

CS193j — Programming in Java

The Class

CS193J is a "Java as a second language" course, designed to introduce a skilled C programmer to the object-oriented paradigm, the Java programming language, and the built-in Java packages.

CS193j is an intermediate course for people with a basic programming background. It is not for complete beginners -- people without significant prior programming experience. Neither is it an advanced course appropriate only for CS majors (those people should consider CS108, also offered this quarter).

Topics to be covered include object-oriented programming (classes, objects, messaging, inheritance), Java language features (interfaces, exceptions, packages, concurrency, garbage collection), the built-in packages (lang, util, io, networking, graphics), understanding applications and applets, security and verification, implementation and the Java virtual machine. The coursework will consist of regular programming assignments and a final exam. By the end, you will have explored the Java language and OOP design principles in solving a variety of problems. The only way to truly learn a language, especially one as rich as Java, is by writing programs, so do expect to invest time into your assignments in order that you may successfully master the language and its built-in facilities.

The Student

The prerequisite for the class is programming and problem solving at the CS106B/X level in C. Thus it is assumed you are proficient in C with a good understanding of arrays, strings, pointers, and dynamic memory allocation and have well-developed debugging skills. In addition, you should be comfortable with recursion, you should be able to construct and manipulate linked data structures (lists, queues, trees, graphs, etc.), you should be familiar with abstract data types and you should have understanding of various algorithmic techniques such as searching, sorting, and hashing. You will be expected to produce elegant, well-decomposed, commented solutions as taught in the CS106's. We will provide some basic style handouts to lower bound the level of polish we are looking for.

Course Page

Being a hip, modern class, CS193j will have all sorts of material maintained at the course page..

<http://www.stanford.edu/class/cs193j/>

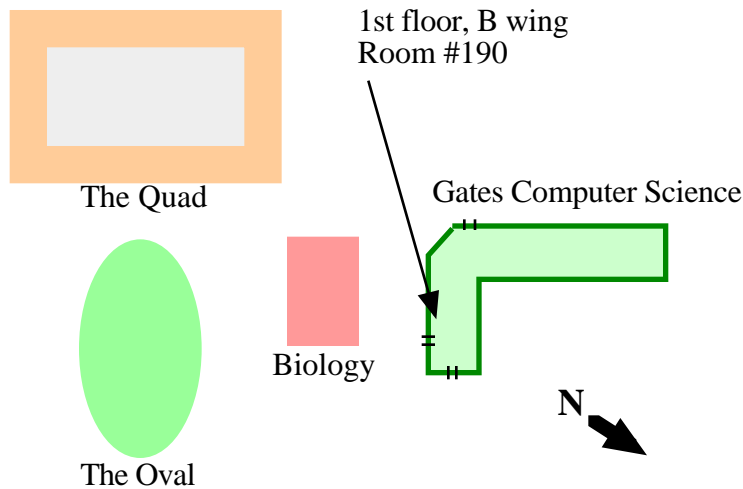
If you are looking for anything having to do with CS193j, the course page will be a good place to start. In particular, handouts, links to online resources, compiler information, and homework materials will all be linked off the course page.

Instructor

- Nick Parlante
nick.parlante@cs.stanford.edu
<http://www-cs-faculty.stanford.edu/~nick/>
 (650) 725-4727

Nick's Office

Gates 190. On the first floor, facing the green Biology building.



Nick's Office hours:

The exact staff office hour scheduling will be published separately, but as a general rule, I'm around and available much of the day Mon-Fri. Feel free to call or stop by. My office hours are often empty, so feel free to come by and get individual help (this is one of many areas where shyness is not going to benefit your college experience).

Books

There are many, many Java books that more-or-less cover our topics. There is no required text, but many people like to have a written text to refer to. Two good choices are: *Core Java, Vol 1 Fundamentals* (in the Stanford bookstore), by Horstmann, and *Java 2: The Complete Reference*, by Schildt. These are fine books, and there are many other reasonable choices available, so you can follow your personal preference. In any case, there will be much material to read in online form (naturally, this will be linked off the course page).

Handouts

Often there will be some sort of outline or source code to go with a day's lecture. These will be available at the course page in PDF format at least 60 minutes before lecture. SCPD students: we send out handout #1 in paper form; please get the others from the web page.

Questions: cs193j@cs.stanford.edu

Please send questions to the staff-collective email address cs193j@cs.stanford.edu. We'll maintain FAQs on the course page for common questions. For debugging questions, try to describe what you've done to narrow down where the bug might be. Short questions work great by email, longer questions probably need to be handled in office hours.

Java

CS193j uses Java. For the most part, this frees you to build on whatever platform you wish that supports Java (Solaris, Windows, MacOS, MacOS X, Linux), although you will need to turn in your work on the leland Unix systems since that's where we do the grading. There are links on the course page explaining how to set up the various compilers. You can use the "VNC" system to run things on leland from elsewhere. On leland, VNC is installed in `/usr/class/cs108/bin` along with some other handy utilities. You may use Java 1.3, Java 1.2, or Java 1.1 with Swing and the collections classes retrofitted. We will emphasize properly structured, cross-platform code (nothing platform specific). We will test your code against the current leland Java install, which is currently Java 1.3. There's a section on the course page that explains how to run Java on the various platforms. Stanford has a site license for CodeWarrior 7 which runs on Macs and PCs and has a decent Java environment. See the course page for more information.

Grading and Exams

The course grading is divided between 4-5 programming assignments, one assigned every 1-2 weeks, and 1 final exam. The approximate grade breakdown is assignments 60%, final exam 40%. To receive a passing grade, you must complete passing work for both the programs and the exams. The final exam will be Thu Mar 22nd, 3:30-6:30 pm. SCPD students may take the exam remotely.

CR/NC teams

The class is offered with either letter or CR/NC grading options. According to university policy, a CR/NC student needs to earn a course grade of C- or better to pass. Course requirements generally don't differ for students under the CR/NC option, however, we will allow CR/NC students to work in a teams of two on the programming assignments if they would like. Working with a partner doesn't cut the work in half, but it does give a bit of relief and perhaps will make the work a little more fun. This seems appropriate for those of you who are taking this class not for a grade or requirement, but just to learn the material for yourself.

If you choose to work with a partner, you may work with only one other student per assignment and both students must be CR/NC. In between assignments, you can switch partners or switch to working on your own. The exams will be taken individually. You may try the newsgroup, `su.class.cs193j`, to find a partner if you need one.

SCPD Students

The initial default due date for SCPD students and all other non in-lecture-in-person-the-traditional-way students is the exact same deadline as everyone else. The handouts and materials go up on the web at the same time planet wide, and the digital videos are available soon after. TVI or other delayed SITN students should work out a mutually acceptable arrangement with their TA to account for their tape delay, once we assign the TA's. We usually shoot for around 3 days.

Late Submissions

Instead of having to ask for extensions on a catastrophe by catastrophe basis, everyone gets three calendar "late days" to extend the due dates of any of the weekly assignments (except possibly the last one). In keeping with the all electronic, 24-hours a day theme of the post-Internet world, late days will be measured in straight calendar days with no distinction for weekends or holidays. All homework deadlines will be at midnight Pacific time.

These late days are intended to deal with the ordinary events of student life, both frivolous and serious: 2 midterms that day, inadvertently spent all night playing Quake, disk crash, med. school interview, illness, started way too late...After your late days are used up, late work loses pretty quickly— about a half a letter grade per day. Come and see me in person in exceptional circumstances. Note that disk failure, network outages and other computer problems probably *do not* represent exceptional occurrences. Hoard your late days "just in case," or spend them early and fly with no parachute— it's up to you.

In the grade database, every homework is recorded with both its score and the number of days late it was turned in. The Great Spreadsheet Reckoning at the end of the quarter figures out if you went over your late day budget.

Giving students their own late-day supply seems more fair since all the students are on the same footing. However it means you now need to make your own decisions about when to use a late day, and when to just turn in what you have. The late days should allow you to do a better job and hopefully learn more in the cases where your schedule gets disrupted. However, three late days do not provide too large a cushion. You should plan to finish your homeworks on time and reserve the late-days for real problems.

Honor Code

You are free to discuss ideas and problem approaches with others, but all the work you hand in should be your own creation (or the creation of your team for a team project). **In particular, sharing or copying code is not OK.** If you feel a particular bit of collaboration may have crossed the line, just clearly cite what

help you got and from whom in your project's Readme. You can never get in Honor Code trouble if the help is clearly credited.

There are tools we may use that do an extremely good job of finding plagiarism within the submissions. Also, I am very tired of having students cry in my office, so don't do something we'll all regret.

If we are using the Foo module, and you find the key 8 lines in the docs or in a book that describe how to call the Foo module best, it's fine to use those lines without comment in your Readme. Using OOP libraries is filled with episodes like that. If I have asked you to implement the Bar module, copying the 200 lines you found that implements Bar is not ok.

Schedule

A rough outline to give you an idea of the topics we are likely to cover —expect adjustments and changes as the quarter develops. We start with fundamental Java structures and the OOP paradigm for the first few weeks, followed by 2 weeks of GUI programming. After that, the topics will vary a little more depending on interest and how things are going, but we will certainly get through concurrency, I/O, and basic networking.

Week	Topics
1	Introduction, Java syntax, simple language built-ins
2	Basic OOP: class, object, message, method, constructors, access control, overloading
3	More objects: inheritance, overriding, polymorphism, class variables and methods, abstract base classes, OOP design, interfaces, class objects, inner classes, packages
4	Finish basic Java and OOP. Start Swing GUIs. Components, drawing, layouts, graphics.
5	Listeners, buttons, mouse tracking.
6	Finish GUIs. Start threads and concurrency. Threads, critical sections, synchronization.
7	Finish Threads. Start I/O, streams, networking.
8	Finish networking.
9	Misc advanced topics, such as XML, VM implementation, and performance techniques
10	More advanced topics.
11	Final Exam: Thu Mar 22, 3:30-6:30 pm