Diverse teams of people bring more perspectives, experiences, and skills to any situation. Such qualities can lead to better problem-solving as well as better need-finding, since diverse perspectives help to identify problems that affect more groups of people.

Unfortunately, computer science programs tend to suffer from a lack of diversity. In particular, they are overwhelmingly male. I care about this issue deeply and have been following it for several years. Some institutions have made major inroads in CS gender diversity in recent years, including my home institutions of Stanford and UC Berkeley. As of 2014, the percentage of CS majors who are women at UC Berkeley was 21%, up from 11% just a few years earlier [5]. At Stanford, CS is now the most popular major for undergraduate women, who comprise 30% of all CS majors [3]. And at Harvey Mudd, an overhaul of the introductory computer science curriculum contributed to an increase in the number of women CS majors from 12% to 40% in just five years [4]. These success stories offer lessons that I believe can be applied elsewhere:

**Treating introductory classes as opportunities to hook students, not weed them out:** Berkeley offers CS 10, “The Beauty and Joy of Computing,” a course aimed at attracting non-majors to computer science. It uses a visual, block-based programming environment before introducing students to Python. The course has been hugely successful and already boasts an enrollment that is majority female [1]. At Harvey Mudd, the introductory computer science course, CS 5, offers two parallel tracks—one for students who have no prior programming experience (Gold), and one for those who do (Black)—that students can freely switch between [2]. I especially like this approach: the two tracks allow students to engage with material at a level that is comfortable for them, and the freedom to switch tracks at any time helps keep the Gold track from seeming ‘remedial.’ These courses are excellent models, but I have been even more impressed with Berkeley’s CS KickStart program, a week-long pre-academic-year program for incoming female students. By immediately surrounding students with a supportive group of others like them who are interested in but uncertain about computing, CS KickStart deals a huge blow against ‘imposter syndrome’—the feeling that many new computer science students have that “everyone else has been programming since they were eight years old.” I would advocate establishing a similar program at any university.

**Immediate and persistent opportunities to apply computing to real-world problems:** Both Stanford and Harvey Mudd promote undergraduate research immediately—both schools have summer programs that recruit students as soon as their first year. All Harvey Mudd students must complete a senior capstone project, or ‘clinic,’ in which they work with and contribute some product of value to a sponsoring company or research lab. And Stanford offers CS 210, a two-quarter course where students develop a software project in collaboration with an industry partner. Experiences such as these can help dispel the perception that computing is stuffy, boring, or too abstract by connecting programming practice with real-world issues students care about.

**Student-to-student mentorship:** Stanford’s introductory computer science courses feature small discussion groups led by more senior undergraduates, called Section Leaders. Berkeley’s CS KickStart program, initially founded by two graduate students, is now run by undergraduate alumnae of the program. I believe that providing such leadership opportunities fosters a sense of community and of investment in the department. More importantly, students in such mentorship positions can serve as role models and can contribute significantly to making the department a safe and welcoming space.
When new students, especially women and other under-represented groups, can look up to senior students like them who have gone through the same introductory programs and have assumed leadership roles, I believe that imposter syndrome has a harder time taking root.

Working in computer graphics, as I do, provides extra opportunity to promote diversity. Graphics is intrinsically interdisciplinary, involving algorithms and data structures, high-performance hardware, human visual perception, aesthetics, and many areas of physics, to name just a few related fields. Rather than focusing myopically on core graphics pipeline issues (such as transformations, rasterization, and shading), introductory graphics courses can be explicitly designed to attract a diverse, interdisciplinary student body. Stanford’s CS 148, as taught by my advisor Pat Hanrahan, embodies this philosophy. It broadly surveys graphics-related topics, from typography to input devices to image processing. The class has in the past attracted diverse groups of students of both sexes, different genders, and various ethnic and racial backgrounds. It has drawn students from multiple majors, from artists curious about new means of expression to scientists interested in new ways to visualize their data. A prominent example of such a student is Victoria Flores, a studio art major who, after taking CS 148, went on to pursue a Masters degree in computer science and to contribute to several research projects in the department. I strongly believe in the power of this approach to introductory graphics education and would seek to implement it myself.

Finally, there are so many small ways to foster diversity and inclusive learning environments. Not encouraging hotshots and know-it-alls who speak out too often in class, or taking them aside after class and informing them about how their behavior can be seen as intimidating. When designing in-class examples, using character names that reflect the diversity of the actual student body (Asuka and Bharath can be used anywhere Alice and Bob usually show up, for instance). Taking care not to casually reinforce heteronormativity, whether that be in constructing stable marriage problem examples, or in refraining from jokes about how a class is so time consuming that “you’ll have to break up with your girlfriend.” These examples might seem small, but I believe that such small-scale, personal interactions can have large, lasting effects on students.

REFERENCES


