CS 1410 — Section 20 — Fall 2010 — Syllabus

August 24, 2010

`http://www.eng.utah.edu/~cs1410-20/`

Lectures:	MWF 9:40–10:30, WEB L112
Lab:	Th 9:40–10:30, MEB 3225
Instructor:	Matthew Flatt
Office:	MEB 3458
Phone:	801-587-9091
Email:	`mflatt@cs.utah.edu`
Consulting hours:	TBA and by appointment
Teaching assistant:	Simon Williams
Email:	`Simon.Williams@utah.edu`
Consulting hours:	TBA and by appointment

1 Course Overview

This course will teach you how to begin with a problem statement and then systematically design a computer program that solves the problem.

The textbook is *How to Design Programs: An Introduction to Programming and Computing* by Felleisen, Findler, Flatt, and Krishnamurthi, MIT Press, 2001. The textbook is available online at

`http://www.htdp.org/`

and will soon be available in the bookstore.

Before using the main textbook, the course will start with an online early draft of the second edition, which is on the web at

`http://www.ccs.neu.edu/home/matthias/HtDP2e/index.html`

For most of the course, you will use a programming language that resembles the programming language Racket (which, in turn, resembles the programming language Scheme). You will use the DrRacket programming environment.

2 Enrolling in Section 20

Section 20 of CS 1410 will cover the same programing concepts as other sections of CS 1410, but through a different approach. Enroll in section 20 if

- you are particularly interested in topics such as operating systems and compilers, or
- you want something different from the other introductory programming courses.

Section 20 of CS 1410 is intended as the first part of a two-semester sequence, and a companion section of CS 2420 will be offered in Spring 2011. Section 20 of CS 1410 will use Racket as the main programming language for learning about program design, while the companion CS 2420 will use C as the main programming language for learning about machine-level details and data structures. Both courses will also use Java, and the courses will explicitly connect concepts learned through Racket and C to Java, but they will not use Java to the same extent as other sections of CS 1410/2420.

Enrolling in section 20 of CS 1410 does not oblige a student to enroll in the companion CS 2420 section; students who finish section 20 of CS 1410 will be prepared for the standard sections of CS 2420, although they may have to learn some Java details to synchronize with students in the main track. (Books and other references on Java are plentiful.) Enrolling in the companion CS 2420 section without having taken section 20 of CS 1410 is not recommended, since the course will build on concepts, terminology, and skills that are specific to section 20 of CS 1410.

3 Why Section 20 is Different

As computing has become ubiquitous and the roles of computer-science graduates ever more diverse, it is natural for students with different interests to start out in different ways.

Among the possible directions for a student of computer science, section 20 in some ways aims to serve a particularly traditional direction. Students who take section 20 should be well prepared for further coursework in operations systems, networking, compilers, and programming languages. Graduates with such traditional skills will always be in demand, and they are currently in short supply.

At the same time, the presentation in section 20 and its companion course will be non-traditional. Although the programming language in a course matters much less than the concepts that are taught in the course, the use of Racket and C (as opposed to Java or other languages that today's graduate would expect to use in a job) signals a different approach to the concepts. Because it is different, students with some programming background may find section 20 more challenging than the normal 1410/2420 track.

CS 1410 section 20 is based on *How to Design Programs*, a textbook for which the instructor is a co-author. Pairing the course with a machine-oriented data-structures course is unusual, but a similar approach is in place at the University of Waterloo.

4 Pragmatics

We will meet for lecture on Mondays, Wednesdays, and Fridays for 50 minutes in WEB L112. In lecture, I will use a projected laptop computer, sometimes with slides and sometime using DrRacket interactively. Slides and DrRacket transcripts will become available on the course web page. For lectures with slides, I will make every effort to put the slides on the web page well before lecture.

You will also meet with the TA and me for one hour on Thursdays in MEB 3225. This room is equipped with a computer for each student. In each section, we will guide you through a set of interactive programming exercises. The laboratory sections will be essential to your success in the course, and all students are required to attend each section.

The schedule web page is the central organizational tool for the course (see the URL at the top of this syllabus)

`http://www.eng.utah.edu/~cs1410-20/schedule.html`

The schedule page contains a tentative day-by-day schedule that will be revised as the semester progresses. Reading assignments are listed on the schedule page for each lecture day. Slides and DrRacket transcripts will be linked from the schedule page, as will programming assignments and solutions.

Most Fridays, I will post a programming assignment for the following week on the schedule page. It will be due on Friday (seven days later) before class by electronic handin. Homework submissions more than 8 hours late count as half credit when final grades are calculated, except for the first two such submissions that are within 48 hours late. No submissions more than 48 hours late will be graded (and the electronic handin program will reject such submissions). Homework is graded on the following scale: “check plus” for essentially perfect work worth 100%, “check” for flawed but acceptable work worth 80%, “check minus” for seriously flawed work worth 50%, and zero for unsubmitted or unacceptable work worth 0%.

The course staff (instructor and teaching assistant) will hold regular consulting hours each week, during which we will be available to help you with questions or problems. The consulting schedule has not yet been determined, and it will be posted on the web page.

There will be two midterms, held in place of lecture, and one two-hour final. The midterms will be on Monday, September 27 and early November. The final will be on Friday, December 17, 8:00-10:00 AM.

Your final grade will be based on the two midterms (15% each), the final (22%), the weekly assignments (45%), and attendance at and participation in the lab sections (3%).

5 Programming Assignments

Most of the programming that you do in this course will be supported by the DrRacket programming environment. DrRacket is available for your use in the College of Engineering’s labs. You can also obtain a free copy of DrRacket to install on your home computer. An installation link is available on the class web page. DrRacket runs under Windows, Mac OS X, and Unix/Linux.

Whether you choose to do your assignments at home or in the CADE Lab, you will need to submit your completed assignment through the **Handin** button in DrRacket. The **Handin** button will not be present the first time that you start DrRacket; you will need to download and install an extension. (The installed extension resides in your personal space on the filesystem.) Instructions for downloading and installing the **Handin** extension appear on the course web page. The first lab will also demonstrate the installation process.

When you click the **Handin** button, you must supply a username and password to the handin server. You will choose a username and password as part of the first homework assignment. When **Handin** communicates with the handin server, it uses SSL, so the connection is secure. Nevertheless, you should not use your CADE password for your handin account, because the handin server is maintained by the course staff, not CADE.

When you submit an assignment, the handin server inspects a few parts of your submission and rejects it if something is obviously wrong. These checks are intended to prevent basic misunderstandings and improve the feedback that you will receive from human graders. A human grader will inspect every submission and assign grades.

6 How To Approach this Class

For most of you, this will be your first class in computer science. For many of you, this will be one of your first classes at the University of Utah. Here is some advice on how to approach this class.

- Skim the relevant chapter in the textbook before you come to lecture, and read more carefully after the lecture. The lecture will not assume that you have read the current text chapter in advance, and the lecture should help you understand the context and key points of the chapter. Nevertheless, the lecture does not fully replace reading, because the lecture will omit fine points that are discussed in the text.

- Concentrate in lecture. The concepts that are presented in lecture are what's important; you will be able to find the details in the book. If you write down everything that is said and then try to figure it out later, you are wasting your time coming to lecture. Instead, think about what is being said. Try to answer all the questions that are asked, even if you only in your head. Raise your hand and ask a question when you don't understand something. Try to understand everything. Don't give up!

- Participate in the labs. You'll be sitting at a computer in a room with a teaching assistant and a few other students. Take advantage of the computer by trying things out. That way, you'll discover the things that you don't understand in a setting where there are plenty of other people to help you out.

- Respect the assignments. Some students expect that if they have done the reading, concentrated in lecture, and participated in the labs, then the assignments will be straightforward. What these students don't understand is that the assignments are designed to challenge you by requiring that you apply the concepts you have learned to new situations. The assignments will be your most important learning experience in the course; they will rarely be straightforward. You should start each problem set days before is is due. This way, you will have time to take a break when you get stuck.

Beginners are often surprised by the amount of human effort that has to go into designing, writing, and testing a program. Complaints from students about the amount of time required by introductory computer programming courses are universal. You should expect to spend three hours outside of class for every hour that you spend in class. In other words, you should expect to spend 12 hours per week reading, studying, and designing programs. Many of you will spend even more time. Please keep this in mind when setting up your schedule for the semester!

7 Getting Help and Information

The class web page, `http://www.eng.utah.edu/~cs1410-20/`, contains a variety of important resources, including the course schedule, course staff consulting hours and e-mail addresses, and links pertaining to the textbook and DrRacket.

When we need to get in touch with you, we will send e-mail to you at an account that you supply when choosing a handin username and password as part of the first homework.

There is a class mailing list `cs1410-20@list.eng.utah.edu` for announcements and discussion. We will use the class mailing list to send urgent messages, such as corrections to problem sets or changes in due dates, to everyone in the class. *Only the course staff will be allowed to send mail to this list.* You *must* subscribe to this list. The course web page contains a link for subscribing.

Questions about the homework usually should be sent directly to the instructor and TA. If we feel that the answer to your question would be of interest to the class at large, we will forward the question and answer to the class list.

We encourage you to seek us out whenever you need help, advice, or encouragement. We will always be available during our regular office hours, and you can make appointments for other times. Simple questions can often be answered by phone or electronic mail. Our consulting schedule will be posted on the class web page as soon as it is finalized.

8 Cooperation vs. Cheating

Working with others on assignments is a good way to learn the material and we encourage it. However, there are limits to the degree of cooperation that we will permit.

When working on programming assignments, you must work only with others whose understanding of the material is approximately equal to yours. In this situation, working together to find a good approach for solving a programming problem is cooperation; listening while someone dictates a solution is cheating. You must limit collaboration to a high-level discussion of solution strategies, and stop short of actually writing down a group answer. Anything that you hand in, whether it is a written problem or a computer program, must be written in your own words. If you base your solution on any other written solution, you are cheating.

When taking an exam, you must work completely independently of everyone else. Any collaboration here, of course, is cheating.

We do not distinguish between cheaters who copy others' work and cheaters who allow their work to be copied.

If you cheat, you will be given an E in the course and referred to the University Student Behavior Committee. If you have any questions about what constitutes cheating, please ask.

9 Students With Disabilities

Reasonable accommodation will gladly be provided to the known disabilities of students in the class. Please let the instructor know of such situations as soon as possible. If you wish to qualify for exemptions under the Americans With Disabilities Act (ADA), you should also notify the Center for Disabled Students Services, 160 Union Bldg.