

Teaching Statement

Nilanjan Banerjee

Finding an optimal trade-off between *course workload* and *enjoyable learning* is core to my teaching philosophy. In ten years as a student, I have learned that instructors who try to enforce a subject on students through heavy course workload and strict evaluation are least successful. Today, many students want to register for courses where they can earn an easy “A”. Given that every student has four or more courses to learn every semester, it is reasonable to assume that they would restore to short cuts. Amidst this murky state of affairs, I believe that it is the teaching philosophy of instructors which needs to change.

My teaching philosophy is to find an optimal balance between “making a course enjoyable” and “assigning a workload adequate for student to understand the course”. Since the primary goal of several students is to earn a good grade, if the workload is too heavy or the evaluation is inflexible, weaker students will lose motivation to learn the subject. I believe that a flexible grading system which provides ample opportunities for students to improve their grade can go a long way in making classroom learning enjoyable and stress free. However, flexible evaluation should not be misinterpreted for zero workload. The primary goal of any teaching exercise is to make sure that students learn the concepts being taught and in experimental computer science, assignments are primary tools of learning core concepts. However, a policy like selecting the top three quiz scores (out of five) for final grades can relieve pressure of performing well in every quiz. Consequently, students can concentrate on learning the course material at their own optimal pace. To compliment a good evaluation policy, an instructor’s lecturing skills can make learning more effective. The most successful teacher, in my opinion, is one who spends the least amount of time talking during his lectures. Simple tactics such as encouraging discussion and student participation during lectures not only attracts the active attention of students but helps them understand the material better.

I had a unique opportunity to test my teaching philosophy as the instructor for an undergraduate-level C++ programming course at University of Massachusetts, Amherst. I was fully responsible for designing the syllabus, lectures, homeworks, grading and office hour policies. This programming course was unique due to multiple reasons: (1) The class strength was 20 with juniors, sophomores and seniors from a wide variety of majors. While some students were computer science graduates with adequate programming background, others were from mathematics and accounting with no prior programming experience. Hence, the class was highly diverse and balancing the needs of students with varied programming expertise was challenging. (2) It was an 8-week course with a weekly one and a half hour lecture. It is impossible to teach an entire C++ syllabus in 8 classes, hence lectures had to be intelligently designed such that students adequately understood the core C++ concepts. (3) The course was for 1-credit. Hence, deciding an appropriate workload was non-trivial. While on one hand I had to make sure that students get appropriate exposure to programming in C++, on the other hand the workload had to be manageable for a one-credit course. (4) The class met once a week for one and a half hours. Keeping students’ attention continuously for such a long time required intelligent lecturing skills.

To meet these challenges I designed the course material from scratch and paid special attention to my teaching strategy. I made sure that my lecture slides were well prepared and every concept was explained through simple examples and humorous analogies. A student in his evaluation commented that the best part of the teaching was that I “put humor into (my) slides”. I encouraged students to bring their laptops to class and during my lectures I demonstrated C++ concepts by compiling and executing code snippets. This helped students with no programming experience get a feel of how programs are written, compiled and executed. I introduced the idea of a “discussion session” at the end of each lecture. The session involved solving a small programming assignment in groups of at least three students. The assignment was based on the day’s lecture. At the end of the session, we discussed the problem on a white board and groups had to submit their solutions. Students were awarded extra credit if they solved the problem correctly. Such a discussion session had a two-fold goal. First, I found that there were students who were reluctant (or shy) to ask questions during lectures. However, during these sessions they felt free to discuss their questions with their peers. This provided an

| Criterion | Average Rating (out of 5.0) |
|------------------------------|-----------------------------|
| Preparedness of Instructor | 4.78 |
| Time Utilization in Class | 4.44 |
| Clarification of Questions | 4.22 |
| Personal Interest in Helping | 4.50 |
| Fairness in Evaluation | 4.78 |
| Overall Teaching | 4.0 |

Table 1: The student evaluation ratings for “Programming in C++” course.

opportunity for collaborative learning—students who understood the material well could help weaker students. Second, the extra credit provided flexibility to the evaluation process. It provided an opportunity for students to make up for previous quizzes and assignments. To compliment the weekly class, I met with students through flexible office hours throughout the week to help them with the programming assignments. Though the course had seven programming assignments, most of them were short and were directed towards understanding the core concepts taught in class. Overall, I received good evaluation rating (see table 1) from the students. I received encouraging comments such as: “One of the best classes I’ve ever taken” and *teacher* “went to great lengths to help me”.

The experience gained through teaching the programming course was very rewarding. For the first time I realized how diverse the perspective of students could be towards the same problem. I believe that teaching is one of the best ways a person can learn a subject really well. I would like to continue applying my teaching methods to classroom learning and student mentoring. I have also had the opportunity of mentoring two undergraduate female students, Maria Kazandjieva and Denitsa Tilkidjieva from Mt. Holyoke College on a research project related to “battery usability”. Mentoring Maria and Denitsa was a great learning experience. Unlike graduate students, I found that undergrads require constant guidance. Floating an abstract research idea to them and hoping that they would come up with an elegant solution is an unreasonable expectation. On the other hand, the best way to expose the youngsters to research is to give them small, clearly defined sub-problems which would help solve the bigger research problem at hand. This strategy of offloading interesting yet well defined research sub-problems worked well with Maria and Denitsa. As an outcome, we published a poster at WMCSA/HotMobile 2006.

In addition to mentoring students and teaching conventional courses (such as Operating Systems, Distributed Systems, and Computer Networking), I am excited about applying my research experience to my teaching. I hope to design and float seminar courses which would help students get a flavor of mobile and sensors systems research. I envision such a course having two components (1) *survey* component with discussions on the latest research papers on mobile and sensor systems (2) *project* component where students design, implement and evaluate a potentially new research idea related to mobile and sensors systems. The survey component would help students get exposure to the interesting research in the area and the research project would help them get hands-on experience with designing and implementing real systems.