Expanding Participation in Computing: Including Programming Principles in a Non-Majors Course

Paul M. Leidig
paul.leidig@gvsu.edu

Roger C. Ferguson
roger.ferguson@gvsu.edu

John H. Reynolds
john.reynolds@gvsu.edu

School of Computing and Information Systems
Grand Valley State University
Allendale, Michigan, 49401, USA

Abstract

Computing has created new ways for people to connect, research, create, and express themselves, and has changed the world in profound ways. However, simply being proficient in computer use often misses the greater power of computing. An empowering experience begins when one learns how to translate new ideas into computer code, thus becoming a producer of information or minimally a much more knowledgeable user of that information. Recent efforts to introduce non-computing majors to these concepts focused either on non-programming computer use skills, or straightforward computer programming languages. Previous research has shown many general education programs offer one of these choices, but few require any programming, nor introduce programming within the larger view of computing. Recent publications seek to stimulate discussion about whether computing courses for non-computing majors should have a programming component. This paper proposes what is most relevant are the big ideas of computing, such as programming fundamentals, using abstractions and algorithms, working with very large data sets, and issues of cybersecurity. The primary purpose of a non-majors course should be to introduce a wider range of students to the expansive world of computing. Recent programs designed to introduce these skills, such as those by code.org and Advanced Placement Computer Science Principles, have been successful and provide a model for university level efforts. To assist in these efforts, this paper proposes creating and delivering a successful non-major, or general education computing course, at the university level patterned after the AP CS Principles course.

Keywords: General Education, Non-Majors, Programming Courses.

1. THE PURPOSE OF GENERAL EDUCATION

With few exceptions, colleges and universities require students to take a collection of courses outside of their major, that allow students to expand their knowledge in areas such as social sciences, history, humanities, mathematics, and sciences. Normally referred to as general education (GE) courses, they provide students with the ability to understand a variety of disciplinary perspectives. This paper uses the generic term "non-major" courses to refer to GE

courses taken outside of the required courses in a chosen discipline. At the micro level, these non-major courses help students understand issues and challenges outside of their chosen major.

A graduate with a general education based degree has the ability to think in broad terms, that is, outside of their respective major. Before graduation, every student will have had a set of courses that prepares them to think critically outside of their major. A graduate with a general education should have the ability to recognize and work with issues and/or problems from multiple perspectives.
For technical majors (computing, engineering, etc.), a GE program serves its intended purpose since the course work is often significantly outside of their major. Courses in ethics and social sciences help future technical professionals make decisions that affect the direction of their respected field. Courses in the arts and humanities offer students a unique perspective in their own culture and the cultures of others. Unfortunately, many students outside of the technical fields may not get the same level of reciprocal benefit from general education. The reason is that many non-technical majors already take courses in areas that general education typically covers. These majors are missing out on courses that would broaden their perspective. For example, courses in computing could begin to bridge this gap; resulting in a significantly enhanced general education for non-computing/non-technical majors. There is a valuable benefit to exposing students in non-technical majors to programming. By having non-majors take courses in the computing field, the number of students exposed to the possibility and value of a minor or major in the computing discipline may increase.

2. COMPUTING AS PART OF GENERAL EDUCATION

The following section reviews the current state of programming offered to non-majors. An obvious place to find these computing courses is in the General Education requirements at each university. Ferguson, Leidig and Reynolds, (2014) showed less than one third of Information Systems (IS) accredited programs offer a programming course in the GE program. Further, virtually no university requires a computing course before students graduate.

Early computing courses focused on programming languages and logic. However, with the advent of personal computers (late 1980's and beyond) and the use of purchased applications, universities shifted the focus to teaching how to use computers, often referred to as "computer literacy." (Rushkoff, 2012) This shift in focus occurred at the expense of understanding how computers work.

To better understand the current state of computing education, this paper expands previous work by analyzing a larger sample size (150 institutions across the United States). An analysis of the requirements of the GE programs shows some of the common components in all the GE programs. Finally, to support previous background work (Ferguson, et. al.) this paper also shows that the most common components of a GE program could be easily satisfied with a programming/computer principles course.

Although there are literally thousands of universities throughout the US, this analysis focused on 150 academic institutions that were represented at the AITP EDSIG IS Education Conference over the past three years. Each university was examined, either by downloading the catalog, or observation of their GE program via the their web page. At each institution, an analysis was done to determine the requirements / goals / objectives of the GE program. The following list shows the many different objectives/goals found at the institutions surveyed. This list shows that there is a wide range of GE requirements across these 150 institutions.

Summary of common terms General Education Objectives and Goals.

- Promote collaboration, teamwork, leadership
- Problem solving skills, critical thinking
- Basic mathematics skills
- Written, oral communication, languages
- Ethics, equality, philosophy, religious studies
- Global history, contemporary issues
- Citizenship, government, social awareness
- Humanities, arts, music, performance
- Multicultural, global concepts
- Scientific, quantitative reasoning
- Using existing and new technologies
- Reading comprehension
- Natural, physical, social sciences
- Information and digital literacy

A common core was created from the GE requirements shared among the different institutions. Similar objectives were totaled and ordered, from most to least common. For

example, the skill of communication is a component shared by almost every institution. This list represents a common theme among all GE programs.

1. Collaboration, teamwork
2. Communication - written and oral
3. Quantitative reasoning, mathematics skills
4. Problem solving skills, critical thinking
5. Humanities
6. Natural, physical, and social sciences
7. Ethics
8. Global history, contemporary issues
9. Information and digital literacy

3. ANALYSIS OF INSTITUTIONAL REQUIREMENTS

How institutions handle computer competency or programming in their GE program is shown in Table 1. The following statement summarizes the overall results of the investigation of these 150 universities: There is an opportunity for universities to change their non-majors requirements (i.e., GE) programs to at least offer some form of computing in GE.

Step	Description	Totals
1	Total number of institutions examined.	150
2	Number remaining after eliminating institutions due to lack of University wide GE requirements and/or poor information availability.	114
3	Number of institutions with no investment of computing, i.e., terms for “computing” not found in GE.	53
4	Number of institutions with some competency requirement within GE	27
5	Number of institutions with significant investment of computing within GE that includes programming.	34

Table 1, Analysis of Institution's GE Programs

Before analysis of the data was done, some of the institutions did not qualify. Taking a closer look at step 1: First, over 36 institutions were eliminated for the following reasons: (1) There were no GE requirements for the university, that is, GE was at the department level and varied across the university. (2) Only students of the universities had access to GE requirements. (3) The university web page access or catalog did not show the requirements. These institutions were eliminated from the analysis and results. Hence, 114 institutions were remaining for analysis (step 2) and included in the results.

In step 3, 53 out of 114 (approx. 45%) of the institution's GE program had no mention of the word computing, no competency requirements or course equivalency. The authors could not find the word computing (or equivalent) in the GE program. These results again support the previous work done by Ferguson, Leidig and Reynolds. Though these results were not surprising, it does show there is an opportunity at most institutions for computing departments to become more involved in the GE program. The benefits to the computing department could be significant. The number of students introduced to computing could be greatly expanded. Perhaps more importantly, every non-technical major would receive the value of a more well-rounded education.

In step 4, institutions are identified that see computer use as a skill that is important for every major. In other words, students are required to take some computing literacy course as a part of their major. This group was 27 out of 114 (approx. 25%) that contained a computer competency statement or class that "most likely" did not have any form of a programming component within GE. Also contained in this group were the GE programs that typically had a course that taught spreadsheets, word processing, etc. that was needed for a non-computing major to function within the field of choice. This section was the GE programs that have the philosophy that computing is a tool, not a field of study.

Step	Description	Totals
5.1	Number of institutions with significant investment of computing within GE (from Table 1, step 5)	34
5.2	Institutions that offer a set of courses that include a programming option in GE. An undefined set of classes.	15
5.3	Offers 1 or 2 courses that map to GE universities requirements.	19

Table 2 An Analysis of Programming in GE

Finally in step 5, we identify 34 out of 114, (approx. 30%) institutions that offer the choice of a programming course within their GE requirements. These institutions typically offer a computer literacy course as well. Unfortunately, there is no "standard approach" on the best way

of offering computing courses (either programming or literacy). Table 2 shows a deeper analysis of the institutions in step 5.

Within this group, about half the universities, 15 out of 34 offer a set of courses (3 or more) in a range of topics within the computing field.

However, the issue with this approach is that students may not be getting a full GE experience. In other words, there appears to be no mapping between what the course offers, to GE requirements for a course to be part of the GE program. The remaining 19 universities have at least one course within the GE program that appears to have a mapping to GE requirements. Thus, less than 20% of the universities that offers programming in GE map the programming objectives into the GE requirements. The reciprocal finding then could be stated, over 80% of students in general education programs do not benefit from an introduction to programming principles.

4. THE NEED FOR COMPUTING SKILLS

Although one may identify many benefits to students and society from learning how computers work, the most obvious is the increasing talent gap in the demand for computing skills, and the number of students interested in majoring in these fields. This skills gap has been well documented over the previous decade. Of all 2010 high school graduates who were tested on the ACT (ACT, 2010), the percentage of those who indicated a career interest in these fields was less than projected demand. This is most obvious when compared to the Bureau of Labor and Statistics projected job openings for 2008-2018 (BLS, 2015). Figure 1 shows this gap, indicating a demand for computer specialists that is 5.5 times the potential supply – the largest gap among all career interests.

Although there is almost universal acknowledgment of the skills gap for computer specialist, there is little agreement on a primary cause of the problem, and even less on the best solution. However, most proposed solutions (Rushkof, 2012) include the need for increasing the exposure to computing principles and programming skills. While there are many calls (Rushkof, 2012) for increasing the number of graduates in computing disciplines, there has not been equal recognition of the benefit of exposing a much broader population to basic computing principles. Recently however, numerous efforts have been taken to generate awareness and understanding of programming as a valuable skill. Some of these efforts have been directed at the general public, such as code.org Code Literacy, and codeacademy to name a few. These efforts have received significant press, including President Obama taking part in "hour of code" lessons to highlight the need for programming knowledge. Certainly these efforts should help raise the awareness of programming. It remains to be seen whether the skills gap will be reduced with these efforts.

Beyond the need for a work force with programming skills, an information based society also calls for a general public with an understanding of how computers do what they are being asked to do. An excellent example of this need is well documented in books such as *Program or Be Programmed (*Rushkoff, 2010), and *The Filter Bubble; How the new personalized web is changing what we read and how we think (*Pariser, 2011). Publications such as these make the case of how important it is that society in general, and an educated workforce, have an awareness of basic computer programming in order to understand how they work and live in their chosen fields. Their arguments help make the case why all students in a general education curriculum should be introduced to computer and programming principles.

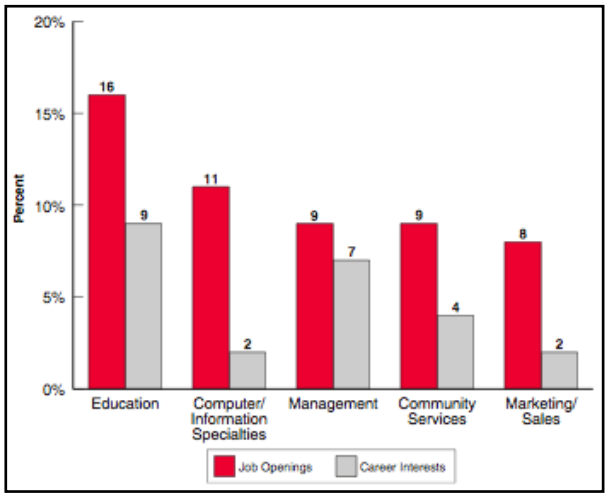

Figure 1 Projected Job Openings versus Career Interests

A less publicly visible effort by The College Board is currently underway to create a new Advanced Placement (AP) computer science course and exam. This new AP Computer Science Principles course is designed to present students with foundational computing skills, an understanding of the real-world impact of computer applications, and programming principles. This

is in contrast to the existing AP Computer Science A course and exam that explicitly addresses programming in Java. Such a course is needed in an effort to increase the number of students interested in computer science and other STEM fields. This new course is designed to introduce a wider perspective of computer science to a wider range of students. The aim is to broaden interest in and participation in computer science related fields. Further, a fundamental concept of AP courses/exams, is that they provide course content and credit for transfer as an equivalent university level course.

5. PROPOSED SOLUTION

With the financial support of the National Science Foundation (NSF), the College Board (CB) began revising its Advanced Placement courses a few years ago. The new AP CS-P course introduces students to the broader concepts of computer science. Focusing on creativity and problem solving, students in this course attempt to solve real-world applications. With this approach, the hope is to attract students who are traditionally underrepresented with this broader, multidisciplinary approach.

General Education Program Goals	maps to	AP CS Principles Computational Thinking Practices
1. Collaboration and teamwork		#6 Collaborating
2. Communication; written and oral		#5 Communicating
3. Quantitative reasoning, mathematics skills		#3 Abstracting
4. Problem solving skills, critical thinking		#2 Creating Computational Artifacts #3 Abstracting #4 Analyzing Problems and Artifacts
5. Humanities		
6. Natural, physical, and social sciences		
7. Ethics		#1 Connecting Computing
8. Global history, contemporary issues		
9. Information and digital literacy		#1 Connecting Computing

Table 3 Mapping of GE Criteria to AP CS-P

The anticipated benefits to the computer science field are potentially enormous. The CB task force identifies several key benefits, including the following:

- Students can experience the "joy, beauty and awe" of computing".
- Traditionally under-represented students in computing will be exposed to the field.
- Students, most of whom have no idea what computer science is, will be able to differentiate between using versus creating an application.
- The creative potential of computing that benefits society and scientific advancement.
- Students who take an advanced placement test in a subject area are, according to the CB, more likely to take a college course in that subject area.
- Students taking the course will be better prepared in their discipline, given the role computing plays in most careers.

With these potential benefits of an AP computing principles course, the authors propose using this course and curriculum for a university-based course within the GE curriculum. Table 3 maps the direct connection of this course to the common criteria in GE programs.

6. CONCLUSIONS

The results of this paper can be summarized in the following: About half the universities have no computing competency or programming class offered in GE. (In many cases, the word computing was not found in the GE program documentation). 30% of the universities have computer programing as an option (not required) within GE, but do not provide a specific path to provide that option. This paper supports the previous work by reviewing a large sample of data (150 universities involved in the analysis). Further, this analysis drills down deeper into how different universities address computing within GE.

In addition, the authors attempt to make the case that computing within all GE programs is needed to produce well-rounded graduates. More specifically this paper also demonstrates the need for computer programming as an option within GE. Finally, the paper has shown through an analysis of 150 universities that there is no agreement about the role of programming within GE. Of the universities that do have programming as a part of GE, the lack

of a standard way to how best offer this option is very apparent from the data.

The reason for a lack of best practices may be as simple as computing has not been part of GE in the past, or computing is not considered to be necessary or a beneficial part of GE. Whatever the reason (more research is needed to determine the reasons) the fact remains, that most institutions just do not offer much in the area of computing within the GE program.

Finally the authors demonstrate the connection between computing programs like the Computer Science Principles course and common general education objectives. The authors propose using the AP computing principles course as the basis of a university course within the GE curriculum. Offering this course provides potential benefits to students of many different career choices, and computing programs in general.

7. REFERENCES

ACT, Educational/Career Aspirations & Economic Development. Retrieved May 16, 2015 from http://www.act.org/research/policymakers/ccr10/pdf/CareerInterestsProjectedJobOpenings.pdf

ACT Research, The Condition of Career and College Readiness: 2010

Adams, A., & Mowers, H., (2013, October 30). Should Coding be the 'New Foreign Language' Requirement? *Edutopia*. Retrieved June 10, 2014 from http://www.edutopia.org/blog/coding-new-foreign-language-requirement-helen-mowers?page=1

BLS, U.S. Department of Labor Bureau of Labor Statistics Downloaded May 16, 2015 from http://www.bls.gov/opub/mlr/2009/11/art5full.pdf

The CollegeBoard, AP Computer Science Principles Curriculum Framework 2016-2017, CollegeBoard, New York, NY www.collegeboard.org

The Economist (2014, April 26). A is for algorithm: A global push for more computer science in classrooms is starting to bear fruit. Retrieved from http://www.economist.com/news/international/21601250-global-push-more-computer-science-classrooms-starting-bear-fruit

Ferguson, R., Leidig, P., & Reynolds, J. (2015). Including a Programming Course in General Education: Are We Doing Enough?. *Information Systems Education Journal*, 13(3) pp 34-42. http://isedj.org/2015-13/ ISSN: 1545-679X

Pariser, Eli (2011) *The Filter Bubble, Penguin Books, New York*

Prensky, Marc (2008, January 13). Programming is the New Literacy. *Edutopia.* Retrieved April 23, 2010 from http://www.edutopia.org/programming-the-new-literacy

Richtel, Matt (2014, May 11). Reading, Writing, Arithmetic, and Lately, Coding. *The New York Times*. Retrieved from http://www.nytimes.com/2014/05/11/us/reading-writing-arithmetic-and-lately-coding.html?action=click&contentCollection=Politics&module=MostEmailed&version=Full&region=Marginalia&src=me&pgtype=article&_r=0

Rushkoff, D., (2010) *Program Or Be Programmed*, OR Books, New York

Rushkoff, D., (2012, November 13). Code Literacy: A 21st Century Requirement. *Edutopia*. Retrieved May 21, 2014 from http://www.edutopia.org/blog/code-literacy-21st-century-requirement-douglas-rushkoff