MONASH University
Information Technology

FIT1029
Algorithmic problem solving

Unit Guide

Semester 1, 2010

Last updated: 13 Feb 2010

Table of Contents

FIT1029 Algorithmic problem solving - Semester 1, 2010 1
- Chief Examiner: 1
- Lecturer(s) / Leader(s): 1
  - Clayton 1

Introduction 2
Unit synopsis 2
Learning outcomes 2
Contact hours 3
Workload 3
Unit relationships 3
- Prerequisites 3
- Co-requisites 3

Teaching and learning method 4
- Teaching approach 4
- Timetable information 4
- Tutorial allocation 4
- Unit Schedule 4

Unit Resources 6
- Prescribed text(s) and readings 6
- Recommended text(s) and readings 6
- Equipment and consumables required or provided 6
- Study resources 6

Assessment 7
- Overview 7
- Faculty assessment policy 7
- Assignment tasks 7
- Examination 8
- Due dates and extensions 8
- Late assignment 9
- Return dates 9

Appendix 10

FIT1029 Algorithmic problem solving - Semester 1, 2010

Chief Examiner:

Dr David Albrecht
Senior Lecturer
Phone: +61 3 990 55526
Fax: +61 3 990 55159

Contact hours: Monday 1pm - 2pm

Lecturer(s) / Leader(s):

Clayton

Dr David Albrecht
Senior Lecturer
Phone: +61 3 990 55526
Fax: +61 3 990 55159

Contact hours: Monday 1pm - 2pm

Introduction

Welcome to FIT1029 Algorithmic problem solving. This 6 point unit is core for Bachelor of Computer Science and the Bachelor of Software Engineering.

The unit has been designed to teach you about algorithms, how to develop them to solve problems, their key components, how to reason about them, how to communicate them, and what are their limitations.

Unit synopsis

Algorithms are recipes for solving a problem. They are fundamental to computer science and software engineering. Algorithms are the formal foundation of computer programming but also exist independently of computers as systematic problem-solving procedures. This unit introduces algorithmics, the study of algorithms. It is not about programming and coding but rather about understanding and analysing algorithms and about algorithmic problem-solving, i.e. the design of systematic problem-solving procedures. The unit is very hands-on and students will develop algorithms to solve a wide variety of different problems, working individually as well as together in groups and as a class.

The unit will not require any knowledge of a programming language. The initial instruction will be performed independently of any programming language and only use simple pseudo-code that will be developed from scratch in the unit. Various means of visualising algorithm execution (manipulating sets of tangible physical object, using turtle graphics, using algorithm visualisations) will be employed to enable the students to trace the execution of algorithms and to complement their formal understanding with an intuitive understanding. Later stages of the unit will make use of the coding knowledge developed in FIT1002 to demonstrate how pseudo-code algorithms can be mapped to concrete programs.

Topics include: What is a computational problem and what is an algorithm; Basic control structures; basic data structures; Modular Algorithm Structure; Recursion; Problem-solving strategies for algorithm development; Arguing correctness of an algorithm; Arguing termination of an algorithm; Understanding the efficiency of an algorithm; and Limitations of algorithms.

Learning outcomes

At the completion of this unit students will have -
A knowledge and understanding of:

- the difference between algorithms and processes;
- basic ways to structure algorithms: basic data structures (simple variables, collections structure, specifically vectors, lists, sets, and tables); basic control structures (sequence, choice, iteration);
- recursion;
- modular algorithm structures;
- the equivalence of recursion and iteration;
- problem solving strategies suitable for algorithm development including top-down design and bottom-up design;
- simple standard patterns for algorithms (eg traversal, search);
- what makes a good algorithm
- limitations of algorithms (high level).

Developed the skills to:

- develop simple iterative and recursive algorithms
- argue the correctness of simple algorithms

- judge the efficiency of simple algorithms, and

Developed attitudes that enable them to:

- value clear specification of problems;
- understand the relation between algorithms and programs;
- appreciate the value of designing abstract algorithms before starting to code a program;
- have confidence that they can develop algorithms to solve computational problems;
- appreciate that seemingly difficult problems can have very simple elegant algorithmic solutions (and vice versa);
- value correctness arguments for algorithms; and
- value the importance of simplicity and efficiency.

Demonstrated the communication skills necessary to:

- solve a problem by discussing possible approaches and solutions as a team; and
- clearly communicate (the specification of) a computational problem, its algorithmic solution and arguments for correctness and efficiency.

Contact hours

2 hrs lectures/wk, 2 hrs tutorials/wk

Workload

For on campus students, workload commitments are:

- two-hour lecture and
- two-hour tutorial (requiring advance preparation)
- a minimum of 2-3 hours of personal study per one hour of contact time in order to satisfy the reading and assignment expectations.
- You will need to allocate up to 2 hours per week in some weeks, for use of a computer, including time for newsgroups/discussion groups.

Off-campus students generally do not attend lecture and tutorial sessions, however, you should plan to spend equivalent time working through the relevant resources and participating in discussion groups each week.

Unit relationships

Prerequisites

Only for students in the Bachelor of Computer Science and Bachelor of Software Engineering, associated Double Degrees and major/minor sequences. Exceptions can be approved by the unit leader after assessment of mathematical background knowledge.

Co-requisites

FIT1002

Teaching and learning method

Teaching approach

This unit will be delivered via two one hour lectures. Lecturers will describe various puzzles and problems, go through specific approaches for solving the problems, present various algorithms, and analyse these algorithms.

In tutorials students will discuss in-depth various puzzles and problems, practice communicating algorithms, and investigate the fundamental and interesting aspects about algorithms which will help them programming in other units.

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.cc.monash.edu.au/

Unit Schedule

Week	Topic	Key dates
1	Introduction to the unit and the type of problems	
2	Understanding the Problem and the requirements for a solution	
3	Getting started	
4	Decomposition; Divide and Conquer	March 22nd: Assignment 1 is due. (10%)
5	Induction and Recursion	
Mid semester break		
6	More Recursion; Introduction to Search Techniques	
7	More Search Techniques	April 19th: Assignment 2 is due. (10%)
8	Strategies	
9	Fundamentals	May 3rd: Assignment 3 is due. (10%)
10	Representation of algorithms, knowledge and data	
11	Reasoning	
12	Limitations of algorithms	May 24th: Assignment 4 is due. (10%)
13	Revision	

Unit Resources

Prescribed text(s) and readings

No text books are required.

Recommended text(s) and readings

1. Michalewicz, Z. and M. Michalewicz, *Puzzle-Based Learning: An introduction to critical thinking, mathematics, and problem solving,* Hybrid Publishers, 2008.
2. Harel, D. with Y. Feldman, *Algorithmics: The Spirit of Computing,* 3rd ed., Pearson Education Limited, 2004.
3. Polya, G., *How to solve it; a new aspect of mathematical method,* 2nd ed., Garden City, N.Y., Doubleday, 1957
4. Bentley, J., *Programming Pearls,* Addison-Wesley, 1986
5. Bentley, J., *More Programming Pearls: confessions of a coder,* Addison-Wesley, 1988
6. Skiena, S., *The Algorithm Design Manual, TELOS--the Electronic Library of Science,* 1998
7. Cormen, T., C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms*, The MIT Press, 1990

Equipment and consumables required or provided

Students studying off-campus are required to have the minimum system configuration specified by the Faculty as a condition of accepting admission, and regular Internet access. On-campus students, and those studying at supported study locations 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 **2** 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:

- A MUSO unit web site where lecture slides, weekly tutorial requirements, assignment specifications, and supplementary material will be available
- Discussion groups via MUSO.
- This Unit Information outlining the administrative information for the unit

Assessment

Overview

Examination (3 hours): 60%; In-semester assessment: 40%

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.

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:
  Assignment 1
  Description:
  This assignment will aim to help you understand how to go about finding algorithms to solve problems.
  Weighting:
  10%
  Due date:
  March 22nd

- **Assignment task 2**

  Title:
  Assignment 2
  Description:
  This assignment will aim to help you the importance of fundamental concepts, such as invariants, divide and conquer, and induction in developing algorithms.
  Weighting:
  10%
  Due date:

April 19th

- **Assignment task 3**

  Title:
  Assignment 3
  Description:
  This assignment will help you understand different search techniques.
  Weighting:
  10%
  Due date:
  May 3rd

- **Assignment task 4**

  Title:
  Assignment 4
  Description:
  This assignment will help you communicate and reason about algorithms.
  Weighting:
  10%
  Due date:
  May 24th

Examination

- **Weighting:** 60%
  Length: 3 hours
  Type (open/closed book): Closed book

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

Assignments received after the due date will be subject to a penalty of 5% per day, including weekends. Assignments received later than one week (seven days) after the due date will not normally be accepted. In some cases, this period may be shorter if there is a need to release sample solutions.

This policy is strict because comments or guidance will be given on assignments as they are returned, and sample solutions may also be published and distributed, after assignment marking or with the returned assignment.

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