Teaching Statement
William Mansky

I look forward to teaching as an important part of my role as a professor. The two courses I taught in grad school – CS 173, Discrete Structures, and CS 421, Programming Languages and Compilers – were both required courses for CS majors, and were difficult for many students. Whether the fundamental properties of numbers in Discrete Structures, or ways to program without mutable variables in Programming Languages, the point was not just to improve students' practical knowledge but to instill new a way of thinking: the discipline of formal logic and strong type systems, the effectiveness of recursion and divide-and-conquer as problem-solving strategies. To this end, I learned to focus my teaching on goals and principles rather than facts: not "zero is an even number" but "why is zero even, and how can we show this?" I also found it helpful to make discussion sections and lectures as interactive as possible; I was always willing to stop for a student's question, and I emphasized practice problems and group work. The basic principles of discrete math and functional programming can seem alien and intimidating, but when reduced to their essence, expressed clearly, and practiced regularly, they are problem-solving tools that can be used far outside the subject matter of the course. Students who truly learn discrete math and functional programming become better imperative programmers as well – and better mathematicians, better logical reasoners, and so on.

As the primary instructor for a summer session of CS 421, I also dealt with the unique challenges of teaching on-campus and online students simultaneously. As part of the Graduate Teacher Certificate, I met regularly with a teaching expert, and we discussed the steady decline in attendance as on-campus students chose to watch the recorded lectures rather than attend in person. My mentor encouraged me to think about why this was a problem, and what I could do to address it. I worried that the fewer students came to class, the fewer questions would be asked, and the less I would be able to adapt my lectures to the class's needs. In future courses, I plan to put more emphasis on solving problems in class, looking for a sweet spot between lecturing and office hours, to make sure that students are actively benefiting from my presence and from time spent in class. The best aspect of the course was the online discussion board Piazza, where students could not only post their questions but also answer each other's. I made a point of being active and responsive on the board, occasionally surprising students with how quickly an answer appeared. I knew I'd truly managed to convey understanding, though, whenever one student explained to another the step they'd missed. I always learn more about a subject when I teach it, and Piazza gave students the chance to learn in the same way.

I look forward to teaching interactive and problem-solving-oriented courses at all levels, and while my experience has been on the mathematical end of computer science, I believe it is more broadly applicable as well. I am able to teach introductory programming in both imperative and functional languages, introductory systems courses, compilers, programming language theory, and discrete math. I am also interested in the ongoing project (by professors including those I have worked with) to develop project-based courses in compilers and operating systems using interactive theorem provers, with the goal of highlighting the mathematical logic behind common programming projects. Whether by incorporating this approach into my own teaching, or simply using it as a guide to the underlying principles of CS topics, I hope to bring an accessible mathematical approach to less traditionally rigorous classes. Formal methods may be intimidating to students, but with clear explanations and interactive discussions, they can be key to pushing students beyond knowledge of facts to deeper understanding.