

CS106X Course Information

Instructor: Jerry Cain
E-Mail: jerry@cs.stanford.edu
Cell phone: (415) 205-2242
Office: Gates 192
Office hours: Mondays, 9:00 a.m. – 10:30 a.m., and by appointment

Lectures: MWF 11:00 – 11:50 a.m.
Skilling Auditorium (and available via scpd.stanford.edu, details TBA)

Prerequisites: CS106X is the more advanced of the two courses teaching basic programming abstractions and algorithms. CS106X is designed as an alternative to the more sensibly paced CS106B, because many students—self-taught hackers, exceptionally strong CS106A students, and AP Java graduates—prefer a more intense treatment in the company of other aficionados.

AP Java and CS106A are all about basic programming practices—expressions, control idioms, decomposition, algorithmic thinking, class design, object orientation, simple inheritance, and basic client use of arrays, lists, and maps. CS106X teaches advanced programming and abstraction techniques, worrying first about C++ language mechanics and eventually focusing on topics such as recursion, C++ lists and maps, and the implementation techniques used to build custom, dynamic data structures.

Sections: In addition to our weekly lectures, you'll also attend a weekly discussion section. The person leading your particular discussion section will be the one grading all of your assignments. There are several discussion sections to choose from. Take a moment to visit <http://cs198.stanford.edu/section> anytime between Thursday, September 24th at 5:00 p.m. and Sunday, September 27th at 5:00 p.m. and state your preferred section times. Come Sunday evening, our computers will work their magic and come up with a master section assignment that will hopefully suit everybody. SCPD students need to visit the web site as well, for no reason other than to register for the televised section, which is tentatively scheduled for Friday mornings at 11:00 a.m.

CS106L: You may have noticed what appears to be a companion lab course, numbered CS106L. It is certainly related to CS106X, but it is completely optional, and you needn't sign up for it if you don't want to or don't have time. CS106L spends time on advanced features of the C++ language that

we just don't have time for in CS106X. In particular, you'll learn the ins and outs of C++'s I/O system, and you'll learn all about the template algorithms and data structures. CS106L is being taught by a veteran section leader who knows C++ just about as well as anyone, so it's definitely worth your time if you expect to be coding in C++ beyond this quarter. CS106L is offered TTh, 4:15 – 5:05 in 260-113, and it starts tomorrow.

Readings: The class textbook is the course reader *Programming Abstractions in C++* by Eric Roberts (with C++ edits by Julie Zelenski). The course reader should already be available at the Stanford Bookstore, so on-campus students can go purchase a copy in person, and remote students can telephone the bookstore to arrange for a copy to be sent.

In addition to the reader, we distribute a good number of handouts, chockfull of additional material and examples. After class, extra copies of the handouts will be placed in the handout bins on the first floor of the Gates Building in the B wing. And electronic copies of all the handouts will be available on the CS106X web site, so that you can print your own copy.

Software: Programming assignments can be written on either Macintosh or Windows PC computers, using either XCode (on the Macintosh) or Visual Studio C++ (on the PC). More information on these two programming environments will be provided in separate handouts.

Mailing List: There is a class mailing list that will be used for important announcements that just can't wait until lecture. All students enrolled in CS106X are automatically subscribed to the `cs106x-aut0910-students@lists` list. The list server is in touch with Axxess, so if you're signed up for the course, you're probably on the mailing list. Please make it a point to sign up for the course as soon as possible, since I tend to send a good number of announcements out during the first week or two, and I don't want you to miss out.

Programs: There are six or seven programming assignments, and it's possible I'll throw in a written problem set for color. The assignments are serious projects, and they get more and more difficult as we cover the more advanced material. The only way to learn programming is to work at it, so expect to spend lots of time in front of a computer. Your assignments are graded interactively in a one-on-one session with your section leader. In general, section leaders will return your assignments within one week of the day you submit it.

Exams: There will be one 120-minute exam during the quarter and a comprehensive 3-hour exam given during the final exam period. Both are

open note, open reader, closed textbook, closed computer exams. The two exam dates are as follows:

Midterm:	Wednesday, November 4 th	7:00 - 9:00 p.m.
Final Exam:	Friday, December 11 th	8:30 - 11:30 a.m.

Because the midterm is being administered at night, I'm happy to accommodate anyone who can't make it then. If you can't take the midterm when everyone else is, then email me at least a week prior and we'll schedule a separate time. You should expect to take your midterm sometime earlier in the day.

I'm much less flexible about alternate final exams, since the December 11th time is dedicated specifically to CS106X. The Educational Affairs wing of the CS Department insists that all students take a course's final exam at the same time during the normally scheduled slot. If you're taking another course that has a final exam on Friday, December 11th at 8:30 a.m., then you should check to make sure that any competing final exams can be taken at another time.

Grading: Your final grade will be computed as follows:

Programs	25 %
Midterm	25 %
Final	50 %

To receive a passing grade, you must complete satisfactory work in all areas. In particular, if you do not pass the final exam, you will not pass my class regardless of your performance on the assignments.

Assignments are graded on a bucket system, because we want to de-emphasize the grade and have you focus more on the assignment and our feedback. However, in the interest of transparency, here is a rough description of the various categorizations that apply to everyone:

- + Given to an exceptionally strong submission that not only meets the requirements, but exceeds them in some very significant way. In general, I see less than 2% of assignments getting +s. The + is ultimately recorded as a 100 in the spreadsheet, since it's clearly A+ work.
- √+ Given to a solid submission that gets the job done and contains at most a very small number of trivial errors. In general, about a 35-40% of assignment submissions get the √+, which roughly maps to an A in terms of a letter grade.

- √ Given to a good submission that gets most of the job done and contains a one or more major errors, or a significant number of minor ones. In general, about 45-50% of assignment submissions get a √, which roughly maps to a B+ come spreadsheet time. This is easily the most contentious grade, because CS106X students don't like getting B+s. However, when we give them, it's because the program wasn't as good as it could have been and there were more impressive submissions.
- √- Given to a submission that does most of the work, but contains enough problems that a √ isn't warranted. In practice, √-'s are rare in CS106X. The √- typically maps to a B- when everything gets put into a spreadsheet.

There are other bucket grades, but they are super duper rare enough that I don't need to describe them. You should try to not get them.

The midterm average tends to be high—someone around 85 – 90%—and the final exam average tends to be in the 70 – 80% range. I don't curve the exams themselves. I only curve the final distribution as I'm assigning grades. In general, I make sure that between a third and a half of the students get some form of an A, and all but a handful of the rest get some form of a B.

Fair Access Students who may 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 (phone: 723-1066).

Late policy: The pace of this course makes it difficult for students to catch up once they have fallen behind, so I encourage you to submit all of your assignments on time. Of course, we're all busy people, so I'm happy to accept late work, provided you keep up with the material. Here's how I handle lateness: You get **three** free late days, and you consume one late day any time you hand in work between 1 second and one class period late. (If, for example, an assignment is due on a Friday and you need to take a late day, you can hand the assignment in on Monday instead.) You can use at most one late day on any particular assignment. Once you've used your three free late days up, they're gone and you can't get any more. You can still hand in your work, but I levy a 10% penalty per assignment for each late day you use beyond the three free ones. My recommendation: plan to get the work done on time, and use a late day if it doesn't come together by the deadline.

Incompletes: I will only grant incompletes in the event of a severe illness or a family emergency that makes it impossible for a student to finish his or her work during the regular academic quarter. Unless your circumstances are particularly extenuating, you must clear the incomplete within 30 days of the final exam, and you must manage to get all outstanding work done.

Honor Code: Although you are encouraged to discuss ideas with others, your programs are to be completed independently and should be original work. Whenever you obtain significant help (from other students, the section leaders, students in other classes) you should acknowledge this in your program write-up, e.g. "The idea to use insertion sort instead of quicksort to alphabetize the list of names was actually my section leader's idea." Any assistance that is not given proper citation will be considered 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 parts of programs from other students. The following three activities are among the many considered to be Honor Code violations in this course:

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.

Unfortunately, the Computer Science Department sees much, much more than its 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 reserve our right to use software tools to compare your submissions against those of all other current and past CS106 students. It isn't our intent to create a Big Brother environment with spies and surveillance cameras. We're not creating that at all. We're just being clear about 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 witnessed a false accusation. 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 this paragraph is here.