Statement of Teaching Philosophy

Rachel Liang

The first time I ever tutored a student was enlightening. I distinctly remember the look on the student's face as they started to understand the subject that we were discussing: excitement and relief of being able to complete their assignments independently in the future. This struck me because I was in the same exact situation only a couple of years ago, relying on the course's TA to grasp certain concepts of computer science that I felt were challenging. The aid of my TA and tutors in the past supported me during dark times when I was struggling with doubts of my ability to stay in STEM. Now, I have grown to love computer science and am glad that I did not give up during those tumultuous years. My previous troubles as a student, as well as my recent experiences as a preceptor[1] and tutor have molded the foundations of my teaching philosophy. Just as I was encouraged to keep pursuing computer science in the past, my aim as a teacher is to inspire and promote different facets of this field to students through facilitating interest and practical guidance.

It is my belief that students respond differently to different types of lecture. Although my experience as a "teacher" thus far has not been through being a lecturer, I have found that during individual tutoring sessions, some students have worked better with a hands on approach to programming whilst others have learned better through discussion and examples. Because of this, I have always gravitated towards classes that incorporate different types of learning in their lessons. Additionally, I believe that when possible, the teacher should pay attention to each student as an individual and note if they are having issues in the course.

As an educator, we have a responsibility to reach out and offer additional help or resources to any student without having to be prompted. However, I recognize that some classes may have too many students to establish this type guidance, and I also realize that the main goal as an instructor is to prepare students for the future. Sometimes, a laissez-faire approach to teaching is just as important. For example, in introductory programming courses when students have not yet learned how to be independent programmers, gently guiding those that are struggling to helpful resources may be the best thing to do in order to keep them interested in the subject. In more advanced courses, having students research on their own and have them independently solve their issues would benefit them far more in the future.

In the Fundamentals of Computing I and II courses that I was a preceptor for in my undergraduate, I had students come to me during office hours asking questions about concepts and assignments. Because these classes were intro courses to programming, I spent more time sitting down and teaching the concepts they had issues with. However, when I was tutoring and students came for help with a course such as Data Structures, I spent more time teaching them how to rely on resources that they had at their disposal, giving tips on how to properly ask questions online and research topics on their own in order to answer their own questions.

The way I see it, the best classroom environment is one where students are comfortable asking questions and are individually challenged (up to a certain point) by the material that they are learning. The former, in my opinion, is facilitated by a student feeling supported by their teachers and other faculty members. The latter can be done by adopting a mix of hands-on teaching and a laissez-faire teaching style. If I can encourage students to find computer science interesting and a viable option for their future, and if I can prepare them for said future, I would be one step closer to achieving my teaching philosophy.

[1] A preceptor preforms the same duties as a teaching assistant, except they cannot grade and are only paid through credits. See https://www.hartford.edu/unotes/2021/11/spring-2022-student-preceptors.aspx for more details.