Technical fields such as computer science provide educators an exciting challenge. In addition to fostering development of analytical reasoning and problem-solving skills, it is equally important to teach students how to effectively convey technical material. Mathematical proofs are only accepted if they can be understood and verified, efficient algorithms will never be implemented if their descriptions are unclear, and brilliant code is largely useless if it cannot be read or maintained. With the development of both analytical and communication skills in mind, I will elaborate on my teaching philosophy using both examples from my past experience and approaches I plan to use in the future.

I believe teaching begins through leading by example. My personal expository style is generally rather informal and as conversational as the format allows, emphasizing clarity without sacrificing academic rigor. This has suited me well in the past, engaging audiences in a sort of dialogue. I believe presentations and other course materials should present problems and solutions in a clear way, discuss motivating applications, and showcase clear technical writing throughout. Given the wide array of learning styles, I would attempt to provide multiple avenues toward learning course material. In addition to lectures, I would provide detailed lecture notes whenever possible and suggest alternative readings—ranging from textbooks and original research papers to notes for similar courses at other institutions. This not only gives students the option to choose their preferred format but also exposes them to many examples of clear technical communication.

In my past coursework, I found the best classes offered feedback early and often. Programming courses offered lab sections where we could work on tasks similar to our assignments and seek individualized help from teaching assistants. Algorithms courses offered problem-solving sessions with worked examples and toy problems to solve in small groups. I think these are a key component to most computer science courses, particularly for developing the necessary technical skills. This approach can be combined with quick turn-around time for grading, ideally with office hours scheduled in order to help students incorporate feedback from one assignment into the next. This is incredibly useful for developing better writing skills, a fact I can attest to comparing the grading cycles for different courses I have helped teach.

I would posit that there are at least as many assessment methods as there are educators. In addition to written assignments and exams, I believe informal student presentations (to instructors or a small group of peers) are very useful. Presenting one’s work in a live setting is a considerably different task than writing about it. Fitting inside time constraints requires distilling the work down to its key ideas, and fielding unscripted questions takes deft
control of the material—both of which require significant reasoning skills. Nearly everyone in a technical job is subject to work review, yet the ability to present technical material is underassessed in most curricula. We have a tendency as scientists to view performance in absolute terms—either the code has a certain functionality or it does not—rather than as a matter of relative progress. In one of the programming courses I helped to teach, a large portion of the rubric for each assignment related to code readability, documentation, and other qualities on which the student was judged relative to her past performance. The students each made considerable progress throughout the semester, regardless of their disparate skill levels coming into the course. This individualized, progress-based grading can be incorporated into other settings, particularly for gauging the growth of communication skills.

In summation, my philosophy of teaching focuses on the development of analytical reasoning, problem-solving skills, and communication skills necessary to effectively convey technical material. I believe the methods I choose to use are transparent with respect to these guiding principles.