Stephen Oney | Teaching Statement

Through my experience as an instructor and mentor, I have established a teaching philosophy centered around two goals. My first goal is to engage every student with the material as much as possible. This means creating material that is interesting and challenging for students from a wide range of backgrounds and with a variety of interests. My second goal is to foster critical thinking skills. Critical thinking is a fundamental skill for Human-Computer Interaction and Computer Science researchers and practitioners. Fostering it in the classroom requires balancing instructive material with hands-on thought-provoking exercises.

One way I engage students in project-based courses is by introducing opportunities for students to personalize their assignments. For every assignment I designed as an instructor at CMU, I isolated the main objectives, made them the "core" of the assignment (in this case, software projects), and required students to further customize their assignments individually for full credit. This encouraged students to not only learn a specific skill set, but also to complete assignments in ways that they found interesting. In faculty course evaluations, this aspect of my class has been highly rated and many of my students included their class projects in their portfolios.

In my classes, I also aim to enhance students' understanding of concepts through critical thinking. During lectures, I engage students by interspersing the material with informal "puzzles". These puzzles have been effective in spurring discussions and helping students clear up misconceptions by challenging their knowledge of the lecture content. I also believe students gain critical thinking skills and perspective when they have the opportunity to see and critique their peers' assignments. To ensure a productive peer-grading environment, I have provided a rubric for peer graders, allowed students to challenge their peer grades, and combined peer grades with instructor grades.

Teaching and Mentoring Experience

So far, I have served as a teaching assistant for three courses at MIT and as an instructor for two courses at CMU. My teaching has benefitted from personalized feedback I've received at CMU's Eberley Center for Teaching Excellence, where I have taken several voluntary workshops for instructors. I have also mentored three CMU undergraduates in conducting independent research, as well as pre-college students through outreach efforts with Women@SCS at CMU.

As an instructor for two classes at CMU, I also had the opportunity to design the course syllabus, assignments, and lecture content from scratch. Both courses were technical in nature and students came from a range of backgrounds. In faculty course evaluations, my students rated my overall teaching among the highest in the School of Computer Science (4.6 and 4.7 out of 5.0). I have thoroughly enjoyed my experience as a teacher and mentor. As I transition to a faculty position, I look forward to creating engaging classroom experiences and fostering students' curiosity and creativity.

Proposed Courses

The interdisciplinary nature of CMU's Human-Computer Interaction Institute has given me the background to teach a range of subjects in Computer Science, Human-Computer Interaction, and Design. A few sample undergraduate and graduate courses could be:

Software Engineering (Undergraduate):

Through my research and previous teaching experiences, I have learned how to introduce software engineering concepts to non-programmers. This course would cover algorithms, abstraction, programming paradigms, engineering patterns, and strategies.

Programming for the Web (Undergraduate):

This course would focus on programming dynamic Web content, covering asynchronous programming, design conventions, event-callback architecture, and Web services. Advanced concepts could also include server-side development, databases, scalability, and popular Web APIs (e.g. jQuery and Node.JS).

Human-Computer Interaction (Undergraduate & Graduate):

Human-Computer Interaction covers techniques for need finding, evaluation, and user-centered design. It would also cover design principles and prototyping techniques. This course would be appropriate for students with a range of backgrounds and interests.

Software Testing (Undergraduate & Graduate):

Testing is a crucial part of real-world software development. This course would teach students how to identify valuable test cases, improve test coverage, test on multiple levels of abstraction, and integrate testing with a variety of development processes.

User Interface Development (Graduate):

User interface development demands a different style of programming than general development. This course would focus on conventions, patterns, and popular frameworks and libraries that aid in user interface development.

End-User Software Engineering (Graduate):

A graduate reading seminar, this course would discuss programming tools aimed at end-users. These tools span several domains, including tools for scientists, prototyping tools, 3D modeling and animations tools, and more.

Human Factors in Software Engineering (Graduate):

This graduate seminar would examine how human psychology and team dynamics affect software engineering practice. These lessons can be grounded in several software engineering case studies and publications.