Richard McKenna Teaching Statement

Computer Science Department
Stony Brook University richard@cs.stonybrook.edu

As a Computer Science instructor, I run my classes the way I wanted them run when I was a student. This means the policies should be clear, the material should be made interesting, the work should be worthwhile, and the grading should be fair. To accomplish this, I make a conscious effort to incorporate the following components into every class I teach:

- Continuous Improvement
- Engaging Lectures
- Interesting Assignments
- Clear Course Policies on Honesty
- Clear and Fair Course Policies on Grading
- Encourage Undergraduate Research Participation

Below I have described these components in greater detail.

- **Continuous Improvement** – Every semester I teach a course, I try to improve it. Teaching the same course for multiple semesters is an opportunity to make gradual progress, and so at the end of each semester I try to think of ways to update the course, including lectures and homework assignments. In so doing, I try to keep the following in mind:
    - *Keep it current* – Computing technologies are in constant flux. Every day there are new techniques, tools, and programming libraries made available that may help me to do a better job of instruction. I feel it is my obligation to stay informed of recent developments in whatever field I am teaching.
    - *Try new approaches to learning* – Straight lectures can be an effective teaching tool, but there are many other approaches that technology is enabling. Films, tutorials, games, and interactive multimedia demonstrations may also be used effectively to teach subject matter.
    - *Use criticism to improve the course* – I like to read all course and teacher evaluations the students submit such that I may get ideas on how to improve the course. Even those criticisms that I feel are unfair may give me some good ideas.

  One example of how I have done this is in CSE 316/416, both of which I teach regularly. In CSE 316 students learn the basics of full stack application development as well as basic software design patterns and principles and are asked to complete an incomplete, but partially architected project. For this I created a MERN stack template that students can use to quickly start working on such a project. The architecture is well-designed, using principles I'm teaching, and I use it repeatedly each semester. In CSE 416, students are then to design their own applications, and most choose to use the architecture I provided as a starting point. In addition, I have developed a design format for this architecture to help students learn how to plan and design software. This took time to develop but saves me time each semester I teach these classes as it provides a common language of design.

- **Engaging lectures** – Students should feel that attending course lectures is a worthwhile use of their time. They should feel they are part of the discussion rather than simply being lectured to, but there is more to it than that. To ensure lectures aren't stale drains on student time, but rather an active and important part of the learning process, I do the following:
    - *Control the classroom* – Classrooms where the lecturer has lost the students are demoralizing. Students feel less serious about learning the material and giving their best efforts in homework and exams. I try to make my classroom open and fun, but also disciplined. No sleeping, unproductive talking, or general class disruptions are permitted. Students breaking such rules are politely asked to answer questions on the lecture material.
    - *Encourage fearless inquisitiveness* – One of the greatest challenges to teaching is dealing with students' fear of failure. Asking and/or answering questions in lecture can be stressful to shy students who fear looking uninformed or even foolish in front of their peers. In lecture I encourage failure. In fact, as a learning tool, answering a question incorrectly and then working our way to a correct solution works better than simply producing the correct solution from the start. Many times, it is the best students who are most willing to participate in question asking and answering, but I try not to limit class discussions to the best students producing correct answers. Instead, I encourage students to go out on a limb and risk answering questions incorrectly, so that we may share the experience of problem resolution, which helps students in learning and remembering important course concepts. In lecture I sometimes specifically ask students for wrong answers, to disarm them a bit, and to release them from the pressure of having to produce a correct solution in front of the class.
    - *Entertain the class* – Lectures must first be well-organized and informative, but it also helps if they are entertaining. Enthusiasm for the subject and providing context and examples the students can relate to go a long way to keeping the students' attention. So does a good joke, or even turning the class into a game show now and then, where students answer questions on the material for prizes. Getting the students thinking in terms of solving problems, developing their own solutions, rather than just presenting material in lecture gets them thinking in a dynamic way about the subject matter.

- **Interesting assignments** – We make you make stuff. That's one way I like to think about our Computer Science Department. Software Engineering is as much about the mechanism of program development and the feedback loops associated with it as it is the algorithms and abstract concepts needed for making programs. When making an assignment, my approach is to first think of an exercise that interests me. It keeps me interested by continuing my learning development, and it results in assignments that most students can relate to. I also try to make sure that there is a payoff for the topics we've discussed. The assignments must be a useful application of the concepts we've studied in the classroom. It also helps if the result is something the students would like to show off to family and friends. Something they, and their peers, can relate to and even enjoy. Finally, I make new

assignments each semester, never rehashing old ones, and so every semester I have a fresh look at material I may have covered many times before, but that may be applied in ways that are new and interesting to me.

- **Clear course policies on honesty** – As a student, one of the things I found most frustrating and demoralizing in a course was when students submitted work done by others. It is impossible to prevent this in every form, but as an instructor one should work to minimize cheating in one's class. As part of this I make course policies very clear, that plagiarism will not be tolerated and that assignments will be verified.

- **Clear and fair course policies on grading** – Grades can be a source of conflict between teachers and students, to minimize this, I do the following:
    - *Clearly state graded course components* – From day one, all work requirements should be made clear for the students.
    - *Be open to any exam re-grading issues* – Students deserve each point they earn, so I never discourage them from correcting possible grading mistakes.

- **Encourage Undergraduate Research Participation** – Over the years I have supervised hundreds of undergraduate students working on research projects of all sorts. A complete list can be found on my homepage. Working on a project for more than one semester is an opportunity to do something really special. Undergraduate students have great ideas and many have great abilities, so it has been incredibly rewarding for me. In recent years I have facilitated this through the College of Engineering's Vertically Integrated Projects program, which has been great. In that program, Ete Chan of Biomedical Engineering and I have had the fortune of working with so many great undergrad students who have done great work and gone on to great careers in industry. I like to encourage such projects for undergrad students because it's important that they learn about the back and forth of starting and running a worthwhile project. It's so much more than just coding.

I have really enjoyed my time teaching at Stony Brook, and look forward to continued success.

Richard McKenna, 2025
http://www.cs.stonybrook.edu/~richard