BEGINNING PROGRAMMING – WHY TEACH IT AND HOW TO TEACH IT?

Jeanne M Baugh, Robert Morris University, baugh@rmu.edu

ABSTRACT

More is attained from learning to program a computer other than just writing correct code. Because computers are infused prominently in all of our daily lives, not only Computer Information Systems majors, but all students should take a programming course. This paper investigates what students really "take away" from an introductory programming course as well as what methods could be employed to successfully teach such a course. A successful model of a large project-based introductory course in computer programming and strategies to allow the students to succeed with such a large project are discussed. Issues of assessment and avoiding student cheating are also stressed. The students in this course had little to no background in programming prior to taking the course defined in this research. The systematic process of creating a well-designed project leads to a significant coding effort on the part of the first time programmer. The students not only learn to program, they are proud that they learned to program!

INTRODUCTION

Traditionally the Information Systems program has had the requirement of programming. The bulk of IS business schools believe in the necessity for programming and database skills (Apigian, & Gambill, 2010). Approximately 99% of these schools offer at least one programming course and all offer database. Likewise, the 47 accredited ABET programs offer multiple programming courses as well as database (Accreditation Board for Engineering and Technology (ABET) http://www.abet.org). In research on programming skills, Aasheim et al found that there is an expectation of industry that students must know programming (Aasheim, Shropshire, & Kadlec, 2012). But in the Information Systems Model Curricula, programming was omitted from the list of requirements for an IS degree (Topi, Valacich, Wright, Kaiser, Nunamaker, Sipior & Vreede, 2010). Many business schools and ABET accredited programs do not agree with this decision. Additionally, many information systems educators find it difficult to understand the thought process of IS curriculum designers in not requiring programming. Few would argue that information systems students should study both systems analysis and data management. Both areas are not only tied to programming, but both are dependent on programming. At the Author's institution, IS students are required to take two semesters of a programming language.

Studying a programming language does more than just teach the student to write a syntactically correct program (Babb, Longenecker, Baugh & Feinstein, 2013). It teaches the student how the computer makes decisions, stores data and basically how the machine that is so pervasive in our daily lives works. We all need to adopt a new attitude: everyone can do it. The web site code.org supports a non-profit organization dedicated to promoting computer science (specifically computer coding) as a requirement for all students (http://code.org). Their vision states that "every student in every school has the opportunity to learn how to code." Additionally, the organization supports the view that "computer science and computer programming should be part of the core curriculum in education, alongside other science, technology, engineering, and mathematics (STEM) courses, such as biology, physics, chemistry and algebra". Leaders in all phases of Industry and Education advocate for all students learning to code. In a study by Wang, programming requires thinking with abstract concepts, which is difficult for novices (Wang, 2010). Secondly, programming includes many different tasks, such as problem solving, algorithm and data structure design, programming language comprehension, testing, and debugging. Writing a computer program is an exercise in logic. Since the basics of all computer work centers around these logical foundations, programming is fundamental to everything in computing (Babb, Longenecker, Baugh & Feinstein, 2013).

STUDENT SURVEY

One hundred eighteen students enrolled in beginning programming courses (C++ and Java) for the Fall 2015 and Spring 2016 semesters were surveyed at the end of the course to ascertain their perceptions of the course. Of the students surveyed, 75% reported that they were required to take the course and 67% will be signing up for another programming course. At the Author's University, CIS students are required to take two semesters of a programming language.

Before taking the course, 58% of the students reported that they had no idea what a computer program was. It does make sense that after taking the course, only 8% reported that they still had little understanding of computer programming. The final grades reflect this finding, in that 11% of the students were given a D or an F. Perhaps an even more telling statistic of these first time programmer's view of the programming course is their view of the job of a computer programmer. They reported that prior to taking the course 40% saw a programmer as being "nerdy". At the conclusion of the course, only 3% of the students felt this way and 95% reported that they actually saw programmers as being skilled. (2% saw programmers both skilled and nerdy)

Figure 1. shows that only 11% stated that the course was not relevant to their area of study. This result is interesting, because 64% of the respondents are NOT Computer and Information Systems majors. The enrollment in these courses comes from Engineering, Education and other majors and only 17% percent were freshman. Students saw that studying programming is relevant to whatever their area of study is. We are living in a digital world and many feel that learning to program is an essential skill to possess.
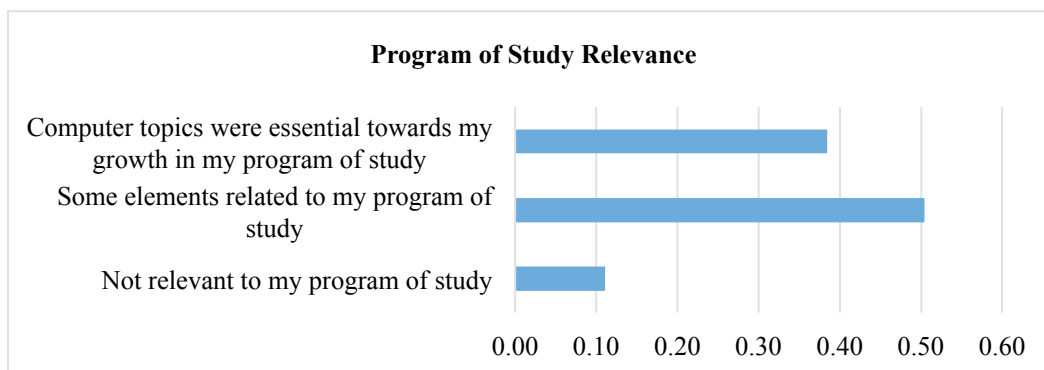

Figure 1. Program Relevance

There are many advocates for requiring programming not only for the Information Systems major, but for all students: (http://www.code.org/quotes)
- Susan Wojcicki, Senior Vice President, Google - "Learning to code makes kids feel empowered, creative, and confident".
- Bill Gates, Chairman, Microsoft -"Learning to write programs stretches your mind, and helps you think better, creates a way of thinking about things that I think is helpful in all domains".
- Dr. Mehmet Oz, Cardiothoracic Surgeon, Author, and TV Personality-"To best prepare for life in the 21st century, today's students should ideally learn basic computer programming".
- President Bill Clinton-"At a time when people are saying "I want a good job - I got out of college and I couldn't find one," every single year in America there is a standing demand for 120,000 people who are training in computer science".
- Mike Bloomberg Mayor, New York City-"We salute the coders, designers, and programmers already hard at work at their desks, and we encourage every student who can't decide whether to take that computer science class to give it a try. New York City's economic future depends on it, and while we're already giving thousands of our students the opportunity to learn how to code, much more can and should be done".

- Tony Hsieh, CEO, Zappos- "I think everyone should get a little exposure to computer science because it really forces you to think in a slightly different way, and it's a skill that you can apply in life in general, whether you end up in computer science or not".

Figure 2. indicates the student's comprehension of how the computer stores data, the binary number system. Many students reported that they had "heard" of bits and bytes, but really never knew what they were until encountering that topic in a programming course. But after the course, their understanding has greatly increased.
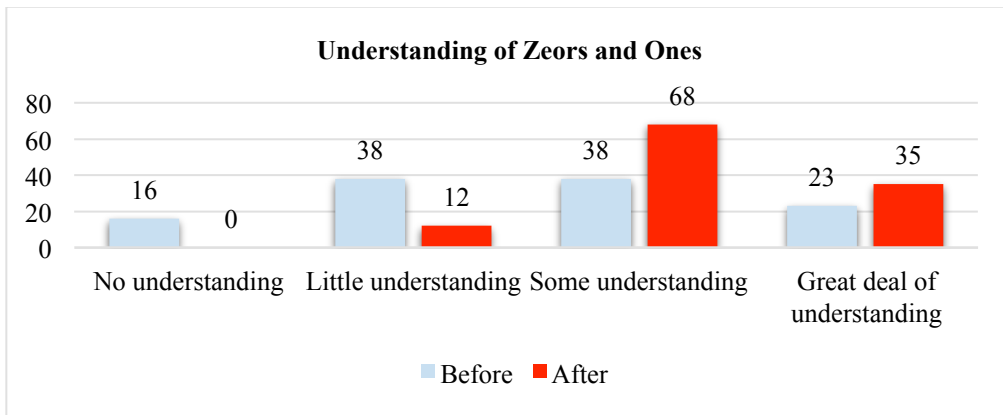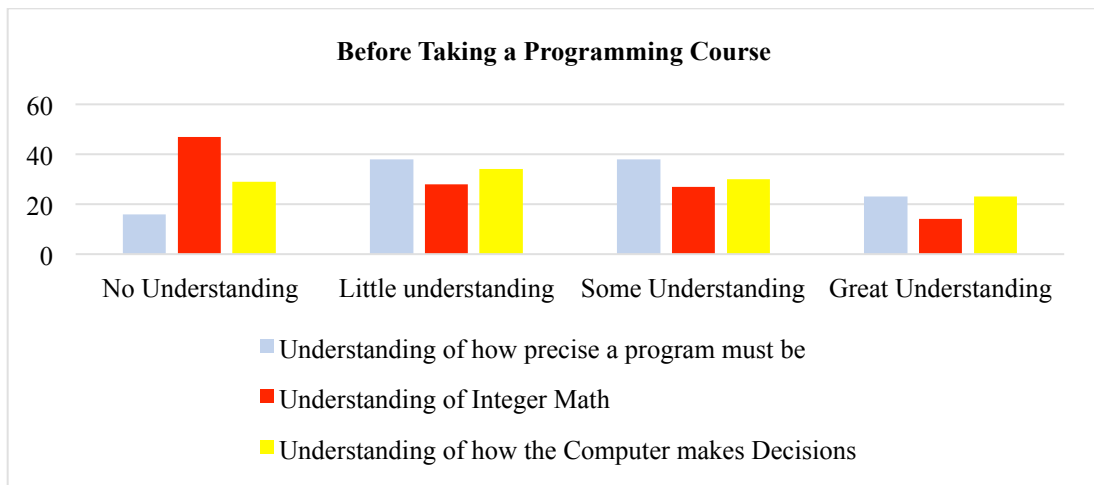

Figure 2. Understanding of Zeros and Ones

Other areas that were surveyed are included in **Figure 3**. In the all of the areas surveyed, students reported that their understanding increased.
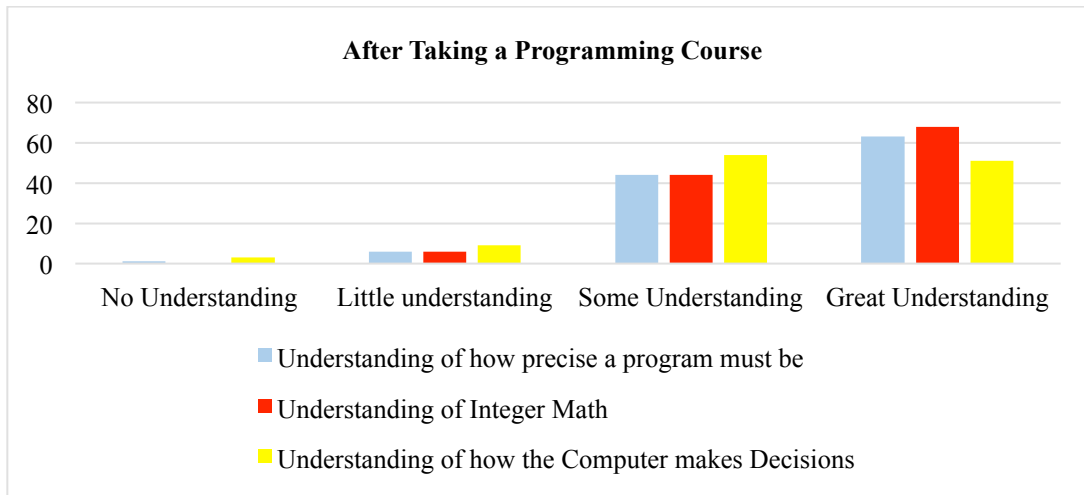
After Taking a Programming Course

Figure 3. Survey Results- Before and After Taking a Programming Class

Many of the students also reported a sense of accomplishment, pride, joy, relief, etc. in getting their program to successfully execute. Those of us who teach programming have experienced this from our students and it is gratifying to see these results. Some of the comments included:

- "The finished product; nothing beats the sense of accomplishment when a program finally works"
- "The fact that I was able to create and execute a working computer program of my own was the greatest feeling"
- "The feeling of actually understanding a concept, and being able to apply it"

TEACHING PROGRAMMING USING A LARGE PROJECT

The Literature presented here and the survey results have shown that programming is important. There are many ways to teach programming: individual projects, group projects, cohort projects; tutoring approach; online training; assignments from the course textbook; assignments from the real world; etc. The survey reported here specifically shows that this set of students learned a great deal more than how to write a program. Therefore, this approach is very successful, but a number of issues must be addressed.

Project Specifications

Assigning a large project to computing students is not new (Joy, 2009). However, large programming projects are usually assigned to a more seasoned programmer such as a capstone course given to CIS seniors (Mitri, 2008). The courses that were surveyed for this paper were taught using an approach that by the end of the semester, these beginning programming students completed a very large programming project. Because students need to learn so many tasks when starting out with programming, many Instructors do not often require their students to write a large project. In fact many advanced concepts must be mastered in order to write a large project, including methods and classes.

The project (for both C++ and Java) included: reading/writing data files, parallel arrays, methods/functions, class definition and intricate searching of multiple data sets. These beginning programming students ended up with a programming project that consisted of at least 1200 lines of code. This alone explains why the students felt very proud of their programming effort. One student said "The sense of accomplishment when you put so much time and effort into it is amazing". Students also reported that they liked learning how computers work. One student actually was in tears when telling this Author how proud they were of the code they had written.

First time programmers can succeed with a large project (Baugh & Kovacs, 2012). This Author has been teaching beginning programming for many years and always assigns a large project as defined here with a great deal of success. The key to all of this is the order in which the programming skills are taught. The students were first given about 5 small programs covering the basic programming skills. When the project started, they knew the basics and new skills would be taught as needed. For example, when the students started the project, they had not been taught file processing, arrays or classes yet. These skills would not be needed at the very beginning of the project. As the project continued, additional skills needed for the next project phase were taught. An overview of the topics taught and assignments given are as follows:

1. Program 1: assignment statements, simple math operations, simple input/output
2. Program 2: more on math operations and input/output
3. Program 3: if statements
4. Program 4: loops
5. Program 5: simple methods/functions
6. Project Phase I: This phase of the project required the student to create a menu that called a specific method stub depending on the menu option selected by the user.
7. Project Phase II: Code was added to Phase I. This phase opened three text files and read data into three sets of parallel arrays. In this phase of the project, the students were also required to create specific reports based on the data read in. Various methods already created as stubs were to do the required work.
8. Project Phase III: The user next added a used defined class and incorporated all methods and major data as members of the new class. Additional requirements were also to add, delete and modify the data in the parallel arrays. All modified data in the three parallel arrays were then written back to the three data files. As stated in Phase II previously, code was added to some of the stubs from previous phases of the project
9. Project Phase IV: This final phase involved writing advanced reports involving all three sets of parallel arrays. The reports required the student to create a loop within a loop, within another loop to search for specific data from all three sets of arrays at the same time.

Project based learning has been shown to increase student engagement (Larmer, 2014). It is the Author's experience that this is truly the case. Most of the students are extremely interested in getting all of the coding to "work". But teaching a course such as this does involve a great deal of time for the instructor. This Author writes many sample programs to illustrate programming topics beyond what is provided with the textbook. Many hours are spent helping students debug code via email as well as during office hours and lab time. It is very important for these first time programmers to feel that there is someone there to help with any programming problems they may encounter. When using a new teaching tactic, such as the project defined here one may need to make adjustments. For example, the Author has made some project changes, such as moving some of the functionality around in the various phases of the project (Voyles, Durik, & Bailey, 2015).

Tutoring on campus can also be extremely beneficial. Good students should be encouraged to sign up to be tutors. Interestingly, it is a wonderful way for the student who is doing the tutoring to increase their own coding skill set. Debugging someone else's code is difficult and helps both students on each side of the tutoring process.

HOW TO PREVENT CHEATING IN A PROGRAMMING COURSE

Some studies suggest that students in many introductory courses are cheating (Marsan, 2010).At Stanford University, 23% of their honor code violations involved introductory computer programming courses; while these computer science students make up only 6.5% of the student body. At the University of Florida, 97 students were caught cheating on programs that were turned in as a final exam. The students were even warned that there were "markers" in the files and they could be easily caught cheating. Various penalties were issued based upon the willingness of the student to admit to cheating (Alcantara, 2012). Some possible reason for cheating in computing programs could be the belief by students that software code is regarded in the same manner as the solution to a math problem (i.e., having one correct answer). Students writing computer programs may feel compelled to provide the "correct" answer rather than applying creative solutions to solve a problem scenario. Other reasons given for the

increased cheating by IS/IT/CS students include the common practice by professors to re-use past programming assignments (Simkin & Mcleod, 2010).

According to a 2014 study, "some students justified helping friends they are close to or who they observe as having difficulties, others considered it immoral and refused to be involved in the activity of cheating".

Those who teach programming know that some of the students do cheat or try to cheat. This Author has employed a few specific methods in the beginning programming course to discourage cheating.

1. Occasionally, students will report to the Author that they worked together, which really could mean that one person did the programming and another copied the code. This Author randomly require the students to explain their code. This can be done either verbally or in a written manner. If the student did not write the code, they can not explain it. This policy of having the students randomly explain the code should be included on the course syllabus.
2. Students are required to turn in their final project code the week before finals. Some of the code is taken from the student's submission and placed on a final for that specific student. An individual final is created for each student by asking them to thoroughly explain each line in the selected section of code. This is time consuming for the instructor, but the results are indisputable. Again, they cannot explain it if they did not write it.
3. As all programming instructors know, there are often multiple correct ways to write a program to solve problem. This Author is very specific about how the program is to be written (data structures, input/output specifications etc…). But, all of the instruction details are not written in the assignment document. They are explained verbally, during class. At first, this might seem like an unusual policy. This Author has implemented this policy because some students have gone outside the University to pay others to write their code. In many cases, the code does work, but uses various concepts that had not been taught and clearly were written by someone other than the student. Often the programs are written using very advanced methods. Just this past semester this Author asked 6 different students to explain very complicated code they had turned in for their project. They had no idea of what their code was doing and therefore received a zero for the project.
4. Having the student write code on the computer is this Author's preferred method to test the student's ability to program. But it is very easy for one student to send their code to another student during the exam. With these types of exams, this Author has written 6 different versions of the exam. This does not stop cheating all together, but it does make it more difficult for the student to copy code from another person. Two different version of such an exam are as follows:

   C++ Version 1:
   In the main function, add parallel arrays to store the data on four monthly bills. You may enter whatever data you like. The data you are to store on these four bills is as follows:
   - Name of the bill   (string)   rent, food, gas, electric, phone etc…
   - Amount of the bill   (double)
   - Days past due   (integer) number of days you are overdue in paying
     a) Change the stub read_data to read in all data
     b) Change the stub print_data to print all data
     c) Change the stub calculate_data to print the total of all bills due and the average past due days. Print these two things in this function.
     d) Change the stub find_data to find the name of the bill that is the highest amount due and pass it back to the main function. Print this bill name from the main function.

   C++ Version 2:
   In the main function, add parallel arrays to store the data on four pets you own. You may enter whatever data you like. The data you are to store on your pets is as follows:
   - Pet Name   (string)
   - Age of Pet   (integer)
   - Veterinarian bill   (double)
     a) Change the stub read_data to read in all data
     b) Change the stub print_data to print all data

c) Change the stub calculate_data to print the total vet bill for all of your pets and the average age of your pets. Print these things in this function.
d) Change the stub find_data to find the name of the pet that has the highest vet bill and pass the name back to the main function. Print this pet name from the main function.

Students can be very creative if they are determined to cheat (Marcoux, 2010). Therefore, instructors must rise to the challenge to combat the epidemic. Preventing cheating is an area for further study. This Author plans on implementing more oral defenses of the students work. If the student cannot explain what he turned in as his own code, then he did not write it!

CONCLUSION

Learning to program is the cornerstone of any Computing Curriculum. Even if the student plans to never program again, they learn how the computer works. In this digital age, it is essential to possess this knowledge. In an old study from 1978, Edwards stated "Using the computer as a problem-solving, teaching tool benefits the process of learning in a variety of ways" (Edwards, 1978). This concept still holds today. Coding helps to nurture problem-solving skills, logic and creativity. The movement "Hour of Code" facilitates the idea that everyone should learn to program. ("The Hour of Code is a one-hour introduction to computer science, designed to demystify code and show that anybody can learn the basics") (https://hourofcode.com/us). "The Labor Department predicts that tech jobs will grow faster than the average for all jobs at a rate of 12 percent this decade" (http://money.usnews.com/careers/best-jobs/rankings/best-technology-jobs)

"Computer programmers are in great demand by American businesses, across the tech sector, banking, entertainment, you name it. These are some of the highest-paying jobs, but there are not enough graduates to fill these opportunities" (Rubio, http://www.code.org). Therefore, we who teach programming must find ways to get our students excited about their programming course experience.

REFERENCES

Aasheim, C., Shropshire, J., Li, L., Kadlec, C. (2012). Knowledge and Skill Requirements for Entry-Level IT Workers: A Longitudinal Study, *Journal of Information Systems Education. Summer, 23*(2), p.193-204.

Accreditation Board for Engineering and Technology (ABET) http://www.abet.org

Apigian, C.H. and Gambill, S.E. (2010). Are We Teaching the IS2009 Model Curriculum? *Journal of Information Systems Education, 21*(4), p.411-420.

Barua, A., & Barua, A. (2012). Gendering the digital body: Women and computers. *AI & Society, 27*(4), 465-477. doi:http://dx.doi.org/10.1007/s00146-012-0371-9

Baugh, J.M., Kovacs,P. (2012). Large Programming Projects for the beginning programmer, *Issues in Information Systems, 13(1*), pp. 85-93.

Baugh, J.M, & Kovacs, P.J., (2014). Programming Not Required? Did The IS-Model Curriculum Get It Right? *Issues in Information System, XV*(1), 399-408.

Chen-yao Kao, Soc. (2015). Mathematically gifted adolescent females' mixed sentiment toward gender stereotypes. *Psychol Educ, 18,* pp. 17–35. DOI 10.1007/s11218-014-9278-2

Davis, G. A., & Rimm, S. B. (2004). Education of the gifted and talented (5th ed.). Boston: Allyn & Bacon.

Kurtz-Cortes, B., Rowley, S. J., Harris-Britt, A., & Woods, T. A. (2008). Gender stereotypes about mathematics and science and self-perceptions of ability in late childhood and early adolescence. *Merrill-Palmer Quarterly, 54*(3), 386–409.

Miles, R. Y. (2009). The challenges and factors that determine female career choices in information technology professions: An exploratory investigation (Order No. 3378954). Available from GenderWatch; ProQuest Dissertations & Theses Global. (305161772). Retrieved from http://search.proquest.com/docview/305161772?accountid=28365

Mishra, S., Caputo, D. J., Leone, G. J., Kohun, F. G., & Draus, P. J. (2014). The role of awareness and communications in information security management: A health care information systems perspective. International Journal of Management & Information Systems (Online), 18(2), 139-138. Retrieved from http://search.proquest.com/docview/1516951161?accountid=28365

Preckel, F., Goetz, T., Pekrun, R., & Kleine, M. (2008). Gender differences in gifted and average-ability students: Comparing girls' and boys' achievement, self-concept, interest, and motivation in mathematics. *Gifted Child Quarterly, 52*(2), 146–159.

Raines, L. (2011). *Technology in need of more women*. Retrieved from http://search.proquest.com/docview/822839988?accountid=28365

postmedia, n. u. (2011*). Number of women in technology fields declining*. Retrieved from http://search.proquest.com/docview/878710386?accountid=28365

Ramnanan, G. P. (2001). *A study on increasing the interest and involvement of females (ages 14 to 18) in technology through skills development and mentoring*. Retrieved from http://search.proquest.com/docview/304772204? accountid=28365

Topi, H., Valacich, J., Wright, R.T., Kaiser, K.M., Nunamaker, J.F., Sipior, J.C., and Vreede, G.J. (2010). *IS 2010 Curriculum Guidelines for Undergraduate Degree Programs in Information Systems*. Retrieved from http://www.acm.org/education/curricula/IS%202010%20ACM%20final.pdf