CS110 Course Information

Instructors: Chris Gregg and Phil Levis
Email: cgregg@stanford.edu / pal@cs.stanford.edu
Offices: Gates 201 (Chris), Gates 409 (Phil)
Phone: 857-234-0211 (Chris)
Office hours (Chris): Tue. 9:00 - 11:00am and Thurs. 10:00am - 12:00pm, or by appointment, in Gates 201
Office hours (Phil): Wed. 2:00pm - 3:00pm, or by appointment, in Gates 409

Lectures: Mondays and Wednesdays from 11:30am – 1:20pm, 320-105. **During the *first week only*, we will hold class on Friday from 11:30am-1:20pm.**
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: Kristine Guo, Wantong Jiang, Clara McCreery, Peter McEvoy, Armin Namavari, Frits van Paasschen
CS110 CA's attend lectures, lead discussion sections, hold office hours, evaluate homework submissions, monitor the online forums, and grade exams. Be glad they're here, because all of them have either taken CS110 before, CA'ed CS110 before, or both. They know the material so well they already know what your questions are going to be.
Prerequisites: Formally, the prerequisite for the course is CS107. Informally, you need to be familiar with the C and C++ programming languages, Unix/Linux, **make**, **Makefile**s, **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 we reference it).

We'll be coding in a mixture of C and C++ throughout the quarter. We rely on C, because the libraries needed 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 language that supports encapsulation and generic programming better than C does. You should understand pointers, dynamic memory allocation (**malloc/realloc/free**), and C strings well enough that you're not intimidated by them. You should understand C++ classes, methods, references, templates, and C++'s **new** and **delete** operators. There are C++ features 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.
The first assignment, which goes out this week, is a systems programming assignment that should bring all relevant CS107 and software development skills back into rotation. If you haven't taken CS107 and/or programmed in C and C++ before, but you're able to work through this first assignment without drama, then you're more than qualified to take CS110.

Readings:

- The first required textbook is <u>Computer Systems: A Programmer's Perspective</u> 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 <u>Principles of Computer System Design: An Introduction</u> 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 online version. Again, you're welcome to purchase a hard copy 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 when not on campus.)

Website: http://cs110.stanford.edu is your new favorite website. There you'll find all reading assignments, lecture slides, homework assignment specifications, and the full list of office hours. If you have any suggestions on how to make the course website even more useful, then drop Chris an email and we'll talk.

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

It's true that we live in a laptop world, and we suspect you'd like to code on your own machines and eventually port everything over to the **myth**s (where you submit your work and we grade your assignments). However, we strongly urge you to code, test, and debug directly on the **myth**s via **ssh**. The **myth**s are outfitted with a hip version of **g++** that supports the advanced C++ features we'll be relying on almost immediately, and it's better to incrementally develop there instead of porting everything over ten minutes before a deadline.

Student Forums: We're using [Piazza](#) for the class forum. If you have a question that might be of interest to other students, please post there for a speedy response. Note, however, that you should never include snippets of code directly from your own homework submissions, since that's code sharing and a huge no-no.

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: 60%

- Discussion Section Participation: 5% (adjusted as per discussion section attendance policy outlined below)
- Midterm Exam: 15%
- Final Exam: 20% (adjusted as per discussion section attendance policy outlined below)

To receive a passing grade, you must pass **both** of our exams. Restated, if you fail either exam, that exam effectively counts 100%. (We'll be clear what a passing grade is for the midterm so you can withdraw from the course well before the deadline.) If you're taking the course **CR/NC**, your final grade is computed precisely the same way as it is for those taking a letter grade, and you need a C- or better to get the **CR**.

Discussion Sections: This quarter, we'll lecture on Mondays and Wednesdays, leaving Fridays open for CS110 discussion sections. Each and every one of you needs to sign up for a discussion section and attend all of them to receive all section participation points for the quarter. Each 80-minute section will have between 15 and 20 students, and most of them will be offered at various locations on Fridays at 11:30am. 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 we'll be lecturing the Friday of Week 1.**. But otherwise, Friday is CS110 Discussion Section Day. To accommodate those who hold conflicts during what would have been our normal lecture time on Fridays, we'll offer a small number of discussion sections on Thursdays at various times.

By the way, your discussion section grade is 100%. If you attend all discussion sections, then that perfect discussion section grade counts for 5% of your final grade. For each discussion section you miss, we reduce the section participation contribution to your final grade by 1%, and transfer that 1% to your final exam. So, if you miss two discussion sections for any reason whatsoever, then your final exam counts for 22% of your grade, and your section participation grade (itself still a 100) only counts for 3% of your final grade. If you miss five or more discussion sections for any reason, then your final exam counts for 25% of your final grade, and your section participation grade doesn't count at all. This policy is an experiment that grants some flexibility to those who feel they can learn the material just fine without discussion sections.

Midterm Exam: The two-hour midterm date will be announced soon. The midterm is **closed book, closed notes, and closed computer**, save for the fact that we'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. We'll 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 we failed to 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 directly emailing Chris or Phil.

Final Exam: The three-hour final exam is scheduled for **Friday, December 13th from 8:30 - 1:30am**. If you're taking a second class whose final exam competes with ours, we may offer an alternate exam time on the same day, but it will not be possible to take the exam on a different day. To be clear, you're expected to take the final at 8:30am unless you have a conflict with another class's final exam, in which case you should email either Chris or Phil to see about taking it at a different time on the same day (please do this as soon as possible)

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 review them 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, depending on how late you submit.

- If you submit an assignment before the deadline, then you can potentially get 100% of the points. Seems right.
- 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 alone. If your assignment is severely 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 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 Chris an email and I'll do what I can to make your life easier while being fair to everyone else.

Students with Documented Disabilities: Students who may need an academic accommodation based on the impact of a disability should initiate the request with the Office of Accessible Education (OAE). Professional staff will evaluate the request with required documentation, recommend reasonable accommodations, and prepare an Accommodation Letter for faculty. Unless the student has a temporary disability, accommodation letters are issued for the entire academic year. Students should contact the OAE as soon as possible since timely notice is needed to coordinate accommodations. The OAE is located at 563 Salvatierra Walk, and their website is located right here.

Honor Code: Although you are encouraged to discuss ideas with others, your programs are to be completed independently and should represent fully 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 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, Sarah Radzihovsky."

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 even minute snippets of programs from other students, past or present. The following activities are among the many we 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**) 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 **github** 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 be identified for the sake of those playing by the rules, we exercise our right to use software tools to compare your submissions against those of all other current and past CS110 students, including any we might find online. 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 we've never seen a false accusation go beyond a single conversation. But if you're ever tempted to share code—whether it's because you don't understand the material, or because you do but just don't have enough time to get the work done—then you need to remember these paragraphs are here.

Final Notes. We want you to succeed in the course, and we will do everything in our power to help you do well. The course is *rigorous* and requires a significant time commitment. Plan on starting the assignments early and working diligently on them. Come to office hours, and do the readings. If you have any concerns about your progress, email Chris as soon as you recognize that you are falling behind. This is even more important if you are a graduating senior who needs CS110 to graduate -- Spring seniors have failed the course numerous times before and we do not make exceptions for students simply because they are going to graduate.

This course has been carefully cultivated by Jerry Cain for many quarters, and the general outline and almost all of the assignments, labs, and course material is credited to him.