Course Outline

Michael P. Fourman

February 2, 2010

CS201 is an introduction to algorithms and data structures, and to the analysis of their behaviour. This course will be taught using the language *Standard ML*, (which has a number of features making it particularly apt for teaching this subject matter). No prior knowledge of SML is assumed, but programming proficiency and experience in a language such as Pascal or C is a prerequisite.

The first section of the course (up to the Easter Break) will introduce the SML language, give a fairly rapid review of concrete data types: integers, strings, booleans, records, lists, and trees, and develop the idea of data abstraction. Small practical exercises will be set to develop your familiarity with these ideas.

The second section of the course will introduce a number of implementations for abstract data types such as sets, dictionaries, and queues, introduce graphs and a repertoire of elementary graph algorithms, and discuss data-structures to support sorting and searching. An initial practical will introduce you to SML modules, then, in a four-week project, you will use a variety of data-structures to build a simple machine-code optimiser.

Assessment will be based on a combination of practical work and two one-hour examinations, given at the end of each section of the course. A detailed schedule of lectures, practical deadlines and examinations is given at the end of this document.

Practical work submitted for assessment is expected to be the result of your individual efforts. You may discuss general principles with others in order to understand the topic—indeed this is encouraged; but, you are responsible for designing and implementing your own solutions.

Practise Classes

These are informal question and answer sessions where a member of the teaching staff is available to answer questions about the course and assignments in a small group setting. These will be held on **Tue 9-10 am**, **Tue 10-11am** and **Tue 4-5pm**, accomodation at these classes is limited. You *may* attend one (1) practise class per week. The practice classes will be discontinued if there is insufficient attendance.

Laboratory Sessions

All laboratory sessions will be held the the second year laboratory, on the second floor of the new computer science building.

There will be supervised laboratory sessions scheduled for CS201, commencing in the second week of semester. Seven sessions are provided, at: **Wed 9-11am**, **Wed 11-1pm**, **Wed 2-4pm**, **Wed 4-6pm**, **Thur 9-11am**, **Thur 2-4pm** and **Thur 4-6pm**. Each student should sign up for one of these sessions during the first week of semester using the sheets outside the Labs. In these sessions, you will be expected to work on the practical exercises, distributed in the Friday lectures, and the project (which will be distributed after the Easter break). Each lab session extends for **2 hours**; students registered for the session will have priority access.

However, you may not able to finish the work during the supervised lab session; you are expected to complete it by coming to the laboratory outside the scheduled lab times. You will normally have 24-hour access to the lab. (However, this will not be available until the end of March, when the card entrance system is expected to become operational. Until then, access will be limited to office hours.)

The practical component of the course is considered compulsory and failure to hand in satisfactory practical work may cause a student to be excluded from sitting the CS201 exam.

Summary Timetable

	Monday	**Tuesday**	**Wednesday**	**Thursday**	**Friday**
9 - 11 am		9am Prac Class (MPF) 10am Prac Class (MJM)	Lab (LB)	Lab (MJM)	
11 - 1 pm			Lab (MJM)	12pm Lecture (MPF)	
1 - 2 pm	Lunch	Lunch	Lunch	Lunch	Lunch
2 - 4 pm			Lab (LB)	Lab (MPF)	1pm Lecture (MPF)
4 - 6 pm		4pm Prac class (MJM)	Lab (RS)	Lab (BB)	

Notes

Thursday CS201 Lecture in Ross Lecture Theatre.
Friday CS201 lecture in Alexander Lecture Theatre.
Labs are in the Second year Lab (Room 2.3).
Practise Classes are in Room 2.28 in the Computer Science Building. Accomodation at these classes is limited.

Key to Abbreviations

MPF Michael P. Fourman. (Lecturer)

MJM Mark J. Messenger. (Assoc. Lecturer)

BB Bruce Backman. (Lab. Supervisor)

LB Luigi Barone. (Lab. Supervisor)

RS Rameri Salama. (Lab. Supervisor)

Staff Availability

In addition to the practice classes, Michael Fourman (2.24) will be available for questions on this course from 4pm to 5pm on Thursdays, and Mark Messenger (2.18) will be available for student consultation from 9am to 11am on Wednesdays. However, questions of general interest should, where possible, be brought to one of the practice classes.

Books and Lecture Notes

The recommended book for this course is, *ML for the Working Programmer,* by Paulson. You are strongly advised to purchase a copy[1]. One important strand of the CS201 course is an introduction to the functional programming language Standard ML (SML). Paulson will provide an essential reference for this material.

Handouts will accompany each lecture. They will contain a summary of the material covered; however, they will often be terse, and may not include diagrams and examples used in the lectures. The notes are intended as a revision aid, to complement the book, and are not a substitute for lecture attendance.

Some lecture summaries in this note refer to particular pages in Paulson's book (e.g., P21-24). Please note that it is not always necessary to understand *all* of the material on these pages — the book sometimes covers a topic in more detail than is required for CS201. You only need to understand the material referred to in the lectures and handouts. The page references will allow you to read ahead if you get time, and you may get more out of the lectures if you do this.

Conversely, not all the material of the course is covered by Paulson's book. If a lecture summary here doesn't have a reference to the book, it usually means that the material is spread over a number of chapters, or is covered later in the book using examples you may not understand at that point. However, in some cases, it means that the material is not covered at all in the book; such topics will be covered in greater detail in the lecture notes.

Another book, *Introduction to Algorithms,* by Cormen, Leiserson and Rivest, (1028 pages) is mildly recommended for further reading on data structures. It is an excellent book and includes most of the topics you will not find in Paulson. However, it will not serve as a reference for ML, and, in general, it is a little advanced for CS2 purposes. If you plan to do a Computer Science Honours degree, it may be worthwhile buying it now. Some handouts will include references to Cormen, Leiserson and Rivest's book for those who want to learn more about the topics covered, but such reading is not necessary for examination purposes.

These books are available on closed reserve in the library. A selection of alternative ML texts will be mentioned in Lecture Note 1.

[1] A limited, but hopefully sufficient, number of copies of Paulson will be available **between the hours of two, and four, pm.** from the Computer Science Office (room 1.31A), for $40 each.

Acknowledgements

The lecture notes are based on material produced, by myself and others, for courses at The University of Edinburgh. In particular, I have drawn extensively from handouts prepared by Gordon Brebner (GB) and Kevin Mitchell (KM) for the CS2Ah course given at The University of Edinburgh. I am also grateful to Stephen Gilmore for several examples taken from his MSc course on Functional Programming, and to Stuart Anderson for his invaluable advice.

The initials of the primary author of each lecture note will appear in its reference number. While credit for the contents of each note is due to this author, it should be noted that I have edited the notes, in some cases extensively, for use at UWA. Responsibility for the mistakes and infelicities I will undoubtedly have introduced is mine.

Michael Fourman, Perth, February 1994.

Assessment

This course will be assessed by a combination of practical work (40%) and examination (60%). *Students who fail to complete the practical component may be excluded from sitting the final examination of CS201.*

CS201 is a prerequisite for many other Computer Science and Information Technology units. You may not be permitted to enrol in other CS units, if you do not achieve a satifactory result in CS201.

Practicals and Examinations

Week 2 Deadline 11.3.94 Introduction to ML – Complexity in theory and practice 4%

Week 3 Deadline 18.3.94 New types – Rational numbers and interval arithmetic 4%

Week 4 Deadline 25.3.94 List manipulation – Polynomials 4%

Week 5 Deadline 31.3.94 Tree manipulation – Syntax trees and algebraic manipulation 4%

Week 6 mid-term examination 14.4.94 25%

This one-hour examination will cover the first section of the course.

Week 7 Deadline 22.4..94 Modular Programming – A basic toolkit 4%

Weeks 8-11 Project Final Deadline 20.5.94 Optimising three-address codes. 20%

Issued 22.4.94, Intermediate Deadline 6.5.94

Week 13 final examination 3.6.94 35%

This one hour examination will cover topics from the entire course.

Examinations will be held in normal class hours, **in a venue to be announced.**

Lecture Plan

1. **SML Basics** 3.3.94

   Expressions, values, and functions. (P15-19)
   Basic types (P21-24)

2. **Recursion and Complexity (review)** 4.3.94

   Recursive functions and pattern matching (P45-49)
   Managing environments (P49-54)

3. **Products and Records** 10.3.94

   Simple structured types; Records. (P25-33)
   Complex Numbers

4. **Specification, Implementation, and Use practical 1 deadline — 11.3.94**

   Signatures,
   Structures,
   Functors
   Limitations of SML Modules

5. **Enumerations and Sums** 17.3.94

   The datatype declaration (P104-106)
   Enumeration types (P107)
   Constructors and pattern matching (P109-110)
   Exceptions. (P114-116)

6. **Lists practical 2 deadline — 18.3.94**

   Basic list operations (P62-75)
   Templates for stack and accumulative recursion
   Sets using lists (P87-89)

7. **Stacks and Queues** 24.3.94

Queues (P236-250)
Philosophy of Abstract Data Types
The `abstype` construct
Examples
Limitations of `abstype`

8. Trees **practical 3 deadline — 25.3.94**

Representing Trees (P120-122)
Traversing trees (P123-125)
Syntax trees

9. Reasoning about Programs **practical 4 deadline — 31.3.94**

Equational reasoning
Proving properties of functions using induction
Induction for lists (P199-203)
Induction for trees (P204-207)

10. Mid-term Exam **Class exam 14.4.94**

This one-hour examination will cover the first section of the course. It will contribute 25% to your final mark.

11. Functions as Data 15.4.94

Association Lists
Sets

12. Advanced SML Modules 21.4.94

Sharing constraints (P264-267)
Views
Styles of use (P268-271)

13. Graphs **practical 5 deadline — 22.4.94**

Definition and applications of graphs
Representation of graphs
Searching graphs: depth-first, breadth-first (P91-92)
Topologically sorting graphs (P93-95)

14. Spanning trees 28.4.94

Kruksal's algorithm
Prim's algorithm

15. Graphs concluded 29.4.94

Finding paths in graphs
Finding shortest and longest paths

16. Arrays 5.5.94

Arrays (P300-304)
Mutable Stacks
Random Permutation
Histograms
Hashing

17. Mutable Data **Intermediate Project deadline — 6.5.94**

References (P280-290)
Marking and clearing
Mutable Queues and Lists
Ring Buffers(P296-300)

18. Infinite Data 12.5.94

Simple sequences (P166-174)
Erastosthenes and other examples
Lazy evaluation (P42-44)
A lazy implementation using references (P292-294)

19. Heaps 13.5.94

The heap invariant
Binomial heaps
Binary heaps

20. Sorting and Searching (review) 19.5.94

Quicksort and Mergesort (P95-101)

21. Binary Search trees **Final project deadline — 20.5.94**

Binary search trees (P126-129)
AVL trees
red-black trees

22. Balanced Search Trees 26.5.94

2-3 trees
B-trees

23. More Reasoning about Programs 27.5.94

Structural induction
Reasoning about functionals (P208-210)
Well-founded induction and recursion (P211-215, 219-225)

24. Design of Data Structures and Algorithms 2.6.94

Algorithm styles
Data structure styles
Further analysis techniques

25. Final Exam **Class exam 3.6.94**

This one hour examination will cover topics from the entire course. It will contribute 35% to your total mark.