3-1997

Teaching Ethical and Social Issues in CS1 and CS2

Kay G. Schulze

Frances Grodzinsky

Teaching Ethical and Social Issues in CS1 and CS2

Kay G. Schulze
Computer Science Department
United States Naval Academy
572 Holloway Road
Annapolis, MD 21402-5000

schulze@nadn.navy.mil

Frances S. Grodzinsky
Computer Science & Information
Technology Department
Sacred Heart University
5151 Park Avenue
Fairfield, CT 06432

grodzin@shu.sacredheart.edu

Introduction

The discussion of whether ethical and social issues of computing should be explored in undergraduate computer science education has resulted in most academic institutions and educators agreeing that they are important topics that must be included. Further support has been provided by Curricula '91 [16], the CSAC/CSAB accreditation [2] and ImpactCS [12]. Many books [7, 8, 9, 10] and papers [6, 14] have discussed what topics should be covered and what techniques can be used either in a dedicated course or in modules across the curriculum. However, explicit detailed examples that have worked successfully, particularly in lower level computer science courses, are still rare. This paper will discuss several examples that have been successfully used in CS1 and CS2 at a medium-sized university.

General Ethical Issues

Many ethical and social issues should be covered in a CS1 course. These topics stimulate discussion and help students develop critical thinking skills. They also provide an introduction to the types of problems students will meet in their professional roles. If students have not had a general philosophy course, it may be necessary to help them learn to argue in a logical manner. It is critical that students understand that irrespective of their viewpoint, ethical solutions must be consistent, coherent and defended with reason rather than emotion or intuition.

Intellectual property rights, particularly with respect to public domain and proprietary software, is one topic that is especially important. Stallman's "Why Software Should be Free" [15] and Nissenbaum's "Should I Copy My Neighbor's Software" [13] can form a basis for a debate on whether software should be free. Even if students take the side for free software, they need to realize that in most institutions software is not free. It is important for them to understand that if the software in the CS1 course has been provided by a site license granted to the institution, they must abide by the licensing agreement. Failure to do so could have two-fold consequences: the institution being sanctioned by the software company and the student being punished by the institution. It also introduces students to the concept that there are legal ramifications they need to be aware of when using software. Hence it is necessary to inform the students about the license and the restrictions it implies.

An intellectual property rights discussion also provides an ideal opportunity for the professor to expand on the department's policy concerning collaboration on programming assignments. This is particularly important if group projects are part of the course.

Most students in CS1 are also using campus-wide networking facilities and must be instructed on the importance of password security. Although the issue of computer security is quite large, at this level it can be limited to a discussion of what type of security a password ensures.

Any student who has had his computer infected with a computer virus is more than willing to discuss the consequences of creating and disseminating viruses. But at the CS1 level, most students lack the computer science expertise for any meaningful discussion concerning the possible positive reasons for having viruses. However, Eisenberg, et al.'s "The Computer Worm" will provide the students with an illustration of the damage that can result when a virus is disseminated over a network [4].

The discussion pertaining to password security and viruses can be broadened to include the notion of hacking. Many students consider hackers to be simply mischievous. Reading "A Dialog on Hacking and Security" by Dorothy Denning and Frank Drake [3] presents security from both the hacker's and establishment's points of view. Students

SIGCSE '97 CA, USA
© 1997 ACM 0-89791-889-4/97/0002..$3.50

are frequently surprised to learn that hackers take their responsibilities very seriously.

The CS1 course frequently provides students with their first exposure to e-mail. This provides an opening to discuss the privacy and abuse issues relative to e-mail. Does the university have the right to read a student's e-mail? Should the e-mail be monitored or censored? What is the university's official policy on monitoring e-mail? Should e-mail be used to harass individuals? It is not uncommon for a student to enter a lab and discover the prior user has forgotten to logoff. This presents an opportunity to abuse e-mail without necessarily being caught. What is the student's professional responsibility if he encounters such a situation? What is his personal reaction? Is there a dichotomy between the two?

Direct Impact Issues

As more and more universities are providing Internet access to their students, CS1 is an appropriate setting to discuss proper netiquette. Since the students are using institutional facilities to access the web, most institutions have a policy statement as to what is permissible and what is not. If no such policy exists, then this is an ideal opportunity for the students to determine what is appropriate.

We used a closed lab early in the semester of CS1 to teach the students how to search the WWW and how to create their own home pages. The students found this to be a very exciting project where they could exhibit their creative tendencies. Considering themselves peons in the scheme of the institution, they thought their home pages would only be of interest to their friends.

Our institution has explicit guidelines as to what may and may not appear on web pages that are housed on institutional servers. These guidelines were explained to the students, but for the most part, they didn't take them seriously. Their attitude was that no one of authority or importance would waste his time browsing a sophomore's home page.

The opportunity for a lively discussion on what was and was not appropriate for a home page arose later in the semester and it captured all the students' attention. A web crawler outside the institution found an inappropriate page belonging to one of the students and reported it to the administration. Suddenly the students realized that others besides their friends were reading their pages. Once the significance of the issue had a direct impact on their lives, the students were more than willing to begin to look at the issues involved.

Although it may not be possible to replicate this scenario on your own campus, one should be aware that students will become much more involved when issues impact them directly rather than appearing to only impact others.

Another area where students can envision a direct impact is security. A classic two-dimensional array programming assignment in CS 1 is the simple encryption problem. Students enjoy programming the computer to read an encrypted message and to translate it into a meaningful one. One of the easiest ways to accomplish this is to design the encrypted message so that when it is read into a two-dimensional array in row-major order and printed in column-major order a meaningful message appears on the screen. This is an easy programming exercise that introduces double dimension arrays and reinforces for-loops.

If a slightly more difficult implementation is desired, a simple Caesar or shift encryption can be used. Each letter in the encrypted message is shifted a fixed amount either forward or backward in the alphabet. This assignment requires that the student "wrap around" the alphabet and requires that the user correctly guess the shift in order to decode the message.

Most students find either of these methods relatively easy to code. Once they have completed the assignment, they are usually more than willing to discuss issues such as the Clipper Chip [11]. After all, if they were able to write a decryption algorithm, then anyone must be able to write one. At this point, the professor can discuss the pros and cons for encryption from a security viewpoint and the necessity for a more robust algorithm than the one the students have coded.

Responsibility, Safety, and Gender Issues in CS1

Most CS1 students realize that the programs they write in the course have been designed for learning and they seldom execute their programs again once they have been graded. As a result, it is difficult to instill in them a sense of responsibility for the reliability of their software. For them, the software is not "real" and no one else will ever use it.

An assignment that begins to bridge this gap is one that simulates a control mechanism. The professor can assign a simple program that simulates the arm of an x-ray machine [5]. The simulation should print the height of the arm in relationship to the table and should respond to user commands to raise or lower the arm. Most students will write the program without any consideration of the patient involved. An inquiry as to how many students' program checked to determine whether there was a patient on the table before lowering the arm to the table, allows them to realize that had their program been "real" it could have resulted in injury to the patient. This should spur a discussion on the software provider's responsibility for the safety of the user and the penumbra. If the student can begin to recognize the penumbra involved in their early programs, they are more likely to consider them when they finally are designing and developing "real" software.

Most students are unaware of the role that gender plays in most software. An interesting investigation project for CS1 is to have the class analyze the design of the male and female home pages belonging to their class or some specific group of students. This will enlighten most students and several gender issue questions can be investigated. What links are provided on male verses female pages? What are the similarities and the differences in colors, icons used, etc.? Do trends exist based on the student's major or on the student's gender? This project enables the student to realize that gender is an issue in designing any type of software.

Specification, Analysis, and Design In CS2

Stacks and queues in CS2 provide an opportunity for the student to consider the specifications and design of good software. The assignment we used in CS2 was a simulation of the parking lot for football games [5]. They were told the Athletic Association had provided the funding for the project. The students were given explicit specifications for the assignment. Vehicles wishing to enter the lot were to be enqueued in a single queue. Due to congestion in the parking lot, each vehicle record contained the time it could enter the queue and the time it could enter the parking lot. Once the vehicle entered the lot, it was parked in one of six stacks based on whether the vehicle belonged to a student from our institution, a student from the opposing institution, a faculty or staff member, an alumni, a guest, or a member of the press. When the game was completed, one stack at a time was allowed to leave the parking lot.

The students were told from the onset that the purpose of the assignment was to ensure that they could work with stacks and queues and that the specifications had been developed for that purpose alone. They were then assigned Collins, et al.'s article "How Good is Good Enough" to read [1]. In addition to writing the program, they were required to write a paper discussing the specifications and design from the viewpoint of the buyer, the provider, the user, and the penumbra.

This was an excellent assignment for several reasons. The students had a great deal of difficulty in determining who were the users and who were the penumbra. Also, having been told from the beginning that the specifications and design had been based on the instructor's requirements and not the buyer's requirements, the students felt free to criticize the specifications and design without feeling any jeopardy.

Before the students started their papers, the class discussed the obvious problems any college student would find with the design. They immediately determined the design provided no mechanism for tail-gating; no one could leave the game early; no provision was made for a mixed group of people in a vehicle, etc. Once these had been discussed, they were told that they could not use any of them in their papers. This forced the students to begin to think more critically and their final papers contained issues such as no provision had been made to provide access to the parking lot for emergency vehicles; no handicap parking had been provided; only one entrance and one exit to the parking lot and the arrangement of the parking lot in stacks were potential hazards to public safety.

This assignment gave the students the opportunity to think critically about the specification and design and to indicate how they could improve it. The assignment also provided concrete evidence supporting the importance of good specifications and design. The students acknowledged they felt as if they were actually participating in the assignment rather than simple programming it. As a side effect the assignment enhanced their writing ability.

A simple analysis project for CS2 is one that involves balancing trees when the data is skewed by gender. The data could involve athletic teams, dormitory rooms, etc. at a fictional university which is heavily populated by either males or females. This same data could also be incorporated into a hashing programming project. The students can analyze the efficiency of their algorithm when part of the key is based on gender and when it is not.

Conclusion

Even though the students in CS1 and CS2 do not have the computer science sophistication to delve deeply into many of the gray areas of ethical computer problems and to construct arguments using utilitarian or deontological reasoning, they do have the ability to recognize some of the ethical and social problems that can arise in the area of computer science. The inclusion of ethical and social issues at the CS1-CS2 level validates, in the eyes of the students, their importance to computer scientists. This is especially true if these issues are also covered in other courses throughout the curriculum. The foundation that students begin to acquire at this level will provide them with an awareness of ethical and social issues that will carry over into their upper-level courses and their work experience as well.

Bibliography

1. Collins, W. R., K. W. Miller, B. J. Spielman, and P. Wherry. How good is good enough?: An ethical analysis of software construction and use. *Communications of the ACM 37*, 1 (1994), 82-91.
2. Computer Sciences Accreditation Commission, CSAC Accreditation Criteria. Stamford CT, (1996).
3. Denning, D. and F. Drake. A Dialog on Hacking and Security. *Computer Ethics and Social Values*, D. G. Johnson and H. Nissenbaum (Eds.), Prentice Hall, Englewood Cliffs, NJ, (1995), 120-125.
4. Eisenberg, T., D. Gries, J. Hartmanis, D. Holcomb, M. S. Lynn, and T. Santoro. Case: The computer worm. *Computer Ethics and Social Values*, D. G.

Johnson and H. Nissenbaum (Eds.), Prentice Hall, Englewood Cliffs, NJ, (1995), 60-89.

5. Gotterbarn, Don, Personal communication, 1995.

6. Grodzinsky, F. S. and K. G. Schulze. Integrating ethics into the computer science curriculum. *The Journal of Computing in Small Colleges 11*, 4 (1996), 84-90.

7. Huff, C. W. and T. Finholt (Eds.). *Social Issues in Computing: Putting Computing in Its Place*, McGraw-Hill, New York, NY, 1994.

8. Johnson, D. G. *Computer Ethics, Second Edition*, Prentice Hall, Englewood Cliffs, NJ, 1994.

9. Johnson, D. G. and H. Nissenbaum (Eds.). *Computer Ethics and Social Values*, Prentice Hall, Englewood Cliffs, NJ, 1995.

10. Kallman, E. A. and J. P. Grillo. *Ethical Decision Making and Information Technology*, McGraw-Hill, New York, NY, 1993.

11. Levy, S. Battle of the Clipper Chip. *Computer Ethics and Social Values*, D. G. Johnson and H. Nissenbaum (Eds.), Prentice Hall, Englewood Cliffs, NJ, (1995), 651-664.

12. Martin D., C. W. Huff, et al. Computers in context: A framework for presenting the social and ethical impact of computing. A Report from the ImpactCS Steering Committee, May 1995.

13. Nissenbaum, H. Should I copy my neighbor's software? *Computer Ethics and Social Values*, D. G. Johnson and H. Nissenbaum (Eds.), Prentice Hall, Englewood Cliffs, NJ, (1995), 201-213.

14. Schulze, K. G. and F. S. Grodzinsky. Teaching ethical issues in computer science: What worked and what didn't. *Proceedings of SIGCSE* (1996) 98-101.

15. Stallman, R. Why software should be free. Free Software Foundation, Inc., (1990).

16. Turner, A. J. A summary of the ACM/IEEE-CS joint curriculum task force report: Computing curricula 1991. *Communications of the ACM 34*, 6 (1991), 69-84.