There are many reasons why I want to become a teaching-track professor in computer science. The most obvious, of course, is that I very much enjoy the teaching process and its resulting student interactions. I have always strived to get to know my students individually, memorizing their names and learning about their respective backgrounds. With that knowledge, I better mold my teaching style and material to accommodate them personally. My goal is to create courses that are accessible and encouraging to all students. The ultimate reward is a room of engaged students in the classroom and a steadily more diverse and collaborative cohort of computing students.

Developing My Philosophy

Through my many years of university education, I have experienced many great courses. In fact, as an undeclared undergrad, a well-presented introductory computer science (CS) course was a large factor in my choosing the major. This brings me to a conviction I hold strongly: solid introductory courses in each major are pivotal in ensuring diverse academic communities. For example, if first-generation and/or students from historically marginalized/excluded racial, gender, identity, and ethnic groups[1] are turned off by a subject from the get-go, they are obviously less likely to continue it. I have seen this firsthand with myself and later in students I taught. These students are also less likely to enter university with prior CS experience, so first impressions are pivotal. If such students become lost, frustrated or feel unguided, their newly conceived negative associations become difficult, if not impossible, to counter [2, 3].

As someone with a long undiagnosed learning disability, I also recognize that well-structured courses are essential. When I was diagnosed with ADHD-PI[2] a few years ago, suddenly many of my academic struggles made sense. I cast doubt on my own diagnosis however, because I felt I could perform well... at least sometimes. Looking back though, I noticed I excelled in courses I would deem *cohesively presented*.

These *cohesively presented* courses:

1. Used logically sequenced and engaging lectures with cogent, fully encapsulated examples and accompanying material.
2. Said lecture material was often available beforehand or soon after each lecture, allowing me to print and mark it up with my own notes and observations from class.
3. Class concepts were described using a mixture of analogies, imagery, or even animation to help describe difficult processes and ideas.
4. If available, courses that were videotaped let me re-view lectures, helping me catch any information I may have missed during lecture. As an added bonus, I could later watch or listen to the recordings at higher speeds—a common learning tool among those with ADHD.

These individual teaching practices are great alone, but together can create a *cohesive presentation* that strengthens its individual components. I have found such varied techniques can help all students (neurotypical and not) and can accommodate students' perceived learning styles and actual learning processes [4]. My application of these teaching practices and my development of two additional practices during my real world teaching experience are described below.

[1] My wording here regarding historic marginalization and exclusion is deliberate, but discourse on the terminology continues. See [1] for details.

[2] Attention Deficit Hyperactive Disorder – Predominantly Inattentive type

Real-World Application

In Fall 2018, I had my first opportunity to put these four practices into action when I became the graduate lecturer of record for three sections (85 students total) of CSCE 121, Texas A&M's Introductory Computer Science & Engineering course.

Though I based my lecture material on those used by fellow lecturers, I modified and restructured them to make them more my own. Thankfully, these materials were an excellent springboard and easily modifiable. (Homeworks and labs, however, stayed same relative to the other instructors and sections.) I ensured the respective material was released before each lecture, allowing students to follow along or read later. Our brand new engineering education building allowed for class recordings. Thus, video recordings were similarly published after class and were available semester-long for review. Over the course of the semester, I found a teaching practice that seemed to engage my students especially well and has education research backing:

5. I commanded engagement through structured yet semi-improvised in-class coding exercises [5] to enforce the concepts I had just presented.

In these, students were invited to ask "what-ifs", e.g. what would happen if I dereferenced this particular pointer value? We then viewed the results together, explaining exactly why the results were so. Such exercises helped students feel they could ask seemingly "dumb" questions, which are usually anything but.

Some of my practices were well-received, while others had unintended consequences. For example, I stopped offering videotaped copies of lectures because some students felt they did not need to attend class anymore. How could I not see this coming?—I always attended my lectures even if they were videotaped! But of course, not all students learn like me. Instead, I had to modify one of my practices:

6. I ended my practice of automatically recording and posting my lectures, but allowed students to personally audio-record lectures in class.

This modification rewarded those who attended and let those who still wanted lecture recaps to have them. In light of education in the time of COVID-19, my policy on in-person attendance and offering taped lectures would be different now, as physically attending class is no longer always feasible. More broadly, I anticipate my teaching style and practices to evolve over time via greater experience and evidence-based computer science education research.

Lessons Learned and Moving Forward

During my lecturing experience, I also learned of the difficult and less-discussed aspects of teaching. A situation of serious plagiarism led to my reporting students to our Academic Honor Council, culminating in a presentation of my evidence in a trial-like manner in front its officials. Never had I considered these responsibilities. More frustrating, it felt like I lost many of my marginalized/underprivileged students throughout the semester. Some simply were not academically ready for the course while others had to leave university outright due to outstanding personal issues. A feeling of having failed them arose. Because of this, I fully expect to assist existing programs (or in their absence, establish new programs) which help support such disadvantaged students so they are not lost in the so-called ”leaky pipeline”. I myself was benefactor of such a CS-focused foundational program and recognize their value and importance[3].

My main takeaway of that Fall 2018 semester was a reminder that students learn in varying manners. It is my job as an educator to provide a diverse yet cohesive learning experience for my students, and actively modify my teaching as necessary to accommodate.

I am excited about computer science and I hope my prior students can attest to that. Attached are excerpts from student evaluations I received at the end of the term from the university's Office of Institutional Effectiveness & Evaluation. (Student evaluations, however, are well-known to not provide the whole picture and suffer heavily from bias [6, 7, 8].) Moreover, as a Latina I hope I am an example for my students, in particular those who belong to historically marginalized or come from disadvantaged racial groups and identities. I cannot wait to continue improving my teaching practice in the future and to shape the next generation of CS academics and professionals.

[3] This was the Columbia University Emerging Scholars Program.
For more information, see `https://www.cs.columbia.edu/esp/`

Office of Institutional Effectiveness & Evaluation Student Reviews

Professor Garza is the only professor I've had this year to express how much she cares about our future. She makes sure we understand the material and even made time to speak with us, her students, about what we can do after taking this course for our professional future.

I went in to office hours almost every week and [Elba] was very helpful every time, helping explain what I was confused about or offering assistance on code issues.

I would say for a first year professor, who hasn't spent their whole life teaching this subject. [Elba] Garza was very professional and making the best to what she was given. She wasn't able to put everything she wanted to say into a class period, but I believe she made up for with office hours, [P]iazza, and the help of her TA, Sharmistha.

Professor Garza is extremely well spoken and is very clear when she explains things. She also makes sure that the class understands what she's saying and often explains things in different ways so we can all understand her.

Professor Garza makes lecture interesting and makes an effort to be friendly.

What was really helpful was the live programming when Elba would show the effects of some code in real time. It was immensely helpful to see a program being built while integrating the topics that we learned that day, or read about the night before.

Elba is always open to discuss a question in lecture, and makes impressive effort to remember every question so that she can further clarify it when it comes up in the slides. Is present on Piazza and has very fast response times.

Class was very interesting and eye opening to the many possibilities of programming and the use of C++. I am glad to have learned it from Elba, and I am excited to see how she further develops her teaching skills as she is already an excellent educator.

Professor Garza is a good teacher for this course because she has plenty of experience in coding. Because of this, the coding assignments are challenging yet entertaining.

[Elba] was very direct and straight to the point in her teaching. Although she may go through some material fairly quickly, the [homework] assignments help to better understand the concepts.

References

[1] T. L. Williams, “Underrepresented Minority Considered Harmful, Racist Language,” *Communications of the ACM*, 2020.

[2] L. J. Barker, K. Garvin-Doxas, and M. Jackson, “Defensive Climate in the Computer Science Classroom,” in *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education*, SIGCSE ’02, (New York, NY, USA), p. 43–47, Association for Computing Machinery, 2002.

[3] S. Cheryan, V. C. Plaut, C. Handron, and L. Hudson, “The Stereotypical Computer Scientist: Gendered Media Representations as a Barrier to Inclusion for Women,” *Sex roles*, vol. 69, no. 1, pp. 58–71, 2013.

[4] C. Riener and D. Willingham, “The Myth of Learning Styles,” *Change: The Magazine of Higher Learning*, vol. 42, no. 5, pp. 32–35, 2010.

[5] M. J. Rubin, “The Effectiveness of Live-Coding to Teach Introductory Programming,” in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE ’13, (New York, NY, USA), p. 651–656, Association for Computing Machinery, 2013.

[6] L. MacNell, A. Driscoll, and A. N. Hunt, “What’s in a Name: Exposing Gender Bias in Student Ratings of Teaching,” *Innovative Higher Education*, vol. 40, no. 4, pp. 291–303, 2015.

[7] K. M. Mitchell and J. Martin, “Gender Bias in Student Evaluations,” *PS: Political Science & Politics*, vol. 51, no. 3, p. 648–652, 2018.

[8] K. Chávez and K. M. Mitchell, “Exploring Bias in Student Evaluations: Gender, Race, and Ethnicity,” *PS: Political Science & Politics*, vol. 53, no. 2, p. 270–274, 2020.