and if he cannot answer your question, it may be that no one can. Again, it may be necessary to approach the lecturer briefly in the first instance to set a time for a consultation. Immediately after lectures is also a reasonable time to ask quick questions. Finally, note that you may also contact the staff using the email addresses above if you need to arrange an appointment.

Use of email

Remember that the LMS forum is your first port of call when contacting the staff. Whenever you send email from outside the university account system, please be sure to identify who you are by full name and student number, and make sure the subject line is *20003 query*: we employ a strict spam filter, and mail from `bigbadbob@hotmail.com` on the subject *Need help* may well get deleted without being read.

Handbook

The University Handbook entry for this subject, available on the web and on the subject LMS site, contains further information about the subject, including generic skills, relevant to areas outside of algorithms and data structures.

Textbook

The *prescribed* textbook for this subject is: S. Skiena, *The Algorithm Design Manual*, 2nd Edition, Springer. This text is available as an e-book in the University of Melbourne library.

The copyright conditions allow storage and printing of the e-book for purposes of private study.

The following books might also be useful, and are held in the ERC library on overnight and 7-day loan.

The overnight copies are held in the High Use area.

- R. Sedgewick, *Algorithms in C, volume 1*, 3rd edition, Addison-Wesley, 2002.
- R. Sedgewick, *Algorithms in C Part 5: Graph Algorithms*, 3rd edition, Addison-Wesley, 2002.
- T.H. Cormen, C.E. Leiserson, and R.L. Rivest, *Introduction to Algorithms*, 3rd edition, MIT Press, 2009.

Lectures

Lecture slides and notes (generated from the same source, and more suitable for printing), will be provided in advance of the the lecture, on the LMS. However, these advance versions should be considered drafts. The lecturer may edit the slides and notes after they are delivered, somewhat like the *Hansard* tradition.

Lectures will be recorded using the Lectopia facility; these recordings will be available to students via the LMS. Please note, however, that this is not an on-line subject. Whiteboard work and in-class exercises will not be recorded, and Lectopia occasionally has glitches in recording.

Syllabus/Schedule

Workshops will generally follow the lecture schedule. Significant deviations will be announced.

Note that this is an approximate schedule, and may be subject to change during the semester.

Week 1 Algorithms: Introduction

Week 2 Complexity analysis; big-O notation; approaches to algorithms

Week 3 Dynamic memory allocation; linked lists; Binary search trees

Week 4 Balanced Trees; Makefiles; Programming Tools

Week 5 Distribution Counting; Hashing

Week 6 Divide and Conquer Algorithms; Mergesort; Quicksort; Master Theorem

Week 7 Graphs and Graph Representations; Mid-Semester Test

Week 8 Tree and Graph Traversal; Priority Queues

Week 9 Graphs Path Problems; P- and NP-problems

Week 10 Minimum Spanning Tree

Week 11 More Graph Algorithms

Week 12 Even more graph algorithms; Review

Assessment

The weighting of assessment components is:

- Projects (30%), spread throughout the semester.
- Mid-semester test (10%)
- End of semester examination (60%).

In order to pass the subject, students must achieve at least 15/30 on the combined marks for the practical work and at least 35/70 on the combined mark for the mid-semester test and the final examination.

Projects

Projects will consist of two programming assignments. These assignments are to be completed *individually*. The programming language is C. At submission time, all submitted programs must compile and run on the School of Engineering computers. Further information will be disseminated at the time the assignments are released.

Late submissions Project work is due at a time that will be made explicit on the project specification. The late penalty is 20% of the available marks for that project for each day (or part thereof) overdue. Requests for extensions on medical grounds will need to be supported by a medical certificate. In general, extensions will not be granted if the interruption covers less than 10% of the project duration. Remember that departmental servers are often heavily loaded near project deadlines, and unexpected outages can occur; in general, these will not be considered as grounds for an extension. Students who experience difficulties due to personal circumstances are encouraged to make use of the appropriate University student support services, and to contact the lecturer, at the earliest opportunity. No student will have more than one submission marked for each project. If you make both on-time and late submissions, please see the lecturer as soon as possible to determine which submission will be assessed.

Test

There will be a *thirty-minute* mid-semester test worth 10% of your final mark. The *intention* is that this test will be held during the lecture: the date will be confirmed at an appropriate time during the semester. Details of the mid-semester test will be made available closer to the date of the test.

Examination

The final examination is a three-hour examination. Unlike previous offerings of this subject, the examination will be wholly paper-based, i.e. *will not* include programming on a computer.

Summary

All marks are provisional until confirmed by student administration through the issue of final results.

Task	%	Note
Projects	30	Must obtain $\geq$15/30 to pass subject
Test	10	Must obtain $\geq$35/70 to pass subject
Examination	60	