My upbringing, degrees, internships, and professional experiences lie primarily in situations where the population distribution is overwhelmingly like me. This leads to blind spots. I have learned and continue to learn that this requires me to actively seek out different experiences, invite conversations, and to critically self-examine my own biases and privileges. In one instance, this led me to identify how my background had led to an inequitable advantage in course projects, and I designed a new course—now a permanent part of the curriculum—to address this achievement gap. In another case, it showed that what seemed a small problem of my own—where to eat late at night—was actually a significant and broader problem and paved the way to a solution that brought the whole community together.

Closing the Achievement Gap for Early-Career EECS Students.
In computer science education, we often use software engineering projects to actualize abstract concepts presented in lectures and evaluate student comprehension. However, such projects inadvertently intertwine evaluation of software engineering skills and computer science principles. The result is that students entering EECS courses without prior background in or exposure to software engineering are placed at an artificial disadvantage. To address this, I created Computing for Computer Scientists (C4CS), a new course targeting early-career EECS students. The goal of the course is to teach software engineering through the lens of computer science principles in a manner that is accessible to all backgrounds. The intent of the course is to provide all students with a common baseline. We further aim to foster skills and approaches for learning new tools. Lectures reason through the objectives of a tool (a build system should create correct, up-to-date software artifacts), how one might design and implement software to achieve these objectives (a dependency graph with rules), and thus build an organic intuition for how a tool operates and how you might debug these failures (missing dependencies misses rebuilds). The goal is to enable mastery of new tools and practices encountered in real-world software engineering in the future.

C4CS was almost not a for-credit course. When I first proposed C4CS, I received strong pushback that the course was not necessary, nor was the material appropriate for credit towards a degree in Computer Science. The counterproposal was to offer a series of "lunchtime lectures," perhaps with sponsored food, covering much of the same material. Such an idea would miss a key aspect of the course mission, however, which was to reduce the impact of the experience gap among incoming EECS students. This requires reaching people who do not have a computer science background or ties to the department. People who do not feel like a part of the computer science community are less likely to join (or even hear about) special lectures to learn more computer science. Those who are feeling overwhelmed and behind are unlikely to allocate their time to a voluntary lecture that does not (directly) lead to better grades in the courses they are struggling with. Folks who have developed their own coping mechanisms and strategies are less likely to be aware of the potential of powerful tools or to seek them out. Free food alone is insufficient to overcome these barriers. It was necessary for such an introduction to software engineering to be *normal*, something that is expected of everyone and everyone is rewarded for, in order to destigmatize learning "the basics."

I first proposed the course that would eventually become C4CS in the fall of 2011. It took several years of persistent effort, and a groundswell of support from a team of highly motivated undergraduates (several of whom would become the first instructional aides) as well as some supportive faculty, to make C4CS a reality.

Chez Betty: A Food Co-op that Became Community.
Graduate life sometimes requires that you work odd hours and days. In some cases this is pushing for a deadline. In other cases this can be cultural, when local holidays appear unexpectedly (or expectedly: over a dozen individuals visited Chez Betty on Christmas Day). In a smaller town or more isolated campus, such as the engineering campus at the University of Michigan, this can make the pragmatics of life surprisingly difficult. It is emotionally exhausting to be working alone and have nowhere to eat. In 2014, Brad Campbell, Zakir Durumeriç, and I created a graduate student food co-op, Chez Betty, to address this need.

Chez Betty is built on a foundation of trust and support from the community. The entire operation runs on the honor system, people simply take food from open shelves and record what they take, and volunteer labor, users restock shelves and unpack deliveries. By setting and expecting positive norms around honesty, service, and inclusion, Chez Betty serves to foster and grow these principles. At Chez Betty, undergraduate students, graduate students, faculty, and staff have natural, quick interactions and impromptu meetings. Watching a senior faculty member rush in to grab a Snickers is a very humanizing experience. A shared passion for sparkling apple cider can spark an unexpected friendship with non-academic staff whom you would otherwise never meet. Additionally, as a student-run co-op Chez Betty can respond to community needs. For example, when we heard that it was difficult to find gluten-free and halal foods or women's health products, Chez Betty sourced and stocked them within days.

Chez Betty also supports teaching. It is open source software, which allows a safe, but real, project for early-career students to contribute to. Students from the Computing for Computer Scientists class I launched, who may not yet have deep software engineering skills, have contributed translations to eight different languages. These translations serve both a pedagogical goal and keep our community welcoming when visitors encounter a small taste of home.

Looking Forward.
As I transition into a new community and a new role, I recognize that the first steps will be driven by listening and learning. However, some challenges are common across much of computer science, engineering, and academia at large. In the classroom, I am interested in understanding how other programs address the breadth of backgrounds and end goals of new computer science students. Several schools are exploring specializing computer science education from the first course by offering a breadth of "Intro to X" courses. How are such courses addressing the background concerns identified by Computing for Computer Scientists, and how can we ensure that all of the introductory courses remain welcoming and accessible to those who may not already have background in the area? Stepping outside of the classroom, in my professional conferences we have struggled with continuity around accessibility concerns. Because organizers turn over every year, it is difficult to ensure the long tail of seemingly small things (e.g. wheelchair access, *all* dietary considerations, and childcare) is realized every time. Installing institutional measures to address these concerns is a high-impact, personally meaningful service opportunity. Finally, in all communities there is a responsibility to identify voices and initiatives, those who have done the legwork to diagnose problems and develop solutions, and amplify them. Many of my efforts would not have been possible without strong faculty allies. I am grateful for their support and am keen to pay it forward.