CS 3 General Information

Spring 2007

Introduction

Welcome to CS 3, "Introduction to Symbolic Programming". This course will introduce you to computer programming, using the Scheme programming language (a dialect of Lisp). Examples and programming assignments will be drawn from non-numeric ("symbolic") applications.

The only prerequisite to CS 3 is high school algebra: more specifically, familiarity with variables and simple functions. We don't assume that you have had any previous computing experience. If you have done some programming, especially involving the technique of recursion, you should seriously consider taking CS 61A instead of CS 3. Many people, however, find CS 3 a good precursor to CS 61A.

You learn programming by doing it rather than by listening to us talk about it. Thus, CS 3 is organized to maximize your time designing and writing programs and experimenting with the programming environment. You'll work hard, but learn a lot. The lab sections and online activities are designed to help you get feedback at the time you need it; we hope to ensure that you're working productively rather than flailing around.

Staff

The instructor is Nathaniel Titterton (`nate@berkeley.edu`). His administrative assistant is Audrey Raya in 385 Soda.

There are three TAs that will run the four lab sessions:

- Alex Elium: `alexelium@gmail.com`
- Bobak Mortazavi: `bobakm@berkeley.edu`
- Roland Carlos: `rollins@berkeley.edu`

There are also readers who grade your homework and lab assistants who work with you during the lab sections. Contact information for them will be posted on the class web site shortly (http://inst.eecs.berkeley.edu/~cs3).

Books and other course material

There are two required books for the course. *Simply Scheme*, by Brian Harvey and Matt Wright (second edition, MIT Press, 1999), is available at local bookstores. The CS3 reader is available at Copy Central on the north side of campus, at 2483 Hearst Avenue, for around $12.

Class activities and scheduling

There is a single lecture each week. It meets in room 160 Kroeber on Mondays, from 5pm to 6pm. There will be no video streams of lectures available on the Internet.

You are each assigned to a lab section that meets twice a week for three hours each meeting. That's right, 6 hours a week.

section	Time
lecture	M 5-6pm
11	TuTh 11am – 2pm
12	TuTh 2 – 5pm
13	TuTh 5 – 8pm
14	WF 11am – 2pm

All sections will meet in room C30 Hearst Field Annex, between Barrows Hall and Bancroft Street. You must attend the section to which you are assigned; if you want to move to another lab section, you must get the approval of the TAs in charge of both the section you wish to leave and the one you wish to transfer to.

For most of the semester, the typical lab period will involve a variety of activities, the majority provided online. It will start with a short quiz based on topics covered on the homework or in the preceding class; each start-of-period quiz will count toward your course grade, and you have to take (the majority of these) in the lab classroom. Following this, you'll be reading, experimenting, brainstorming, evaluating each other's ideas, and sometimes working with partners. In the Thursday and Friday sections, T.A.s may set up impromptu discussion sections to clarify student confusion they've seen earlier in the week; please tell us what you would prefer on this issue, since we will respond to feedback. There will be three "mini-projects" during the semester, to which some of the lab meetings will be devoted. The last several weeks of the semester will be less structured, as you'll be working on a large programming project.

A short set of homework exercises will be typically be assigned at the end of each lab. The exercises will involve writing or analyzing programs and contributing to online discussions about typical programming misconceptions. Answers to the programming exercises and contributions to the discussions will be submitted online. You should expect to put in four or five hours of work per week outside of class. If you finish the online exercises early, you may leave early or work on your homework. Some of the work later in the course, along with the project, may be done in partnership with other students in your section. (Cross-section partnerships will not be allowed.)

Lectures will introduce and motivate new material or review confusion noticed during the previous week's lab sections.

There will also be three exams. Two 1.5-hour midterms will be during the normal lecture slot and the following hour, from 5-7 pm, on Monday, February 26th and Monday, April 9th. These *may* be in the lecture room, but may be in a different room. A two-hour final exam will be given on May 17th, from 8 to 11 am. The final will *not* take place in the lecture hall; its location is to be determined.

In CS 3, you will be using programming tools and course material devised by a research group of computer science and education researchers. To determine the effectiveness of these tools and material, we are gathering data on your background and performance, via questionnaires, interviews, and analysis of your work. You will be expected to complete several surveys through the course of the semester.

Computing

Most of your work for this course will be done in class in C30 Hearst Field Annex. Outside of class, you may work in any EECS lab room in which a lab section is not meeting. You may also work at home, of course! You may obtain a card key to work in C30 on weekends or late at night by going to 387 Soda Hall and filling out the relevant forms.

To do work for this course on your own computer, you will need to use a recent Firefox browser. Internet Explorer will not work (or at least work well).

You can run scheme by connecting to the lab machines via a secure telnet connection, or by getting a scheme environment for you home computer. Check the course website (http://inst.eecs.berkeley.edu/~cs3/) for more information on getting your computer setup properly.

Grading

The various course activities will contribute points to your grade as follows.

activity	course points	percent of total grade
project	30	15%
all mini-projects	24 (8 each)	12% (4% each)
all other homework	scaled to 24	12%
all on-line quizzes	scaled to 16	8%
random on-line step	6	3%
midterm exams	60 (30 each)	30% (15% each)
final exam	40	20%

You are expected to keep up with the classwork! There will occasionally be time devoted in lab to helping you catch up or solidify your understanding of the material. Homework assigned on Tuesday/Wednesday is due at the start of the Thursday/Friday lab section; homework assigned on Thursday/Friday is due at the start of the Tuesday/Wednesday lab section. You will at most earn half-credit for homework turned in after the due date but

before the next lab meeting; you will not earn any credit for any homework exercise submitted more than one class meeting after it is due

There will be more than 24 points worth of scaled homework points to earn; your homework score, however, will be capped at 24. As such, you can miss some homework assignments and still earn the full amount that homework can count towards your final grade.

Quizzes are online, and while they may be taken outside of the lab room, you will receive credit for at most four quizzes taken outside of your lab section. You will not receive any credit for quizzes taken after the lab-section in which they were assigned, whether or not you take them in the lab room or out of the lab room. As with homework assignments, there will be more than 16 points worth of scaled quiz points to earn, but at most 16 will count toward your course grade.

Each week one of the steps in the on-line materials will count towards your grade. Which step this is won't be known to you (or to us, beforehand), but will generally be a step for which you are asked to answer a question or create a file, although it may be a simple reading step for which we will log requests from the UCWISE server. The purpose of grading this is to encourage you to keep up-to-date on the lab materials; generally, the grades for these steps will consider whether you attempted it, rather than whether you did it well.

Your letter grade will be determined by total course points, as shown in the table to the right. There is no curve; your grade will depend only on how well you do, not on how well everyone else does.

Points	Grade
185-200	A+
165-185	A
155-165	A-
145-155	B+
135-145	B
125-135	B-
115-125	C+
105-115	C
95-105	C-
75-95	D
< 75	F

Incomplete grades will be granted only for dire medical or personal emergencies that cause you to miss the final exam, and only if your work up to that point is satisfactory. Copying and presenting another person's work as your own constitutes cheating. It will be penalized at least by a 0 on the work in question and notification of the incident to the Office of Student Conduct.

Approximate topic/activity schedule

Coverage will proceed roughly as follows. (Case studies are included in *CS 3 Readings*.) Make sure to check regularly the announcements calendar on the UCWISE course portal for the latest information, as topics and sequences may change somewhat.

week of ...	**lecture topics/activities**	**lab topics**	**reading**
Jan 15	(holiday, no lecture)	Beginning Scheme programming	*Simply Scheme* chapters 3-5
Jan 22	Early Scheme concepts (functions, words, sentences), conditionals	Conditionals	*Simply Scheme* chapter 6
Jan 29	Review; Conditional expressions; introduction to case studies	"Difference Between Dates" case study	"Difference between dates" case study, part I
Feb 5	Data abstraction	"Difference Between Dates" miniproject	
Feb 12	Introduction to recursion; Midterm Review	Recursion	*Simply Scheme* chapter 11 "Difference between dates" case study (recursion)
Feb 19	(holiday, no lecture)	Recursion	*Simply Scheme* chaps 12-13 "Roman Numerals" case study
Feb 26	*Midterm Exam 1*	More advanced recursion	*Simply Scheme* chap 14
Mar 5	Recursion	"Number Spelling" miniproject	
Mar 12	Introduction to higher-order procedures	Higher-order procedures	*Simply Scheme* chaps 7-9 "Difference between dates" case study (hof)
Mar 19	Higher order procedures	Higher-order procedures, tic-tac-toe	*Simply Scheme* chap 10, 15 "Change making" case study
Mar 26	(spring break)		
Apr 2	Higher order procedures	"Election" miniproject	
Apr 9	*Midterm Exam 2*	Lists	*Simply Scheme* chap 17, 19
Apr 16	Lists, background for projects	Lists, start on project	*SS* chap. 20 (recommended)
Apr 23	Advanced Lists, Project review	Work on project	
Apr 30	Course summary, Guest Lecture (CS at Berkeley)	Finish project (due this week)	
May 7	Exam Review	(no labs)	
May 17	*Final Exam (Thursday, 5-8pm, location TBA)*		

Alternative courses

Other courses in which you can learn to program include CS 3S, CS 4, IDS 110, and CS 61A.

CS 3S is the self-paced version of CS 3. Students may enroll for fewer than four units of CS 3S, in order to take only a portion of the course or to spread the complete course over more than one semester. (You need only three units of CS 3S to prepare for CS 61A, and two units would probably suffice.) The textbooks used in CS 3S differ from those we'll use, so if there's a chance you might want to switch, you should think about switching sooner rather than later. For further information, contact the Self-Paced Center, 642-9920, in room C10 Hearst Field Annex.

CS 4 is an introductory programming course for science and engineering students, and is similar to CS3 in scope. Programming exercises and class examples are drawn mostly from numeric applications (as opposed to the nonnumeric applications covered in CS 3). The programming language used is Java. It is not being offered this semester, however, and departmental politics may keep it on the shelf for a while.

In IDS 110, students use programming tools like spreadsheets and data base managers as well as write programs. Programming in IDS 110 is done using Javascript. IDS 110 also has scheduled lab sections, held in the Tolman Microcomputer Facility in 1535 Tolman. Course staff makes an effort to get students with similar interests to work together in discussion and lab sections. IDS 110 satisfies the computing course requirement for entry into the School of Business Administration.

CS 61A is the first of a sequence of courses aimed at students with a particular interest in computer science. Its prerequisite is computing experience roughly equivalent to the first half of CS 3; thus students with no previous experience often take CS 3 or 3S to prepare for CS 61A. Scheme is also used for programming in CS 61A. If you've done more than a little programming, particularly if your experience includes exposure to recursion, you should take CS 61A rather than CS 3. If you are thinking of becoming a computer science major, you should think about taking CS 61A. CS 61A covers more material, and in a more rapid fashion, than CS 3.