Programming Without Computer: Revisiting a Traditional Method to Improve Students' Learning Experience in Computer Programming

Mr. S. Cyrus Rezvanifar, University of Akron

S. Cyrus Rezvanifar is a Ph.D. student in Biomedical Engineering at The University of Akron. He has also served as a research assistant in Cleveland Clinic Akron General since 2016, where he conducts research on biomechanics of human knee joint and patellar instability. In 2016, he received a doctoral teaching fellowship from the College of Engineering at The University of Akron. Through this teaching program, he has served as an instructor for several undergraduate-level courses, and he has conducted educational research on the effect of various learning techniques on improving students' self-efficacy and overall learning experience.

Programming Without Computer: Revisiting a Traditional Method to Improve Students' Learning Experience in Computer Programming

Introduction

During the past three decades, computer programming has been recognized as an essential skill and a necessary element in education. Previous studies have reported numerous cognitive outcomes from learning to program [1]. Feurzeig et al. [2] presented an extensive list of cognitive benefits of learning computer programming and argued that "the teaching of the set of concepts related to programming can be used to provide a natural foundation for the teaching of mathematics, and indeed for the notions and art of logical and rigorous thinking in general". Additionally, computer programming is an increasingly required skill in science, technology, engineering, and mathematics (STEM) fields, and there is a pivotal demand for up-to-date techniques in its teaching to enhance students' learning experience and professional competency. Nevertheless, teaching and learning programming has remained a challenge for both educators and students of all levels. The differences between expert versus novice programmers, and knowledge versus strategy approach investigated in the literature [3-5] are among the reasons behind the foregoing challenge in learning to program. Many studies have thus far investigated challenging topics in programming from both students' and educators' perspective [6] and have introduced different approaches such as "syntax-free", "literacy", and "problem-solving" to address those issues [7]. In the present study, we have investigated the hypothesis that practicing the knowledge of programming without computer can substantially improve students' learning experience with various core topics of programming such as loops, arrays, and conditional statements.

Method

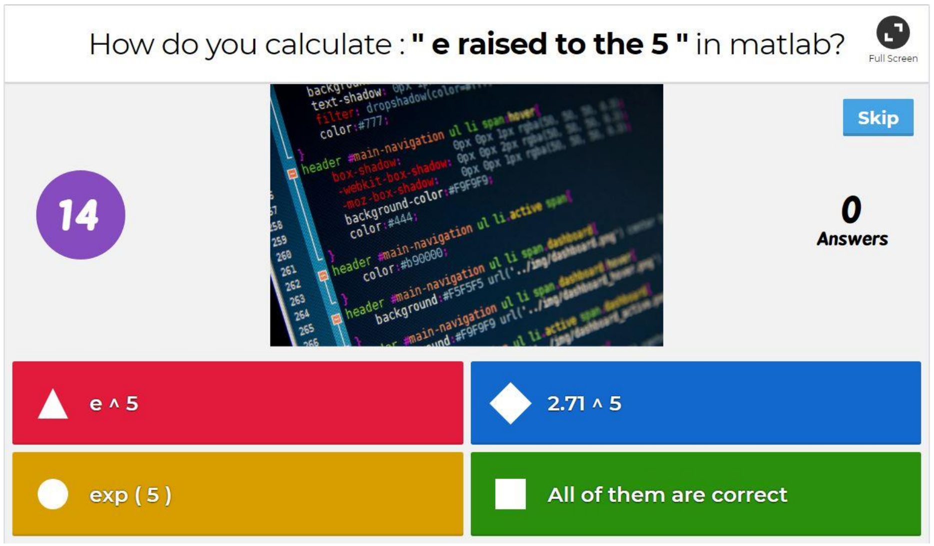
In our sophomore-level Biomedical Computing course (4800:220), we integrated a "programming without computer" approach into teaching MATLAB programming. Many students in our class, as discussed in the literature [6], had issues with basic concepts such as defining variables, selection structures, and loops. In addition, we noticed that many students were capable of writing a loop to do a certain task, however, they were often not able to correctly predict the output of a given program. We believe that such observations are quite common in computer programming classes, and typically occur due to the memorization of syntax and quasi-templates, as opposed to deep retention. Hence, we required the students to complete in-class assignments without using MATLAB and write down the output of a very short program—with MATLAB syntax—on a piece of paper. These short programs were meticulously designed by the course instructor, each addressing a specific detail in using a single concept such as *loops* (Fig.1). For each program, the students were given a few minutes to write down their responses, with the course instructor and teaching assistants walking around the class and helping those who were struggling with the problem.

```
j = 1;
i = 4;

while i
        x ( j ) = i ^ 2 ;
        i = i - 1 ;
        j = j + 1 ;
end
disp ( x )
```

Figure 1. Example of a "programming without computer" problem. Students were required to write down the output of this program as an in-class exercise in less than 5 minutes.

The same approach was also incorporated in short quizzes in the beginning of each session. For each quiz, a free game-based online learning platform (http://kahoot.com) was used to display three multiple-choice questions on the screen for all students, asking them to use their cell phones as response remotes and participate in the quiz. Each question consisted of a very short program followed by potential outputs presented in multiple choices. Students were required to go through each program and select the correct answer in 30 to 60 seconds—depending on the complexity of the question—without using MATLAB (Fig.2). Students inserted their names in the online platform as participants at the beginning of the quiz. After each question, the number of people who selected each of the choices was revealed. Before moving on to the next question, we encouraged students to initiate discussions, explain their reasoning for selecting the correct answer, and discuss why other choices were incorrect. This active learning approach allowed students to evaluate their retention of the previous lecture material, and also learn about common mistakes and pitfalls through an interactive and exciting experience. As an incentive for studying before each class session, students who provided correct answers to all three questions of a quiz received bonus points toward their final grade.


How do you calculate : **" e raised to the 5 "** in matlab?

e ^ 5

2.71 ^ 5

exp (5)

All of them are correct

Figure 2. Example of a multiple-choice quiz/game question (http://kahoot.com)

In both activities, the questions can either focus on lower levels of Bloom's taxonomy by addressing basic concepts and syntax, or assess students' deep learning by questions that require a high retention of the core topics, yet can be answered in less than a minute without using the computer. Of note is the fact that designing the questions in a way that would allow the foregoing outcomes can be done by the course instructor in a relatively short time and would not demand a significant amount time or preparation.

We also incorporated the "programming without computer" technique into the midterm and final exams. In both tests, "Question 1" consisted of three short pieces of programs—similar to the one demonstrated in Fig.1. The students were given 15 minutes to write down their answers to those questions without using computer, submit their solutions, and then start working on the rest of the exam—typically two programming questions with multiple sections—using MATLAB.

To statistically investigate the effect of this approach on students' performance and learning experience, two-way ANOVA along with post hoc Tukey's tests were conducted to compare students' grades in "Question 1" and "Other Questions Combined" (total grade – Question 1 grade) in midterm and final exams. This subtraction would eliminate the effect of "Question 1" grade on the overall grade and serve as a control against other influential factors on students' performance. Statistical significance level was set at α=0.05.

Results

Students' grades in "Question 1" significantly improved ($p < 0.0001$) from Midterm (53% ± 23%, N=24) to Final Exam (79% ± 16%, N=24), while only a slight improvement was observed in "Other Questions Combined" grades between the two exams (Midterm = 76% ± 11% vs. Final = 86% ± 13%). On the other hand, there was a significant difference between students' grades in "Question 1" versus "Other Questions Combined" in the Midterm Exam. However, students demonstrated an almost equal performance in the two categories of questions in the Final Exam.

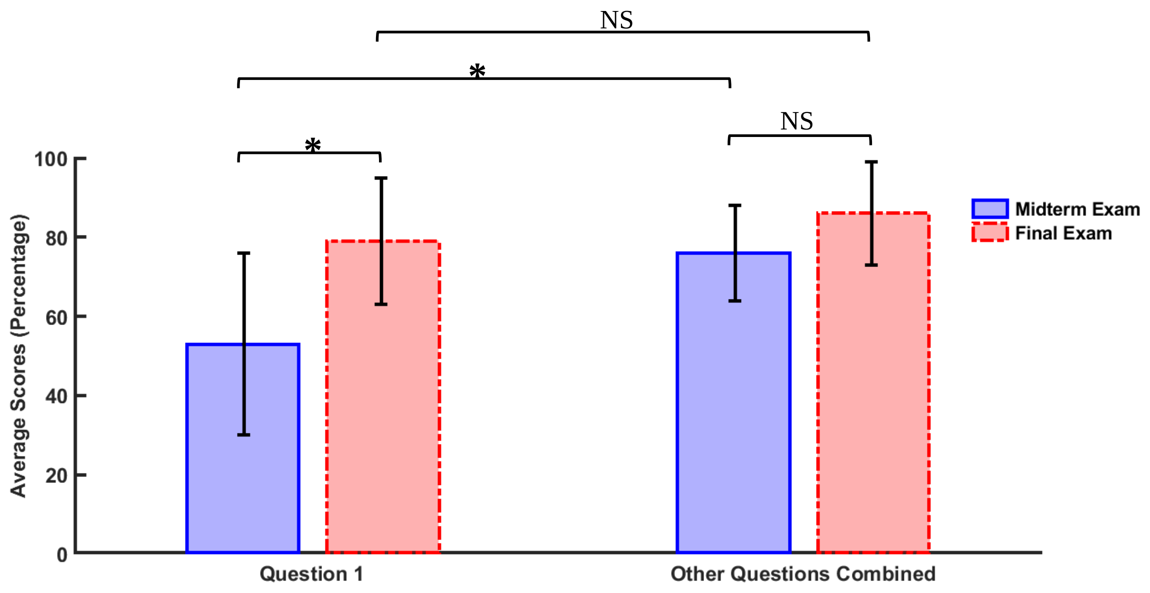


Figure 3. Students' average scores (±SD) as a percentage for "Question 1" vs. "Other Questions Combined" in Midterm and Final Exams. "*" indicates statistically significant difference

Discussion

The results confirm our hypothesis that the proposed “programming without computer” approach significantly improves students’ performance and deeper understanding of core programming topics, without an adverse effect on their learning experience of programming in MATLAB. The proposed approach agrees well with previously introduced techniques for teaching programming [3, 4, 6-8]. Although writing computer programs on paper were previously incorporated when limited number of computers were accessible, it should be noted that “programming without computer” in class exercises and exams is substantially different from solely writing the same piece of code on paper. A multitude of students—mostly novice programmers—memorize sections of code as templates and try to “plug and chug” those templates into different programs without realizing the fundamental syntax and formulation differences. This method encourages students to ponder upon each and every question separately and avoid forming unconscious mental patterns in programming. Consequently, students will learn the fundamentals of each programming concept in depth, before getting involved with numerous details of syntax. Furthermore, a deep retention of computer programming as opposed to memorization of syntax and templates would result in aforementioned cognitive benefits in other courses such as mathematics.

A limitation in our study is that other factors such as a probable difference in difficulty level of the two exams might have affected students’ performance. Moreover, the effect of “programming without computer” technique on other programming questions cannot be assessed in isolation using the currently available data. An independent control group with the exact same conditions except for the “programming without computer” experience could enable more reliable statistical inferences. However, since this technique has indicated significant improvements on students’ performance and learning experience, it would not be ethically justifiable to eliminate this experience in a future course for the purpose of having a control group. Nonetheless, an improved study design will be incorporated in our future study to more accurately and specifically investigate the effect of this technique on students’ performance and learning experience in computer programming.

As a side note, since using cell phones is typically forbidden during class time and exams, allowing students to use their cell phones as response remotes broke the routine of the class, and made students excited to take a quiz. We believe that this behavior is rarely observed among college students and should be considered by educators. In 2002, Guzdial and Soloway [9] referred to college students of that time as “Nintendo generation”. They argued that wisely implementing students’ preferences and hobbies into the *theme* of course syllabus and assignment deliverables would enhance students’ involvement, motivation, and deep learning. We believe that the current generation could be referred to as “cell phone generation”. Hence, using cell phones as response remotes for quizzes, acquiring cell phone sensors data for computing projects, and teaching image processing and two-dimensional filters in a *social media* theme are examples of numerous approaches that would engage students and lead into their deep learning.

Acknowledgment

The author would like to acknowledge the Doctoral Teaching Program in College of Engineering at The University of Akron for providing teaching fellowships for S. Cyrus Rezvanifar.

References

[1] Pea, R. D., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New ideas in psychology*, 2(2), 137-168.

[2] Feurzeig, W., et al. (1981). Microcomputers in education. *National Institute of Education*. Venezuela Departmentof Health, Education and Welfare.

[3] Robins, Anthony, et al. (2003). Learning and teaching programming: A review and discussion. *Computer science education* 13.2. 137-172.

[4] Winslow, Leon E. (1996). Programming pedagogy—a psychological overview. *ACM Sigcse Bulletin* 28.3. 17-22.

[5] Davies, Simon P. (1993). Models and theories of programming strategy. *International Journal of Man-Machine Studies* 39.2. 237-267.

[6] Milne, Iain, and Glenn Rowe (2002). Difficulties in learning and teaching programming—views of students and tutors. *Education and Information technologies* 7.1. 55-66.

[7] Fincher, Sally. 1999. What are we doing when we teach programming? *fie. IEEE*

[8] Wulf, T. (2005). Constructivist approaches for teaching computer programming. *In Proceedings of the 6th conference on Information technology education*. 245-248

[9] Guzdial, M., & Soloway, E. (2002). Teaching the Nintendo generation to program. *Communications of the ACM*, 45(4), 17-21.