https://doi.org/10.48009/1_iis_2008_101-104

LESSONS LEARNED: THE CREATION OF A CS/CIS GATEWAY CLASS

Shannon Duvall, Elon University, sduvall2@elon.edu
Michele Kleckner, Elon University, mkleckne@elon.edu

ABSTRACT

Creating a Computing Sciences class for non-majors has many well-known challenges: What topics should be covered? How do we meet the various needs of non-majors? What do the students already know? How can we make the class engaging? In this article we uncover how these challenges were addressed at Elon University, leading to a successful CS0 course now in its fourth year.

Keywords: accessibility, computer literacy, computer skills, CS0, curriculum development, engagement, motivating factors, programming, relevance, student learning, women in computing.

DEFINING OUR CURRICULAR GOALS

Several factors motivated the decision to create a computing course for non-majors. First, the only course in the Computing Sciences department that allowed non-majors to earn General Education credit was the introductory programming class that also serves as the gateway into the major. This led to a wide gap in the skill sets and interest levels of the students in the introductory class. It was extremely difficult to teach students who were not really motivated by their major to learn to program while challenging the students who were majors. The class was divided into the "haves" and the "have nots" in terms of mathematical ability and previous experience with computers, making the class awkward and unable to fully meet the needs of either group.

Furthermore, in a Computing Sciences department that encompasses both Computer Science and Information Science, we desired a course that gave a more balanced overview of both. We wanted a course that could serve as a gateway to either major in the department. Non-majors could take the course and not only understand the differences in the two majors, but also come away with knowledge and skills that would help them in either, should they be interested in learning more. The hope was that students who found themselves struggling in the introductory course in either major could instead take the non-majors course and be more prepared and confident to enter the major.

Finally, we were motivated to create this course based on a university-wide need. Exit surveys given to seniors across the university asked how they would rate themselves in computer skills. As seen in Figure 1, only 16.15% reported "Very Good" out of 483 students surveyed.

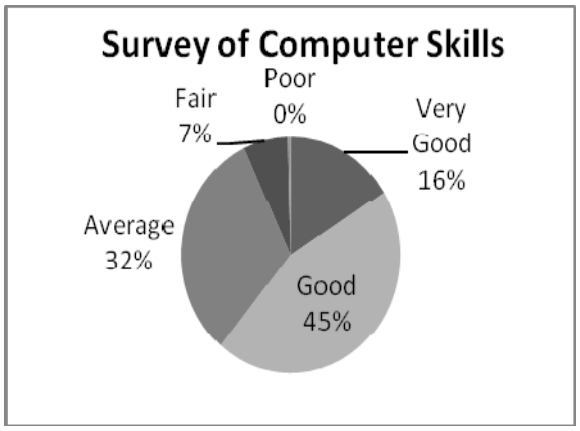

Figure 1: Survey of Computer Skills

In our own student surveys we asked what the university could do to make students more technically proficient. While the answers varied, most answers fit into one of two categories: increase the use of technology in every class, and offer classes or workshops on computing for non-majors.

With these goals in mind we set out to make a non-majors course that would act as a service course to the university as well as become a means for making computing more accessible and visible on campus. In the rest of this article we will describe how we met the remaining challenges of course design and implementation: what to teach, how to deal with a variety of student experience, how to meet the needs of the non-majors, how to make the class engaging, and how to make the class accessible to those who are traditionally turned off by computing.

CHALLENGE: WHAT TO TEACH

To decide on the topics our course would cover, we held a student focus group made of first-semester freshmen. We presented them with several CS0 textbooks currently in use and asked them to skim a chapter and the table of contents of each and tell us which they preferred and why. From this we learned that the students wanted to learn *everything* – programming, website creation, office applications, hardware, and graphics. Most students wanted to know how to fix various computer problems they might encounter. It was obvious that we needed a breadth-first approach to computing in the class. It was apparent that we should expose our students to many areas but not go into depth in any one area. Hopefully in this way we could cover several topics of interest to each major.

From the focus group's feedback we defined the major themes for the class: Global Applications, Business & Science Applications, Programming and Problem Solving, Hardware, and Societal Issues in Computing. The focus group chose the textbook that we use, *The Analytical Engine*, and encouraged us to hold class in a computer lab for hands-on learning [4].

We also compared ourselves to other Associated New American Collages (ANAC) institutions to see how our peers were teaching CS0. We categorized the different modes of teaching into four categories: literacy, skills & literacy, skills, and programming. On the literacy end of the spectrum, students are exposed to an overview of hardware, software, history, and ethics. The skills taught in these CS0 courses often include Microsoft Office and Internet software. On the programming end of the spectrum, Java and C++ tend to be the programming language taught. As shown in Figure 2, of the twenty universities, the majority of them focused their CS0 course on literacy.

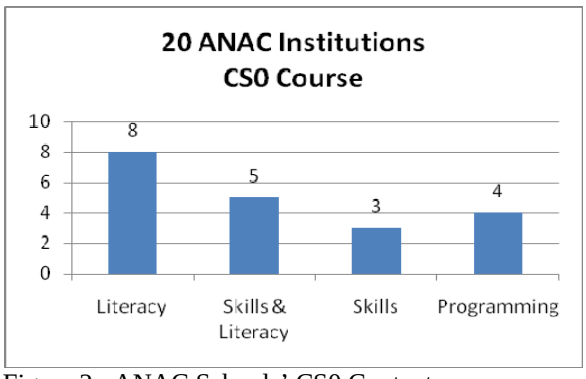

Figure 2: ANAC Schools' CS0 Content

CHALLENGE: VARIETY OF EXPERIENCE

One of the challenges of teaching a non-majors service course is that all types of students take it. Freshman through seniors are found in the course. Some have programming experience and some do not. Some have experience with various applications and some do not. To gauge the level of expertise of the students, we surveyed them on the first day of the class. They were asked about their proficiency with various applications, their programming experience, and their familiarity with various operating systems. We also asked them to provide technical questions they would like to be able to answer by the end of class.

We found that not only do students have a wide variety of previous technical experience, but also they are not sure what they do and do not know. As class professors we learned that we cannot assume the students know something just because they all indicated on their entrance surveys that they know it. Sometimes being in class teaches the students not only new knowledge, but also just how much knowledge is left to be gained.

Just like we do not assume that a student is as proficient as he thinks, we also do not assume that every student is at a beginner level for every topic. Since we cover such a wide variety of topics in the class, students usually find at least one section challenging and one simple. As a result, we give very open-ended assignments. Projects in the course tend to have a low baseline for minimal requirements with lots of optional ways to earn more points. For an assignment on creating a website, for example, a student with experience with Flash or Photoshop can add these elements to his site. Other students may instead do more research for the content of the website or in studying the university's Acceptable Use policy for web pages. Projects allow the student to exercise his or her own strengths while learning new technical material.

CHALLENGE: MEETING THE NEEDS OF NON-MAJORS

Having options for project requirements also helps the students in the course focus on acquiring skills that will be most relevant to their specific majors and interests. We encourage the students to put computing in the context of specific occupations by doing case studies and role-playing. Sometimes students are assigned roles, and other times they can choose roles to play. For example, one exercise is to have a mock "technology forum" to discuss the

effects of computing. Students are randomly assigned occupations such as librarian, computer programmer, artist, scientist, and teacher. The various groups then research the use of technology in their assigned fields, focusing on the changes in productivity, job availability, and job skills. They then report to the group, calling for changes in technology regulations or making suggestions for changes in their own field to respond to and leverage the ubiquity of computing. Exercises like these encourage the students to always be thinking about how the concepts in class affect them personally both now and after graduation.

One challenge of teaching applications in the non-majors course is knowing which applications the students will find most helpful in their careers. Some schools provide CS0 classes that are specific to a major. We found that the University of Evansville teaches their CS0 class "by special request" for specific disciplines. At our university we do not currently have support for creating different field-specific sections.

In order to give the students the opportunity to learn the applications of their choice, we decided to leverage the technology services already available on campus. The university's technology center offers workshops on applications such as Photoshop, Dreamweaver, Expression Web, Excel, Access, and Flash, all taught by students who work in the center. We require that the students attend a workshop, but we do not dictate which one. Each person not only learns a skill of his choosing, but also is exposed to the services that are freely available each semester.

CHALLENGE: MAKING THE CLASS ENGAGING

Our student focus group made a definite recommendation to make the class fun and engaging. We teach the class in a computer lab, and we are always using the computers in various ways. We not only use them to practice developing technical skills, but we also use them for communicating in class, accessing podcasts or other media, and for researching. Often when a student asks a question in class, we encourage him to look up the answer online and report back to the class. They learn to use the computer as a tool to acquire knowledge themselves.

We also use other hands-on techniques that do not involve the computer. To learn about circuits, we use "Snap Circuits", a toy by Elenco, which lets the student snap together AND, OR, and NOT circuits as an introduction to logic. When we teach the fundamentals of programming, we use LEGO Mindstorm robots. The CS1 class programs them in Java, while our CS0 class uses the drag-and-drop software interface to build flowcharts. They learn basic control flow constructs in a hands-on environment.

We also have several activities to get the students out of their seats and active in a very literal sense. For example, when learning about the Internet, students play the role of routers and pass around envelope "packets" with addresses on the outside and partial messages on the inside. Not only do students actively become familiar with TCP/IP, but the exercise provides a nice way to discuss Internet issues. For example, if a student drops an envelope, we discuss what happens if a packet is lost. The envelopes are intentionally left unsealed, and usually at least one student admits to sneaking a peek at a message not addressed to them. This leads to a discussion of Internet security.

CHALLENGE: ACCESSIBILITY FOR ALL

Making the class engaging and active also makes it more attractive to groups not traditionally interested in computing. Having hands-on objects like the "Snap Circuits" in class helps pull the focus away from using the computer for those who are intimidated. Also, using these objects in class helps give a tangible metaphor for abstract concepts. For example, Hasbro's "Lite Brite" illustrates the use of pixels to make a graphic. Combination padlocks are used to illustrate public and private keys in RSA encryption. Cards are printed with powers of two to introduce binary arithmetic. These techniques help those who learn visually as well as those who learn through metaphor. There is evidence that these techniques make computing more accessible to women in particular [7].

It is especially important for us to make our CS0 course accessible to attract and retain women. An Arthur Andersen survey of teens found that "while both boys and girls agreed that it's important to understand computers for future employment, boys were five times more likely to be interested in majoring in computer science or computer engineering than girls" [6]. Anita Borg created the Institute for Women and Technology in order to work with Universities to make their programs more appealing and relevant to women. "If women are not in an environment in which their own style of genius and contribution can be recognized," warns Borg, "they will not thrive and may not stay" [2].

Finally, we try to convince the students that the field of computing is exciting and relevant to everyone. As Dan Reed, chair of the Computing Research Association, reports in his blog, "I believe we must rethink our computing education approaches in some deep and fundamental ways. First, as researchers and technologists we seek to reproduce students in our technical image, failing to acknowledge that most of our students will not develop compilers, write operating systems or design computer chips. Rather, they benefit from training in logical problem solving, knowledge of computing tools and their applicability to new domains." [3] Literacy seems to be a key trend here, making computing relevant to all aspects of education.

In our CS0 class, the students read a current article on technology each week. These articles are on issues like green computing, the digital divide, security, and electronic voting. These articles are written for general audiences and do not require technical knowledge. The students then post their opinions on the issue in a blog and we hold a discussion of the issue in class once a week for about fifteen minutes. The students are graded on their blog post as well as their discussion participation, ensuring that they tie the article into the class topics.

CONCLUSION: OUR GOALS REVISITED

We have addressed many challenges in our CS0 class and feel that we have ended up with a course that meets the needs of our non-majors. For others facing this task, we advocate a breadth-first approach, teaching mainly literacy, with a very small coverage of programming and application skills. We also recommend stressing societal issues and using open-ended projects that allow students to use their personal interests and strengths to make computing relevant to them. Finally, we support using hands-on, active learning to make the topics interesting and fun to those not traditionally attracted to computing.

The CS0 course is in its fourth year and always fills to capacity. The "success" of the course can be measured when we convert non-majors to majors. This has happened with several students, mostly women. We have also had several majors enroll in the course. The course will continue to evolve as the topics and roles in computing change.

REFERENCES

1. Beyer, S., Rynes, K., & Haller, S. (2004, Spring Volume:23, Issue 1). Deterrents to Women Taking Computing Science Courses. *Technology and Society Magazine, IEEE* , pp. 21-28.

2. Borg, A. (2002, June Volume 34, Number 2). Computing 2002: Democracy, Education, and the Future. *SIGCSE Bulletin*. pp 13-14.

3. *Computing Research Policy Blog*. (2008, February 11). Retrieved February 24, 2008, from Computing Education and the Infinite Onion: http://www.cra.org/govaffairs/blog/archives/000657.html

4. (2003). In R. Decker, & S. Hirshfield, *The Analytical Engine: An Introduction to Computer Science Using the Internet* (p. 384). Course Technology.

5. *Girls Geek Chat*. Retrieved February 24, 2008, from GirlGeeks Best of Chats Series with Anita Borg: http://www.girlgeeks.org/chat/borg.shtml

6. Keller, L. (2001, January 17). *Working on the pay gap - He clicked, she clicked*. Retrieved February 29, 2008, from CNN.com - Career - He clicked, she clicked: http://archives.cnn.com/2001/CAREER/trends/01/17/techies/index.html

7. Pollard, S., & Duvall, R.C. (2006). Everything I Needed to Know About Teaching I Learned in Kindergarten: Bringing Elementary Education Techniques to Undergraduate Computer Science Classes. *Proceedings of the Thirty-Seventh SIGCSE Technical Symposium on Computer Science Education*. 224-228.

8. *Women in CS: the dance remix version* . (2005, December 20). Retrieved February 24, 2008, from See Jane Computer: Women in CS: the dance remix version : http://seejanecompute.blogspot.com/2005/12/women-in-cs-dance-remix-version.html