MONASH University
Information Technology

FIT1008
Introduction to computer science

Unit Guide

Semester 1, 2011

The information contained in this unit guide is correct at time of publication. The University has the right to change any of the elements contained in this document at any time.

Last updated: 27 Feb 2011

Table of Contents

FIT1008 Introduction to computer science - Semester 1, 2011

This unit introduces students to core problem-solving, analytical skills, and methodologies useful for developing flexible, robust, and maintainable software. In doing this it covers a range of conceptual levels, from high level algorithms and data-structures, down to abstract machine models and simple assembly language programming. Topics include data structures; algorithms; object-oriented design and programming; and abstract machines.

Mode of Delivery

- Clayton (Day)
- Sunway (Day)

Contact Hours

3 hrs lectures/wk, 3 hrs laboratories/wk, 1 hr tutorial/wk

Workload

For on campus students, workload commitments are:

- three one-hour lectures,
- one one-hour tutorial
- one 1 and 1/2 hour computer lab prac (requiring advance preparation) followed by an extra 1 and 1/2 hour for prac marking
- a minimum of 6 hours of personal study per week in order to satisfy the reading and assignment expectations.
- You will need to allocate up to 4 hours per week, for use of a computer, including time for newsgroups/discussion groups.

Unit Relationships

Prohibitions

CSE1303, CSC1030, FIT1007, FIT1015

Prerequisites

FIT1002 or equivalent and FIT1029 or equivalent

Chief Examiner

Maria Garcia de la Banda

Campus Lecturer

Clayton

Maria Garcia de la Banda

Sunway

Choon Ling Tan

Tutors

Clayton

Learning Objectives

At the completion of this unit, students will have

Developed the ability to:

- understand abstract data types and, in particular, data structures for stacks, queues, lists, and trees, as well as their associated algorithms for creating and manipulating them. Evaluate the appropriateness of different data structures for a given problem;
- understand basic searching and sorting algorithms and implement them. Understand the concept of algorithmic complexity. Analyse the complexity of these searching and sorting algorithms as well as other basic algorithms. Compare the complexity of different algorithms for solving a given problem;
- analyse different implementations of abstract data types and determine their implications regarding complexity, functionality, and memory usage;
- understand the uses of recursive algorithms and data structures, their advantages and disadvantages. Analyse the complexity of simple recursive algorithms, and their relationship with iteration. Understand basic recursive algorithms for lists and trees, and develop new ones;
- understand the basic concepts in the object-oriented (OO) programming paradigm;
- understand the basic concepts in testing;
- understand the requirements for "good programming practice";
- understand the different compilation targets, including abstract machine code, assembly language, object code, and machine code. Understand the relationship between simple code in a high level imperative language and its low level translation into assembly code;
- learn the structure and design of a particular processor simulator. Analyse the execution in this simulator of simple iterative algorithms learned before, thus gaining a deeper understanding of the connection between software and hardware, between an algorithm and its execution;
- understand how the simulator implements function calling, and use it to reinforce the connection between recursion and iteration.

Developed attitudes that enable them to:

- conform to programming standards when writing software;
- use good design principles when constructing systems;
- take a patient and thorough approach to testing;
- acknowledge any assistance they have received in writing a program;

- search for information in appropriate places when necessary.

Developed the skills to:

- implement their own data-structures. Design and implement Java programs using a variety of data structures and algorithms;
- implement an object-oriented program consisting of several interacting classes requiring not only basic but also advance object-oriented concepts;
- construct a test harness for testing an object-oriented program;
- debug and modify an existing program (written by somebody else);
- use the Java API classes as part of their programs.

Demonstrated the communication skills necessary to:

- document a program correctly;
- explain how parts of a program work.

Graduate Attributes

Monash prepares its graduates to be:

1. responsible and effective global citizens who:

a. engage in an internationalised world
b. exhibit cross-cultural competence
c. demonstrate ethical values

critical and creative scholars who:

a. produce innovative solutions to problems
b. apply research skills to a range of challenges
c. communicate perceptively and effectively

Assessment Summary

Examination (3 hours): 70%; In-semester assessment: 30%

Assessment Task	Value	Due Date
Mid Semester Test (1 hour)	10%	Monday 18 April 2011
Pracs (1 and 1/2 hours each)	20%	There is a prac every week except when the Mid-Semester Test is held. Each prac will be marked during the hour-and-a-half immediately after the first hour-and-a-half of that prac session. You must remain in the prac session until your prac is marked.
Examination 1	70%	To be advised

Teaching Approach

Lecture and tutorials or problem classes

This teaching and learning approach helps students first encounter the information at lectures, discuss and explore them at length during tutorials, and practice them in a hands-on environment during labs.

Feedback

Our feedback to You

Types of feedback you can expect to receive in this unit are:

- Informal feedback on progress in labs/tutes
- Graded assignments without comments
- Test results and feedback
- Quiz results
- Other: Solutions to tutes

Your feedback to Us

Monash is committed to excellence in education and regularly seeks feedback from students, employers and staff. One of the key formal ways students have to provide feedback is through SETU, Student Evaluation of Teacher and Unit. The University's student evaluation policy requires that every unit is evaluated each year. Students are strongly encouraged to complete the surveys. The feedback is anonymous and provides the Faculty with evidence of aspects that students are satisfied and areas for improvement.

For more information on Monash's educational strategy, and on student evaluations, see:
http://www.monash.edu.au/about/monash-directions/directions.html
http://www.policy.monash.edu/policy-bank/academic/education/quality/student-evaluation-policy.html

Previous Student Evaluations of this unit

If you wish to view how previous students rated this unit, please go to
https://emuapps.monash.edu.au/unitevaluations/index.jsp

Required Resources

Eclipse Platform. One of the two recomended platforms. It can be downloaded from http://www.eclipse.org/downloads/

NetBeans IDE. The other recommended platform. It can be downloaded fro http://netbeans.org/downloads/

Java Development Kit, Version j2sdk-1_5_0_06 or later, Sun Microsystems, Inc. You should download the freeware version. You have no need for the fuller facilities provided in JCreatorPro, and would have to pay for it as well.

The MIPS R2000 simulator **SPIM S20**. This, and all the other above, are included as part of the Standard Operating Environment used in Faculty computer Labs.

Unit Schedule

Week	Date*	Activities	Assessment
0	21/02/11	Register for tutorials, pracs and lectures	No formal assessment or activities are undertaken in week 0
1	28/02/11	Lectures on Revision of BigO and List (arrays). Tute & Prac.	Prac 1
2	07/03/11	Lectures on List Sorting & Other Array Data Structures. Tute & Prac.	Prac 2
3	14/03/11	Lectures on Linked Data Structures: Tute & Prac	Prac 3
4	21/03/11	Lectures on Object Oriented Basics. Tute & Prac.	Prac 4
5	28/03/11	Lectures on Advanced Object Oriented. Tute & Prac.	Prac 5
6	04/04/11	Lectures on Recursion and Recursive Sorts. Tute & Prac.	Prac 6
7	11/04/11	Lectures on Binary Trees and Heaps. Tute & Prac.	Prac 7
8	18/04/11	Lectures on Testing & Debugging. Tute & Mid-Semester Test. No Prac.	Mid Semester Test, 18th April
Mid semester break			
9	02/05/11	Lectures on Hash Tables & Number representation. Tute & Prac.	Prac 8
10	09/05/11	Lectures on Computer Systems & MIPS. Tute & Prac	Prac 9
11	16/05/11	Lectures on MIPS execution of selection and iteration. Tute & Prac.	Prac 10
12	23/05/11	Lectures on MIPS function call/return. Tute & Prac	Prac 11
	30/05/11	SWOT VAC	No formal assessment is undertaken SWOT VAC

*Please note that these dates may only apply to Australian campuses of Monash University. Off-shore students need to check the dates with their unit leader.

Assessment Policy

To pass a unit which includes an examination as part of the assessment a student must obtain:

- 40% or more in the unit's examination, and
- 40% or more in the unit's total non-examination assessment, and
- an overall unit mark of 50% or more.

If a student does not achieve 40% or more in the unit examination or the unit non-examination total assessment, and the total mark for the unit is greater than 50% then a mark of no greater than 49-N will be recorded for the unit

Assessment Tasks

Participation

Students must actively participate in at least 7 out of the 11 tutorials. Participation includes contributing opinions to a discussion, providing an answer to some question/exercise, or posing a unit-related question.

- **Assessment task 1**

  Title:
  Mid Semester Test (1 hour)
  Description:
  This test will evaluate your understanding of the material provided during the first six weeks of semester, your capability to code simple algorithms given a clear specification, and to analyse the behaviour and complexity of simple fragments of code.
  Weighting:
  10%
  Criteria for assessment:
  See previous paragraph (Description).
  Due date:
  Monday 18 April 2011

- **Assessment task 2**

  Title:
  Pracs (1 and 1/2 hours each)
  Description:
  Each week you will need to complete a prac assignment (possibly together with another student). Prac assignments are long and are designed to take a significant part of your 6 "home study hours" (usually, up to 4 hours). This means that you must have a significant proportion of the prac completed before attending the scheduled computer lab. The aim of the 1 and 1/2 hour computer lab practical is to iron out any bugs, ask any questions about the prac you have not been able to solve on your own, and improve the parts that your demonstrator points out as lacking (including comments, algorithms, etc). If you do nothing before the1 and 1/2 hours scheduled, you will soon realise that you do not have enough time to complete it. The prac sheets will be released every Thursday morning and made available in the unit's web page.
  Weighting:
  20%
  Criteria for assessment:
  Every prac sheet contains the assessment criteria used to assess that prac. In addition, demonstrators carry with them a marking guide prepared by the lecturer which indicates how exactly to mark each prac question. You can request the demonstrator to show you the marking guide after he/she has marked your prac.
  Due date:
  There is a prac every week except when the Mid-Semester Test is held. Each prac will be marked during the hour-and-a-half immediately after the first hour-and-a-half of that prac session. You must remain in the prac session until your prac is marked.
  Remarks:
  There are two hurdles associated to the pracs. First, you must attend at least 7 out of the 10 pracs. Second, you must score at least 50% of the prac mark.

As mentioned above, there is also one hurdle associated with tutorials. You must attend at least 7 out of the 11 tutorials.

A student who does not meet all these hurdles can get a maximum of 49-N for the unit.

Examinations

- **Examination 1**

  Weighting:
  70%
  Length:
  3 hours
  Type (open/closed book):
  Closed book
  Electronic devices allowed in the exam:
  None
  Remarks:
  There is a hurdle associated with the exam mark: you must score at least 50% of the exam mark. Furthermore, you must score at least 50% overall (i.e., for the mid semester test, pracs and exam).

Assignment submission

Assignment coversheets are available via "Student Forms" on the Faculty website: http://www.infotech.monash.edu.au/resources/student/forms/
You MUST submit a completed coversheet with all assignments, ensuring that the plagiarism declaration section is signed.

Extensions and penalties

Submission must be made by the due date otherwise penalties will be enforced.

You must negotiate any extensions formally with your campus unit leader via the in-semester special consideration process: http://www.infotech.monash.edu.au/resources/student/equity/special-consideration.html.

Returning assignments

Students can expect assignments to be returned within two weeks of the submission date or after receipt, whichever is later

Policies

Monash has educational policies, procedures and guidelines, which are designed to ensure that staff and students are aware of the University's academic standards, and to provide advice on how they might uphold them. You can find Monash's Education Policies at: http://policy.monash.edu.au/policy-bank/academic/education/index.html

Key educational policies include:

- Plagiarism (http://www.policy.monash.edu/policy-bank/academic/education/conduct/plagiarism-policy.html)
- Assessment (http://www.policy.monash.edu/policy-bank/academic/education/assessment/assessment-in-coursework-po
- Special Consideration (http://www.policy.monash.edu/policy-bank/academic/education/assessment/special-consideration-policy.h
- Grading Scale (http://www.policy.monash.edu/policy-bank/academic/education/assessment/grading-scale-policy.html)
- Discipline: Student Policy (http://www.policy.monash.edu/policy-bank/academic/education/conduct/student-discipline-policy.html)
- Academic Calendar and Semesters (http://www.monash.edu.au/students/key-dates/);
- Orientation and Transition (http://www.infotech.monash.edu.au/resources/student/orientation/); and
- Academic and Administrative Complaints and Grievances Policy (http://www.policy.monash.edu/policy-bank/academic/education/management/complaints-grievance-policy.

Student services

The University provides many different kinds of support services for you. Contact your tutor if you need advice and see the range of services available at www.monash.edu.au/students The Monash University Library provides a range of services and resources that enable you to save time and be more effective in your learning and research. Go to http://www.lib.monash.edu.au or the library tab in my.monash portal for more information. Students who have a disability or medical condition are welcome to contact the Disability Liaison Unit to discuss academic support services. Disability Liaison Officers (DLOs) visit all Victorian campuses on a regular basis

- Website: http://adm.monash.edu/sss/equity-diversity/disability-liaison/index.html;
- Telephone: 03 9905 5704 to book an appointment with a DLO;
- Email: dlu@monash.edu
- Drop In: Equity and Diversity Centre, Level 1 Gallery Building (Building 55), Monash University, Clayton Campus.

READING LIST

(1) *Data Structures and Algorithms in Java*. Robert Lafore, SAMS. This book provides a very simple approach to understanding data structures and algorithms. While the book uses Java to illustrate the implementation, its focus is on the actual data structures and algorithms, rather than on Java, which is very useful for first year students. Very basic and simple.

(2) *Data Structures and Algorithms in Java.* Adam Drozdek, Brooks/Cole. More advanced but still appropriate for average and high-end students.

(3) *Algorithms in Java*. Robert Sedgewick. Parts 1-4. This book is a more in-depth book. It is recommended for advanced students who want to learn more about the complexity of the algorithms and data structures used.

(4) *The Java Programming Language.* Fourth Edition. Ken Arnold, James Gosling and David Holmes. Prentice Hall. This book is useful for students who have questions about the Java language. The reason for using the fourth edition is because it is the earliest edition that includes parametric polymorphism (called Generics in Java).