Introductory Computer Science for Non-science, Non-engineering Students

Kenneth Steiglitz, Professor of Computer Science, Princeton University

Computer fear is a common malady among students. New users typically must learn long commands with difficult syntax, or lock themselves into a computer that is initially easy to use but limits users to an underpowered environment. When the Computer Science Department at Princeton University decided to offer an introductory computer science course for non-science, non-engineering students, NeXT immediately became the platform of choice. The department felt the platform would help students overcome their initial fears via the NeXT environment: an easy-to-use graphical interface, as well as all the power of a traditional UNIX environment.

Goals of the Course

1. To teach theoretical and practical subject matter, and to relate the two whenever possible.
We want to expose students to the fundamental concepts and problems in computer science, not just teach them to use computers. The course is divided into two threads: theoretical and practical. The theoretical thread covers subjects such as computability (using the Halting Problem), big-oh notation, and NP-completeness. We associate this to the practical portion of the course (the labs) as much as possible. Big-oh notation is examined by looking at different sorting algorithms. Students relate to this because they spend one lab programming a selection sort algorithm in C. The NeXT environment provides a multitude of examples, such as SortingInAction, that graphically illustrate different techniques. These examples are teaching and learning aids that help integrate the various parts of the course, which is the way we feel computer science should be taught—many threads, all related to one another, interweaving to form one body of knowledge.

2. To allow students to try what they learn in a hands-on lab environment.
Lecturing offers a limited understanding when teaching computer science. Exploring practical problems helps to forge a better understanding of material and is far more rewarding for students.

The lab manual is divided into labs and a set of "CheatSheets" to explain details of applications and concepts. There are nine labs, each focusing on a specific topic:

- Introduction to the NeXT environment
- Communications—UNIX mail, NetNews, ftp, telnet, NewsGrazer, NeXTmail
- Graphics—PostScript programming, drawing applications (e.g. ,TopDraw)
- Presentation—troff and combining multiple applications to produce more complex documents. Students use any applications or tools they like, and CheatSheets are available for WriteNow, Improv, TopDraw, Diagram!, Yap, Scene, and Grab.
- Sound—recording and playing sounds from UNIX, editing sounds in SoundEditor, recording to and from cassette using Digital Ears, mixing with rt.app and SoundWorks
- Programming in C—"Hello World," printing out your name in a loop, building a simple application using Interface Builder, basic input using scanf
- Selection Sort—writing a selection sort in C. As expected, this is by far the most difficult lab for most students.
- ein—Using the ein application to write sound filters and analyze sound spectrograms
- Information Processing—grep, Digital Librarian, regular expressions

There is also an introductory section on setting up a user account and basic NeXT concepts such as windows and GUI versus command line interface.

It is important to point out that there is no formal UNIX lab with COS111. We view UNIX as a method of accomplishing tasks—not something that should be learned as an end in itself. On the other hand, we also see UNIX as an extremely powerful tool serving many functions. Therefore, we use UNIX in nearly all of the labs and expect students to use it. It is not surprising, however, that students overwhelmingly prefer NeXTstep applications.

3. To be authentic in the presentation of subject matter through real world examples.
We do not want COS111 to be a survey course. We want students to feel they are dealing with real problems and not just doing useless exercises. For example, students are expected to produce a useful

program (a selection sorting program) when learning C. Although not the best sorting algorithm, it does a fair job and is easy to understand. In addition, it gives the students a feeling that they are actually doing something useful by solving a real problem. We do not want to create an overview course that merely talks about programming. We want students to try everything, and see for themselves how it all works.

4. To teach through interest, not force.
We want students to understand what computers can do for them. This is far more interesting than a traditional introductory programming course, that emphasizes making the computer do what you want, rather than learning what the computer can already do. Fortunately, NeXT computers offer a tremendous amount to the students. Sound, for example, is an instant attention-getter. By the second lab we expose the students to voice attachments within NeXTmail. Then we introduce sound waveforms and sound editing, and encourage further exploration. This is in distinct contrast to requiring students to write one specific program after another.

We designed the labs so there is a standard, required section that each student must complete. This is usually fairly easy to accomplish in the lab period, however, so most students had time left over to explore on their own. Suggested topics and pointers are found in the "Above and Beyond" section of each lab.

Course Structure

Due to the large amount of information to cover, we spent a considerable amount of time designing the course's format. Lectures are held twice a week and cover theoretical material. Lectures also cover some practical material. Lab periods are held twice a week, and we recommend that students attend at least one period. If students prefer to do the labs on their own, we do not require them to attend at all, though they are still responsible for completing the lab. Teaching assistants are on duty during the lab periods to answer questions.

Small group meetings with the teaching assistants are optional. There are two meetings per week in addition to the teaching assistants' office hours. Students can come to these sessions to ask questions, receive extra help, find out more information about the labs, or go over theoretical material in an informal group setting.

The Platform of Choice for Introductory Computer Science

It is important to point out that the entire set of course materials are presented on a single platform—NeXT. Putting so many topics into a one-semester course for beginning computer users makes time an extremely valuable commodity. Teaching the fundamentals of more than one platform to non-science majors is not an option. NeXT allowed us to create a course about computer science—not computer hardware.

We do not think that any other platform could work as well as NeXT. The variety of tasks that it can handle makes it the best general use machine for this course. Sound, graphics, programming, communication, UNIX. . . NeXT has all of them in one platform. Without NeXT, we would have been forced to either limit the scope of the course or to use multiple hardware platforms.

For more information on the use of NeXT computers for COS111, please contact:

Professor Kenneth Steiglitz
Department of Computer Science
35 Olden Street
Princeton University
Princeton, NJ 08544-2087
Internet: ken@princeton.edu; Bitnet: kenn@pucc

Professor Kenneth Steiglitz and three Princeton University undergraduates, Jeff Blum '94, Rich Feit '94, and Jonathan Thompson '93, developed COS 111 during the summer of 1991. Steiglitz oversaw the content of the entire course and was primarily responsible for the theoretical portions. Blum, Feit, and Thompson focused on the laboratory manual—writing all of the labs and their corresponding CheatSheets. Did we succeed in creating an introductory Computer Science course that can interest non-science majors? According to the student reviews, 14 out of 17 respondents said they would recommend the course to a friend. Hopefully, this success will spur such courses to be taught at other universities.