Use of Online Tools in Teaching C++ Programming to Freshmen in All Engineering Majors

Grace Ni, David Bishop, Anthony Donaldson
College of Engineering
California Baptist University
8432 Magnolia Avenue, Riverside, CA, 92508
gni@calbaptist.edu, dabishop@calbaptist.edu, adonaldson@calbaptist.edu

Abstract **- As computer software becomes increasingly used in analysis and design in all engineering disciplines, more engineering programs have started including computer programming in their common core for all engineering majors. C++ is a popular programming language that's been chosen for teaching engineering students programming. At California Baptist University, EGR 121 Introduction to Computer Programming in C++ is a required course for all engineering students. Most of our engineering students take this course in their first year. This course was taught using traditional means of lecture, text book reading and exercises along with labs and programming projects. Since the fall of 2012 we incorporated two online resources, an online interactive content resource and an online exercise tool to replace the previous textbook problems as homework. We discuss our experience in the classroom along with survey feedback from our students. Although no statistically significant difference in final grades was detected, we did find anecdotal indication that students benefited from these tools particularly the online homework problems.**

Index Terms - C++, Introduction to programming, Online learning, Problem based learning.

INTRODUCTION

The development of the digital computer has facilitated many significant engineering achievements over the past five decades. For example, computer processing of the images collected by the Mars Reconnaissance Orbiter, computer simulation of advanced composite materials, prediction of weather, climate, and global change with computer analysis, computer speech recognition, etc. [1]. Therefore, programming skills are now deemed essential in most engineering schools. Both structured languages, such as C/C++, and computational tools such as MATLAB, have been used in engineering curricula.

The College of Engineering at California Baptist University was established in 2007. Besides three ABET accredited Bachelor of Science degree programs in Electrical and Computer Engineering, Mechanical Engineering, and Civil Engineering, programs in Chemical Engineering, Software Engineering, and Construction Management have also been developed in recent years. Programming skills are emphasized in our college of engineering curriculum starting from freshmen year through multiple courses, for example, EGR121 Intro to Computer Programming in C++, EGR 122 Visualization Languages which covers Excel and CAD, EGR182 Introductory Mathematics for Engineering Applications which covers beginner level Matlab.

Since its first offering in 2008, EGR121 Intro to Computer Programming in C++ was offered to all engineering students as a required course. Most of our engineering students take this course in their first year. In this paper, we discuss our experience in teaching this course, especially the adoption of two online resources: an online interactive text website and an online homework system.

This paper is organized as follows: we will first present the rationale behind offering a C++ programming course to all engineering students and our teaching methods in general; next, following a literature survey on online teaching resources and online homework, we will introduce the two online-tools we adopted: CodeLab online homework system and the Zyante interactive online text; then we will summarize the student survey results on their experiences of using CodeLab and the Zyante website and their preference on using online materials only; finally, we will conclude the paper with discussions on lessons learned, concerns, and possible solutions.

TEACHING C++ PROGRAMMING TO ENGINEERING STUDENTS

In this section, we focus on why and how we offer a C++ programming course to all engineering students at CBU, mostly in their first year of college.

I. Why C++ for Freshmen Engineering Students

At California Baptist University, one pragmatic value of requiring C++ programming for freshman is that it fits into a common first year curriculum designed to expose students to the diversity of engineering program opportunities. At CBU first year students take a common curriculum that then enables them to defer specialization until their second year in the program. This gives students an opportunity to explore the different aspects of our available engineering

programs prior to requiring a commitment to a particular track. Students and parents appreciate the ability to defer affinity until the sophomore year. Another consideration that is pragmatic in nature is to leverage courses that are taken by all engineering students and taught by any engineering faculty, thus reducing the need for specialized staff in the early development of the engineering programs.

In addition to pragmatic concerns, the introduction to programming in C++ also affords the development of a logical thinking and problem solving process in an applied environment. The course emphasizes top-down decomposition of problems and bottom-up construction of solutions. This approach is then leveraged throughout the engineering curriculum. By developing these skills through computer programming the students get to see tangible results in the output of their programs.

One further consideration that factored into including computer programming for all freshmen was the reality of today's business environment. Due to the ubiquity of software in business and engineering, engineering students need to become comfortable with adapting to using a variety of software applications as well as having a rudimentary understanding of what goes into developing software, so they can effectively communicate with other software developers and understand some of the challenges and potential limitations of software. The effectiveness of this approach is demonstrated in the survey results from our internship program. Internship supervisors consistently rate CBU engineering students highly on their ability to adapt to and use a variety of software on the job.

Finally, one goal for the first year curriculum is to impart life skills to students. Many students may come with a preconceived negative attitude toward software development and desire to focus on other more physical forms of engineering. But by taking the C++ programming course they learn to do something that perhaps they didn't initially like but come to understand that it is a stepping stone to move to other things that they do like. This deferment of enjoyment for achieving a larger goal is a valuable life lesson and many students have come to appreciate this principle through the inclusion of a programming course in their required curriculum.

II. Structure of the Course

Since its first offering in 2008 until the spring semester of 2012, EGR121 Intro to Computer Programming in C++ was offered using traditional means of lectures and computer labs. Nine computer labs were embedded in the schedule of three lectures per week, with fifty minutes per lecture. Homework problems were chosen mainly from the textbook by Dale and Weems [2]. Six programming assignments that are more difficult than homework and lab exercises were assigned throughout the semester. In the fall semester of 2012, we switched to a textbook by D. Malik [3] and slightly reorganized the course. Table I shows the topics covered in this course, and the lab projects and programming assignments accompanying those topics.

TABLE I
TOPICS AND ACCOMPANYING LABS/PROGRAMMING ASSIGNMENTS

Topics	Lecture Hours	Lab Projects	Programming Assignments
An overview of computers and programming languages	2	Lab #1: Introduction to development and execution of C++ programs	None
Basic elements of C++	3	Lab #2: Arithmetic expressions and I/O statements	PA #1: Data analysis of an electronic circuit experiment
Input/output	2	Lab #3: Interactive I/O and file I/O	
Selection control	3	Lab #4: Logical expressions and selection control	PA #2: Computation of tax rates
Repetition control	3	Lab #5: Repetition control and loop design	PA #3: Conversion of digital data to analog tones
User-defined functions	3	Lab #6: Functions, scope and lifetime	PA #4: Computation of parking fee
Enumeration type, namespaces and the string type	2	Lab #7: Enumeration type, string operations, and multi-file program	PA #5: Password strength check
Arrays and C-strings	3	Lab #8: Arrays	PA #6: Image processing
Structs and intro to classes	4	Lab #9: Structs and classes	None

III. Programming Examples and Assignments in the Engineering Context

To motivate and engage engineering students in learning C++ programming, we incorporated programming examples and assignments related to engineering problems. Engineering related programming examples can be found in several C++ textbooks oriented to engineers and scientists [1][4][5]. However, since most of the students taking EGR121 are freshmen, we cannot assume that they have gained much engineering knowledge. Too many engineering related problems or problems beyond their knowledge can possibly cause frustration instead of interest. With this consideration in mind, we developed or adopted some examples and assignments related to engineering practice that are easy to understand. A few of them are listed below:

- Programming Assignment #1: Suppose the ECE junior students conducted an experiment on an electronic device and obtained three sets of data as the measurements of input and output voltages of the device. Write a C++ program that will: allow a user to enter the measurements with interactive I/O; calculate the theoretical output voltage based on each input voltage and the mathematical model of the device;

calculate the percentage error of each actual output compared to the theoretical output; record all the data to a file with proper labeling and formatting.

- Programming Assignment #3 (adopted from an example in [4]): A calling modem transmits each data bit 1 as a 1270-hertz tone lasting one time unit, and each data bit 0 as a 1070-hertz tone also lasting one time unit. The file "digital.dat" contains a sequence of zeros and ones separated by spaces. Write a program that displays messages indicating the frequency and length of tones that would be emitted for the digital data in "digital.dat".
- Programming Assignment #6: Given an image stored in a file as asterisks and spaces, write functions to load the image to a two-dimensional array, to print the image, to flip it horizontally and vertically, to rotate it clockwise and counter-clockwise, and to generate the negative of the image. Test all the functions.
- Programing example: In a circuit simulation software, the user interface normally allows a user to choose the unit of capacitance from a list of units, fF, pF, nF, uF, mF, etc. Use a switch statement to convert the character 'f', 'p', 'n', 'u' or 'm' to the corresponding factor 10^{-15}, 10^{-12}, 10^{-9}, 10^{-6}, or 10^{-3} such that the capacitance value will be multiplied by the factor before being passed to the next stage.
- Programing example: When direct measurements are not available, velocity can be computed numerically from position data as the difference between current and previous position divided by the sampling time. Write a while loop to compute velocity from position data stored in a file.

In the future offering of this course, we plan to include more examples related to mechanical engineering, civil engineering, chemical engineering, and biomedical engineering.

USE OF ONLINE TOOLS

In the fall semester of 2012, we started using two online tools in teaching C++ programming: CodeLab and the Zyante website. In this section, we first review the benefits of blended-learning as a combination of face-to-face instruction with online/web-based learning, then introduce the two online-tools we adopted.

I. Background

Learning how to program is a difficult task [6]. Many approaches have been utilized. Thevananth [7] suggests that a blended learning approach combined with problem based learning is a useful tactic for overcoming the inherent difficulties of acquiring a first programming language. Blended learning is the notion of combining face-to-face instruction with online/web-based learning [8]. Problem based learning encourages students to find the requisite knowledge and skills to solve a problem [9] providing a self-directed learning experience.

Online learning has demonstrated benefits in a variety of contexts from second language learning [10] to computer programming language learning [11]. Some of these benefits are believed to be derived from ease of access and instant feedback [10] and a tight integration of content and exercises [11]. Roberts [12] highlights the need to differentiate between codified knowledge and tacit know-how type knowledge. Lucas [11] suggests that online problem based learning is an effective mechanism for developing the tacit know-how type of knowledge in students.

Two additional dimensions of online learning emerge from the literature in the areas of cognitive load theory and increased homework engagement. Cognitive load theory proposes two types of burdens on our thinking while learning: intrinsic and extraneous [13]. Intrinsic cognitive load is the level of difficulty that a learner associates with a particular topic and cannot be addressed by external factors in the control of the instructor. Extraneous cognitive load are the peripheral issues that surround the learning activity and can be influenced by the instructor. Things like organization and relevance of examples have an impact on extraneous cognitive load. Heo and Chow [14] advocate that online problems are a way to reduce extraneous cognitive load by using a worked example approach rather than a means-ends analysis.

Numerous research results have demonstrated a positive correlation between the amount of time spent on homework and good student performance [15]. Studies focused on engineering students have indicated that feedback on homework and homework that has an impact on course grade also have positive correlation with improved student test scores [16]. The thesis of Arora et al. [16] is that graded online well-aligned homework enhances long term retention of information and results in improved student learning outcomes.

This brief literature review indicates that student engagement, learning and performance can be enhanced through well-aligned and graded online homework. Our experience with online resources in our introductory C++ course for engineers corresponds well with the literature.

II. CodeLab

CodeLab from Turing's Craft is a web-based interactive programming exercise system for introductory programming classes. It is offered for several programming languages including C++. CodeLab has over 200+ short exercises, each focused on a particular programming idea or language construct. CodeLab also supports the creation of custom exercises developed by the instructor. The student enters source code statements and the system immediately evaluates its correctness, offering hints when the submission is incorrect. We adopted CodeLab as one part of our homework system for EGR 121 – Introduction to Computer Programming in C++ in the fall of 2012 and continued use it for the spring 2013 semester.

We interviewed the co-founder of Turing's Craft, David Arnow, on May 22nd, 2013 as part of this study. CodeLab became commercially available in August of 2002 and is now used by 186 different institutions and over 15,000 students. The original goal for creating CodeLab was to create a tool to lift student scores in introductory programming classes, especially those that were receiving below average to average grades. The concept was to provide students with the opportunity for repeated practice on increasingly sophisticated exercises for specific language constructs. Based on the continued use (85% to 100% annual re-adoption rate) and anecdotal feedback, the creator of CodeLab believes the product is achieving its design goals. An additional unanticipated benefit is that since students are able to develop fundamental skills independently instructors are able to focus on more advanced concepts and material in class because they are relieved from covering the basics benefiting both low end and higher achieving students.

A typical sequence of student interactions is pictured in Figures 1-3 below. Figure 1 shows an exercise with a common mistake (incorrect Boolean condition).

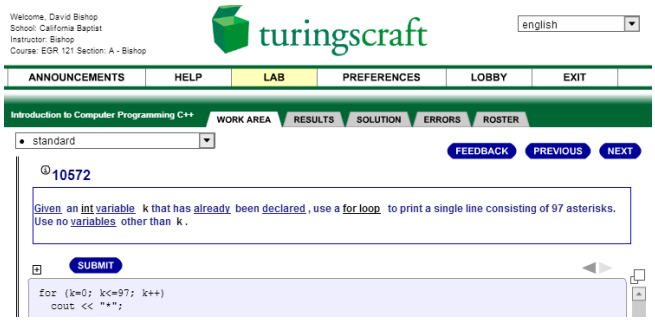

FIGURE 1
TYPICAL CODELAB EXERCISE

Figure 2 illustrates the "hint" or help feature providing feedback to the student highlighting the incorrect relational operator in the Boolean expression.

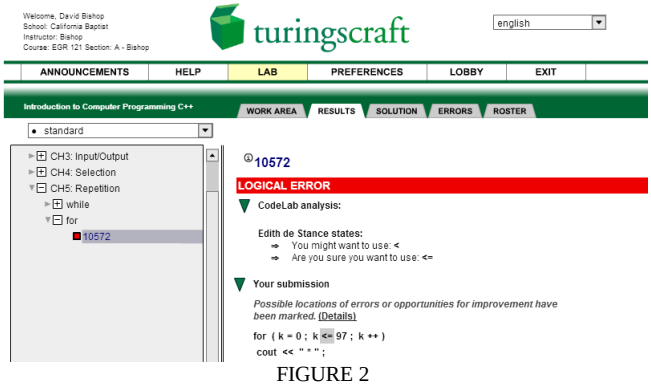

FIGURE 2
HINT FEATURE EXAMPLE

Finally, Figure 3 below shows the results of submitting the corrected solution.

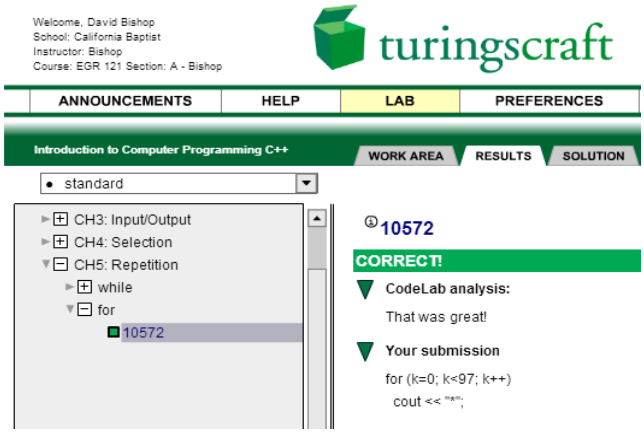

FIGURE 3
CORRECT SUBMISSION

Instant feedback, helpful hints and confirmation of correctness all contribute to enhanced learning for students. The ability of CodeLab to automatically evaluate code snippets and provide feedback is a unique characteristic of the product.

Another feature of CodeLab is its prompt and detailed feedback to the instructors. When a set of exercises are due, CodeLab provides the instructor a class roster with the status "correct, on time", "correct, late", "incorrect", or "unattempted", on each assigned problem for each student. The instructor can then only look at the incorrect submissions to see the mistakes made by students and then summarize the common mistakes during the next lecture.

III. Zyante Website

Another online resource/tool we utilized in teaching C++ is the website developed by a start-up software company called Zyante. Compared to CodeLab, Zyante is a new player in the online computer programming education market. Zyante entered the field in 2012. The website is essentially an online textbook which combines text, pictures, videos, animation, exercises and interactive elements. The interactive elements allow a student to see how a system responds to the inputs provided. The animation allows a user to walk through a new concept step by step. During the fall 2012 semester, we offered credit (5% of their total grade) to students for completing assigned Zyante exercises. Figure 4 below shows a typical content and animation section in Zyante.

In addition to content and animations Zyante offers exercises with immediate feedback. The feedback differs from that of CodeLab. In CodeLab the student is provided an analysis of their proposed solution with some hints regarding possible improvements. With Zyante the exercise is assessed immediately. Two options emerge if the student answered incorrectly. For some questions, like true/false, the student can just change their answer to get it correct. For more complex problems hints may be provided and a "show" button which will show and explain the correct answer. An example of the exercise feature is shown in Figure 5.

FIGURE 4
EXAMPLE ZYANTE C++ CONTENT AND ANIMATION

FIGURE 5
ZYANTE EXERCISE EXAMPLE

The Zyante website had several major updates throughout the academic year. Some of them were based on problems or suggestions reported by students or instructors.

STUDENT SURVEY RESULTS

At the end of the past two semesters, we conducted surveys to collect student feedback on this course. Besides questions on the course learning outcomes and teaching methods, we also included the following three questions:

- How do you like CodeLab?
- How do you like the Zyante site?
- Would you recommend using the CodeLab and Zyante systems in addition to existing teaching materials (PowerPoint slides, labs, programming assignments, sample code, etc.), but without any hardcopy textbook, in future offerings of this course? Why or why not?

A total number of 48 students took the surveys, with 29 in the fall and 19 in the spring. 46 students responded to the first question, 45 responded to the second question, and 40 responded to the third.

I. CodeLab

The answers to the first question which is related to CodeLab were summarized in the Figures 6 & 7. Figure 6 shows that 36 out of 46 students commented that their experiences with CodeLab were positive. As shown in the pie chart on the right, 7 of them stated specific features of CodeLab that helped them learning, namely immediate feedback and multiply tries. Further explanations on these comments were listed in Table II. Figure 7 shows the areas for improvement suggested by students. Further explanations on these areas were also listed in Table II.

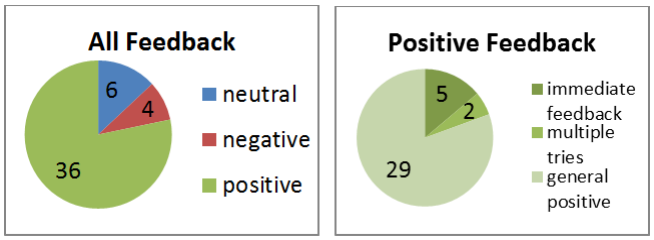

FIGURE 6
STUDENT FEEDBACK ON CODELAB.

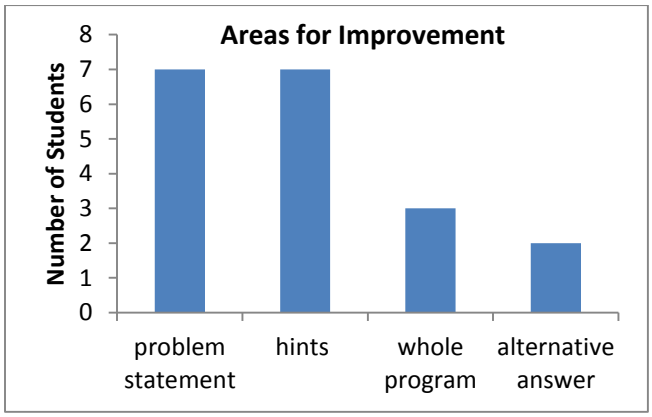

FIGURE 7
STUDENT SUGGESTIONS ON AREAS FOR IMPROVEMENT.

TABLE II
LEGEND AND AXIS DESCRIPTION FOR FIGURE 6 & 7

Legend/Axis	Description
Student Feedback on CodeLab	
neutral	Students whose attitudes were neither like nor dislike CodeLab.
negative	Students who disliked CodeLab.
positive	Students who liked CodeLab.
immediate feedback	Students who commented that they liked CodeLab because it gives immediate feedback.
multiple tries	Students who commented that they liked CodeLab because it allows a user to try multiple times.
general positive	Students who liked CodeLab but did not give any specific reason.
Areas for Improvement	
problem statement	Students who commented that sometimes the problem statements for CodeLab exercises were difficult to understand.
hints	Students who commented that sometimes CodeLab does not give hints on incorrect answers and sometimes the hints were not helpful.
whole program	Students who commented that it was not clear to them how the short code segments they wrote as CodeLab exercises were used in complete programs.
alternative answer	Students who commented that CodeLab sometimes does not accept alternative answers to certain problems.

II. Zyante Website

As mentioned earlier, we offered 5% credit to students for completing assigned Zyante exercises in the fall semester of

2012. Among the 26 students who responded to the Zyante question on the survey, 24 students actually used the Zyante website, but less than 60% of them indicated that their experiences were positive. A couple of students mentioned that they didn't like the fact that Zyante exercises gives the user correct answers immediately. One student stated:
"It doesn't help that you get credit for doing the homework just by clicking 'check'. It shows you the correct answer even when you haven't written anything."

Therefore, in the spring of 2013, we removed the requirement of using the Zyante website, but recommended it to the students as a supplement to the textbook in the beginning of the semester. As expected, the participation rate dropped dramatically from above 90% in the fall to slightly above 40% in the spring, based on the survey results. However, among those who did use the Zyante website, the percentage of favorable comments increased by almost 30%. The comparison of student responses on Zyante between the fall and spring semesters is shown in Figure 8.

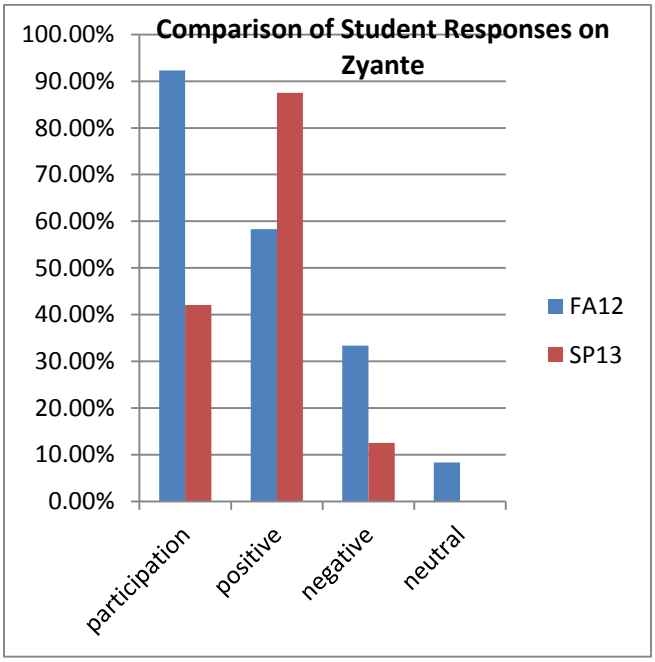

FIGURE 8
COMPARISON OF STUDENT RESPONSES ON ZYANTE BETWEEN FALL 2012 AND SPRING 2013 SEMESTERS.

III. Preference on Using Online Materials Only

The responses to the third question, regarding whether the online materials are sufficient in delivering the course, together with the existing lectures, computer labs, and programming projects, were quite mixed. As shown in Figure 9, about half of those who responded were in favor of using online materials only, but 38% recommended keeping the textbook.

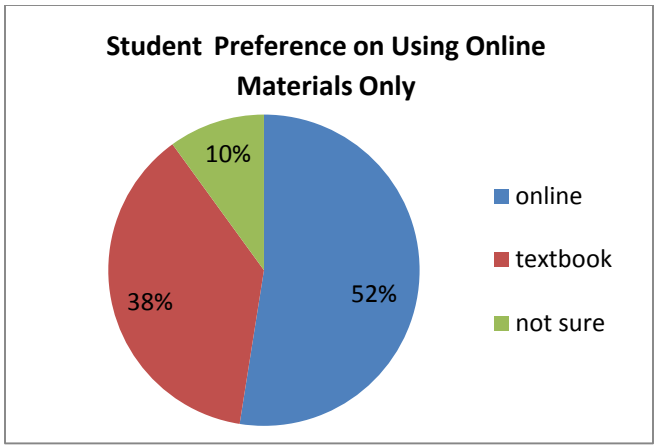

FIGURE 9
SURVEY RESULTS ON STUDENT PREFERENCE ON USING ONLINE MATERIALS ONLY

Most of those who felt that a hardcopy textbook is unnecessary for learning C++ programming mentioned that they rarely used the book throughout the semester. Among those who recommended keeping the textbook, they indicated in their comments that the textbook is an important resource, for example:

- *"I found that the textbook was a more reliable source than Zyante or CodeLab."*
- *"I think the text book helps. It gives the students more information while CodeLab and Zyante provide hands-on learning."*
- *"I would like to have both because the book helps me review the material more and CodeLab helps me apply what I have learned."*
- *"I found the book to be very helpful. I used it when studying for exams."*

DISCUSSIONS AND CONCLUSIONS

Obviously the interactive practice provided by CodeLab and multimedia contents on the Zyante website are valuable additions to traditional textbook based teaching of computer programming. However, we did not see significant improvements in students' testing scores after the adoption of these online tools. The reasons why there was not as much improvement as we anticipated were not reflected directly by the survey responses. Here we would like to discuss a few concerns that we identified as possible obstacles that may have hindered students in taking full advantage of these online tools.

- There were gaps in the integration of existing course materials with the newly adopted online tools. For example, for certain topics covered in the lectures, such as the enumeration data type, there were very few exercises in CodeLab. Some exam problems were similar to exercises in the textbook, for example, read a code segment and predict the results, however, were quite different from CodeLab exercises focusing on writing code. Also the lectures were organized based on

the textbook and could not be mapped to the chapters on the Zyante website exactly.

- We need more effective approach to encourage students to utilize the Zyante website than either giving credit for going through some exercises on the website, or simply recommend it as an option. For example, some animations on the Zyante site can be linked to PowerPoint slides and shown to students during lectures. Based on feedback from instructors and students, Zyante improved various areas such as user interface and account management from the original version in the fall of 2012 to the new version in the spring of 2013. We hope to see continuous improvement in the Zyante website that makes it beneficial to more students.
- One more concern is plagiarism. Some students found solutions to CodeLab problems from online forums or tutorials. This kind of plagiarism was not detected through grading because of the auto-grading mechanism in CodeLab. Those students earned credits on their CodeLab homework assignments but did not really master the programming skills. We have discussed this with Turing's Craft and have received action plans and suggestions. Turing's Craft warns the "solution providers" of their violation of copyright, and also works on features in the auto-grading process to catch plagiarism. On the instructor's side, giving short quizzes with CodeLab problems (or slightly modified versions) may prevent plagiarism.

Our experience and student feedback indicate the value of online tools in introductory programming courses. There is room for improvement but the increase in ability of students to exercise their learning styles; the increased engagement of students with the material and the reduced load on instructors are all good reasons to pursue incorporation of online learning tools in introductory programming courses.

REFERENCES

[1] Etter, D. M., & Ingber, J. A., *Engineering Problem Solving with C++*, 3rd Edition, Prentice Hall, 2012.

[2] Dale, N., & Weems, C., *Programming and Problem Solving with C++*, 5th Edition, Jones and Bartless Publishers, Inc., 2010.

[3] Malik, D. S., *C++ Programming: Program Design Including Data Structures*, 6th Edition, Cengage Learning, 2013.

[4] Hanly, J. R., *Essential C++ for Engineers and Scientists*, 2nd Edition, Pearson Education, 2002.

[5] Bronson, G. J., *C++ for Engineers and Scientists*, 4th Edition, Cengage Learning, 2013.

[6] Lahtinen, E., Ala-Mutka, K., & Järvinen, H.-M., " A Study of the Difficulties of Novice Programmers", *the ACM SIGCSE Bulletin*, 2005.

[7] Thevananth, J., "Using a Blended Learning Approach to Support Problem-based-learning for Teaching Computing and Programming with Second Year Students - A Case Study Based on C-programming", Annual International Conference on Education and E-Learning, Singapore, 2011.

[8] Oliver, M., & Trigwell, K., "Can 'Blended Learning' Be Redeemed?", *E-learning and Digital Media*, Vol. 2, No. 1, 2005, pp.17-26.

[9] Kilroy, D., "Problem Based Learning", *Emergency Medicine Journal*, Vol. 21, No. 4, 2004, pp. 411-413.

[10] Sagarra, N., & Zapata, G. C., "Blending Classroom Instruction with Online Homework: A Study of Student Perceptions of Computer-Assisted L2 Learning", *ReCALL : the Journal of EUROCALL*, Vol. 20, No. 2, 2008, pp. 208-224.

[11] Lucas, A. R., "Using WeBWorK, a Web-Based Homework Delivery and Grading System, to Help Prepare Students for Active Learning", *Primus : Problems, Resources, and Issues in Mathematics Undergraduate Studies*, Vol. 22, No. 2, 2012, pp. 97-107.

[12] Roberts, J. , "From Know-how to Show-how? Questioning the Role of Information and Communication Technologies in Knowledge Transfer", *Technology Analysis & Strategic Management*, Vol. 12, No. 4, 2000, pp. 429-443.

[13] Cooper, G., "Cognitive Load Theory as an Aid for Instructional Design", *Australian Journal of Educational Technology* Vol. 6, No. 2, 1990, pp. 108-113.

[14] Heo, M., & Chow, A., "The Impact of Computer Augmented Online Learning and Assessment Tool", *Journal of Educational Technology & Society*, Vol. 8, No. 1, 2005, pp. 113-125.

[15] Cooper, H., "Synthesis of Research on Homework", *Educational Leadership*, Vol. 47, No. 3, 1989, pp. 85-91.

[16] Arora, M. L., Rho, Y. J., & Masson, C. , "Longitudinal Study of Online Statics Homework as a Method to Improve Learning", *Journal of STEM Education : Innovations and Research*, Vol. 14, No. 1, 2013, pp. 36-44.

AUTHOR INFORMATION

Grace Ni Associate Professor of Electrical and Computer Engineering, College of Engineering, California Baptist University, gni@calbaptist.edu

David Bishop Assistant Professor of Software Engineering, College of Engineering, California Baptist University, dabishop@calbaptist.edu

Anthony Donaldson Dean and Professor, College of Engineering, California Baptist University, adonaldson@calbaptist.edu