Teaching Statement
Ryan Dougherty
Ryan.Dougherty@asu.edu

I have been involved with the “instructional” part of the educational process since my undergraduate degree. Starting in 2013, I have been a Teaching Assistant, as well as a grader for various courses, and ended up supervising over 1100 students over the course of two years. By far, the course of which I have been a part most often is CSE110 at Arizona State University, which is dubbed “Principles of Programming with Java,” and the professor I have worked with is Farideh Tadayon-Navabi. This course is the very first that new students take in the Computer Science department, and there are no assumptions made on previous knowledge of computer usage, programming, etc. The most important achievement of ours was our design of the ASU Online version of the course, which is still widely popular today. However, even with this popularity, many students still had the struggles that come with engineering majors and only learning materials and concepts over the Internet.

CSE110 Lecture Notes

Therefore, in August 2014, I developed my own series of lecture notes for CSE110. There are several motivations for why I did so. The first is that, in addition to the assigned textbook the students used, there are many different materials available online, in the form of PDFs and videos on Youtube (among others). Given this vast array of available information, it is very difficult for a beginning student to progress. Since the notes are only approximately 50 pages long, it not only offers every concept in the course, but is compact and easy-to-use for the students. The second motivation is that Java requires a computer to use; what if the student is away from his/her computer for an extended period of time (either while taking CSE110 or not)? The notes give plenty of examples, with code given, as well as definitions and usage patterns for various Java concepts. At the end of each section, I give both written and programming exercises to further enhance the concepts. These notes have helped many students in subsequent sections of 110, and some have thanked me for taking the time to write them. The writing of these notes have given me some important insights:

1. **Students are very (possibly too much) trusting of information presented to them.** Many will take what they see and hear from a professor or TA as face value and not re-evaluate the correctness of concepts being covered. This is a natural intuition that students should make, but the major downside is the frequency of which this occurs. In fact, as of writing this, no one has sent me corrections or errors of my notes. Students should be able to challenge ideas and concepts that do not fully make sense, because the whole purpose of their taking these classes is to have them critically analyze the world around them. The whole purpose of the scientific/engineering process is to be able to think in conceptual terms and to improve the literature around them by analyzing and improving on what others have done. We (the other instructors and I) have recognized this problem, and have started to change our way of teaching; however, we still have a long way to go.

2. **Conveying Information is Very Difficult.** The whole purpose of the notes is to convey the CSE110 concepts to students who have no background in computers or programming at all, and this is extremely difficult. Imagine trying to teach a child how to ride a bicycle for the first time–there are going to be bruises, cuts, many failed attempts, balancing issues, and many other struggles before he/she can successfully ride without direct aid. I used the many email and in-person conversations that I have had with other students in previous offerings of CSE110 (as well as my experiences taking the class) to gain an understanding of how first-year students learn concepts, and to fit the approach to the concepts using this understanding.

3. **Students Love Step-By-Step Examples.** This speaks for itself–theory is nice, but concrete examples, as well as any intermediate steps, are much more important in solidifying knowledge. Going from 0 to 100 very quickly in terms of being able to understand Java and common programming language paradigms (e.g., indices in strings, arrays, etc. start from 0) is one of the hardest parts of the entire undergraduate process. After I first released my notes back in August 2014, I realized that although I do provide examples in some chapters, I have neglected others, especially important ones such as arrays. I have now made it a priority to update these notes with many more examples with step-by-step analysis and reasoning.
FSE100

In August 2015, I was offered a teaching position for the FSE100 class, which is dubbed “Introduction to Engineering.” Like CSE110, this is a first-year course in which students, in the weekly lecture, learn about the Engineering Design Process, as well as other topics like Computers and Testing. However, they also have a weekly lab where they work in teams to apply these concepts, with the use of their imagination and creativity, to building LEGO robots that will compete against the other teams in two competitions. This is often the most fun course a student will take at ASU for several reasons: (1) the hands-on nature of the class, (2) competition against other students, and (3) working with people that they have not ever met before in a team. Unlike before, when I was only a TA or grader, there are many new challenges that I faced, almost immediately after I started the position:

1. One has to be absolutely clear in every capacity. The labs that the main instructors use for FSE100 are recycled each semester. Even though they are improved each iteration, there are still ambiguities in how some instructions can be interpreted. If you are not “rock solid” in every possible way of requirements and instructions, then if they can be interpreted a different way, they will be. Of course, this is not possible for one instructor (or even a few) to do. However, the feedback I receive from my students has been immensely helpful in clearing these ambiguities in the later labs.

2. The balance of strictness and leniency is important. This is a point that every instructor has to deal with at some point in his/her career. On one hand, one needs to have requirements, rules, deadlines, grade cut-offs, and so much more. On the other hand, approachability and leniency are important to keep students motivated within the educational process. There are advantages and disadvantages to weighing either side more than the other, and I believe there can be a balance. However, this balance is person- and course-dependent, and finding an appropriate balance seems only possible through teaching the course itself, which seems like a circular argument. There are several approaches to overcome this obstacle, and one notable one is by observing how other lecturers teach, as well as ask questions about their methodology. These approaches are what I plan to do when I teach courses in the future, because they are extraordinarily helpful in improving lecturers as a whole.

3. Students are afraid of being wrong. This is, unfortunately, something that exists in courses, especially undergraduate ones. My style of teaching involves asking many questions to the class about concepts, including ones such as: “If instead of doing X we do Y, how does the result change?”. The first few times I ask this sort of question, almost no one raises their hand or answers. There are many possible reasons as to why, but one way that I know helps students overcome their fear of answering is through positive feedback. Saying “No,” “Incorrect,” or “I don’t think so” to an incorrect student answer is absolutely unproductive for the student, the instructor, and the other students. However, saying “X is a good idea, maybe we can improve it by adding Y”, or even “I haven’t thought of that, but that is a good approach also”, is much more productive in that it not only gives the student more motivation to participate in answering, but also gives all the students a way of thinking about the question that they may not have realized before. Providing feedback such as the latter examples, I believe, are an absolute necessity to teach effectively.

Teaching Philosophy

My philosophy with regard to teaching, in any capacity, is that there must be:

1. Consistency. A large percentage of the students in these classes work really hard at fulfilling the requirements, and need to be taught a strong foundation of the concepts. If one is inconsistent with regard to any part of the course (be it the syllabus, exam questions, assignment or lab criteria, etc.), then there will not only be struggles on the student side, but also the instructor and TA side (grading, answering student questions, etc.). One can achieve consistency in several ways: by (1) observing other instructors’ examples and build off of them, or (2) use one’s own intuition and past student experience in an attempt to improve one’s consistency. I have (and will continue to) used a combination of these techniques to be as consistent as possible throughout.

2. Motivation. No matter the course material, there needs to be motivation from both the instructor and TAs in order to be successful. The reasoning is that if there is no motivation, then there cannot be any from the students. Also, having motivation allows one to present material in a clearer manner because not only do the students pay closer attention, but also that the instructor him/herself is able to convey the material more effectively. What I do for some material is present examples (such as how finite state machines are useful everywhere, such as compiler design), and for each, I try to motivate the concept through the examples, not the other way around.

3. Instructor-Student connection. Sometimes, student motivation does not have to occur through motivational examples during lecture. As an example, for the FSE100 labs, the TAs and I would walk around the
room to ask questions from the student teams if any problem arose, or if they needed to be signed off. If this is all that the TAs and I did, then the labs would be more boring than they have to be. What we have done is to engage with the students more, to ask questions about what they currently are doing, even if they do not currently need help. More than that, we emphasize important concepts through asking questions in a “progressive” way (i.e., help the students progress toward the desired result through reasoning, rather than giving the answer directly).