Volume 13, No. 3
May 2015
ISSN: 1545-679X

INFORMATION SYSTEMS EDUCATION JOURNAL

In this issue:

The **Information Systems Education Journal** (ISEDJ) is a double-blind peer-reviewed academic journal published by **EDSIG**, the Education Special Interest Group of AITP, the Association of Information Technology Professionals (Chicago, Illinois). Publishing frequency is six times per year. The first year of publication is 2003.

ISEDJ is published online (http://isedjorg). Our sister publication, the Proceedings of EDSIG (http://www.edsigcon.org) features all papers, panels, workshops, and presentations from the conference.

The journal acceptance review process involves a minimum of three double-blind peer reviews, where both the reviewer is not aware of the identities of the authors and the authors are not aware of the identities of the reviewers. The initial reviews happen before the conference. At that point papers are divided into award papers (top 15%), other journal papers (top 30%), unsettled papers, and non-journal papers. The unsettled papers are subjected to a second round of blind peer review to establish whether they will be accepted to the journal or not. Those papers that are deemed of sufficient quality are accepted for publication in the ISEDJ journal. Currently the target acceptance rate for the journal is under 40%.

Information Systems Education Journal is pleased to be listed in the 1st Edition of Cabell's Directory of Publishing Opportunities in Educational Technology and Library Science, in both the electronic and printed editions. Questions should be addressed to the editor at editor@isedj.org or the publisher at publisher@isedj.org.

INFORMATION SYSTEMS EDUCATION JOURNAL

Including a Programming Course in General Education: Are We Doing Enough?

Roger C. Ferguson
roger.ferguson@gvsu.edu

Paul M. Leidig
paul.leidig@gvsu.edu

John H. Reynolds
john.reynolds@gvsu.edu

School of CIS
Grand Valley State University
Allendale, Michigan 49401, USA

Abstract

General education is more than a list of required courses a student must take to complete their degree. For most universities, general education is the groundwork for the student's university experience. These courses span multiple disciplines and allow students to experience a wide range of topics on their path to graduation. Programming classes, e.g., Introduction to Programming, have not typically been an option as part of a general education course sequence at most universities. This study found that, only half of universities offer any kind of programming course in the General Education (GenEd) Program. The data also show that only two-thirds of institutions offer a computing class of any kind as a general education option. Institutions with ABET accredited Information Systems (IS) programs are significantly lower in both of these categories. This paper demonstrates the reasoning and process for including a programming class as an option in a GenEd Program, thereby showing how a programming class can be used to satisfy the requirements of a general education course. This results in two significant advantages to the computing department and university since the departments expand their reach to many more students, with a potential of increasing the number of majors or minors within the department; and non-computing majors have the opportunity to take courses that have not traditionally been offered in the GenEd Program. The latter results in students receiving a more comprehensive education and exposure to skills in high demand.

Keywords: General Education, Programming Courses.

1. THE PURPOSE OF GENERAL EDUCATION

With few exceptions, colleges and universities require students to take a sequence of courses outside of their major, that allow students to expand their knowledge in areas such as ethics, social sciences, history, the arts, humanities, mathematical sciences, natural sciences, and many other areas of interest. These general education courses provide students with the ability to understand a variety of disciplinary perspectives. In addition, general education has a set of goals that help student understand one's own culture and the cultures of others as well.

A graduate with a general education based degree has the ability to think in broad terms, that is, outside of their respective major. Simply put, general education prepares students for life. Before a student graduates, every student in the college or university will have had a set of courses that prepares them for responsible citizenship, and the ability to think critically outside of their major. A generally educated person should have the ability to recognize and work with issues and/or problems from multiple perspectives.

For computing majors, a GenEd Program serves its intended purpose since the course work is significantly outside of their major. Courses in ethics and social sciences help future computing professionals make decisions that affect the direction of the computing field. Courses in the arts and humanities offer computing students a unique perspective in their own culture and the cultures of others. Unfortunately, many students outside of the computing field may not get the same level of 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. Courses in computing could begin to bridge this gap; resulting in a significantly enhanced general education for non-computing majors. There is an additional benefit in exposing non-traditional students to programming. By having non-computing majors take courses in the computing field, the number of minors or majors within the computing department may increase.

2. NON-COMPUTING MAJORS ARE AT A TECHNOLOGICAL DISADVANTAGE

The issue with general education is not with the concept of creating students with a well-rounded, life preparing experience. The problem is that general education does not produce this type of student in all majors. For a major in a technical field, general education serves these students very well, the courses are truly outside of their major. However, the problem is that for many majors, typically in a non-technical field, such as: history, foreign language, arts, philosophy, religious studies, etc., general education fails to give these majors a well-rounded, life-preparing experience with relevant skills. Regardless of choice of major, students should learn to use computing systems to access, process, and analyze information as an essential aspect of critical thinking and problem solving. In many disciplines, students should also learn how to design algorithms, to write programs, and implement computing solutions applicable to their professions. Recent papers and articles show that computer competency courses or basic "code literacy" is becoming a requirement for 21st century culture (Rushkof, 2012). Due to efforts of organizations such as Code.org and many educational institutions around the country, courses in coding look less like an extracurricular activity and more like a basic life skill. Some school districts have expanded such efforts to as early as second grade (Richtel, 2014). This literature shows the trend toward basic "code literacy" for a generally educated person will only get stronger.

Increasingly, an understanding of programming logic is seen as a requirement for participation in today's digital world. Prensky (2008) makes the point that when people acquired language, they didn't just learn how to listen, but also how to speak. When people acquired text, they didn't just learn how to read, but also how to write. Now that people have computers, they (i.e., non-computing majors) are learning to use them but not how to program them. Without this understanding, people must accept the devices they use with the limitations, or worse, the agendas their creators have built into them. Douglas Rushkoff in Program or Be Programmed says the real question is, "do we direct technology, or do we let ourselves be directed by it and those who have mastered it?" (Rushkoff, 2012)

As non-computing majors spend an increasing amount of time in digital environments where others have written the rules, not understanding these rules puts them at a disadvantage. Knowledge of the fundamental concepts of how computer code works can help users understand the limitations or intentions behind the code. Being literate of the logic behind the systems might encourage readers to stop accepting the products (e.g., websites, apps, etc.) at face value, and begin to engage critically and purposefully with them.

As Rushkoff (2012) claims, learning to code familiarizes people with the values of a digital society: how people collaborate and share information. This new way of thinking and processing is quickly replacing the industrial age value of the hoarding of knowledge. Learning how software is developed and how computer technology really works helps everyone understand how they will be working and living as a society.

With the advent of personal computers and off-the-shelf applications in the 1980s, education efforts saw a shift to teaching how to use computers for office productivity tasks such as word processing and creating presentations. The result was that pupils left school with little idea how computers work. The focus had shifted from teaching programming and creating applications, to teaching how to use software, yet provided little insight into how software was actually works. The use of digital technology is now so ubiquitous that many think a liberal education requires a foundation in understanding programming, just as much as biology, chemistry or physics. That is one reason for the increased interest in teaching coding. The fewer people who know the basics of computing fundamentals, the smaller the number of potential technically skilled employees there are. A growing percentage of jobs require "computational thinking"; the ability to formulate problems in such a way that they can be tackled by computers. (The Economist, 2014) The U.S. Chief Technology Officer, Todd Park, said,

> [T]technology and computers are very much at the core of our economy going forward. To be prepared for the demands of the 21st century — and to take advantage of its opportunities — it is essential that more of our students today learn basic computer programming skills, no matter what field of work they want to pursue (Adams and Mowers, 2013).

Many current open jobs requiring computing skills are outside of traditional technology fields. Simply knowing and understanding the programming process can enhance job skills and careers in areas outside of actual programming careers. In response to this growing need, and while everyone may not need to be a "coder", all students will benefit by learning enough about programming to communicate with programmers in the digital information age. As a place to start, providing one of two valuable core skills would help students become more code literate. First, is learning basic programming concepts, such as "if" conditional branches and "for" loops. This provides an understanding of the automation provided by computer programs and helps understand the lingo used by programmers to explain the logic required in an application. Secondly, with the ubiquitous nature of the internet and living on the web, knowing the basics of how web pages are created (e.g. basic java script) is a skill that would be very helpful, while demonstrating rudimental programming logic and syntax. While either of these skills is beneficial, taking a general programming course provides an ability to communicate better in the digital society.

3. COMPARING GENERAL EDUCATION PROGRAMS

To provide a snapshot of the current general education requirements and options as they relate to computing courses in general and programming courses in particular, the authors chose two specific and different groups. The first group contains two sub-groups, the first consisting of the 14 other public institutions in the state, while the second sub-group is made up of 11 additional peer institutions. Peer institutions, in this case, were defined by the institution's accreditation and assessment process as having similar degree offerings, mission, but not geographically located near the institution. GVSU has chosen to benchmark itself against these two sub-groups as one group of peer institutions and they will be treated as such for comparison purposes in this study.

The second group consists of the 38 individual institutions that have ABET accredited Information Systems (IS) programs as of April 2014. An ABET accredited group of schools was chosen as the group most likely to have institutions that place a premium on technology education and, therefore, would be the most likely to have a computing technology requirement and/or a programming course included as a general education options. The IS accredited group represents a subset of institutions that have ABET accredited programs that are most relevant to the audience for this paper, Information Systems educators.

In each case, the general education requirements and approved courses for each institution were analyzed using the content provided by that institution's web site.

1) Is any computing course on the approved list of general education courses?

H_1: Universities that have ABET accredited IS programs will be different in the percentage that have a computing course on the approved list of general education courses.

2) Of these approved courses, are any a programming class (defined as a course that teaches programming logic as part of its content)?

H_2: Universities that have ABET accredited IS programs will be different in the percentage that have a programming course on the approved list of general education courses.

3) Of these approved courses, are any an introductory computing course (defined as a course in general computer literacy, use of office productivity tools, and/or information literacy and use)?

H_3: Universities that have ABET accredited IS programs will be different in the percentage that have an introductory computing course on the approved list of general education courses.

4) Is at least one of the approved general education computing courses required of all students?

H_4: Universities that have ABET accredited IS programs will be different in the percentage that require a computing course on the approved list of general education courses of all majors.

Regarding creation and enforcement of general education requirements, some institutions allow general education to be controlled at the department level, where others allow easy substitution within the GenEd Program. These later schools were counted as those who had the choice of a programming course in the GenEd Program.

Some institutions had a specific programming class in the GenEd Program, but others allowed a choice of any programming class as a general education class. Both of these were counted as having a programming course in the GenEd Program; however when a course was defined as having some programming content in addition to other computing literacy content, it was counted as an introductory computing course.

Statistical Methodology
For each of the four hypotheses, the null hypothesis will be accepted or rejected using the significance level of .05. To compare two independent groups based on binary variables, most statistics guidelines suggest using the chi-square test of independence as long as the sample sizes are large enough. Sauro and Lewis (2008) contend, however, that the "latest research suggests that a slight adjustment to the standard chi-square test, and equivalently to the two-proportion test, generates the best results for almost all sample sizes" (p. 75).

To determine whether a sample size is adequate for the chi-square test, calculate the expected cell counts in the 2x2 table to determine if they are greater than 5. Since the values in this study pass this test, the data was evaluated using the standard chi-square test. Next, since some might classify the sample sizes too small, the N-1 chi-square test was also run. The results were nearly identical, but the actual p-values used were from the latter test since they gave a slightly more conservative result. The formula for the N-1 chi-square test (Sauro and Lewis, 2008) is shown below using the standard terminology from the 2x2 table:

$$\chi^2 = \frac{(ad-bc)^2(N-1)}{mnrs}$$

Test Results
Hypotheses are supported when the null hypothesis is rejected. In this study, the null hypothesis is rejected when there is a statistically significant difference between the proportions represented by $p<.05$. Accordingly, the first hypothesis (H_1) is supported since there is a significant difference between the 45% of IS Accredited Schools and the 72% of GVSU Peer Institutions that offer any computing class that is also eligible for general education credit. In addition, the second hypothesis (H_2) is supported since there is a significant difference between the 26% of IS Accredited Schools and the 56% of GVSU Peer Institutions that offer a programming class in the list of approved general education courses. Further underscoring the significance of these results is that these differences are not in the direction that was expected. Universities with ABET accredited IS programs have significantly lower percentages in

both comparisons. Chart 1.0 shows the comparison of the proportions.

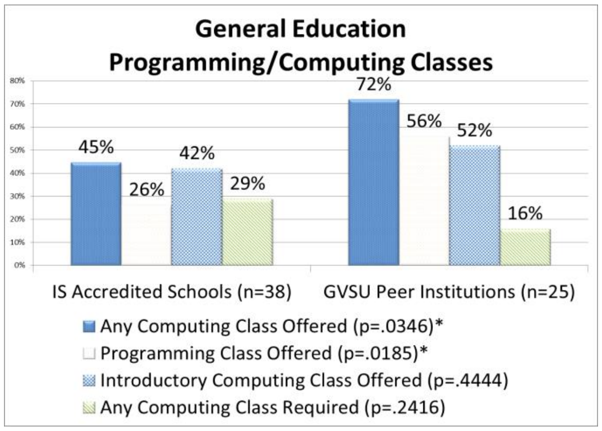

Chart 1.0 Survey results

4. FINDINGS

Based on these results, one can conclude there is an opportunity for universities to include a programming course in their general educational system. The graph shows only 28 percent of ABET accredited IS programs offer a general education approved programming course. If this number is representative of the general population of IS programs, then a significant number of institutions (the remaining 72 percent) could improve their GenEd Program for ALL majors.

The literature demonstrates that early computer education focused on teaching programming languages and logic. However, with the advent of personal computers and the use of purchased applications, universities shifted the focus to teaching how to use computers, often referred to as “computer literacy.” This shift in focus occurred at the expense of understanding how computers work, and how to program them. The increased demand for people with computing (i.e. programming) skills is important in many different disciplines and careers shows the importance of returning basic computer knowledge to an understanding of programming, or how to make computers work.

These conclusions from the literature are confirmed by the analysis of GenEd Programs at peer institutions (survey result above). Not only were few institutions offering general computing courses (45% of IS accredited schools), even fewer were offering a programming logic course as part of the GenEd Program.

Based primarily on this review and analysis, the authors determined that it would be valuable to review the history of the computing courses in the GenEd Program at GVSU. The intent of this process is to provide a case study that might shed light on how institutions offer or require programming logic in GenEd Programs.

5. AN INTRODUCTORY PROGRAMMING CLASS IN THE GENERAL EDUCATION PROGRAM

This section examines GVSU’s general educational program and the requirements for a course to be included in this program and demonstrates how a programming class can be included in general education. Since the GenEd Program at GVSU was recently revised in 2011 and since the programming class listed in that curriculum had not been revised in many years, it was decided that a thorough review should be undertaken. The process was to scan the environment, evaluate the current course and its history as a General Education course, and then evaluate the course from the point of view of the revised GenEd Program and current best practices in educational pedagogy related to that course.

The General Education Program - 1980’s
The structure of the GenEd Program at Grand Valley State University remained largely unchanged from the 1986-1987 through the 1998-1999 academic catalogs. The requirements were divided into four major sections: College, Arts and Humanities, Natural Sciences, and Social Sciences. Each section had sub-groups of courses from which students were to choose individual courses based on the specified requirements as shown below.

1. College Section (one course in each)
   - Study of logical and mathematical quantitative reasoning
   - Foreign and multicultural approaches
   - History of Western civilization
   - Critical examinations of values and ideas
2. Arts and Humanities (one course in each)
   - Exploration of art, music, and theatre
   - Exploration of literature
3. Natural Sciences Section (one course in each and one of these must include a lab)
   - Physical sciences
   - Life sciences

4. Social Sciences Section (two courses, each from a different group and discipline)
   - Human behavior and experience
   - Social and cultural phenomena
   - Formal institutions

First General Education Programming Course
A course in BASIC was approved in fall of 1983 and added to the GVSU computing curriculum the next academic year. Three years later in the fall of 1986, the course was added to the GenEd Program for the following academic year under the College Section category of Quantitative and Logical Reasoning. The rationale for this is summed up in this one statement from the justification document submitted with the proposal, which says

> While the students do spend time in learning the syntax of a specific computer language, BASIC, the bulk of the time in this course is spent in learning problem solving and in relating the logical constructs of flow of control, organization of data, and inter-relationships between the sub-problems, to the given problem.

Revised General Education Program
As stated above, general education at GVSU has been part of the university since its inception. The "revised" main focus stated on the University's web site is: "…the general education program is to provide students with an education that balances depth with breadth, the specialized with the general. The general education program helps students become literate in a sophisticated way in a number of disciplines." There are two main goals in the GenED Program: Knowledge and Skills.

For Knowledge:
1. A graduating person from the GenEd Program is able to understand a variety of disciplinary perspectives, and understands the growth of knowledge and the various approaches through which the knowledge was acquired.
2. A generally educated person understands one's own culture and other cultures as well.
3. A graduating person understands how academic studies connect to current issues.

For Skills:
1. Learn the process of working together, i.e., collaboration through sharing of ideas towards a common project.
2. Critical and creative thinking using systematic reasoning to examine and evaluate ideas.
3. Use Ethical reasoning in the decision-making process based on defining systems of value.
4. Information literacy using multiple forms of information.
5. Integration, that is, the ability to synthesize and apply existing knowledge to complex problems.
6. To use effective practices in oral communication across a wide variety of public audiences.
7. Problem solving as it relates to open ended questions through the use of designing and evaluating the designs.
8. Quantitative literacy is competency working with numbers.
9. Written communication: the ability to create and refine messages that an educated reader would value.

The goals outlined above, knowledge and skills, are at the heart of the program. Students typically take 11 - 13 courses (over 30 credits) in the program. This is a significant commitment the university is making towards the GenEd Program.

The programming course that has been offered in the GenEd Program has been available to students since the 1987. The programming course satisfied the original requirements and continues to satisfy the requirements for the new general educational program criteria. The next section demonstrates the mapping of the programming course to the original set of criteria.

Mapping a Programming Course to the New General Education Requirements
By mapping the course content with the GenEd Program criteria, this section shows how a programming course can be used to satisfy the requirements of the GenEd Program at GVSU. Specifically, it addresses how a programming course might be used "to provide students with an education that balances depth with breadth, the specialized with the general. In particular, would such a course help students become literate in a sophisticated way in a number of disciplines."

Assignments/ projects in the Programming Course	**Maps** $\Longrightarrow$ **Criteria in a general education course**
Group Projects Create exercises to establish effective groups	Skill 1: Collaboration
Use Graphical User Design (GUI) with multi-buttons, text fields, labels, combo-boxes, data fields, etc. Visual studio is an excellent tool for this since it is a drag and drop, and prototyping environment	Skill 4: Information literacy, multi-forms of data Skill 7: Problem solving as it relates to open ended questions
Use complex programming structures The project should use nested while loops, nested if statements, etc.	Skill 2: Critical and creative thinking using systematic reasoning
Have periodic Code reviews Have each group gives an oral presentation of their code	Skill 6: Practice and refine oral communication
Assignments statements Have the project do non-trivial equations, e.g., have the program calculate the quadratic equation	Skill 8: Working with numbers
Comments within the code Require multi-iterations of the internal comments; consider the first set of comments as a draft	Skill 9: Written communication
Make the group project solve a real world problem Perhaps a simulation of a voting booth machine, discussion of the ethics behind this project	Skill 5: Integration; the ability to synthesize and apply existing knowledge to complex problems Skill 3: Ethics, what information should be retained.

Table 1 – Mapping a GenEd Course

Finally, this course should contribute to the two goals of providing both Knowledge and Skills. Table 1 provides a mapping of the projects in the programming course with the Skills criteria of GVSU's GenEd Program.

6. CONCLUSIONS

The process used in this analysis demonstrates that having a programming class in the GenEd Program has several benefits. First, the computing department expands their reach to non-majors that may result in an increase in majors or minors. Second, there is an immediate increase in the number of students serviced by the department, thus receiving exposure to additional skills. Finally, non-computing majors benefit by expanding their general educational course selection with relevant computing knowledge. This gives non-computing majors a broader multi-discipline degree, including skills that are in demand for many different careers.

The results also demonstrate that most institutions are not taking advantage of this opportunity. Only 56 percent of the peer institutions are utilizing a programming class in the GenEd Program. Furthermore, only 28 percent of ABET IS accredited universities are taking advantage of this opportunity. This does not mean that every university would benefit having a programming class in the GenEd Program. However, there does seem to be an opportunity for many computing departments to offer a programming course within the general education environment, benefiting both the department and the students taking the course.

The data and analysis from this study does not provide additional insight into the difference between simply offering versus requiring a programming course. However, based on the findings of the literature regarding the value of understanding programming logic for students in many non-computing majors, these authors suggest that requiring all students to take a general programming course provides an ability to communicate better in the digital society.

7. REFERENCES

Adams, Anna and Mowers, Helen (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

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

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, Douglas (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

Sauro, Jeff and Lewis, James (2008). Quantifying *the User Experience: Practical Statistics for User Research.* Morgan Kaufmann, Burlington, Massachusetts.

Editor's Note:

This paper was selected for inclusion in the journal as an ISECON 2014 Meritorious Paper. The acceptance rate is typically 15% for this category of paper based on blind reviews from six or more peers including three or more former best papers authors who did not submit a paper in 2014.

Appendix A - List of Universities in Study

A ABET Accredited IS Programs
(as of April 2014)

1. Arkansas Tech University
2. California State University, Chico
3. California University of Pennsylvania
4. Drexel University, College of Information Science & Technology
5. East Tennessee State University
6. Fitchburg State University
7. Florida Memorial University
8. Gannon University
9. Grand Valley State University
10. Illinois State University
11. Jacksonville State University
12. James Madison University
13. Kennesaw State University
14. Metropolitan State University of Denver
15. New Jersey Institute of Technology
16. Quinnipiac University
17. Radford University
18. Regis University
19. Robert Morris University
20. Rowan University
21. Slippery Rock University
22. Southern Utah University
23. State University of New York at Brockport
24. The University of Tampa
25. University of Houston - Clear Lake
26. University of Houston, College of Technology
27. University of Nebraska at Omaha
28. University of North Alabama
29. University of North Florida
30. University of Puerto Rico at Bayamon
31. University of Puerto Rico, Rio Piedras Campus
32. University of South Alabama
33. University of South Carolina
34. Utah State University
35. Utah Valley University
36. Virginia Commonwealth University
37. West Texas A&M University
38. Wright State University

GVSU Peer Institutions
(as defined by Institutional Analysis Office)

1. Appalachian State University
2. Boise State University
3. Central Michigan University
4. CUNY Hunter College
5. Eastern Michigan University
6. Ferris State University
7. James Madison University
8. Lake Superior State University
9. Michigan State University
10. Michigan Technological University
11. Montclair State University
12. Northern Michigan University
13. Oakland University
14. Portland State University
15. Saginaw Valley State University
16. Towson University
17. University of Michigan-Ann Arbor
18. University of Michigan-Dearborn
19. University of Michigan-Flint
20. University of Nebraska at Omaha
21. University of Northern Iowa
22. Wayne State University
23. Western Michigan University
24. Western Washington University
25. Youngstown State University