COMP 210, Spring 2001
Lecture 1A: Administrivia & Introduction

This course is titled "Principles of Computing and Programming".

Philosophy

This is a course intended to teach you to design and implement small programs. It is not a course focused on programming mechanics but, instead, a course focused on teaching you a program design principles. COMP 210 teaches a data-driven approach to programming. This contrasts directly with most programming courses, which are syntax-directed—that is, they teach you a collection of syntactic constructs in some programming language (C, C++, Java, Ada, Fortran, Python, …). The principles that we teach scale to large programs, but we do not address all of the issues involved in writing large software systems.

You are here to learn program design. I am here to teach program design. You may already have experience programming computers. I know that I do. In this class, you must put aside your experience and focus on developing programs that conform to the design principles taught in the course. Because the course focuses on design, some answers that "work" are not acceptable in COMP 210–the course is about good systematic design, not about hacking together a solution that produces the correct output for some test cases. To succeed in this course, ask yourself (at every step) "What do the course design principles say that I should do?"

We'll defer an extensive discussion of "Why use Scheme?" for a later lecture.

The Web Site

http://www.owlnet.rice.edu/~comp210

is a critical resource for the course. All assignments and announcements will be posted there, as will copies of the lecture notes. I will post the notes after class. I will strive to do this in a timely fashion. The notes from last semester (which are very similar) are already posted.

Instructors:	Robert "Corky" Cartwright,	3104 DH,	cork@rice.edu
	"Zung" Nguyen,	3098 DH,	dxnguyen@rice.edu

Teaching Assistants:	Scott Schaefer	sschaef@rice.edu
	Jamie Raymond	jraymond@rice.edu
	Cheryl Hom	chom@rice.edu

Plus a phalanx of undergraduate laboratory assistants. (Check the web site for details.)

The Book:

Draft of a new textbook (MIT PRESS, 2001)
How to Design Programs, by Felleisen, Flatt, Findler, & Krishnamurthi

Available at the bookstore
Older draft is online; you want the newer draft, with corrections.

Requirements:

Details on the web site

- Attend class
- Attend lab section
  - Wednesday afternoon & Thursday afternoon
  - Taught by Dr. Nguyen, Scott Schaefer, Jamie Raymond, Cheryl Hom, Sajeev Verma (struan@rice.edu), and L. Almagor (lalmagor@rice.edu).
  - Covers some topics skipped in class, reinforces others
  - One question on each exam will be drawn from lab material!
  - Homework returned in lab section
  - First labs are this week (including today!); assigned lab sections will be posted by noon today on the course web site. Try to attend your assigned section this week, but attend some section regardless!
- Weekly homework
  - Handed out Wednesday due following Wednesday (start of class)
  - Some programming
  - Some paper exercises
  - Done in teams (more later on this subject)
  - Lab section should help prepare you for homework
  - Homework should prepare you for the tests

Half-credit homework this week …

- Homework assignments will include a final project –- a larger program
- Three exams
  - Roughly five week intervals
  - First one is in-class, others may be at night

Basis for Grading:

50% from homework	10% from first exam
(10% for final project)	20% from second exam
	20% from third exam

Homework Teams

We encourage you to work with a partner on your homework. You pick the partner; if you're having trouble finding one, we'll find you one.

There are two approaches to doing homework jointly:

- Do every step of the assignment jointly---alternating the person who does the keyboarding or writing every half-hour. Every word or keystroke should be generated jointly.
- Do the entire assignment separately, compare your final results, and hand in the best solution.

Hand in exactly one solution, with both your names on it. Each person can only participate in one team. You should not hand in two copies of the same homework.

If you **NEED** to work alone, come talk to either Zung or me.

If you don't get your work back in a timely fashion (1 to 2 weeks), come tell me!

Programming

All programming will be done in Scheme, using the Dr. Scheme programming environment. It is installed on OwlNet; when you run the registration software, it will set up your UNIX PATH so that Dr. Scheme is on it.

If you want a copy of Dr. Scheme for your personal computer (or your work computer), you can retrieve it from the World-Wide Web. Follow the pointers on the COMP 210 web page. It runs on almost any contemporary computer system: Windows 98, Windows NT, MacOS, many varieties of Unix (Linux, Solaris, etc.)

Dealing with the Graders

This may be the first class where you deal with undergraduate graders. Here a few pointers.

- Remember that it is a friendly relationship, but an adversarial relationship. Your grader wants to get his or her work done. Your grader is taking too many classes, just as you are.
- Your grader is a 19 to 23 year old (except Jamie). The graders have received no formal training in how to deal with your questions or problems, so they are feeling their way along and learning as they go.
- Use the anonymous evaluation method (on the web site, of course) to submit a complaint, if necessary. Those forms go to Dr. Nguyen and me.
- My goal is for all students to pass this course. But I expect you to work hard.

Keeping Up with Class

Check the web site regularly.
Check the newsgroup. Post your questions, comments, and musings.
Attend class and attend your lab section.
Read the text **BEFORE** we cover it in class.

ACTION ITEMS for Students

Get an OwlNet account
Visit the course web site and register for a lab section
Start thinking about a partner for the homeworks