Laura (Wendlandt) Burdick

Teaching Statement

As the primary instructor for a new one-credit, freshmen-level computer science (CS) course, "EECS 198: Discover CS," several students approached me excitedly, "Through this class, I've realized how much fun CS is! I can't wait to take more programming classes." These interactions encapsulate what I love most about teaching: helping students to grow in their abilities and discover new, exciting ideas.

Teaching Philosophy
Here, I highlight four themes of teaching, which I believe are important for supporting students and equipping them to tackle real-world challenges.

1. Thorough Understanding of the Building Blocks of Programming. An undergraduate degree builds a foundation that will allow students to successfully pursue post-graduate work and studies. One important skill that students need is to be able to quickly pick up new programming languages. An effective way to teach this skill is to require students to focus on a single programming language, such as C++, for multiple semesters. This gives students the opportunity to deeply engage with and understand one framework. Once students have a firm grasp of the basic building blocks of programming (e.g., control flow, object-oriented classes), they will be able to transfer these core ideas to other frameworks and settings.

In Discover CS, I spent the first half of the class introducing foundational programming concepts, such as variables and loops. During class time, I used live coding to work through examples, and asked students to solve small programming problems in pairs. Working through problems in small groups encourages participation from students who may not feel comfortable participating in larger activities. Targeted programming assignments completed after class allowed me to evaluate whether or not the students had a working understanding of the underlying concepts. Teaching core concepts additionally opens up an opportunity to highlight contributions of women and people of color to CS. In future introductory classes, I will ask students to research and prepare a short presentation about a historical computer scientist of their choice.

2. Ability to Adapt to Current Trends and Technologies. At the same time that we teach fundamental ideas, new technologies are being developed frequently, and, in order to be competitive, students must have a working knowledge of new tools and frameworks. One way to do this is to show how underlying concepts enable new advances in technology. In the natural language processing (NLP) class I co-taught, we asked students to form small groups and find a recent article about "one cool thing" in NLP. Every class period, one of the student groups gave a five-minute presentation about what they had found. This generated excitement for current advancements in the field, and it also gave us opportunities to tie in class material with current trends. Sometimes the presentations would show a new application of a method that we were talking about in the class, and we could highlight this. In the NLP class, we also chose what we saw as the most important current trends in the field, and we made sure to give adequate class time to covering these ideas. One way that we assessed this was through a group project, where students were asked to do a research project on a topic of their choice. Part of the requirements of the project included using new tools and technologies, which we assessed through a poster presentation and a written report.

3. Curiosity and Independent Learning. Because of the rapidly changing nature

of technology, it is infeasible to teach every new tool in the classroom. In order to prepare students for this environment, we must impart to them the values of curiosity and resilience in the face of failure. This involves giving students practical methods for learning new skills independently. One of the goals of Discover CS was for students to gain basic proficiency in Python, a high-level programming language. To give students practice learning independently, I asked them to complete interactive tutorials outside of class. These tutorials introduced key concepts, such as lists and functions, in Python and then gave students a chance to practice on simple examples. To solidify the students' knowledge, I also covered these concepts in class and worked through more complex examples together. These tutorials gave students the experience of working through programming problems that they had not seen before. After I finished a new skill, I assessed students' comprehension using homework assignments. These assignments asked them to apply the skill that they had learned in a novel setting. For example, after learning about basic operations (e.g., adding, removing, and changing elements) on small Python dictionaries, the homework required students to construct a dictionary with multiple elements and manipulate the information in the dictionary in response to user input.

Asking students to explore new concepts independently requires a supportive classroom environment. To encourage students to seek help when needed, I gave students class credit for coming to office hours and asking questions on an online forum (with the option to post anonymously). This contributes to making the classroom a welcoming place for all students, because it gives students multiple ways to ask for an alternative explanation or seek more instruction on new concepts. Having an anonymous avenue to ask questions helps engage students who are uncomfortable coming to office hours or posting on the forum publicly. Additionally, regular homework assignments provides a way for me to see how students were progressing and to follow up with individual students that might be falling behind in the class.

4. Ethical Consideration with Multiple Perspectives. As students build technology that will alter the lives of people, it is imperative that they learn to think ethically about their creations. For future upper-level classes, I want to incorporate case studies asking students to evaluate the impact of certain technologies being deployed by companies, such as automatic identification of fake news and in-home digital assistants. Students will need to consider questions such as: how is technology being used currently? How does it have the potential to be used? Who does this technology have the potential to harm? Does the community that this will affect have a voice in its creation? Students will be assessed on their ability to clearly articulate, in the form of a written report and a short presentation, the major ethical concerns surrounding a piece of technology, as well as their personal opinions on the proper use of this technology. Furthermore, students will be asked to complete this assignment with instructor-formed, diverse groups, so that they will have the opportunity to interact with peers with different viewpoints and concerns regarding the technology in question. This will encourage students to consider a variety of perspectives when forming ethical judgments. The in-class presentations will also help foster an environment of listening to and respecting the opinions of others.

Goals

One of the things that I am excited about is teaching introductory programming classes that introduce students to CS. I am also interested in teaching upper-level classes in areas related to my expertise. Additionally, I am willing to teach other core computer science classes. Teachers have a responsibility to do more than impart knowledge; they must also

encourage, support, and listen to their students, to build an environment where all can thrive.