CS110 Course Information

Instructor: Jerry Cain

Email: jerry@cs.stanford.edu

Phone: 415-205-2242 (between 7:00am and 9:00pm)

Office: Gates 192

Office hours: TBD

Lectures: MWF 1:30pm – 2:50pm, Gates B01.

Units: 5 units. Only matriculated graduate students may register for fewer than five. The requirements are the same for all students, including those who take the course for 3 or 4 units.

Course Assistants: Mahesh Agrawal, Chase Basich, Kushaagra Goyal, David Hershey, Audrey Ho, Raunak Kasera, Michael Painter, Michael Precup, Pamela Toman. CS110 CA's attend lectures, lead discussion sections, hold office hours, evaluate program submissions, and grade exams. Be glad they're here, because virtually all of them have either taken CS110 before, CA'ed CS110 before, or both, and they know the material so well they already know what your questions are going to be.

Prerequisites: The prerequisites for the course are CS107 and CS103. You need to be familiar with the C and C++ programming languages, Unix/Linux, `make`, `Makefiles`, `gcc/g++`, `valgrind`, `gdb`, and have some experience with basic computer architecture (x86 as it's taught in CS107, or exposure to some other architecture with the confidence and ability to pick up x86 as I reference it).

We'll be coding in a mixture of C and C++ throughout the quarter. We rely on C, because the libraries we rely on to interface with system resources are written in C. We rely on C++, because the projects become large enough that I prefer to go with a more mature language that supports encapsulation and generic programming better than C does. You should understand pointers, dynamic memory allocation (`malloc/realloc/free`), and C strings perfectly. You should understand C++ classes, methods, references, templates, and C++'s `new` and `delete` operators. There are some features of the C++11 version of the language that you're not expected to know, but you should have enough programming maturity to pick those features up and search the web for reference materials as needed.

Readings:

- The first required textbook is Computer Systems: A Programmer's Perspective by Bryant and O'Hallaron, either the 2nd or 3rd edition. Both CS107 and CS110 teach from a subset of the B&O textbook, so I've arranged for a custom reader—with just the chapters we need—to be sold at the Stanford Bookstore. Of course, if you want to purchase the entire textbook, you're free to do that as well, though you'll need to buy from Amazon or some other online retailer.

- The second required textbook is Principles of Computer System Design: An Introduction by Jerome H. Saltzer and M. Frans Kaashoek. Stanford has university-wide digital access to the textbook, and because it's free, I assume most of you would just prefer to go with the free online version. Again, you're welcome to purchase a hardcopy if you'd like. It's available for purchase on Amazon. (If you're living off campus, then you should read this so you can view the textbook online.)

Website: **http://cs110.stanford.edu** is your new favorite website. There you'll find all reading assignments, lecture slides, homework assignments, and the full list of office hours. If you have any suggestions on how to make the course website even better, feel free to send Jerry an email.

Software: The shared UNIX workstations (`myth` machines in Gates B08, available via `ssh`) provide all of the development tools needed for lecture examples and all of the assignments (although I may occasionally reference the more powerful `corn`, `rye` and `barley` machines to clarify the impact that more processors and larger caches have on execution).

I recognize we live in a laptop world, and that you'd prefer to code on your own machines and eventually port it over to the `myths` (where you submit your work and we grade your assignments unless otherwise stated in the assignment handout). However, I strongly urge you to code, test, and debug directly on the `myths` via `ssh`. The `myths` are outfitted with a hip version of `g++` that supports the advanced C++11 features we'll be relying on almost immediately.

Forums: We are using Slack for class discussion forums (and a direct link is available on the CS110 website). When you have a question that might be of interest to other students, please post it there for a speedy response. Please note that you should never include snippets of code directly from your own homework submissions, since that's code sharing with others and a huge no-no. Sign up for the CS110 Slack team by hitting http://cs110.slack.com/signup once you know you're taking the class.

Grading: You can take the course for a letter grade, or **CR/NC.** The course grading is divided between several programming assignments, discussion section participation, a midterm during Week 6, and a final exam. The grade breakdown is:

- Programming assignments: 40%
- Discussion Section Participation: 5%
- Midterm Exam: 20%
- Final Exam: 35%

To receive a passing grade, you must pass **both** of my exams. Restated, if you fail either exam, that exam effectively counts 100%. (I will be clear what a passing grade is for the midterm so you can withdraw from the course before the deadline.) If you're taking the course **CR/NC**, you need a C- or better to get the **CR**.

Midterm Exam: The two-hour midterm is tentatively scheduled for Friday, May 12th during our normal lecture time. The midterm is **closed book, closed notes, and closed computer**, save for the fact that I'll allow you to prepare and refer to a single 8.5"-by-11" cheat sheet containing any information you can cram onto each of its two sides. I will include all relevant prototypes and type definitions (C functions, C++ classes, etc.) on the exam, and you're welcome to ask a staff member for a function or method prototype if I don't include it. If you can't take the midterm during that time because of a competing class, then you can arrange to take the exam sometime earlier that same day by emailing me directly.

Final Exam: The three-hour final exam is scheduled for Monday, June 12th at 3:30pm, location TBD. If you're leveraging the SCPD system, taking two classes at the same time, and watching my lectures online, then and only then can you take the final exam on Monday, June 12th at 7:00pm instead. To be clear, you're expected to take the final Monday June 12th at 3:30pm unless you have a conflict with another class's final exam, in which case you can take it during the following time slot at 7:00pm. If you need to take the final exam during the later time, then email me as soon as possible.

The final is also **closed book, closed notes, and closed computer**, but I'll allow you to bring and refer to the **four sides** of the **two** 8.5"-by-11" cheat sheets you prepare ahead of time. As with the midterm, I will include all relevant prototypes and type definitions (C functions, C++ classes, etc.) and you're welcome to ask a staff member for any function or method prototype.

Discussion Sections: This quarter, I'll be experimenting with a new teaching model where I lecture on Mondays and Fridays, leaving Wednesdays open for CS110 discussion sections. Each and every one of you needs to sign up for a discussion section and attend the vast majority of them. Each 50-minute section will have between 20 and 30 students, and most of them will be offered at various locations at 1:30pm and 2:30pm. You'll be expected to bring your laptop or pair up with someone who has one, as the section will be a combination of written problems, coding exercises, and software engineering tips to ensure that you understand the material and how to successfully complete the assignments with minimal drama. **Note that I'll be lecturing on Wednesday of Week 1**, but after that, Wednesday will be CS110 Discussion Section Day.

Late Policy: We understand everyone here is busy. But falling behind on assignments just leads to more problems, and it interferes with our ability to grade and turn around grades in a timely manner.

All programming assignments are due at the stroke of midnight, and you'll always be given at least 7 days to complete any one of them. If you need to submit an assignment after the deadline, you still can. But doing so places a cap on the maximum number of points you can get for that assignment, depending on how late you submit.

- If you submit an assignment before the deadline, then you can potentially get 100% of the points. Woo.
- If you submit an assignment after the deadline, but within 24 hours, you can get at most 90% of the points. This doesn't mean we impose a 10% penalty regardless of your final score. It means that all scores between 91% and 100% are demoted to 90%, but all other scores are left as is. If your assignment is really very broken at the time you would normally

need to submit, then you have a good reason to take an additional 24 hours to increase your score, as it can only go up. If your program is pretty much working with no obvious flaws, then you probably should submit it as is by the published deadline.

- If you submit an assignment between 24 hours and 48 hours after the deadline, you can get at most 60% of the points.
- You can never submit an assignment more than 48 hours after the deadline.
- **The first assignment must be turned in by the published deadline, without exception.** We want to grade your first assignment as quickly as possible so you can get feedback well before your second assignment falls due.

Note that requests for extensions will be denied unless something truly extenuating—a family emergency, severe illness—presents itself, in which case you can send me an email and I'll do what I can to make your life easier while being fair to everyone else.

Disabilities: Students who need an academic accommodation based on the impact of a disability must initiate the request with the Student Disability Resource Center (SDRC) located within the Office of Accessible Education (OAE). SDRC staff will evaluate the request with required documentation, recommend reasonable accommodations, and prepare an Accommodation Letter for faculty dated in the current quarter in which the request is being made. Students should contact the SDRC as soon as possible since timely notice is needed to coordinate accommodations. The OAE is located at 563 Salvatierra Walk, and their phone number is 650-723-1066.

Honor Code: Although you are encouraged to discuss ideas with others, your programs are to be completed independently and should represent truly original work. Whenever you obtain help (from current or previous CS110 students, the CA's, students in other classes, etc.) you should credit those who helped you directly in your program, e.g. in a program comment, type ”The idea to use a `mutex`-guarded linked list of file descriptors came from a discussion with my CS110 CA, Mike Precup”.

Any assistance that is not given proper citation is considered plagiarism, and a violation of the Stanford Honor Code. To be even more specific, you are not allowed to collaborate on the coding of your programs, nor are you allowed to copy programs or even minute snippets of programs from other students, past or present. The following four activities are among the many I consider to be Honor Code violations:

1. Looking at another student's code.
2. Showing another student your code.
3. Discussing assignments in such detail that you duplicate a portion of someone else's code in your own program.
4. Uploading your code to a public repository (e.g. `github.com` or `bitbucket.com`) so that others can easily discover it via word of mouth or search engines. If you'd like to upload your code to a private repository, you can do so on `bitbucket` or some other hosting service that provides free-of-charge private hosting.

Unfortunately, the CS department sees more than its fair share of Honor Code violations. Because it's important that all cases of academic dishonesty are identified for the sake of those playing by the rules, we use software tools to compare your submissions against those of all other current and past CS110 students. While we certainly don't want to create some Big Brother environment, we do need to be clear how far we'll go to make sure the consistently honest feel their honesty is valued.

If the thought of copying code has never crossed your mind, then you needn't worry, because I've never seen a false accusation go beyond a heated conversation. But if you're ever tempted to share code—whether it's because you don't understand the material, or you do understand but just don't have enough time to get the work done—then you need to remember these paragraphs are here.