Cosc 241
Programming and Problem Solving
Lecture 1 (24/2/2020)

Michael Albert
michael.albert@cs.otago.ac.nz

UNIVERSITY of OTAGO
Te Whare Wānanga o Otāgo
NEW ZEALAND

Who (and where) we are

Lecturer, Paper Coordinator

Michael Albert

Owheo G.31

michael.albert@cs.otago.ac.nz

Lecturer

Lech Szymanski

Owheo 2.49

lechszym@cs.otago.ac.nz

Lab Coordinator

Iain Hewson

Owheo G.37A

ihewson@cs.otago.ac.nz

Assessment

Internal

- One assignment
  - due end of week 10 (14%)
- Two practical tests (in labs)
  - week 6 (5%)
  - week 12 (5%)
- Assessed weekly labs
  - Weeks 1, 2, 3, 4, 5, 7, 9, 11 (2% each)

External

- Three hour final exam (60%)

Detailed instructions for the assessed labs (including submission method and deadlines) will be available in the lab.

Workload expectations

- The generic equivalence for workload in an Otago paper is that “1 point = 10 hours”.
- This is an 18 point paper.
- Allowing a generous 50 hours for final exam preparation that means that in an average week during semester you should be spending 10 hours on COSC241.
- Lectures are two hours, labs are four hours, so that still leaves four hours for review, preparation, extra lab work etc.
- Obviously, the load is not completely uniform and you will be spending a bit more time in weeks leading up to the practical tests and assignments, and a bit less in others.

Academic integrity

- For COSC241 and indeed most COSC papers, the most basic principle of academic integrity is write your own code and do not share it.
- We encourage you to discuss your work with your peers, ask questions, and help out where you can but just don't ever give anyone a copy of code that you've written (at least not until after the due date!)
- Submissions will be checked for similarity, and we have unfortunately needed to pursue cases of academic misconduct in the past. You really don't want to get tied up in that procedure (and neither do I).
- Please read the material on academic integrity that is linked through the paper's web pages.
- If you're at all in doubt about whether something is acceptable – just ask.

The textbook

- There isn't one, though the COMP160 text ("Java Foundations" by DePasquale and Chase) is useful as a reference for basic Java concepts.
- We will provide specific links where appropriate to online materials.
- Additionally, there are a number of free online texts that cover the same (or similar) material to COSC241. Links to some of these are included in the course webpage.
- Google is your friend.

Lecture notes

- We are attempting to be as paperless as possible – all information is provided via email or the course webpage (link below).
- No lecture handouts will be produced.
- Lecture slides will be available on the course webpage by Thursday p.m. at the latest for the following week.
- Also available there will be any relevant supplementary material (including code samples from code discussed during lectures) and all the information that would normally be included in a course outline.

www.cs.otago.ac.nz/cosc241

Keep in touch!

If problems arise or something is going particularly well then we want to hear about it as quickly as possible.

- Regarding lab work or internal assessment get in touch with Iain.
- Regarding lecture materials get in touch with either Michael or Lech.
- If you wish to remain anonymous then contact a class representative who will forward the message (contact details for class representatives will be on the paper's web page).

If you have a more general issue to raise then the department has an email alias feedback@cs.otago.ac.nz for that purpose.

Learning objectives

When you successfully complete this paper, you will be able to demonstrate:

- an understanding of the nature of algorithms and how to analyse their efficiency,
- an understanding of random number generators and how to make use of them,
- an appreciation for abstract data types (ADTs) and a knowledge of the ADTs most commonly used in software development (e.g., stacks, queues, lists),
- an understanding of some of the most common data structures used to represent ADTs (e.g., arrays, linked lists, heaps), and the algorithms that operate on them,
- a detailed knowledge of various types of sorting algorithms and their characteristics, and
- an increased proficiency in Java programming.

Topics

- Review of Java programming
- Recursion
- Analysis of algorithms and programs
- Array algorithms
- Random number generators and their application
- Simple ADTs including stacks, lists, and queues
- Basic sorting algorithms (selection and insertion sort)
- Quicksort and merge sort
- Heaps, priority queues and heapsort

Basic Java review

- Java programs consist of class definitions.
- Classes are templates for data types, defining the structure of each object in the class (data fields) and the operations defined on them (methods).
- An application class contains a main method, and directs the execution of the program.
- In 241 we are concentrating on data structures and algorithms, so we limit ourselves for the most part to simple text based input/output.

Example

- Point and Circle, two support classes which could be part of a ‘virtual geometry’ package forming the backbone of an application such as Geogebra which allows the manipulation of geometric objects.
- Data fields: visibility, static (instance or class variables).
- Constructors: defaults, use of this.
- Methods: visibility, return types, length, don’t duplicate.