Teaching Statement **Pat Pannuto** **ppannuto@berkeley.edu**

I am passionate about understanding how students learn, how to design curricula that makes electrical engineering and computer science accessible, and how to build a student's capacity and enthusiasm for future innovations. From an aggregate nine semesters as an undergraduate and graduate teaching assistant, I identified an underserved need in the curriculum and created a self-designed course targeting early-career EECS students. I served as the primary instructor for two semesters, with a combined enrollment of 475 students. I received teaching awards from both the College of Engineering and the Graduate School for this course. The University of Michigan has made the course a permanent part of the curriculum and an advised co-requisite for first-year CS students, EECS 201: Computer Science Pragmatics (Course homepage).

As a faculty member, I am excited to bring together diverse groups of graduate students under the broad umbrella of systems to identify cross-cutting opportunities. For undergraduates, my background in embedded systems has prepared me to teach a diverse array of courses, including circuits, signals and systems, digital logic, computer architecture, operating systems, or networking.

Computing for Computer Scientists.

In computer science education, many complex concepts such as operating systems, networking, or artificial intelligence leverage "learning by doing" with supporting projects that help students to develop a deep intuition of what they are creating. In my early career as a teaching aide, however, I observed a troubling trend in these courses. The mechanics of software engineering—how to author, edit, track, build, test, and debug code—were standing in the way of developing projects for some students. While students grasped the concepts with the same degree of comprehension they could not express them in software as effectively as their peers. The root cause of this knowledge gap was surprisingly simple; these mechanics were simply not part of the curriculum.

To address this gap, I designed and developed a new course, Computing for Computer Scientists (C4CS). The design of C4CS makes the implicit practice of software development explicit and transparent. By deconstructing the tools used by computer scientists and returning these tools to first principles, the course develops practical and conceptual foundational knowledge and demonstrates the utility of core concepts being learned across the curriculum.

"Lectures" in C4CS are unique. Once a new concept or tool is introduced abstractly, we explore its usage together in an interactive forum. This involves real-time examples and questions, which produces a shared space to critically respond to mistakes as a group. One of the most consistent and positive pieces of feedback from the course is the learning that comes from noticing a mistake, watching how it is debugged, and learning how it can be corrected and worked through.

C4CS has exhibited early success in supporting underrepresented minorities. The design of the course destigmatizes failure and normalizes the errors and difficulties that exist outside of source code by making them an explicit and transparent part of a student's experience and workflow. We break into discussions on dangers of certain archetypes like the "rockstar developer" and explore the very personal nature of development environments. The goal is to help understand and enforce that developing an identity as a computer scientist will take time. Encountering a strong, and different, identity does not necessarily invalidate your own or make you any less capable. This pays dividends for all computer science students, but it is especially rewarding for individuals with less prior experience.

Student Survey Quotes

> A good course! The lecturers were very engaging and friendly, which may not seem important, but really really is. A lot of times, computational skills like this have a high learning curve, but asking for help can be stigmatized. Simply put, it's hard to be a newcomer to CS. I think the lecturers did a really good job of erasing some of the barriers of learning new skills.
> – Student 5, W16::005

> Pat is super passionate about the material and really wants to help people learn. He made things that weren't really naturally interesting interesting. [...]
> – Student 60, W16::005

> This course is invaluable for students who want to be in EECS but don't have the background that independent programmers come in with. Classes like this are what enable students who feel behind to enter classes at the same level as others with more experience. I really appreciate everyone who set up this course
> – Student 2, F16::003

Mentoring.

I am particularly excited to begin forming the lifelong connections with new students, to nurture their interests and fan their passions, and to shepherd their growth over the course of their degree and beyond.

Formally, thus far I have had the pleasure of supervising the master's thesis of Andreas Biri, who is completing their *MSc ETH EEIT* with our research group at UC Berkeley. Andreas is one of brightest and most talented individuals I have ever met. As part of the thesis work, Andreas has taken charge of a inter-institution collaboration with a team of early childhood development psychologists at Vanderbilt University. Following the deployment thread of my research philosophy, together we have navigated the balancing act of providing a useful system for a targeted application while identifying a principled set of parameters that apply to the localization system at large, forming the foundation of the thesis. I have watched with immense joy and pride as Andreas has mastered the clear articulation of a clean hypothesis and developed taste for the design of experiments that will most effectively make the case. Unsure at the commencement of the master's thesis work, I am thrilled to report that Andreas has elected to pursue a Ph.D., beginning at ETH next spring.