Volume 9, No. 1
April 2011
ISSN: 1545-679X

INFORMATION SYSTEMS EDUCATION JOURNAL

In this issue:

EDSIG
Serving Information Systems Educators

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 quarterly. The first year of publication is 2003.

ISEDJ is published online (http://isedjorg) in connection with ISECON, the Information Systems Education Conference, which is also double-blind peer reviewed. Our sister publication, the Proceedings of ISECON (http://isecon.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 about 45%.

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

Editors

ISEDJ Editorial Board

Student Perceptions of Instructional Tools in Programming Logic: A Comparison of Traditional versus Alice Teaching Environments

Leah Schultz
lschult@tarleton.edu
Computer Information Systems, Tarleton State University
Stephenville, Texas 76401, United States

Abstract

This research investigates the implementation of the programming language Alice to teach computer programming logic to computer information systems students. Alice has been implemented in other university settings and has been reported to have many benefits including object-oriented concepts and an engaging and fun learning environment. In this study, students were surveyed on their opinions about the effectiveness of the Alice environment and were compared to responses of students who had been taught in a more traditional environment using flowcharts and pseudocode. Analysis of data revealed there were no statistically significant differences in responses, with both groups reporting high levels of satisfaction with their respective learning experience. Discussion includes ways to potentially improve the implementation of Alice and its benefits in the classroom.

Keywords: Alice programming language, introductory programming, student attitudes

1. INTRODUCTION

Researchers have been discussing the best approach to teaching the introductory computer class for almost as long as computing languages have been around (Pears, et al., 2007). The first programming class is a fundamental part of the learning process for new programmers but many students find the class to be difficult and uninteresting, leading to high rates of failure or drop rates. As a result, faculty members in computer science and information systems programs continually look for better ways to instill basic programming skills into tomorrow's programmers.

From studies of motivation and cognition to experimentation with new programming environments, researchers approach the problem from many directions. One way that has garnered some attention over the past few years has been changing the environment in which the basic constructs of programming are taught. Some programs have implemented introductory courses taught in scripting languages such as Python (Nikula, et al., 2007) while others have adopted new environments such as the programming language Alice.

This study looks at the use of Alice in the classroom and compares student attitudes about programming confidence, success, applicability, and preparedness. These attitudes are compared with students in the same program who were taught programming fundamentals using more traditional methods including flowcharts and pseudocode only.

2. BACKGROUND

The concern about introductory programming courses begins with the widely reported failure rates of first time programmers that range from 30 to 60% in computer science programs (Dehnadi & Bornat, 2006). Researchers speculate

that there are many reasons for failure and success in the programs that range from cognitive to motivational. In his thesis *The Motivation of Students of Programming*, Jenkins (2001) postulates that the reasons for failure or success stem from who the student is, what the teacher does, and what the student does. In other words, there are many factors that come into play when deciding why students fail or succeed in these classes. When asking students why they believe they failed an introductory computer class, Hawi (2010) found that most students cited learning strategies and teaching methods as the main reasons for poor performance, followed closely by lack of appropriate effort on the part of the student.

When examining learning strategies, we must look at the way many introductory courses are taught. Many times the examples and strategies used to explain basic concepts are the same teaching methods that have been around for years. Although the iconic "hello world" example lives on, it may not be the most exciting or relevant programming for students today. Combine these strategies with a learning environment that requires high levels of abstraction in a complex modern language and many students will not or cannot comprehend the material being presented (Nikula, et al., 2007).

Given these many barriers, there are still many students who succeed in the introductory programming course and are highly motivated to perform well. Law, Lee, and Yu (2010) discovered a complex relationship between individual attitude, direction, social pressure and competition, and reward or recognition. In other studies, there was a significant importance placed on the importance of value to the student and the expectancy of the outcomes, although this motivation was found to be short lived and affected mostly the student's motivation in that one course and did not extend to the rest of the academic or professional career (Jenkins, 2001). Keeping students interested in programming is difficult because many come to the class with preconceived ideas of difficulty and many view programming as boring. The job of changing these attitudes and motivating students to succeed is daunting to many faculty members but takes on greater importance as enrollment in computer science and information systems programs continues to decline while the demand for skilled workers increases (Kelleher & Pausch, 2007).

Alice Programming Environment

In order to motivate students and make first programming experiences more interesting for students, some universities have integrated the software program Alice. Alice was developed by Randy Pausch at Carnegie Mellon in an attempt to encourage younger girls to take an interest in computer programming. Kelleher and Pausch (2007) reported success in motivating girls through the storytelling element with an easy user interface and state that in addition to simplifying the learning environment, tools need to be developed that attract students to the programming field.

In the university environment, the implementation of Alice has taken many forms from modules in beginning programming classes to entire semester courses teaching fundamental concepts through Alice. In addition to tangible strengths of presenting object oriented concepts, intangible benefits of Alice include better visual rewards and reinforcement, intrinsic motivation from storytelling aspect, self-confidence, and increased interest in computers (Goulet & Slater, 2009). Many researchers have reported success stories in the use of Alice and resulted in increases in positive attitudes towards programming in an introductory computer class for non-majors. Many comments from students in the class commented on the creativity and the fun nature of the programming environment. (Courte, Howard & Bishop-Clark, 2006). In a study where Alice was used in pair programming exercises, students found Alice to be "simple" and the success of the students increased when students worked in pairs with the tool. (Howard, Evans, Courte, & Bishop-Clark, 2009).

There are however drawbacks to the use of the tool in the university environment. Some critics believe it is not appropriate for undergraduate learners because the program is not a fully developed program that is considered "useful" in industry (McKenzie, 2009). Because the focus of Alice is storytelling using 3D visual objects, other critics do not believe it is an appropriate tool to use particularly when teaching business programming because examples and programming problems do not lend themselves to the environment.

3. PROBLEM

Because learning many times occurs on the edges of what we already know and prior knowledge can affect the success of a first year programmer, Computer Information Systems (CIS) majors at Tarleton State university begin their programming sequence with an introductory class focusing on programming logic (Robins, 2010). At times the course was offered in the same semester as the first programming languages and other times it preceded enrollment in the first language. As interest in object-oriented languages increased and enrollments in procedural languages such as COBOL began to decline, the logic course began to deviate somewhat from the concepts being presented in the first language course. At the same time, enrollment overall began to decline in the program and fewer student were exploring CIS as a degree option. Anecdotally, students in the course were bored and attendance was dismal. As state budgets constricted and the legislature mandated lower core curricula, computer literacy courses were dropped from many degree plans and the CIS department had even fewer opportunities to attract new students.

Because of these factors, faculty began to explore new ways to teach the logic course to better prepare students for their first programming class, to engage students in the process, and hopefully attract more students to the programming degree. At the time, many other universities were reporting moderate success with Alice and textbooks were developed to facilitate the formal teaching of Alice. Faculty hoped that Alice would be the tool to teach object-oriented fundamentals in a fun, engaging manner that would motivate students to learn and possible attract more majors.

Alice has been used to teach programming logic since 2008 and approximately half of the students in the program were instructed using the old method and newer students were instructed using the Alice environment. All students, whether they concentrate their studies in programming or network administration, are required to complete the course before beginning their first programming language. Currently, the students can choose to take JAVA or Visual Basic as their first programming course. To determine the student's perceptions of the changes and the effectiveness in meeting some of the departmental goals, this research project was undertaken.

4. METHODS

A survey (Appendix A) was administered to students enrolled in CIS classes beyond the logic course. The survey was conducted face to face and was not administered to students in online courses. Students were asked to provide some demographic information as well as indicate the semester in which they took the logic course. Then, students were asked to provide their opinions on their experience in the logic course. In addition to an open ended question allowing students to provide additional comments, questions on the survey covered the following concepts:

- Relatedness to concepts in first programming class
- Confidence in abilities to program
- Enjoyment or fun in programming
- Preparedness for first programming class
- Improvement in problem solving
- Influence on pursuing a CIS degree
- Applicability in other CIS classes
- Improvement of programming skills
- Perceived value of class
- Recommendation of class to others

Students were asked to indicate their agreement with positively phrased questions covering these concepts on a Likert scale from 1 to 5.

Forty-three students completed the survey in the spring of 2010. Of the students who participated in the survey, three were female and forty-one were male. Thirty-eight of the students were aged 18-25, with four students in the age group 25-34 and one student over 35 years of age. Twenty-seven students indicated a concentration in network administration, ten in systems development, and six did not indicate their area of concentration. Of the students responding, 20 completed the logic course before the implementation of Alice and 23 used Alice.

Survey results were compiled in Microsoft Excel and imported into SPSS for analysis. The responses to questions about student attitude were compared between students who were taught using the older method and students who were taught using the Alice environment. Because of the small population size and the use of ordinal

data, tests were conducted to determine differences in population using the Mann-Whitney test.

5. RESULTS

When analyzing the data, each question was examined to determine if differences existed between the two groups of students who had been instructed in different learning environments (see Appendix B). Surprisingly, on all ten questions there are no statistically significant differences in the responses given by students in the two different groups.

In addition to the differences in learning environment, other factors such as first programming language and concentration were also analyzed to see if these had an impact on responses. An analysis of responses between students who had used Alice in the introductory program and concentrated their studies in network administration and students who were interested in programming revealed no significant differences in their attitudes towards Alice as a learning tool. (see Appendix C)

In order to determine if responses may have been affected by choice of first programming language when determining efficacy of Alice as a teaching tool, data were analyzed and compared students who took Microsoft Visual Basic against those students who enrolled in Java for their first programming language (see Appendix D). However, there were no significant findings between responses except for one. Students who took Visual Basic as their first programming languages provided more positive responses when asked if the logic class had influenced their decision to pursue a CIS degree ($U=7.5$, $p=>02$).

6. DISCUSSION

When examining the data, the results indicate that the inclusion of Alice has not brought much change in students' perceptions of the programming and logic course. However, there are many other variables to consider when examining this issue. Previous iterations of the logic course had been taught for many years previous to the respondents' enrollment in the course. The curriculum and the preparation had been fine tuned by many of the professors teaching the course and was many times tied directly to the problems being presented in the programming courses.

As the tool continues to be used in the logic class, perhaps better coordination between instructors teaching the logic tool and instructors teaching programming languages could leverage some of the advantages Alice has been previously been reported to provide. Also, as with any new software program, installation issues and program functionality issues may have had an impact on attitudes towards Alice. Some students reported in the comment section of the survey that they had many technical issues with installation and stability of Alice that frustrated them throughout the process. Once these technical issues are resolved or perhaps better installation instructions are provided for students there may be an in improvements to students' attitudes toward the tool.

Overall, students appear satisfied with the usefulness and applicability of concepts learned in the logic and programming class regardless of the tool implemented to teach the course. A comparison between the two groups does not necessarily discount the use of Alice as a tool but demonstrates that perhaps the potential of the tool has not yet been maximized. Anecdotally, students have reported a disconnect between the types of problems being solved in Alice and the business problems being presented in the first programming course. Perhaps a blending of traditional flowcharting and pseudocode for business problems combined with the fun and engaging environment of Alice to teach basic concepts and logic structures may increase relevancy and value for students in the course. For example, an initial assignment in Alice may introduce students to the concept of iterations by having a penguin flap its wings in Alice. Then, once the logical concept has been presented, a second example relating to business may help students bridge the gap between the 3D storytelling environment into a framework that will more closely mimic the types of problems they will encounter in their first programming class and ultimately in their career.

Further research needs to be conducted in this area to go beyond the students' expectations and attitudes towards the tool. Measures of success whether they be tied to grades in programming classes or performance in capstone courses needs to be analyzed as well. In depth analysis of student learning on various concepts may reveal strengths and weaknesses of both environments that may lead to blended uses of tools to teach different concepts. For example, research may reveal that Alice is very effective in presenting the ideas of objects but might not

be as strong at presenting the logical structures of programming such as decision structures or looping structures.

In addition to its efficacy in teaching introductory concepts, further research needs to be conducted on the usefulness of the tool in attracting and retaining students. Because Alice was initially designed to generate interest in younger girls through storytelling and interactivity, studies need to be done to determine if this is indeed drawing more women to a field that is traditionally male and also if the concepts of storytelling and interactivity appeal to male students as much as they might to female students. Unfortunately, only three respondents in this study were female and did not allow for analysis of this type.

In conclusion, the implementation of Alice in the programming logic class was not a failure. Students reported high levels of satisfaction with the tool similar to those reported by students in previous, more traditional introductory course. With improvements to the content presented and with tweaks to the use of the tool, both technically and pedagogically, there is hope that many of the benefits of Alice may be realized in future course offerings.

7. REFERENCES

Courte, J., Howard, E.V., & Bishop-Clark, C. (2006). Using Alice in a computer science survey course. *Information Systems Education Journal*, 4(87), 3-7.

Dehnadi, S. & Bornat, R. (2006). The camel has two humps: paper presented at PPIG 2006 Workshop. Retrieved May 2010 from http://www.cs.mdx.ac.uk/research/PhDArea/saeed/paper1.pdf.

Goulet, D.V., & Slater, D. (2009). Alice and the introductory programming course: an invitation to dialogue. *Information Systems Journal*, 7(20), 3-16.

Hawi, N. (2010). Causal attributions of success and failure made by undergraduate students in an introductory-level computer programming course. *Computers & Education*, 54(2010), 1127-1136.

Howard, E.V., Evans, D., Courte, J. & Bishop-Clark, C. (2009). A qualitative look at Alice and pair-programming. *Information Systems Education Journal*, 7(80), 3-10.

Jenkins, T. (2001). The motivation of students of programming. Online thesis retrieved on May 13, 2010 from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.95.5098&rep.

Kelleher, C. & Pausch, R. (2007). Using storytelling to motivate programming. *Communications of the ACM*, 50(7), 59-64.

Law, K.M., Lee, V.C.S., & Yu, Y.T. (2010). Learning motivation in e-learning facilitated computer programming courses. *Computers & Education*, 55(2010), 218-228.

McKenzie, B. (2009). Introductory programming with ALICE as a gateway to the computing profession. *Information Systems Education Journal*, 7(38), 3-8.

Moskal, B., Lurie, D. & Cooper, S. (2004). Evaluating the effectiveness of a new instructional approach. *Proceedings of SIGCSE*, USA, 35, 75-79.

Nikula, U., Sajanieme, J., Tedre, M. & Wray, S. (2007). Python and roles of variables in introductory programming: experiences from three educational institutions. *Journal of Information Technology Education*, 6 (2010), 199-214.

Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J. (2007) Literature review of teaching first year programming. *SIGCSE Bulletin*, 39(4), 2004-223.

Robins, A. (2010). Learning edge momentum: a new account of outcomes in CS1. *Computer Science Education*, 20(1), 37-71.

Appendices

Appendix A – Survey Questions

Students were asked to provide their responses on a 5 point Likert scale from 1 (Strongly Disagree) to 5 (Strongly Agree)

1. The concepts I learned in the programming and logic class (CIS110) directly related to concepts used in my first programming class,.
2. The programming and logic class increased my confidence in my abilities to program.
3. The programming and logic class made programming seem like fun.
4. The programming and logic class prepared me for my first programming class.
5. The programming and logic class helped me solve problems in my first programming class.
6. The programming and logic class influenced me to pursue a CIS degree.
7. The programming and logic class has helped me in other CIS classes.
8. The programming and logic class made me a better a programmer.
9. The programming and logic class was a valuable experience.
10. I would recommend the programming and logic class to new CIS students.

Appendix B – Analysis Results of Responses Between Students in Different Learning Environments

Question	Environment	N	Mean Rank	Sum of Ranks	Mann-Whitney U	Significance
1	Alice Non-Alice	20 23	22.00 22.00	440.00 506.00	230.000	1.000
2	Alice Non-Alice	20 23	21.30 22.61	426.00 520.00	216.000	.724
3	Alice Non-Alice	20 23	20.85 23.00	417.00 529.00	207.000	.564
4	Alice Non-Alice	20 23	23.34 19.98	443.50 459.50	183.500	.361
5	Alice Non-Alice	20 23	22.12 21.89	442.50 503.50	227.500	.950
6	Alice Non-Alice	20 23	23.00 21.13	460.00 486.00	210.00	.612
7	Alice Non-Alice	20 23	21.25 22.65	425.00 521.00	215.00	.708
8	Alice Non-Alice	20 23	21.22 22.67	424.50 521.50	214.500	.698
9	Alice Non-Alice	20 23	21.48 22.46	429.50 516.50	219.500	.791
10	Alice Non-Alice	20 23	20.70 23.13	414.00 532.00	204.00	.507

Appendix C - Analysis of Responses of Students in Alice Environment by First Programming Language

Question	1st Programming Language	N	Mean Rank	Sum of Ranks	Mann-Whitney U	Significance
1	Java Visual Basic	12 7	9.50 10.86	114.00 76.00	36.000	.650
2	Java Visual Basic	12 7	9.04 11.64	108.50 81.50	30.500	.340
3	Java Visual Basic	12 7	8.33 12.86	100.00 90.00	22.000	.100
4	Java Visual Basic	12 7	8.58 12.43	103.00 87.00	25.000	.167
5	Java Visual Basic	12 7	9.21 11.36	110.50 79.50	32.500	.432
6	Java Visual Basic	12 7	7.12 14.93	85.50 104.50	7.500	.002
7	Java Visual Basic	12 7	8.75 12.14	105.00 85.00	27.000	.227
8	Java Visual Basic	12 7	9.83 10.29	118.00 72.00	40.000	.902
9	Java Visual Basic	12 7	9.25 11.29	111.00 79.00	33.000	.482
10	Java Visual Basic	12 7	9.08 11.57	109.00 81.00	31.000	.384

Appendix D – Analysis of Responses of Students in Alice Environment by Concentration

Question	Concentration	N	Mean Rank	Sum of Ranks	Mann-Whitney U	Significance
1	Networking Systems Dev	11 7	9.00 10.29	99.00 72.00	33.000	.659
2	Networking Systems Dev	11 7	9.14 10.07	100.50 70.50	34.500	.724
3	Networking Systems Dev	11 7	8.41 11.21	92.50 78.50	26.500	.285
4	Networking Systems Dev	11 7	9.00 10.29	99.00 72.00	33.000	.659
5	Networking Systems Dev	11 7	8.64 10.86	95.00 76.00	29.000	.425
6	Networking Systems Dev	11 7	9.09 10.14	100.00 71.00	34.000	.724
7	Networking Systems Dev	11 7	8.91 10.43	98.00 73.00	32.000	.596
8	Networking Systems Dev	11 7	9.14 10.07	100.50 70.50	34.500	.724
9	Networking Systems Dev	11 7	8.68 10.79	95.50 75.50	29.500	.425
10	Networking Systems Dev	11 7	8.23 11.50	90.50 80.50	24.500	.211