TEACHING STATEMENT

Michael Hilton (hiltonm@eecs.oregonstate.edu)

During my ten-year career as a developer, I frankly found myself quite unprepared, and was forced to learn many things on my own. Fortunately, I was in a situation where I was able to do that, and eventually I was able to be successful. However, my vision is to help students be better prepared than I was, and give them a foundation of success for them to build upon. My philosophy of teaching centers on striving for student engagement, listening to students, and creating an atmosphere where all students are put into a position to succeed.

During my career, I participated in many mentoring situations, and I always found it very rewarding to help others succeed. During my time in graduate school, I have been able to teach two classes as instructor of record, *Fundamentals of Computer Science I* at Cal Poly San Luis Obispo, and *Software Engineering I* at Oregon State University. I am also currently teaching an introduction to graduate school seminar for new graduate students at Oregon State University. I continuously learn about teaching methods, and strive to improve my teaching effectiveness.

My Teaching Approach

Active Learning

I believe that for computing education to be effective, students must have a high degree of engagement with the material. One tool I use to help the students engage with the material is *active learning* [1]. When preparing lectures, I try to always build in an active learning activity as a way to break-up the lecture. I will generally ask students to break up into groups of around 4 students, and then present a problem for them to solve as a group. Once I feel that the groups are reaching a consensus, I will randomly choose a group to present their findings to the rest of the class. If the problem is open-ended, I will also ask another group to compare and contrast their results with the first group. Sometimes, instead of asking students to come up with a solution, I ask them to brainstorm, where there are no bad ideas. I have observed that certain students prefer there to be a "right" answer, while other students engage more when brainstorming.

Random Calling

Another tool I use for helping the entire class engage in learning is calling on the students at random. This practice is commonly used in law schools [3]. At the start of the term, I create a deck of notecards with each students name on them. At the beginning of each class, I shuffle them in front of the students. When it comes time to call on the students, I will randomly choose the next name on the deck. This ensures that I am not simply calling on the same students over and over again, but that all students get a chance to participate. While this approach has proved very motivating to students, and noticeably keeps them engaged, I do acknowledge that this is stressful for some students who prefer not to talk in class. In order to mitigate this, I emphasize to the students that when I call on them, there are no bad answers. I make a point to reinforce to them that if they are lost, saying "I don't understand the question", or "I am a little lost right now" is perfectly acceptable. When a student answers along those lines, I thank them for their honesty and explain the concepts behind the question again. This provides me with valuable feedback, because if a random student doesn't understand the question, it is very likely many other students are lost as well.

Listening to Students

Another tool I use to help students stay engaged with the class is simply giving them a voice, and then listening to them. The first day of class, I ask them to answer two questions; i) what do they expect from the class, and ii) what do they want to learn from the class. I then present to the class what they told me they hope to learn, and tie the course materials back to their learning objectives. In my experience, this helps students connect much better with the learning goals than if they are simply presented to them without their involvement at all. Additionally, at the mid-point of the class, I conduct early informal feedback, where I ask students how they feel about the class, the instruction, course activities, as well as general comments. I also ask them for suggestions for improvement. I then divide the feedback into those that I will change and those I will not. For things that I can change, I make changes for the benefits of the students. For example, in one class, I was only posting the due dates for each assignment once the assignment was given. Several students described how they would rather know all the due dates for the rest of the term, in order to better plan for other classes. Based on this feedback, I posted all remaining due dates, and several students approached me afterwards and thanked me for being responsive. There is other feedback though, where students suggest changes that I am not willing to make. For example, some students complained about how I randomly call on them in class. Despite the fact that I did not stop randomly calling on students, this was valuable feedback for me as well, because it was clear that they did not understand *why* I was randomly calling. Because of their feedback, I was able to have a discussion with the class, and clearly present the reasons why I believe that random calling is important. I have found when you treat your students with respect, and listen to and address their concerns, it can have a significant improvement on the level of engagement and appreciation students have with the class.

Inclusive Teaching

I am committed creating an environment where all students feel equally valued. My goal is to teach in a way that addresses the needs of students with a variety of backgrounds, learning styles, and abilities. I believe that inclusive teaching should not be seen as providing special accommodations for a specific group, but as a way to improve computing education for everyone. For example, by introducing a CS1 class for scientists, [2] researchers at Harvey Mudd were able to retain more female students, and at the same time help the male students improve their scores.

Not only am I committed to the ideals of inclusive teaching, but I have already incorporated it into my teaching. When teaching *Software Engineering I* at Oregon State University, I took specific steps towards making my classroom a more inclusive place. One component of the class was that I interviewed different professionals in order to give the students a picture of some ways that they could apply their degree. My goal was to present the students with a variety of possible career options, including some that they might not have thought of before. I was proud to have half (2 of 4) of our interviews be females who are working in computing. We also were able to show a diversity of career paths, including the head of a startup, a junior developer at a local mid-sized company, a researcher at Google, and a program manager at Microsoft. I also made a point to use gender neutral pronouns when describing hypothetical situations, so that no group would feel left out.

As faculty, I plan to continue on this path, as well as continue to make my classrooms more inclusive by following established research in this area, from such places as NCWIT[1].

Experience

Teaching

Software Engineering I At Oregon State University, I had the opportunity to teach *Software Engineering I*, a core requirement class. There were 85 students enrolled in the class. I was responsible for all aspects of the course, from creating the syllabus, writing exams, developing assignments, and coordinating with the two TA's that were allotted for the class. In this class we covered classical Software Engineering topics such as requirements definition, risk management, UML diagrams and design patterns. However, we also covered innovative topics such as Git flow, Continuous Integration, and Mob Programming. They also worked on a term project in groups of about four students. This allowed them to put into practice proper version control, peer review, and more. The students rated the course as a whole 5.4/6. The department median was 4.8/6, and the course median was 5.0/6. The students rated the instructor's contribution as 5.4/6. The department median was 4.9/6, and the course median was 5.0/6. The entire course materials are available online here: `http://web.engr.oregonstate.edu/~hiltonm/classes/cs361/`

Fundamentals of Computer Science I I taught *Fundamentals of Computer Science I* at Cal Poly. This class was an introduction to programming class, taught in C, with no programming experience expected. The section that I taught was offered off-cycle, and so most of the students were not CS students, but still needed a programming class, (e.g., math

[1] https://www.ncwit.org

majors). This class was taught in C, and included a lab period as well as in-class instruction. The topics covered included variables, functions, syntax, IO, simple data structures, etc. There were 31 students in the class, most of which had never programmed before. By the end of the term the students had written a ray-tracer. I was responsible for the development of all the lecture material, as well as making significant changes to the lab assignments. The term project was shared among multiple sections of the class, which were offered with different instructors, as well as a common final exam. The students assigned me an overall instructor rating of 3.62/4.

Introduction to Graduate School Currently I am teaching an introduction to graduate school seminar with 12 graduate students. I am developing the entire curriculum from scratch, but as it is a credit/no-credit seminar class, it is more of a mentoring opportunity than a traditional technical instruction class. In this class we covered how to read and write research papers, strategies for having successful relationships with advisors, and in-depth discussions about the process of progressing through graduate school.

Mentoring

I have had various opportunities to do mentoring throughout my career and in graduate school.

Professional Mentoring As a professional developer, I actively sought out entry level developers to mentor. I was able to establish many mentor relationships with new hires via the New Professional program, where new hires would take three month rotations with different projects, in order to learn more about the organization as a whole. Once I was a technical lead for various projects, I was able to mentor newer members of my technical team.

Cal Poly REUs During my summer at Cal Poly, I was able to help provide mentorship for two students via the Research Experience for Undergraduates Summer Program (REU). They were both between their freshman and sophomore years, and they provided assistance on my research project I was completing for my Masters. I worked with them over the summer, and they were able to make significant contributions to the project.

Oregon State REUs While at Oregon State, I was able to mentor five students over two summers. All of these students were able to make significant contributions to research projects, and were co-authors on papers with me. One went on to join the PhD program at Oregon State. He has since told me that my mentoring was part of the reason why he decided to pursue a PhD in CS. Two students have gone on to accept jobs in industry, and both of them have since told me that they feel that their time with me was able to prepare them to be successful from day one at their jobs. The last two students are still in the undergraduate program.

Course Preferences

I am qualified to teach any *beginning programming* class, as well as any class on *Software Engineering* topics. Some classes that I think I would be especially suited for include: *Software Engineering*, *Web Development*, *Mobile Development*, *DevOps*, *Software Visualizations*, *Programming for non-CS majors*, and *Building a startup.*

References

[1] C. C. Bonwell and J. A. Eison. *Active Learning: Creating Excitement in the Classroom. 1991 ASHE-ERIC Higher Education Reports.* ERIC, 1991.

[2] Z. Dodds, R. Libeskind-Hadas, C. Alvarado, and G. Kuenning. Evaluating a breadth-first cs 1 for scientists. In *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '08, pages 266–270, New York, NY, USA, 2008. ACM.

[3] J. M. Rogers. Class participation: Random calling and anonymous grading. *Journal of Legal Education*, 47(1):73–82, 1997.