Teaching Statement

Paul "Will" McBurney

I have been teaching as a profession for three years, and taught as a graduate student as early as 2010. Over this time, my philosophy on teaching has continued to grow and shift to meet new challenges. Specifically, in the last three years, my philosophy on the need to provide students with rapid, detailed feedback through the use of frequent small assignments, has been reinforced. Further, I have developed numerous techniques for managing the large courses I have instructed at the University of Pennsylvania. Above all, I am committed to being the best teacher I can be to my students, which is ultimately why I chose a teaching career over a tenure-track career.

Frequent Assignments

One of my strongest beliefs that has been reinforced in my courses (notably CIS 110, CIS 350 CIT 590, and CIT 594 at the University of Pennsylvania) is that the most effective learning comes not from lecture, but from assignments. Lecture is vital, as it is often where students first encounter a topic, and get their first experience seeing a topic applied. However, I find students ultimately become most confident in a topic when they apply it independently in an assignment.

Frequent and regular assignment deadlines can be used to detect if a student's performance declines. This helps both instructors and teaching assistants identify and help a student in need. If assignment windows are irregular or too infrequent, it becomes difficult to monitor student progress. A decline in performance may be caught too late into the course to correct, and lasting damage may be done to the students understanding, which will carry over into future courses. In classes I have taught, I have ensured students are always aware when the next assignment is due. Gaps between assignments are as small as possible, as having an assignment pending, even if not due in the immediate future, keeps students attentive in class to topics. This is not to say I support artificially inflating a course's workflow just to have assignments. Rather, this idea changes how I write assignments to ensure they are highly targeted to a small set of learning objectives.

Student Feedback

An additional benefit of frequent small assignment is the ability to provide rapid student feedback. For example, in my CIS 110 courses, I have found it is easy for students to develop bad habits in computer programming, such as not thoroughly testing code, poorly documenting code, using global variables unnecessarily, etc., if they are not given consistent feedback. To this end, it is important to have an objective, detailed, easily understood rubric for all assignments, as the grading rubric is the primary means of feedback to a student.

As much as possible, rubrics should be objective. This is not always possible (for example, quality of code documentation is largely subjective), but removing subjectivity when possible from a rubric, and using precise detailed language to limit subjectivity, gives students more faith that an assignment was graded correctly. This faith in the fairness of grading is critical, as if a student believes an assignment is unfairly graded, they are likely to not take away the lessons the deductions were

meant to teach. I view grading as an incentive structure to facilitate learning, rather than a system of rewards and punishments. The primary purpose of grading an assignment is not to give a student a grade; the primary purpose is to help the student identify where mistakes were made and *how they can be corrected*. Thus, the grading rubric must be understandable to the student.

Managing Large Courses

A consistent concern across Computer Science departments relates to the swelling enrollments in the field. It is, of course, not at all bad news that our field is in demand. There is a significant national need for more people with a computer science skill set. However, this has created a strain, as class sizes are increasing with these enrollments. While the definition of a large course is subjective to an institution and an instructor, I have met very few lecturers and professors not concerned with the increase in class size.

In my role as a faculty member, I have encouraged the use of existing department infrastructure, by helping fellow instructors use the existing infrastructure and improving documentation of that infrastructure. Additionally, I have expanded our infrastructure by looking for off-the-shelf products, such as Gradescope and Piazza, to provided single access platforms where students can view detailed grading feedback, discuss problems with assignments or concepts, etc. Automatic grading scripts (which I have used frequently in CIS 110, CIT 590, and CIS 350), can provide clear indications of success or failure, which students can access *before* assignment deadlines. All of these tools help students understand where they stand in terms of understanding class material, and give them tools to improve their understanding.

Integrating infrastructure into a course is not a painless process. Often, in the first semester a specific tool is integrated, the breaking of an existing workflow can be taxing. However, once infrastructure is integrated correctly and thoroughly, it can provide significant benefits to maintaining a large course. This is why I volunteered to be a member of the Infrastructure Committee in the CIS Department at Penn: to help other instructors in the department get through the growing pains of increasing enrollments and infrastructure integration.

Teaching Interests

I am interested in teaching a variety of primarily undergraduate courses. I have also instructed graduate courses, but feel particular in-tune with the pedagogy of undergraduate courses. The following is a ranking, from, in order of preference from most prefered to least, of courses I feel immediately prepared to teach: 1) Software Engineering 2) Introductory Programming 3) Discrete Mathematics/Mathematical Foundations 4) Data Structures 5) Programming Languages. Further, while I would need to prepare the course, I feel I could teach introductory courses in algorithms, operating systems, data mining/pattern recognition, and artificial intelligence.

I have course materials, including lectures and assignments, prepared for Software Engineering and Introductory Programming. I have been very happy with many of the assignments I have used, and have continually refined these assignments to best communicate course material. I would be happy to share this material on request, and would be willing to work with colleagues teaching or co-teaching these courses to continue refining and applying them.