15–212: Principles of Programming

Spring 2011

Carnegie Mellon University
Department of Computer Science

URL: http://www.cs.cmu.edu/∼me/212/

(file last updated: January 10, 2011)

1 Teaching Staff

Professor

	Michael Erdmann
URL:	http://www.cs.cmu.edu/~me
Email:	me @C
Phone:	268-7883
Office:	Gates Hillman 9203
Office Hours:	To be announced

Teaching Assistants

	Pablo Chavez	Mark Hahnenberg	Nathan Herzing	Roger Su	Ian Voysey
Email:	pchavez @A	mhahnenb @A	nfh @A	rmsu @A	iev @A
Office:	TBA	TBA	TBA	TBA	TBA
Office Hours:	TBA	TBA	TBA	TBA	TBA

Note: Email addresses listed above should be interpreted as follows:
        @A — @andrew.cmu.edu   @G — @gmail.com   @C — @cs.cmu.edu

Date	Topic	Reading
11 Jan	Introduction; Evaluation & Typing	Chapter 1 & *handout*
13 Jan	Binding, Scope, Functions	Chapter 2
18 Jan	Recursion & Induction	*handout*
20 Jan	Datatypes & Patterns; Lists	Chapters 3, 5, 7.1–7.5
25 Jan	Structural Induction and Tail Recursion	Chapter 17 & *handout*
26 Jan	*Assignment 1 due*	
27 Jan	Higher Order Functions and Staged Computation	Chapter 9
1 Feb	Data Structures	Chapters 8, 10, 11.1–11.3
3 Feb	Representation Invariants	*as above*
8 Feb	Continuations	*handout*
9 Feb	*Assignment 2 due*	
10 Feb	Regular Expressions	*handout*
15 Feb	*Review*	
17 Feb	*Midterm*	
22 Feb	Combinators	
24 Feb	Exceptions, n-Queens	Sections 7.6–7.7
1 Mar	Functors and Substructures	Chapter 11
2 Mar	*Assignment 3 due*	
3 Mar	Game Tree Search	
8 Mar	*No Class* (Spring Break)	
10 Mar	*No Class* (Spring Break)	
15 Mar	Mutation and State	Chapter 18, 5.5, & *handout*
17 Mar	Ephemeral Data Structures	*as above*
22 Mar	Streams, Demand-Driven Computation	Chapter 14
23 Mar	*Assignment 4 due*	
24 Mar	Streams, Laziness and Memoization	
29 Mar	Lexical Analysis & Grammars	
31 Mar	Grammars & Parsing	*handout*
5 Apr	Evaluation, Interpreters, and Recursion	
6 Apr	*Assignment 5 due*	
7 Apr	Language Properties	
12 Apr	Interpreters and Recursion	
14 Apr	*No Class* (Spring Carnival Break)	
19 Apr	Computability	*handout*
20 Apr	*Assignment 6 due*	
21 Apr	Computability	*as above*
26 Apr	TBA	
28 Apr	*Review*	
TBA	*Final*	

Table 1: Schedule of Lectures and Readings (subject to change)

2 Class Schedule

The schedule of lectures and associated readings for the class is given in Table 1 on page 2. Also, please be sure to read the schedule posted on the course webpage. That schedule contains various links to further readings and handouts.

The schedule is subject to change during the semester. The webpage version of the schedule will be updated to reflect schedule changes as the term progresses.

3 Meeting Times

Lecture: Tuesday and Thursday, 3:00 – 4:20, in Wean Hall 7500.

Recitations: Wednesday of each week at the following times and locations, with the teaching assistants shown:

- Sec E: 11:30–12:20   in Gates 4215: Mark Hahnenberg

- Sec A: 12:30– 1:20   in Wean Hall 5302: Ian Voysey

- Sec B: 1:30– 2:20   in Wean Hall 5302: Pablo Chavez

- Sec C: 2:30– 3:20   in Porter Hall 226C: Roger Su

- Sec D: 3:30– 4:20   in Wean Hall 5312: Nathan Herzing

4 Textbooks

Required: The required textbook for this course is *Introduction to Programming Using SML*, by M. R. Hansen and H. Rischel, Addison-Wesley, 1999.

Web Text: Professor Robert Harper has written a textbook *Programming in Standard ML*, which may be found at `http://www.cs.cmu.edu/~rwh/smlbook`
Older webnotes are at `http://www.cs.cmu.edu/~rwh/introsml`
This web text is also required reading. It will provide additional perspective on the course material.

Supplementary: As a supplementary reference text, take a look at *ML for the Working Programmer*, 2nd edition, by L.C. Paulson, Cambridge University Press, 1996. This book was the course text prior to Spring 2000.

5 Objectives

The objective of 15–212 is to provide the student with a firm foundation in the fundamental principles of computer science, focusing in particular on high-level programming techniques. You will learn how to decompose problems, how to compose the solutions into complete programs, and how to reason about programs to ensure that they are correct.

The course will be taught using the Standard ML programming language, a relatively new language supporting higher-order functions, exceptions, polymorphism, data abstraction, and modularity. Using Standard ML as a vehicle we will cover the following topics:

- Induction and recursion, loop invariants.

- Symbolic computation.

- Data abstraction, representation invariants.

- Higher-order functions, continuations.

- Assignment, mutable data structures.

- Modularity and the structure of large programs.

- Streams and demand-driven programming.

- Exceptions, advanced control.

- Grammars and parsing.

- State encapsulation, objects.

- Computability.

6 Course Requirements

Participation in this course consists of the following activities:

- Attending and participating in lectures and recitations.

- Reading the textbooks and handouts.

- Carrying out homework exercises.

- Taking written examinations.

Each of these activities will have a bearing on your final grade in the course.

Attendance at lecture and recitation is *strongly* encouraged. You are responsible for all material presented in lecture and in recitation. The lectures are *not* based directly on the textbook. You should *not* expect that all lecture material will be given to you in written form, nor should you expect that the material presented in lecture will be drawn from the text. The recitations are a supplement to the lectures consisting of both new material not presented in lecture and review of both lectures and homework problems. Active participation in recitation is strongly encouraged, and will increase your chances for getting a good grade.

7 Homework

7.1 Overview

Homework assignments are a critical part of the course work. A full understanding of the material in the textbooks and lectures can only be gained by applying them to solve problems. Assignments may involve both written work as well as programming. All assignments will be handed out electronically, approximately every other week.

Due Dates: Programming assignments are to be handed in using the electronic submission procedure described below. Assignments are due at *2:12AM* (that's **AM**) on the due date.

Caution: Programming assignments are due at 2:12 AM on Wednesdays. That means overnight Tuesday/Wednesday. Don't get confused and hand in your assignment Thursday morning, unless you want it to be a day late.

Late Policy: Late homework will be accepted up to the *start* of lecture on Thursday, with a 25% penalty assessed for lateness. No homework assignments will be accepted after the start of the Thursday lecture.

Extensions: There will be no homework extensions beyond the automatic one-day late policy. *It is your responsibility to plan ahead and start assignments early enough so that you will finish on time even in the face of unexpected but common disruptions.* Common disruptions include minor illnesses such as colds and flus, computer or network failures, responsibilities in other classes, etc.

Extra Credit: Some homework assignments may include extra credit. Any extra credit earned will be recorded separately from your nominal grade, and will be used in determining your final letter grade for the course.

Assignment Return: Graded assignments will be returned either electronically or at recitation, generally one week after they are submitted. Requests for re-grades are handled by the teaching assistant responsible for the assignment, and should be made directly to him or her.

7.2   Homework Requirements and Grading Basis

Summary: Assignments will be graded based on comments, style, and correctness. You should hand in code that runs, but a working program is *not* sufficient for full credit. Make sure your code runs, is well-implemented, *and* is properly annotated with invariants and comments.

Purpose: The main purpose of this course is to teach you advanced programming techniques as outlined in Section 5. SML is a convenient vehicle by which to explore these techniques, because of its functional nature and methods of abstraction. The primary purpose of the course is not to make you super SML hackers. Rather, the purpose is to instill in you an understanding of advanced programming techniques not easily seen in other languages such as Java or C++, including methods of reasoning about programs, "spec"ing programs, and proving program correctness. After you graduate from CMU, we hope that some of these techniques will help you in designing new languages, implementing new systems, or creating new programming methodologies.

Style: The homework assignments are tools to help you explore some of the power and subtleties encountered with strong typing, functional programming, and recursive thinking. Naturally, SML's expressive power allows you to ignore its features and instead program in an imperative C-like style. If you do that, you will miss the whole point of the semester. For this reason we will grade you on style. You are required to program in a functional style, using higher-order abstraction, modularity, and typing. In the last third of the course we will permit a modicum of imperative programming in certain contexts.

Create Working Programs AND ... For each assignment you will be asked to electronically hand in your code. Please make sure that any file you hand in compiles and runs, even if you have only finished part of the assignment. If you hand in a file that does not compile and run we will assume that you did not understand your own code, which generally is a sign of someone trying to hack together an assignment at the last minute. DO NOT DO THAT. Instead, start assignments early and spend plenty of time thinking about the assignment and outlining your solution prior to coding.

... Explain Yourself (VERY IMPORTANT): Always annotate your code with comments. State the types of variables and functions. Describe the values expected by and produced by functions. State invariants assumed by your functions. Explain all auxiliary functions similarly. [We will provide you with examples of the types of comments we expect during the first couple of lectures.] — If you are unable to complete portions of an assignment, comment out the part of the code that does not work properly, and explain what you did, what worked, and what didn't. It is your responsibility to explain as carefully as you can why you think you were unable to get the code working, what you think is wrong, and how you might go about fixing it. The quality of such an explanation will be important to us in deciding whether to give you partial credit.

Homework Points: We expect that there will be a total of six assignments. The first assignment will be worth 50 points, the other five will be worth 100 points each.

8 Examinations

There will be one midterm and one final examination in this course. The midterm will be given during class on Thursday *February 17* and will be a closed-book examination.

Missed examinations count as zero credit. Except in the case of dire medical or family emergencies, no make-up examinations will be administered.

9 Course Grade

The raw score for the course is determined by a weighted average of each student's scores on homework assignments and examinations, according to the following formula:

Homeworks	50%
Midterm	20%
Final [*]	30%

The final letter grade for the course will be determined by evaluating each student's performance relative to that of the class as a whole. The raw scores are plotted as a histogram which is then used to determine the letter grade. Individuals whose raw scores lie near the borderline between letter grades may have their grade adjusted upwards or downwards based on such factors as attendance at lecture and recitation, earned extra credit, participation in recitation, and any special circumstances.

A grade of "incomplete" will be assigned only in the most extraordinary circumstances.

[*] You *must* take the final to pass the class. Failure to take the final examination will result in a failing grade.

10 Collaboration, Doing Your Own Work, Cheating

10.1 Collaboration

We encourage you to *discuss* methods and problems amongst yourselves. Learning from each other is potentially very beneficial. You should feel free to discuss at a *high-level* the basic ideas involved in functional programming, basic approaches to an assignment, nuances of SML, and so forth.

However, please do not write code together. Coding, specs, and written answers to assignments must all be "in your own words."

10.2   Doing Your Own Work

Any work submitted as a homework assignment or examination must be entirely your own and may not be derived from the work of others, whether a published or unpublished source, the worldwide web, another student, other textbooks, materials from another course (including prior semesters of this course), or any other person or program.

You may not copy, examine, or alter anyone else's homework assignment or computer program, or use a computer program to transcribe or otherwise modify or copy anyone else's files.

It is possible that we may reuse assignments in part or in whole from previous years, much like some courses repeatedly assign the same problems from a textbook. Do not read, copy, or discuss solutions from any prior semester of 15–212, even if your friends have such solutions handy and want to share them with you.

To repeat: All work submitted must be your own.

Score Penalties

We may sometimes run automatic code comparison programs (such as MOSS). These programs are very good at detecting similarity between code, even code that has been purposefully obfuscated. Such programs can compare a submitted assignment against all other submitted assignments, against all known previous solutions of a problem, etc. The signal-to-noise ratio of such comparisons is usually very distinctive, making it very clear what code is a student's original creative work and what code is merely transcribed from some other source.

If we run MOSS (or any similar program) and detect similarity between your code and any other source beyond the similarity expected due to code we have given you, you will receive a penalty as follows:

1. For the first such incident, the penalty will be a net score of -50 for the assignment.

2. For the second such incident, the penalty will be a net score of -100 for the assignment.

3. For the third such incident, the penalty will be a failing grade in the course.

Advice

Start early on assignments. One reason students copy from each other is that they do not take seriously the creative effort required to work through a problem set. Some students start the night before an assignment is due, discover that they are too tired to think properly or simply haven't mastered the skills to code properly in a new language, and in desperation resort to copying someone else's code. *DO NOT PUT YOURSELF IN THAT POSITION.*

For those of you who do start early, please be careful that no one copies your code. It is your responsibility to protect your programs, homework assignments, and examinations from illicit inspection or copying. You are expected to use the standard file system protection mechanisms to render your course materials unreadable to anyone other than yourself. Failure to do so may be regarded as evidence of improper collusion on homework assignments, subject to the penalties described above.

10.3   Cheating

We have tried to separate out momentary lapses of judgment, leading a student to submit work not entirely his/her own, from more serious forms of cheating.

For example, purposefully breaking into a computer system in order to obtain a solution to an assignment, either from another student or a member of the teaching staff, is an example of an extremely serious violation, likely to result in expulsion from the University. There are many other such examples. Any such violations will be handled in accordance with the University Policy on Cheating and Plagiarism.

11   Computing Facilities

Programming Language

All programs in this course will be written in SML using SML/NJ. You should familiarize yourself as soon as possible with the use of SML/NJ. There are various pointers and tutorials available underneath the course webpage. You are responsible for downloading and installing SML/NJ on your PC. If this is not possible, you should use the SML/NJ implementation available in the clusters.

Worldwide Web

The webpage for the course is accessible by the URL

> `http://www.cs.cmu.edu/∼me/212/`

The webpage contains links to important course material, such as the schedule, bulletin boards, assignments, handouts, code, and notes, as well as information about using SML. You should monitor the course webpage regularly.

Course Directory

The course directory is `/afs/andrew/course/15/212sp/`
Note the "`sp`" in the path, for "spring". The course directory contains the following two important subdirectories (plus a few others):

> `handin`   electronic submission directory (for turning in assignments)
> `www`   course material (assignments, notes, code, handouts, etc.)

The course material is accessible via the web through `http://www.cs.cmu.edu/∼me/212/`
Of course, you can also browse the directory tree `/afs/andrew/course/15/212sp/www/` by using the Andrew filesystem.

Electronic Hand-in

All programs are to be handed in using the electronic hand-in procedure. To do this, copy the appropriate file(s) into the following submission directory:

> `/afs/andrew/course/15/212sp/handin/`*userid*`/hw`*n*`/`

where *userid* is your Andrew user identification code and n is the assignment number. You must be authenticated with your Andrew user identification to have access to your submission directory.

The submission directory will be automatically scanned at *2:12AM* on the due date to retrieve whatever files are present. You may freely copy files into the submission directory at any time up to the deadline for the assignment; the copy present when the automatic sweep occurs at the submission deadline will be regarded as your submission for that assignment. Do not delete or modify your file after the sweep deadline. A second late sweep for late assignments occurs at *3:00pm* on the Thursday after the due date. If the timestamp on your file has changed it will be regarded as late.

Bulletin Boards

There are two bulletin boards devoted to this class, each with a title of the form

$$\text{academic.cs.15-212.}\textit{topic},$$

where *topic* is one of the following:

announce This bulletin board is used for official announcements by the teaching staff to publicize clarifications, corrections, or changes to the homework assignments, and to make announcements of changes to the class schedule. Anyone can read this bulletin board, but only the teaching staff may post to it. *Read this bulletin board regularly.*

discuss This bulletin board is to be used by the students to discuss homework assignments, and may be used to pose questions to the teaching staff regarding the homeworks. Anyone can read and post to this bulletin board.

The bulletin boards are your first line of communication with the teaching staff, and should be used in preference to electronic mail to make inquiries about the homework exercises. Electronic mail to the teaching assistants or professor should be limited to urgent communications requiring private handling or immediate response. Start early on your homework assignments. Do *not* expect the teaching assistants to be available for questions or read the bulletin boards late on Tuesday night before the deadline!

12   Taking Notes

Please take notes by writing or typing. Do not record or tape lectures electronically.

Specifically: No student may record or tape any classroom activity without the express written consent of Professor Erdmann. If a student believes that he/she is disabled and needs to record or tape classroom activities, he/she should contact the Office of Disability Resources to request an appropriate accommodation.

13   Advice

Learn by Doing

The homework exercises provide the most important part of your educational experience. Write out rough forms of your solutions, think them over, and write them out again in final form. Don't be afraid to try different approaches to the same problem.

Think Before You Hack

The homework exercises are designed to stress conceptual issues, and cannot ordinarily be solved just by sitting at a computer. The grading criteria stress readability of your code; it is *not* sufficient for a good grade that your programs work correctly. Omission of comments or invariants will also lead to deduction of points, so make sure your code is properly documented.

Start Assignments Early

Homework assignments are due approximately every other week and require a substantial amount of work. Start your assignments early so you can get help in time if you need it. Cluster space is limited and you may not be able to get on a machine close to the deadline.

Attend Lecture

The bulk of the course material is presented in lecture. *Most of the lecture material does not appear in the textbook*, so students are *strongly advised* to attend every lecture.

Attend Recitations

The recitations are the best time for you to review material, ask questions, and get personal help mastering the course material. The teaching assistants are friendly and enthusiastic. Supplementary material to the lectures will be presented in recitation; you are expected to attend and will be responsible for anything covered in the sections.

Keep Up

The course material is cumulative in nature and it is important to keep up. Experience has shown that once a student falls behind the pace, he or she tends to stay behind and suffers severely in the final grade. Keep up the pace, attend recitations and office hours for help, and see the instructor if you run into serious difficulties before they get out of hand.