A Toolkit for Assessments in Introductory Programming Courses

Eric Xing
eric.xing819@topper.wku.edu
Western Kentucky University
Bowling Green, KY, USA

Guangming Xing
guangming.xing@wku.edu
Western Kentucky University
Bowling Green, KY, USA

ABSTRACT

Traditional paper-based exams and LMS-provided online exams for introductory programming courses are not aligned with learning objectives that emphasize problem-solving and coding skills. In this poster, we present a cloud-based assessment solution for introductory programming courses. First, we discuss the requirements and challenges of conducting frequent assessments. We then outline the functions in our online exam toolkit that allow instructors to administer versatile assessments. Instead of relying on a traditional lockdown browser, the plagiarism and cheating detection in our toolkit allows instructors to administer exams in any modern browser for face-to-face classes.

1 INTRODUCTION

It has been shown that frequent assessments help improve knowledge retention and may improve understanding of course content [1]. However, it is difficult to conduct frequent assessments for large programming courses in dedicated computer labs due to limited resources. This has motivated bring-your-own-device exam[2], but most exams are restricted to only multiple-choice or fill-in-the-blank questions. Additionally, these exams require the use of a lockdown browser to prevent cheating. Setting up these lockdown browsers unnecessarily increases the complexity of assessments, creating hassle for both the student and teacher. It is therefore important to develop methods to detect and deter student cheating without introducing additional software. Additionally, containerized technologies have increased the popularity of web IDE, making them common in computer science classrooms. These advancements naturally call for their use in computer science assessments. Lastly, educators spend a large amount of time grading assessments. While LMSs provide a degree of automation, their lack of diversity in question types limits the usefulness of current auto-grading.

To address the aforementioned limitations, we propose an online assessment toolkit for introductory programming courses. We summarize its contributions as follows:

- Our toolkit augments traditional question types by using web IDEs to provide realistic coding questions in an assessment.
- Our toolkit eliminates the need for a lockdown browser by incorporating deterrents and cheating detection.
- Our toolkit vastly reduces the time needed for instructors to author and grade assessments by using AI technologies.

SIGCSE 2023, March 15–18, 2023, Toronto, ON, Canada

ACM ISBN 978-1-4503-9433-8/23/03.
https://doi.org/10.1145/3545947.3576231

2 OVERVIEW

Ideally, assessments in programming courses should determine student understanding of programming concepts as well as their ability to write actual code. Assessing basic programming concepts is usually achieved well through multiple choice, fill-in-the-blank, and matching questions, which are available in most LMSs. Coding questions, however, are generally not supported by LMSs. This is because they require more than simple text boxes, as most students need IDE features to write code. Recent efforts by Computer Science education systems, like PrairieLearn [3], offer auto-grading of coding problems, but they lack one or more features in our toolkit.

The main component of the student exam module is the exam web interface. Instead of relying on a lockdown browser, we present the exam content in full-screen mode, which forces the student to stay on the exam. When a student switches to a different tab or app for more than two seconds, a color change will be triggered in the web interface, and the instructor will be notified of this deviation.

Additionally, all student interactions with the web IDE are recorded, so any improper use of the web IDE for code tracing problems is identified through cheating detection.

Lastly, the authoring modules and grading module provides multiple tools that increase the convenience and flexibility of frequent assessments. The system supports auto-grading of all question types, with the capability to handle partial credit. Additionally, it supports manual grading of the coding questions, which allows code editing and running test cases.

3 CONTRIBUTIONS AND FUTURE RESEARCH

We have piloted the system in two courses for weekly quizzes and exams with more than 150 students. The setup for an exam takes significantly less time when compared to a traditional LMS, such as Blackboard, and auto-grading reduces the amount of time needed to return assessment results. In our pilot, quizzes are completely auto-graded, while exams are mostly auto-graded, with the exception of giving partial credit on coding questions. In our experiments, this system has cut the time needed for grading by over 60%.

Research to quantify the impacts on the instructors and the students is being conducted. In the future, we hope to find more collaborators by sharing our experience so that we can build a repository of assessment questions that can be shared by educators.

REFERENCES

[1] B. J. Huelser K. B. McDermott M. A. McDaniel, P. K. Agarwal and H. L. I. Roediger. 2011. Test-Enhanced Learning in a Middle School Science Classroom: The Effects of Quiz Frequency and Placement. *Journal of Educational Psychology* 103, 2 (November 2011), 399–414. https://doi.org/10.1145/161468.161469

[2] Norman Tiong Seng Lee Oka Kurniawan and Christopher M. Poskitt. 2020. Securing Bring-Your-Own-Device (BYOD) Programming Exams. In *51st ACM Technical Symposium on Computer Science Education (SIGCSE '20)*. ACM SigCSE, 880–886.

[3] prairielearn 2022. *PrairieLearn*. Retrieved Jan 2, 2023 from https://www.prairielearn.org/