Course Placement Information

Much of this handout written by Eric Roberts.

Learning to program computers unlocks the full power of computer technology in a way that is both liberating and exciting. At the same time, programming is an intellectually challenging activity that comes easily to very few people. Taking a programming course requires a great deal of work and commitment on your part, but you will not be able to master programming without putting in that level of work somewhere along the way. The payoffs, however, are quite real. If you make the effort and keep up with the material, you will be able to make computers do amazing things. As you begin your journey in computing at Stanford, it would serve you well to decide what are the best options for you along this route.

What introductory programming course should I take?

Almost 70 percent of Stanford students take a programming course at some point during their undergraduate career. To accommodate students with a wide range of backgrounds and interests, the CS department offers several different introductory classes:

- *CS105—Introduction to Computing*. This course is designed as a general-education introduction to computer science. It attracts an audience of approximately 300 students a year, many of whom take the course primarily to meet the Stanford distribution requirement in applied mathematical science GER:2b. If your only interest is in meeting that requirement, CS105 is likely to be the most appropriate course. Like any programming course, CS105 requires a reasonable amount of work, but not as much as CS106A. CS105 is taught Fall and Spring quarters.
- *CS106A—Programming Methodology*. This course is the largest of the introductory programming courses and, with its enrollment at ~600 a year, one of the largest courses at Stanford. The course is taught using the Java programming language, which is widely used in academia and industry. CS106A is explicitly designed to appeal to humanists and social scientists as well as hard-core techies. In fact, most CS106A graduates end up majoring outside of the School of Engineering. The course requires no previous background in programming, but does require considerable dedication and hard work. The course is also appropriate for students with some programming background, but who are not advanced enough for CS106X or CS106B. CS106A is offered every quarter, including summer.
- *CS106B—Programming Abstractions*. This course is the natural successor to CS106A and covers more advanced programming topics such as recursion, algorithmic analysis, and data abstraction. This course is taught using the C++ programming language, which is similar to both C and Java. In the past when both CS106A and CS106B were taught in C/C++, the coupling between the two classes was very tight and it was unheard for students to take CS106B without having completed our CS106A (we recommended 106X instead). We believe it is feasible for students to start in CS106B nowadays, but we don't yet have much experience on how well this works. CS106B could be appropriate for a student who scored well (4 or 5) on the CS AP exam and has sufficient familiarity with good programming style and software engineering issues (at the level of CS106A) to use this understanding as a foundation on which to tackle advanced topics. Please talk to me if you are thinking of enrolling in CS106B without having taken CS106A and we can discuss whether this is the right choice for you. CS106B will be taught all quarters this year.

- *CS106X—Programming Methodology and Abstractions (accelerated).* This course combines 90 percent of the programming concepts in CS106A and CS106B into a single course and is taught using the C++ programming language. In order to get through that much material in a quarter, CS106X moves at an incredible pace. If you've had previous programming experience, this class is an excellent way to learn the core ideas behind programming, get coverage of programming style and software engineering issues, and further hone your C++ skills. If you haven't done much programming before, you should take the CS106A/B sequence instead. Don't let anyone tell you that "real engineers take CS106X." These days, most computer scientists and engineers start with CS106A, where they do just fine. The last thing you want to do is get in over your head. CS106X will be offered Fall and Winter quarters this year.

I already know how to program—should I skip the intro courses?

Many students entering Stanford today have had considerable programming experience in high school or from their own independent work with computers. If you are in that position, the idea of starting with a beginning programming course—even an intensive one like CS106X—seems like a waste of time. Your perception may in fact be correct. In my experience, there are a handful of students in each entering class who should start at a more advanced point in the sequence. For most of you, however, the right place to start is with the CS106 series. Many high-school computing courses are quite weak and provide very little background in modern software engineering techniques. By taking CS106X, you will learn how the CS department at Stanford approaches programming and get a solid foundation for more advanced work. If you're unsure as to where you should start the programming sequence, please come and talk to me in person.

Other courses

As computers become more powerful, it is possible to use them for increasingly sophisticated tasks without engaging in programming, at least in a traditional sense. Programming and using computers are two very different skills, and so if your goal is knowing more about how to *use* computers, you should investigate the following course:

- *CS1C—Introduction to Computing at Stanford.* This one-unit course is offered in the autumn quarter only and is designed to ensure that you have a level of "computer literacy" that will allow you to function effectively in Stanford's highly computerized environment. It does not teach programming at all. In the CS106 courses, we assume that you are familiar with basic computer use (e.g., word processing and using a web browser.) If you want to learn more, contact your local Residential Computer Coordinator (RCC.)

If, on the other hand, you already have programming experience and want to learn about specific languages and tools, here are a few other courses to consider. Note that CS106 courses teach programming fundamentals. Although we happen to use Java or C++ languages, these courses are not focused on just the mechanics of the languages.

- *CS193C—Client-Side Internet Technologies.* This course offers an introduction to web browser client technologies, including the JavaScript programming language.
- *CS193D—C++ and Object-Oriented Programming*. This course thoroughly covers the C++ programming language and the object-oriented paradigm. If you really want to learn the ins and outs of this complex language, this class is it. (CS106B/X will use some C++ features but not cover the language in depth).
- *CS193I—Internet Programming.* This course explores the core technologies behind modern Internet tools from a programmer's perspective.

- *CS193W—Microsoft Windows Programming*. This course covers the fundamentals of programming for the Microsoft Windows platform, focusing on the use of the Microsoft Foundation Class (MFC) package.
- *CS193N—C# and the .NET Platform*. This course covers the C# programming language and the Microsoft .NET framework, including topics such as database access and internet-based programming.