FIT1008
Computer science

Unit Guide

Semester 1, 2010

Last updated: 12 Feb 2010

Table of Contents

FIT1008 Computer science - Semester 1, 2010

Chief Examiner:

Associate Professor Maria Garcia De La Banda
Head of School
Phone: +61 3 990 31058
Fax: +61 3 990 31077

Lecturer(s) / Leader(s):

Clayton

Associate Professor Graham Farr
Associate Professor
Phone: +61 3 990 55201
Fax: +61 3 990 55159

Additional communication information:

The preferred communication method for questions regarding the unit's material and/or organisation is through the on-line discussion forum (that way, everyone can benefit from it). For more in-depth help, students can either talk to the lecturer during consultation hours, or through the "Help Room" sessions.

If students have a personal matter that they wish to notify the lecturer about, then they should email cl-fit1008-admin@infotech.monash.edu.au.

Notices related to the unit during the semester will be placed on the News of the Unit's Website.Please, check this regularly. Failure to read the Notices newsgroup is not regarded as grounds for special consideration.

IMPORTANT: for any e-mail contact regarding this unit, please make sure your subject line starts with "FIT1008:". Otherwise, the e-mail might go undetected.

Introduction

Welcome to FIT1008 Computer Science. This is a 6 credit point unit core in the Bachelor of Computer Science and the Bachelor of Software Engineering. The unit is designed to develop the student's understanding on how to implement and use the basic data structures and algorithms, and also to explore how simple programs that use these basic components are actually executed by the computer.

Unit synopsis

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.

Learning outcomes

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;
- gain a deeper understanding of basic object-oriented (OO) concepts, and learn more advanced ones such as inheritance, polymorphism, information hiding and encapsulation;
- understand the design principles for building an object-oriented program, such as identify classes, and determine how and when to use inheritance;
- understand what a programming language paradigm is, and learn to distinguish among some of the major paradigms, including imperative, object oriented, functional and logic;
- understand the basic concepts in testing, including execution vs non-execution based testing, glass box and black box testing, correctness proofs, and test case selection;
- understand the requirements for good programming practice;
- understand how numbers are represented on a computer;
- 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 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 the trade-offs regarding simplicity, efficiency and memory usage when designing the architecture of a computer;
- 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:

- create their own data-structures. Design and implement Java programs using a variety of data structures and algorithms;
- implement an object-oriented program consisting of many 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;
- produce appropriate documentation for designing and testing a program;
- explain how parts of a program work.

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

Prerequisites

FIT1002 or equivalent

Prohibitions

CSE1303, CSC1030, FIT1007, FIT1015

Teaching and learning method

Teaching approach

The main teaching mechanisms are through the material covered in lectures, and the questions and discussion promoted during tutorials. Learning is also expected to occur through discussions with the prac partner, interaction with the demonstrator, participation in the on-line discussion forum for the unit and, importantly, through the reading of the book chapters recomended for each topic in the lecture slides.

Timetable information

For information on timetabling for on-campus classes please refer to MUTTS, http://mutts.monash.edu.au/MUTTS/

Tutorial allocation

On-campus students should register for tutorials/laboratories using the Allocate+ system: http://allocate.its.monash.edu.au/

Unit Schedule

Week	Date*	Topic	Key dates
1	01/03/10	BigO. List (arrays): Addition, Deletion, Search	
2	08/03/10	List Sorting & Other Array Data Structures	
3	15/03/10	Linked Data Structures	
4	22/03/10	Object Oriented Basics	
5	29/03/10	Advanced OO	
Mid semester break			
6	12/04/10	Testing/Debugging	Wednesday 14th April
7	19/04/10	Recursion and Recursive Sorts	
8	26/04/10	Binary Trees & Programming Languages	
9	03/05/10	Architecture, number representation	
10	10/05/10	MIPS	
11	17/05/10	Translating to assembler	
12	24/05/10	Function call/return	
13	31/05/10	Revision	

*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.

Improvements to this unit

Most pracs have been slighly shortened.

Unit Resources

Prescribed text(s) and readings

There are no required texts for this subject since there is no single text that contains all the material. Please read the appropriate parts of the recommended texts.

Text books are available from the Monash University Book Shops. Availability from other suppliers cannot be assured. The Bookshop orders texts in specifically for this unit. You are advised to purchase your text book early.

Recommended text(s) and readings

(1) *Data Structures and Algorithms in Java*. Second Edition. 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*. Third Edition. 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) *Absolute Java*.Second Edition. Walter Savitch. Addison Wesley. This book also contains some data structures and algorithms, but it uses them to illustrate the use of Java. It is useful for students who have questions about the Java language.

Required software and/or hardware

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.

Equipment and consumables required or provided

Students may use the facilities available in the computing labs. Information about computer use for students is available from the ITS Student Resource Guide in the Monash University Handbook.

You will need to allocate up to **4** hours per week for use of a computer, including time for newsgroups/discussion groups.

Study resources

Study resources we will provide for your study are:

found at the FIT1008 site, including:

- lecture slides,
- code for the lectures in Java class format,
- weekly tutorial exercises,
- weekly assignment specifications,
- weekly tutorial solutions (available *after* the tutorial), and
- supplementary material.

Assessment

Overview

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

Faculty 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.

In addition, to pass this unit you must:

- attend at least 7 out of the 11 pracs;
- attend at least 7 out of the 11 tutorials;
- score 50% or better in pracs
- score 50% or better in the exam, and
- score at least 50% overall.

If these five hurdles are met, your score for the unit will be calculated by:

0.7(Total Exam Mark) + 0.2*(Total Prac Mark) + 0.1*(Total Test mark)*

Otherwise, the maximum score is 44N

Assignment tasks

Assignment coversheets

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.

Assignment submission and return procedures, and assessment criteria will be specified with each assignment.

- **Assignment task 1**

  Title:
  Mid Semester Test (1 hour)
  Description:
  This test will evaluate your understanding of the material provided during the first five 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%
Due date:
Wednesday 14th April

- **Assignment task 2**

Title:
Pracs (1 and 1/2 hours each)
Description:
Each week you will need to complete a prac assignment 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%
Due date:
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 11 pracs. Second, you must score at least 50% of the prac mark.

There is 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.

Examination

- **Weighting:** 70%
  Length: 3 hours
  Type (open/closed book): Closed book
  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).

See Appendix for End of semester special consideration / deferred exams process.

Due dates and extensions

Please make every effort to submit work by the due dates. It is your responsibility to structure your study program around assignment deadlines, family, work and other commitments. Factors such as normal work pressures, vacations, etc. are not regarded as appropriate reasons for granting extensions.

Students are advised to NOT assume that granting of an extension is a matter of course.

Students requesting an extension for any assessment during semester (eg. Assignments, tests or presentations) are required to submit a Special Consideration application form (in-semester exam/assessment task), along with original copies of supporting documentation, directly to their lecturer within two working days before the assessment submission deadline. Lecturers will provide specific outcomes directly to students via email within 2 working days. The lecturer reserves the right to refuse late applications.

A copy of the email or other written communication of an extension must be attached to the assignment submission.

Refer to the Faculty Special consideration webpage or further details and to access application forms: http://www.infotech.monash.edu.au/resources/student/equity/special-consideration.html

Late assignment

If you miss a prac or tutorial class for any reason you *must* do the following to obtain an exemption for the missed class:

- Submit an "In-Semester Special Consideration" form no more than one week after you return to University. These forms are available from and should be handed in to the General office (Clayton) in building 63.
- Attach any documentary evidence, for example, medical certificate covering the date of your missed class, letter of explanation, police report or plane boarding pass.

Failure to do the above will result in you being marked absent for the class and receiving zero marks. Exemptions will not be granted automatically, and will be considered on a case by case basis.

Return dates

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

Appendix

Please visit the following URL: http://www.infotech.monash.edu.au/units/appendix.html for further information about:

- Continuous improvement
- Unit evaluations
- Communication, participation and feedback
- Library access
- Monash University Studies Online (MUSO)
- Plagiarism, cheating and collusion
- Register of counselling about plagiarism
- Non-discriminatory language
- Students with disability
- End of semester special consideration / deferred exams