CMSC 15100: Introduction to Computer Science I

The University of Chicago, Summer 2016

Nicholas Seltzer

http://www.classes.cs.uchicago.edu/archive/2016/summer/15100-1

Welcome! In CMSC 15100 (informally *CS151*), we introduce a selection of major computer science topics through instruction in computer programming and various analytical techniques.

CS151 is designed for students intending to major or minor in the subject, although others are welcome.

The specific goals of the course are these:

- to understand solving computational problems in terms of identifying, and, when necessary, designing, relevant abstractions,
- to process data structures in several ways, most importantly by the technique of structural recursion,
- to learn to recognize and exploit common computational patterns through
- code organization and higher-order programming,
- to learn to use simple and polymorphic types as a powerful approximation
- of correctness in computer programs, and
- to analyze the efficiency of certain algorithms.

In pursuing these goals, students will become acquainted with a selection of classic data structures and algorithms. Broader, more technical treatments of these topics, in particular algorithm analysis, are presented in later undergraduate courses.

We use the Racket programming language in our studies. Racket is a dialect of Scheme, a language with a long history in the field of computer science generally and college-level instruction specifically. This year, for the second time, we will use the Typed Racket variant of Racket, for reasons we will articulate during the quarter.

Having completed this course, students will be able to use computer programming as a robust and efficient method for analytical problem solving and creative endeavors, and develop a sense of the relationship between computer programming and computer science. Students will discover, in future work, that the experience gained in this course applies to programming generally, in any programming language; that is, CS151 should not be thought of as a course in programming Scheme. Furthermore, students will have gained experience with some best practices in the discipline.

Piazza: Online Support
Register with *piazza*. Piazza is an online question-and-answer system that we use for that purpose as well as distribution of course materials on occasion. You will receive an email about piazza registration, with instructions, at your uchicago email address at the start of the quarter, so make sure you check that email address by June 20.

Instructors
Nicholas Seltzer
email: nseltzer@cs.uchicago.edu
office: Ryerson 177

Contacting Us
If you have questions about the course, and those questions are in a sense impersonal - that is, they are about course material or course logistics – we ask that you post those questions publicly on piazza, rather than contacting any of the staff members directly. This ensures you will receive the fastest, most consistent possible response from the staff. Since students usually have common questions, posting public questions is efficient for your classmates as well. As yet another advantage, it avoids duplication of work on the part of the staff. In cases where you have a question that is about your own personal situation and not relevant to the class as a whole, you may ask a "private question" on piazza, which is invisible to your classmates, or send email to your instructor directly.

A few piazza rules:

- Do not post any more than a snippet of code to piazza. Any post that says "Here's my hundred lines of code - what's wrong with it?" will be deleted immediately.

- Please do not post anonymously to piazza. Piazza posts are better, more thoughtfully written, and more courteous when the author is identified. We reserve the right to delete anonymous posts from piazza.

Lectures
Lectures are held Monday, Wednesday, and Friday 1:30 – 3:30 p.m. in Ryerson 251. The first lecture is on Monday, June 20; the last is on Friday, July 22. There will be no lecture on July 4.

Lab Sessions
Students must attend lab session each week. Lab is Wednesday 4 – 6 p.m. in the Computer Science Instructional Laboratory (also known as the CSIL). The CSIL is located on the first floor of Crerar Library. Attendance at the lab session is mandatory.

Office Hours
To be announced on the web once the quarter starts.

Optional Text
How to Design Programs, Felleisen *et al.*, ISBN 0-262-06218-6. Over the years and as our curriculum has evolved, we have departed from this text to such an extent that we no longer require you to buy a copy. Moreover, the full text of the book is available online at http://www.htdp.org free of charge. It is a useful reference and a good book in its own right, and, despite various differences with this course in its present form, has given our curriculum its basic shape.

Software
All the software we use in this course is available free of charge for all common platforms. We will mainly use *DrRacket*, available at `http://racket-lang.org`, and subversion. Macintosh and Linux users very likely have subversion on their machines already. Windows users will need to download and install Cygwin, and will be able to include subversion in their Cygwin installations.

Grading
Coursework is comprised of lab exercises, homework assignments, and exams. Each student's final grade will be computed according to the following formula: lab exercises 15%, homework 20%, quizzes 30%, final 35% each. What precisely constitutes an A, B, etc. will be determined by the collective performance of the class.

Homework
There will be weekly homework assignments. These will be assigned on Monday or Tuesday (usually) and will be due the following Monday.

Final
The final exam will be held during lecture on the last day of class.

Exam Accommodations
If you are a student who has special exam-taking arrangements with Student Disabilities Services (SDS), you must contact SDS to arrange for a time, date and proctor for the exams in this course. We will do our part by delivering the exams to the SDS office as needed.

If you do not have any arrangements with SDS but believe you should, please contact them directly (`https://disabilities.uchicago.edu/`) or talk to your college adviser to open the conversation.

To be clear, we, your instructors, are not in a position to judge who needs what sort of special accommodations; we leave those judgments to the professionals. SDS, along with the college administration, makes these determinations, and we abide them. Therefore, although you may ask one of us about making an accommodation for an exam, we will simply refer to your adviser or SDS in such cases.

If you have a conflict with an exam because of a personal event (dinner with friends, night at the theater, meeting with a study group, etc.), we expect you to make every effort to put the exam first and plan around the exam rather than the other way around. If your exam conflict is important and cannot be postponed or rescheduled, we will accommodate you as best we can.

Late Work
Deadlines in this course are rigid. Since you submit your work electronically, deadlines are enforced to the minute. Late work will not earn credit.

(We will occasionally accept late work in the case of special circumstances, when those circumstances are extraordinary.)

Academic Honesty
In this course, as in all your courses, you must adhere to college-wide honesty guidelines as set forth at `http://college.uchicago.edu/policies-regulations/academic-integrity-student-conduct`. The college's rules have the final say in all cases. Our own paraphrase is as follows:

1. Never copy work from any other source and submit it as your own.
2. Never allow your work to be copied.
3. Never submit work identical to another student's.
4. Document all collaboration.
5. Cite your sources.

We are serious about enforcing academic honesty. If you break any of these rules, you will face tough consequences. Please note that sharing your work publicly (such as posting it to the web) definitely breaks the second rule. With respect to the third rule, you may discuss the general strategy of how to solve a particular problem with another student (in which case, you must document it per the fourth rule), but you may not share your work directly, and when it comes time to sit down and start typing, you must do the work by yourself. If you ever have any questions or concerns about honesty issues, raise them wit your instructor, early.

Advice
Writing code that does what it is supposed to do can be joyful, even exhilarating. By contrast, fighting for hours with broken code is misery. We would like you to experience more of the former and less of the latter. Work methodically. Start your work well ahead of time. Beyond a certain point, it is not profitable to be stumped. If you have made no progress in some nontrivial chunk of time, say, one hour, it is time to stop and change your approach. Use one of our many support mechanisms to get some assistance. We will help you get going again when you are stuck.

Last revised 2016 June 19 12:00pm