2018 ASEE Mid-Atlantic Section Spring Conference: Washington, District of Columbia Apr 6

A Study of Online Assessment Tools to Practice Programming and Their Effect on Students Grades

Dr. Jose M Reyes Alamo, CUNY - New York City College of Technology

José M. Reyes Álamo did his undergraduate studies in Computer Science at the University of Puerto Rico at Bayamón. After graduation he worked in the industry but his strong passion for education and research brought him back to school a year later. He went to Iowa State University (ISU) to pursue his Ph.D. in Computer Science. During his time at ISU he worked in the Smart Home Lab researching topics in Software Engineering, Smart Homes, Pervasive Computing, Formal Methods, and Wireless Sensors. He published several papers and completed his dissertation entitled ”A framework for safe composition of heterogeneous SOA services in a pervasive computing environment with resource constraints”. In ISU he was also involved in several other activities like organizing the GMAP Symposium, participating in student organizations, and working as research assistant. After graduation he worked for a trading software company in New York City but once again his love for academics brought him back, this time as a professor in Computer Engineering Technology at CUNY - New York City College of Technology.

A Study of Online Assessment Tools to Practice Programming and Their Effect on Students Grades

José M. Reyes Álamo

CUNY - New York City College of Technology

Abstract

"Practice makes perfect" is an old phrase that proves true in many aspects of the life of a computer engineer. Students in programming courses are reminded constantly by their instructors to practice in order to become better developers. Traditionally, book exercises have been used or assigned to students for practicing programming. However unless these exercises are counted for credit, some students will lack the motivation to do them. On the other hand, assigning too many problems for credit can become a time consuming activity for both students and faculty as well as a grading burden for instructors. It is also known that there are a lot of online and electronic resources for practicing programming, but students can get overwhelmed with so many tools. In this paper we present our preliminary results of how using online assessment tools can help student practice and improve their programming skills. The tools used provide immediate feedback and automatic grading. The hypothesis is that these tools help students to practice more and by giving them immediate feedback and quick grading, they get better at programming and consequently get better test scores. Preliminary data collected shows this to be the case. In this paper we present different scenarios how these tools were used and their effect in the final exam results in different semesters.

Keywords

Assessment, online, practice, learning, programming.

Introduction

Computer engineering students are well aware that "practice makes perfect", especially when it comes to programming. Many students when they take their first programming class are exposed for the first time to writing code. To master the first programming language it is crucial to practice the commands and strategies learned. Many times the textbooks have a set of exercises at the end of each chapter. They may also have a separate set of lab problems that faculty can assign. This has been the traditional way of giving students homework. However, unless these problems are counted for credit some student may lack the motivation to practice using them [1]. Other students may feel overwhelmed with the assignments or get frustrated when using a compiler or IDE as they may not understand where the problematic code might be [2].

On the faculty side of things, assigning programming problems is always a challenge. Assigning too few problems poses the risk for students not practicing enough. Assigning too many problems may become a burden at the time of grading and providing feedback [3]. More faculty are relying on online system for practicing programming in order to provide students with more problems to practice, give them immediate feedback, and allow them to work based on their

schedules. The research on electronic assignments usually focuses on the assignments themselves. However in this paper, we study the possible effects that giving students extra tools may have in their exam results. The hypothesis is that these tools help students to practice more, provides them immediate feedback, and by practicing at times that are convenient for them they get better at programming and consequently get better test scores. The rest of the paper is organized as follows: first we survey some of the tools available, then we present how some of these tools were implemented into one of our courses, followed by the results of the students' exams during several semester, and finally we present our conclusions and future work.

Survey of assessment tools available

There are several tools to learn and practice programming online. Many of them consist of a set of exercises divided by topics such as CodingBat [4], but these are more stand-alone, independent tools for developers in general. Other tools are more appropriate for classrooms as they provide a better interface and even have a way for the instructor to access the students' completed work. Some of these tools are the following:

- CodeCademy [5] – a very popular online platform to learn programming languages such as Python, JavaScript, Java, Ruby, and HTML. Most CodeCademy courses that teach the basics are free while other courses are premium and paid. The CodeCademy style is more like a guided tutorial where the student have to complete a set of particular tasks as well as other problems where the student needs to complete the missing code. Good tool for beginners and for those wanting to refresh the knowledge.

- Programmr [6] – provides coding simulators for several languages. Their courses are free and in the platform students can solve problems in Java, C, C++, Objective C, Angular JS. At Programmr the student can code, compile, and run projects in the browser in many of the most popular programming languages available. They have several courses as well as independent problems with instant feedback.

- Turing's Craft CodeLab [7] – describes itself a "web-based interactive programming exercise system for intro programming classes in Python, Java, C++, C, JavaScript, C#, VB and SQL." CodeLab was designed and developed by a group of faculty to reduce attrition and provide the tools for students to practice and become better programmers. CodeLab has been adopted by several Colleges and Universities across the world. In CodeLab students are presented with a problem, they enter the code, run it and the systems provides feedback and hints.

- Practice-It [8] – another web application especially designed to learn and practice Java programming. This application was created by the University of Washington to help students in their introductory Java class. Many of the same problems found on the textbook can also be found in this online tool, with the advantage of instant grading and feedback in case there are errors.

- Code Step by Step [9] - from the creators of Practice-It, comes CodeStepByStep. It is also an online tool to help students learn how to program but supports multiple programming languages. It is also intended to a broader audience and provide tools for instructors to

create courses, select exercises, as well as to create their own problems. Very useful tool for anyone teaching Java, Python, C/C++, JavaScript, and PHP among others.

Other tools that are similar to the ones explained above are Computer Science Circle[10] and PySchools[11] specifically focused on Python programming as well as CloudCoder [12] an open source exercise system for instructors of introductory programming courses in C/C++, Java, Python, and Ruby to easily assign short exercises to students. In the next section we explain how some of these tools were introduced and used in a Java programming course.

Implementation of Assessment Tools by Semester

The Department of Computer Engineering Technology at the New York City College of Technology has a Java programming course where some of these tools were used. In an effort to reduce the attrition rate and improve the students' final exam scores and overall grade, we concluded that students were not practicing enough. Therefore we looked for ways to give our students more opportunities to practice and tried out online tools. We collected results in the Fall semester for the academic years 2013-2014 to 2016-17 of the final exam results before any curve. Every Fall semester we introduced different tools and below we explain how they were integrated into the class:

- Fall 2013: We introduced the tool Programmer as extra credit assignment. Students created an account, several exercises were assigned and they showed the results to the instructor to receive the extra points.

- Fall 2014: During this semester we used CodeCademy as part of their lab assignment as well as their extra credit. They completed some of the sections and these were counted for credit, while other sections were optional and counted as extra credit. Because the CodeCademy Java problems are limited, we kept Programmr as a reference, and recommended it to students who wanted to practice more.

- Fall 2015: In this semester we moved to Turing's Craft as made a significant number of labs count for credit using this tool. The advantage with CodeLab was that it was offered free for our College and faculty was able to always see the students' results. We kept CodeCademy as the tool for the extra credit and as a review for the final exam. The fact that CodeCademy Java exercises are fewer, it served well for that purpose. We also recommended Programmr as a reference to students who wanted even more practice.

- Fall 2016: That semester we changed the book and with it the tools available. We started using CodeStepbyStep for the assignments for credit. We also kept CodeCademy for extra credit and as reference we recommended Pracite-It and Programmr.

Informal surveys and conversations with students about these tools showed that they appreciated the fact that they could practice more, practice outside of the classroom and having extra time to finish homework. To get evidence of the feedback we received from students we collected data about the results in their final exams, and we present it in the next section.

Results

We received informal feedback from students about the benefits of using these online programming practice tools. To formally acknowledge these, we collected data of their final exam grade before any curving. In Table 1 we show the mean and median statistics of the final exam for the Fall 2013, Fall 2014, Fall 2015, and Fall 2016 semester, after each tool was introduced.

Semester	Fall 2013	Fall 2014	Fall 2015	Fall 2016
Final Exam Mean	58.25%	70.77%	74.68%	82.86%
Final Exam Median	53.50%	67.50%	74.50%	86.40%

Table 1: Final Exam mean and median data by semester

This data can also be seen in the Figure 1 as a chart. As you can see in the chart the final exam scores mean and median have been increasing as we rely more on these tools for our students to practice. Students went from a final exam mean of 58.25% in the Fall 2013 to a 82.86% in Fall 2016 for a 24.61% increase, which we believe is a great improvement. The median also rose from 53.50% to 86.40% for an increase of 32.50%, another great improvement. Students in general performed better in their final exam and that change was gradual and increasing throughout the semesters as can be appreciated from Figure 1.

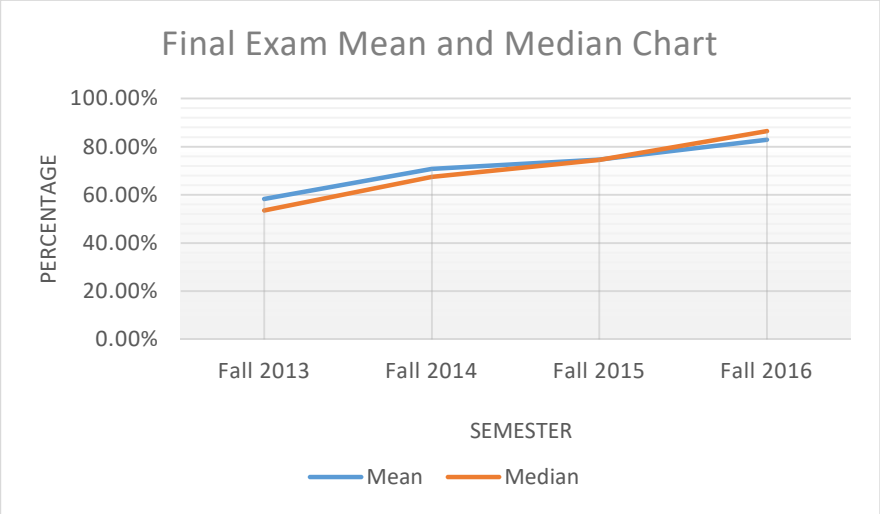

Figure 1: This is a caption for the first figure

Conclusions and Future Work

We believe that online programming practice tools have a positive impact on student's learning. We also believe based on our preliminary data that it helps them in their exam grades, as they have more opportunities to practice and get immediate feedback. Also they can easily review the problems where they struggle and reinforce those skills. We plan to expand this work by studying the effect in multiple sections of the course, as well as including data from other programming courses.

References

1. Bergin, S. & Reilly, R. The influence of motivation and comfort-level on learning to program. (2005). (Accessed: 20th February 2018)
2. Rodrigo, M. M. T. & Baker, R. S. J. d. Coarse-grained Detection of Student Frustration in an Introductory Programming Course. in *Proceedings of the Fifth International Workshop on Computing Education Research Workshop* 75–80 (ACM, 2009). doi:10.1145/1584322.1584332
3. Cheang, B., Kurnia, A., Lim, A. & Oon, W.-C. On automated grading of programming assignments in an academic institution. *Comput. Educ.* **41,** 121–131 (2003).
4. CodingBat Java. Available at: http://codingbat.com/java. (Accessed: 20th February 2018)
5. Codecademy - learn to code, interactively, for free. *Codecademy* Available at: https://www.codecademy.com/. (Accessed: 19th February 2018)
6. Programmr | Your Online Code Lab. Available at: http://www.programmr.com/. (Accessed: 19th February 2018)
7. Turing's Craft. Available at: https://www.turingscraft.com/. (Accessed: 20th February 2018)
8. Practice-It, a web-based practice problem tool for computer science students. Available at: https://practiceit.cs.washington.edu/. (Accessed: 20th February 2018)
9. CodeStepByStep, a web-based practice problem tool for computer science students. Available at: https://www.codestepbystep.com/. (Accessed: 20th February 2018)
10. Computer Science Circles | 01000011 01010011 01000011.
11. Python Tutorial. Available at: http://www.pyschools.com/. (Accessed: 20th February 2018)

12. CloudCoder - Welcome to CloudCoder! Available at: https://cloudcoder.org/. (Accessed: 20th February 2018)

José M. Reyes Álamo is an Assistant Professor in the Department of Computer Engineering Technology at the New York City College of Technology of the City University of New York. His research interest include smart homes, internet of things, embedded devices, software engineering, and information retrieval. He received his Ph.D. in Computer Science from Iowa State University and his B.S. in Computer Science from the University of Puerto Rico at Bayamón.