MONASH University

FIT3082
Programming languages and paradigms

Unit guide

Semester 1, 2009

Last updated : 13 Feb 2009

Table of Contents

FIT3082 Programming languages and paradigms - Semester 1, 2009

Unit leader :

Bernd Meyer

Lecturer(s) :

Clayton

- Bernd Meyer
- Maria Garcia de la Banda

Introduction

Welcome to FIT3082. At this point in your studies you should have achieved proficiency in at least one programming language, but you have probably also started to wonder whether there are fundamentally different ways of programming. The answer is that there are many, and for any computer scientist/software engineer knowledge of more than just a single programming paradigm is essential. This unit gives you an in-depth introduction to the two most important ones (functional programming and logic programming) and at least a glimpse of some other alternative models of computation that call for entirely different ways of programming.

Unit synopsis

ASCED: 020103 Programming

Building on knowledge in imperative and object-oriented programming, FIT3082 provides a thorough understanding of the main programming alternative language paradigms: functional and logic programming. In doing this, it reviews different example languages and the kinds of applications each language is intended for, focusing on how the choices made during the design of a language affect the resulting ease of programming. Students will gain practical programming experience in languages that use these paradigms, such as SML and Prolog.

Learning outcomes

Knowledge and Understanding

K1. Knowledge of the history of programming languages and the reasons for their introduction. Understanding of the fact that programming languages and programming language paradigms have evolved and will almost certainly change in the future.

K2. Knowledge and comprehension of programming language paradigms including imperative, functional and logical. Comprehension that the choice of programming language affects the ease of programming. Ability to evaluate whether a particular paradigm or programming language is well-suited for a particular problem.

K3. Appreciation of the importance of formal mathematical calculi as the basis for programming languages (in particular, Lambda Calculus and First Order Predicate Calculus). Understanding of how these calculi can provide the basis for programming language semantics and understanding of their limitations, in particular regarding

interactive programs.

K4. Knowledge of the core aspects of modern functional programming languages: Functions as first class objects, type systems, higher-order functions, eager and lazy evaluation.

K5. Knowledge of a particular functional programming language, such as ML or Scheme, which exemplifies most of these aspects and the ability to use it for implementing non-trivial algorithms.

K6. Knowledge of the core aspects of logic programming: logical inference, deduction, resolution, negation as failure, logic variables and unification.

K7. Knowledge of a particular logic language, such as Prolog or CLP, which exemplifies most of these aspects and the ability to use it for implementing non-trivial algorithms.

K8. Knowledge of the main issues in programming language design and comprehension of the various design alternatives and of the possibility to create mixed-paradigm languages, as exemplified in Python.

Attitudes, Values and Beliefs

A1. Value the important role that theory (in this case logic, type theory and lambda-calculus) can play in the development of practical software applications (in this case compilers and program verification).

A2. Carefully consider the characteristics of the different programming languages available whenever developing a new application.

Practical Skills

P1. Become an informed user of programming languages, choosing the right programming language for a particular task.

P2. Be able to implement and debug non-trivial algorithms in both a functional and a logic language.

Workload

two 1-hour lectures, weekly

one 1-hour tutorial (combined with practical work at the computer), weekly

9 hours of personal study (incl reading, lecture preparation and assignment preparation)

Unit relationships

Prerequisites

Before attempting this unit you must have satisfactorily completed

FIT2004 or CSE2304, FIT2014 or CSE2303

or equivalent.

Students beginning FIT3082 are assumed to be competent programmers in at least one programming language based on a paradigm other than procedural Students are required to have a thorough understanding of datatypes and

recursion and the fundamental concepts of formal languages and grammars. Knowledge of first order predicate calculus is required.

Relationships

FIT3082 is an elective unit in the Bachelor of Computer Science and the Bachelor of Software Engineering.

Before attempting this unit you must have satisfactorily completed

FIT2004 or CSE2304, FIT2014 or CSE2303

or equivalent.

You may not study this unit and

CSE3322

in your degree.

Continuous improvement

Monash is committed to 'Excellence in education' (Monash Directions 2025 - http://www.monash.edu.au/about/monash-directions/directions.html) and strives for the highest possible quality in teaching and learning.

To monitor how successful we are in providing quality teaching and learning Monash regularly seeks feedback from students, employers and staff. One of the key formal ways students have to provide feedback is through Unit Evaluation Surveys. The University's Unit Evaluation policy (http://www.policy.monash.edu/policy-bank/academic/education/quality/unit-evaluation-policy.html) requires that every unit offered is evaluated each year. Students are strongly encouraged to complete the surveys as they are an important avenue for students to "have their say". The feedback is anonymous and provides the Faculty with evidence of aspects that students are satisfied and areas for improvement.

Faculties have the option of administering the Unit Evaluation survey online through the my.monash portal or in class. Lecturers will inform students of the method being used for this unit towards the end of the semester.

Student Evaluations

If you wish to view how previous students rated this unit, please go to http://www.adm.monash.edu.au/cheq/evaluations/unit-evaluations/

Improvements to this unit

Some material has been removed from the syllabus (advanced material from meta programming and CLP) in favour of treating the remaining material in more depth.

Unit staff - contact details

Unit leader

Associate Professor Bernd Meyer
Associate Professor
Phone +61 3 990 52240
Fax +61 3 990 31077

Lecturer(s) :

Associate Professor Bernd Meyer
Associate Professor
Phone +61 3 990 52240
Fax +61 3 990 31077
Associate Professor Maria Garcia De La Banda
Associate Professor
Phone +61 3 990 55777
Fax +61 3 990 55157

Teaching and learning method

Lectures will be used to present new concepts, introduce the fundamental approaches to problem solving with particular paradigms, compare different approaches, analyse their advantages and disadvantages, and propose general questions. The aim is to give students an overview of the concepts and to challenge them to think further. Tutorials and practicals will be used to link the theory with practice and deepen the students understanding and practical abilities. To this end, tutorials and practicals are tightly integrated into sessions where theoretical discussion and practical programming work complement each other.

Tutorial allocation

On-campus students should register for tutorials/laboratories using Allocate+.

Communication, participation and feedback

Monash aims to provide a learning environment in which students receive a range of ongoing feedback throughout their studies. You will receive feedback on your work and progress in this unit. This may take the form of group feedback, individual feedback, peer feedback, self-comparison, verbal and written feedback, discussions (on line and in class) as well as more formal feedback related to assignment marks and grades. You are encouraged to draw on a variety of feedback to enhance your learning.

It is essential that you take action immediately if you realise that you have a problem that is affecting your study. Semesters are short, so we can help you best if you let us know as soon as problems arise. Regardless of whether the problem is related directly to your progress in the unit, if it is likely to interfere with your progress you should discuss it with your lecturer or a Community Service counsellor as soon as possible.

Unit Schedule

Week	Topic	Key dates
1	Overview, History of Programming Languages	
2	ML I: Expressions, Functions, Basic Datatypes, Lists, Pattern Matching	
3	ML II: Type Definitions, Polymorphism	Hurdle Prac 1
4	ML III: Higher order functions, Structures	
5	ML IV: Input/Output, Destructive Update, Binding Mechanisms, Scope and Extent	Hurdle Prac 2

6	ML V: Type Inference, Overview of Abstraction Mechanisms	
Mid semester break		
7	LP I: Rule-based Programming, Deductive Databases, Basic LP Syntax, Logic Semantics	April 24, Assignment 1 due
8	LP II: Deduction, Search & Backtracking, Unification	
9	LP III: Basic Lists and Difference Lists, Cut, Negation by failure, Impure Features (I/O, Arithmetic)	Hurdle Prac 3
10	LP IV: Metaprogramming	
11	LP V: CLP basics	Hurdle Prac 4
12	LP VI: CLP	
13	Revision	June 5, Assignment 2 due

Unit Resources

Prescribed text(s) and readings

Jeffrey D. Ullman, "Elements of ML Programming", Prentice Hall. Leon Sterling and Ehud Shairo: "The Art of Prolog", MIT Press.

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

Kenneth C. Louden "Programming Languages, Principles and Practice", Thomson.

David A. Watt "Programming Language Design Concepts", Wiley.

The following technical reports / journal articles may also be helpful and interesting:

(LP III) K. Knight: Unification: A Multidisciplinary Survey. ACM Computing Surveys 21(1)1989:93-124

(LP V) T. Fruwirth et al: Constraint Logic Programming - an informal introduction (ECRC Tech Report 93-5)

(LP V) P. van Hentenryck: Constraint logic programming, The Knowledge Engineering Review 6(3)1991:151-194

(Tutorial and Labs for LP) D.H.D. Warren: Logic Programming and Compiler Writing. Software - Practice and Experience 10, 1980:97-125

(Tutorials and Labs for LP) J. Cohen and T.J. Hickey: Parsing and Compiling Using Prolog. ACM Transactions on Programming Languages and Systems 9(2)1987:125-163

Required software and/or hardware

You will need access to

* SML/NJ (latest version)

* SICStus Prolog (latest version)

both are installed in the labs. For home use you can download SML/NJ from http://www.smlnj.org . Several free Prolog Systems are publicly available on the web (unfortunately, SICStus is not free).

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 **n** 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:

- Weekly detailed lecture notes outlining the learning objectives, discussion of the content, required readings and exercises;
- Fortnighlty tutorial and laboratory tasks and exercises;
- Assignment specifications and sample solutions;
- A sample examination and suggested solution
- Access to past examination papers when available
- Discussion groups;
- This Unit Guide outlining the administrative information for the unit;
- The unit web site on MUSO (Moodle), where the resources outlined above will be made available.

Library access

The Monash University Library site contains details about borrowing rights and catalogue searching. To learn more about the library and the various resources available, please go to http://www.lib.monash.edu.au.

The Educational Library and Media Resources (LMR) is also a very resourceful place to visit at http://www.education.monash.edu.au/library/

Monash University Studies Online (MUSO)

All unit and lecture materials are available through MUSO (Monash University Studies Online). Blackboard is the primary application used to deliver your unit resources. Some units will be piloted in Moodle. If your unit is piloted in Moodle, you will see a link from your Blackboard unit to Moodle (http://moodle.monash.edu.au) and can bookmark this link to access directly. In Moodle, from the Faculty of Information Technology category, click on the link for your unit.

You can access MUSO and Blackboard via the portal: http://my.monash.edu.au

Click on the Study and enrolment tab, then Blackboard under the MUSO learning systems.

In order for your Blackboard unit(s) to function correctly, your computer needs to be correctly configured.

For example:

- Blackboard supported browser

- Supported Java runtime environment

For more information, please visit: http://www.monash.edu.au/muso/support/students/downloadables-student.html

You can contact the MUSO Support by phone : (+61 3) 9903 1268

For further contact information including operational hours, please visit:
http://www.monash.edu.au/muso/support/students/contact.html

Further information can be obtained from the MUSO support site:
http://www.monash.edu.au/muso/support/index.html

Assessment

Unit assessment policy

The unit is assessed with two assignments (assessed in the pracs) and a three hour closed book examination. To pass the unit you must:

- attempt at least one assignment and the examination
- achieve no less than 40% of the possible marks in the total non-examination assessment
- achieve no less than 50% of the possible marks in the examination
- achieve no less than 50% of possible marks overall

Note that there are also four "Hurdle Pracs" assessed as pass/fail. You will only be eligible to submit Assignment 1 if you pass at least one of the Hurdle Pracs 1, 2 and you will only be eligible to submit Assignment 2 if you pass at least one of the Hurdle Pracs 3, 4.

Assignment tasks

- ### Assignment Task

  Title : Functional Programming

  Description :

  This will consist of programming tasks in ML including advanced functional concepts, such as higher order functions, list comprehensions, simulated laziness etc.
  Weighting : 15%

  Criteria for assessment :

  Your programs will be marked on correctness, style, efficiency, clarity and documentation. Specific tasks and marking criteria will be distributed at the appropriate time during the semester.
  Due date : April 24
- ### Assignment Task

  Title : Logic Programming

  Description :

This will consist of programming tasks in Prolog/CLP ranging from introductory concepts, such as lists, terms, and unification, to advanced concepts, such as meta-programming, DCGs, constraint solving, extra-logical features etc.
Weighting : 15%

Criteria for assessment :

Your programs will be marked on correctness, style, efficiency, clarity and documentation. Specific tasks and marking criteria will be distributed at the appropriate time during the semester.
Due date : June 5

Examinations

- **Examination 1**

Weighting : 70%

Length : 3 hours

Type (open/closed book) : Closed book

Assignment submission

Assignments will be submitted by **mixed electronic and paper** submission**.** On-campus Students Submit the assignment in the pracs, with the appropriate cover sheet correctly filled out and attached Off Campus (OCL) students [OCL only] Mail your assignment to the Off-Campus Learning Centre with the cover sheet attached. Do not email submissions. The due date is the date by which the submission must be received/the date by which the the submission is to be posted

Assignment coversheets

All assignments must include an assignment coversheet. For the paper submission, the coversheet can be found in http://infotech.monash.edu.au/resources/students/assignments/ For the electronic submission, the coversheet can be found in MUSO (Moodle).

University and Faculty policy on assessment

Due dates and extensions

The due dates for the submission of assignments are given in the previous section. 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 seldom regarded as appropriate reasons for granting extensions. Students are advised to NOT assume that granting of an extension is a matter of course.

Requests for any other extension must be made to the unit lecturer or head tutor at your campus at least two days before the due date. You will be asked to forward original medical certificates in cases of illness, and may be asked to provide other forms of documentation where necessary. A copy of the email or other written communication of an extension must be attached to the assignment submission.

You are, however, generally allowed to submit your work in your allocated laboratory during the week of the due date even if you laboratory is after that due date.

Late assignment

Assignments received after the due date will be subject to a penalty of 10% per day of late submission. No submissions will be accepted later than 1 week after the due date.

Return dates

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

Assessment for the unit as a whole is in accordance with the provisions of the Monash University Education Policy at http://www.policy.monash.edu/policy-bank/academic/education/assessment/

We will aim to have assignment results made available to you within two weeks after assignment receipt.

Plagiarism, cheating and collusion

Plagiarism and cheating are regarded as very serious offences. In cases where cheating has been confirmed, students have been severely penalised, from losing all marks for an assignment, to facing disciplinary action at the Faculty level. While we would wish that all our students adhere to sound ethical conduct and honesty, I will ask you to acquaint yourself with Student Rights and Responsibilities (http://www.infotech.monash.edu.au/about/committees-groups/facboard/policies/studrights.html) and the Faculty regulations that apply to students detected cheating as these will be applied in all detected cases.

In this University, cheating means seeking to obtain an unfair advantage in any examination or any other written or practical work to be submitted or completed by a student for assessment. It includes the use, or attempted use, of any means to gain an unfair advantage for any assessable work in the unit, where the means is contrary to the instructions for such work.

When you submit an individual assessment item, such as a program, a report, an essay, assignment or other piece of work, under your name you are understood to be stating that this is your own work. If a submission is identical with, or similar to, someone else's work, an assumption of cheating may arise. If you are planning on working with another student, it is acceptable to undertake research together, and discuss problems, but it is not acceptable to jointly develop or share solutions unless this is specified by your lecturer.

Intentionally providing students with your solutions to assignments is classified as "assisting to cheat" and students who do this may be subject to disciplinary action. You should take reasonable care that your solution is not accidentally or deliberately obtained by other students. For example, do not leave copies of your work in progress on the hard drives of shared computers, and do not show your work to other students. If you believe this may have happened, please be sure to contact your lecturer as soon as possible.

Cheating also includes taking into an examination any material contrary to the regulations, including any bilingual dictionary, whether or not with the intention of using it to obtain an advantage.

Plagiarism involves the false representation of another person's ideas, or findings, as your own by either copying material or paraphrasing without citing sources. It is both professional and ethical to reference clearly the ideas and information that you have used from another writer. If the source is not identified, then you have plagiarised work of the other author. Plagiarism is a form of dishonesty that is insulting to the reader and grossly unfair to your student colleagues.

Register of counselling about plagiarism

The university requires faculties to keep a simple and confidential register to record counselling to students about plagiarism (e.g. warnings). The register is accessible to Associate Deans Teaching (or nominees) and, where requested, students concerned have access to their own details in the register. The register is to serve as a record of counselling about the nature of plagiarism, not as a record of allegations; and no provision of appeals in relation to the register is necessary or applicable.

Non-discriminatory language

The Faculty of Information Technology is committed to the use of non-discriminatory language in all forms of communication. Discriminatory language is that which refers in abusive terms to gender, race, age, sexual orientation, citizenship or nationality, ethnic or language background, physical or mental ability, or political or religious views, or which stereotypes groups in an adverse manner. This is not meant to preclude or inhibit legitimate academic debate on any issue; however, the language used in such debate should be non-discriminatory and sensitive to these matters. It is important to avoid the use of discriminatory language in your communications and written work. The most common form of discriminatory language in academic work tends to be in the area of gender inclusiveness. You are, therefore, requested to check for this and to ensure your work and communications are non-discriminatory in all respects.

Students with disabilities

Students with disabilities that may disadvantage them in assessment should seek advice from one of the following before completing assessment tasks and examinations:

- Faculty of Information Technology Student Service staff, and / or
- your Unit Coordinator, or
- Disabilities Liaison Unit

Deferred assessment and special consideration

Deferred assessment (not to be confused with an extension for submission of an assignment) may be granted in cases of extenuating personal circumstances such as serious personal illness or bereavement. Information and forms for Special Consideration and deferred assessment applications are available at http://www.monash.edu.au/exams/special-consideration.html. Contact the Faculty's Student Services staff at your campus for further information and advice.