Statement of Teaching Philosophy

Shreya Kumar
shreyak@mtu.edu

I believe in empowering people for success by facilitating their journey to achieve the right skills. For CS undergraduate students, that empowerment may come from being able to create things they envision being used in the world and being able to visualize themselves as viable contributors to society. For senior citizens, that empowerment may come from being less afraid of technology, being less embarrassed to ask questions and explore, and from feeling more independent. For high schoolers and middle-schoolers, that empowerment may come from imagining positive possibilities for their future with computing and realizing that seemingly esoteric knowledge and skills are all achievable. In teaching, I have found that I am able to combine my enthusiasm for computing and my love for guiding people to find their own answers. I enjoy helping others realize that technology is for them to use as a tool, instead of having to struggle to find a way around it.

My teaching philosophy is guided by three core principles :

- an understanding that students learn better when they feel engaged—both with the instructor and their peers,
- an understanding that there is no “one size fits all” pedagogy, and that it is important that one keep growing pedagogically,
- a need to maintain an open channel of communication, to assure students that there is no challenge that cannot be discussed and resolved through active, timely communication.

All the courses I teach are designed with a critical pedagogy that recognizes that students learn better when they understand the real world impact of their work and their field.

I teach the team software project and introductory-level CS courses. These core courses allow students to acquire the technical and practical skills they need to succeed in industry and, in turn, for the industry to succeed. From my experience in industry, working in rigorously Agile environments, I realize the need for well-rounded skills in new graduates with an understanding of the value of software process through authentic exposure. Even working in a CMMI Level 5 Agile software process, I found that there was room for more rigor and clarity. This inspired my Ph.D. research which has two foci—to understand how software engineering communities sustain themselves, and improve the integration an authentic software development experience in the computing curricula in a way that is relevant to the field.

In the Team Software Project course, I have integrated Process Oriented Guided Inquiry Learning (POGIL) based activities which uses the inverted classroom format [1, 2, 3]. The students perform their reading as homework; in the classroom, they work in groups where each student has a specific role and the group answers timed questions structured to guide them towards reflection and exploration. In designing these activities, we learn from and join the growing movement of CS-POGIL [4]. I also encourage and guide students to choose a project topic for a specific type of user and incorporate the notion of fulfilling a need or providing a service.

The POGIL based activities are part of a larger scaffolding where the students are guided to examine first some common software development industry practices, then excerpts from the experiences of other computer science student teams, followed by real-world examples of communication in open source software development. The students then reflect on their own practices as part of sprint retrospectives in the structure of Scrum. Student teams compare their practices with other teams through interviews with other student teams. Teams also interview real/mock software users to increase their understanding of user requirements. These practices help students bridge the transition from programmers to well-rounded software developers.

For teaching programming-centric courses, I believe that creating assignments that serve a real-world purpose and relate to real world users—especially users with different expectations than ourselves— helps relate value to the material being taught. I have created assignments where young student programmers are asked to first create a small application for other young users and then create something equally useful for the elderly, combining my passion for working with both demographics.

I have helped design and teach summer courses to attract more young women to computing by giving them a realistic picture of the field [5]. The course combined mobile app programming sessions interlaced with real-time interviews of successful female computing professionals, interactive computing lab demos and introductory lectures on different areas of computing, linking back to the notion of real-world impact.

In teaching vastly different types of learners like CS undergraduate students, middle and high school students and senior citizens learning how to use computers, I have had the opportunity to work with different types of pedagogical techniques and resources. In the last few years, I have become quite excited about new pedagogical paradigms that mix traditional classroom methods with new techniques. I have regularly participated in campus wide discussions on a move to more blended forms of learning with the Center for Teaching and Learning. With my research group, I have explored the option of hosting a connectivist MOOC on digital literacy for the elderly. However, I do still value the traditional classroom lecture approach. I believe that the flipped classroom model works well for specific types of material and certain types of learning styles and more rigorous, traditional settings work for others.

References

1. Kumar, Shreya and Charles Wallace, “Instruction in software project communication through guided inquiry and reflection,” in *proceedings of the IEEE Frontiers in Education Conference*, IEEE, 2014.
2. Wallace, Charles and Shreya Kumar, "Engaging Software Engineering Students in Communication Design through a Pattern Language" at the International Writing Across the Curriculum Conference, Minneapolis, MN, June 2014.
3. Kumar, Shreya and Charles Wallace, “A tale of two projects: A pattern based comparison of communication strategies in student software development,” in *proceedings of the 2013 IEEE Frontiers in Education Conference*, 2013, pp. 1844–1850.
4. Kussmaul, Clifton, Helen Hu, and Martin Lang, "Using POGIL to help students discover CS concepts and develop process skills." in *proceedings of the 44th ACM technical symposium on Computer science education*, ACM, 2013.
5. Kumar, Shreya and Linda Ott, "Encouraging Talented High School Girls toward a Career in Computing through a Broader Understanding of the Field", in *proceedings of the 121st ASEE Annual Conference*, June 2014.