CMSC 15100: Introduction to Computer Science I

The University of Chicago, Autumn 2020

Adam Shaw and Timothy Ng

`https://www.classes.cs.uchicago.edu/archive/2020/fall/15100-1`

Welcome! In CMSC 15100 (informally *CS151*), we introduce a selection of important computer science topics and techniques by way of instruction in computer programming and software development. We do so by guiding students through a series of interesting, colorful, dynamic, challenging exercises in building increasingly complex software systems.

CS151 is designed for students intending to major or minor in computer science. Having said that, students in other areas who would like to sample something of this elegant subject, for whatever reason and with whatever degree of curiosity, are pointedly welcome. We assume no particular academic background on anyone's part; any student working in any area of study should be able to keep pace in this course with sufficient effort and commitment. Computer science is closely aligned with mathematics, and while no special mathematical expertise is needed for CS151, a reasonable degree of comfort and familiarity with high school mathematics, up to calculus, is basically assumed.

The specific goals of the course are these:

- to understand how to solve computational problems in terms of identifying, and, when necessary, designing, relevant abstractions,
- to process data structures in several ways, most importantly to process recursive data structures with structural recursion,
- to learn to recognize and exploit common computational patterns through code organization, code factoring 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. We use the *Typed Racket* variant of Racket, for reasons we will articulate during the quarter. In a few words, the emphasis this course places on *types* makes for a richer, more horizon-stretching, more interesting, more rigorous, more varied curriculum than otherwise. You will find that having worked under a rigorous typing discipline is beneficial even when one works in an untyped setting.

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 (an activity, useful and profitable, which can be done poorly or well) and computer science (an academic discipline with mathematical foundations and timeless insights). Students will discover, in future work, that the experience gained in this course applies to programming generally, in any programming language; that is, CS151 must not be thought of as a course in Racket (Scheme) programming, specifically. Furthermore, students will have gained experience with best practices in the discipline, including many of immediate practical value.

Piazza: Online Support

You will need to register with *Piazza*. Piazza is an online question-and-answer system that we use for that purpose as well as distribution of most course materials. 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 that time.

Instructors

Adam Shaw, email: `ams@cs.uchicago.edu`.

Timothy Ng, email: `timng@uchicago.edu`.

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. We prefer to be contacted on Piazza to streamline our communications.

A few Piazza rules:

- Do not post any more than a snippet of code to Piazza. Any post that consists of "Here's my hundred lines of code – what's wrong with it?" will be deleted immediately. Reflect on and focus your question before you write it.
- Please do not post anonymously to Piazza. Piazza posts are better, more thoughtfully written, and more courteous when the author is identified. Furthermore, we are committed to building a community where questions are asked openly and enjoy respect from all constituencies — fellow students and teachers alike. We reserve the right to delete anonymous posts from Piazza.

Lectures There will not be lectures this quarter in the usual sense, but we will produce a selection of videos to be viewed online as fits your schedule. We expect you will view each video within a day or so of its publication. Prerecorded videos will be mixed with live interactions with your instructor. Real-time interactions will occur roughly weekly throughout the quarter.

Lab Sessions Students must register for and attend lab sessions each week. We will have a signup mechanism for you that will enable you to choose a time that works for you. Lab sessions will be held online and guided by one or both of your instructors. Attendance at the lab session for which you are registered is mandatory.

Office Hours The office hours schedule will be announced on the web once the quarter starts. In addition to the office hours we provide, the College Core Tutor Program employs computer science tutors Sunday through Thursday nights from 7pm–11pm, starting in the second week of the quarter. The College Core tutors are specifically charged with support of CS151 students, and are often alums of the course themselves.

Schedule of Topics by Week (subject to change)

Optional Text *How to Design Programs*, Felleisen *et al.* Even though this book is the undisputed primary ancestor of this curriculum, over the years and as our curriculum has evolved, we have departed from the text to such an extent that we no longer refer to it directly all that often. The full text of the book is available online at `https://www.htdp.org` free of charge; it is also available as a physical book if that interests you. It is a useful reference and a very good book in its own right.

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 `https://racket-lang.org`, and *git*. Macintosh and Linux users very likely have git on their machines already; Windows users might need to take a few steps. We will provide some assistance for various platforms once the quarter starts.

Grading Coursework is comprised of lab exercises (initiated at lab sessions,

Week-Day	Lecture	Topics
1-W	1	expressions, types
1-F	2	images, booleans, conditionals
2-M	3	functions, tests
2-W	4	structures, local definitions
2-F	5	recursion, pattern matching
3-M	6	unions, images
3-W	7	monomorphic binary trees
3-F	8	polymorphic binary trees
4-M	9	linked lists
4-W	10	higher-order programming I
4-F	11	higher-order programming II (parametric polymorphism)
5-M	12	EXAM higher-order programming III (λ, currying)
5-W	13	universes
5-F	14	lambda, build-list
6-M	15	universes, modality, more λ
6-W	16	graphs I
6-F	17	graphs II
7-M	18	graphs III
7-W	19	search trees I
7-F	20	search trees II
8-M	21	sorting
8-W	22	accumulators and tail recursion I
8-F	23	accumulators and tail recursion II
—	—	*Thanksgiving break*
9-M	24	vectors I
9-W	25	vectors II
9-F	27	EXAM onward to CS152

discussed above), homework assignments, projects, and two exams. Each student's final grade will be computed according to the following formula: lab exercises 20%, homework 20%, project work 20%, exams 20% each. What precisely constitutes an A, B, *etc.* will be determined by the collective performance of the class. Having said that, earning 92% assures at least an A-; 82% a B-; 72% a C-; 62% assures a D. We reserve the right to adjust that scale, but we will not adjust any of those thresholds upward (that is, a 92% will not quietly become a B+).

We know your grades are important to you. Nevertheless, when a grader has deducted points on one of your pieces of work, your first response should be to do your best to understand the reason for and the instructional lesson to be learned from that deduction. In other words, a deduction is constructive criticism and is therefore an opportunity for you to improve. Please try to view it that way.

Any requests for us to reconsider the way a piece of work was graded must be communicated to us within one week of having received the evaluated work.

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

Projects There will be a longer multipart project during the latter part of the term. The final part of this project will be due during exam week.

Exams There will be two exams for all students. At the time of writing this syllabus, we have not decided the exact mechanism by which the exam will be administered, but it will, of course, be something you will be able to participate in with any computer, wherever you are. Nevertheless, please circle the following dates on your calendar: October 26 and December 4. Those are the dates planned for the exams. Any changes to these dates will be announced well in advance.

Exam Accommodations If you are a student who has special exam-taking arrangements with Student Disabilities Services (SDS), please let us know so we can discuss accommodations. 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 start a 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 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.

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. However, to allow for whatever inevitable deadline-related difficulties arise for individuals throughout the quarter, we will drop the lowest scoring homework exercise and the lowest lab exercise from our final calculation. Please note that no project work will be dropped: only one homework and one lab.

We will occasionally accept late work in the case of special circumstances, when those circumstances are actually 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/advising/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.

If you break any of these rules, you will face tough consequences. The minimum consequence is that the work in question will be considered an undroppable 0. Beyond that, any student who is determined to have participated in a major violation of academic honesty will not be allowed to withdraw and will receive a course grade no higher than a C. Furthermore, if determined to have violated our expectations of academic honesty, the student will not be allowed to record a pass/fail grade, and any pass/fail designation the student may have requested and been granted up to that point shall be retroactively canceled.

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 with your instructor, early.

Advice Writing code that does what it is supposed to do can be enlivening, joyful, even uplifting. By contrast, fighting for hours with broken code is the opposite: discouraging, frustrating, and generally miserable. We would like you to experience more of the former and less of the latter.[1] 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.

2020 Sept 27 4:30pm. This is revision 2 of this document.

[1] In particular, we never want you to come to us and say "I spent six hours fighting with git." Do that under no circumstances whatsoever! This campus is full of people who know how to negotiate version control systems generally, and git specifically. Please seek help from any one of them.