The CS1 Reviewer App: Choose Your Own Adventure or Choose for Me!

Anshul Shah
anshul.shah@duke.edu
Duke University
Durham, North Carolina, USA

Jonathan Liu
liujon23@gmail.com
Duke University
Durham, North Carolina, USA

Kristin Stephens-Martinez
ksm@cs.duke.edu
Duke University
Durham, North Carolina, USA

Susan H. Rodger
rodger@cs.duke.edu
Duke University
Durham, North Carolina, USA

ABSTRACT

We present the CS1 Reviewer App - an online tool for an introductory Python course that allows students to solve customized problem sets on many concepts in the course. Currently, the app's questions focus on code tracing by presenting a block of Python code and asking students to predict the output of the code. The tool tracks a student's response history to maintain a "mastery level" that represents a student's knowledge of a concept. We also provide an option of answering auto-generated quizzes based on the student's mastery across concepts. As a result, the tool provides students a choice between creating their own learning experience or leveraging our question selection algorithm. The app is supported on traditional webpages and mobile devices, providing a convenient way for students to study a variety of concepts. Students in the CS1 course at Duke University used this tool during the Spring and Fall 2020 semesters. In this paper, we explore trends in usage, feedback and suggestions from students, and avenues of future work based on student experiences.

CCS CONCEPTS

• **Applied computing** → **Computer-assisted instruction**; **Interactive learning environments**; **E-learning**.

KEYWORDS

CS1; intelligent tutoring system; auto-generated questions; auto-generated quizzes; knowledge tracing; adaptive learning

ACM Reference Format:

Anshul Shah, Jonathan Liu, Kristin Stephens-Martinez, and Susan H. Rodger. 2021. The CS1 Reviewer App: Choose Your Own Adventure or Choose for Me!. In *26th ACM Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2021), June 26–July 1, 2021, Virtual Event, Germany*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3430665.3456333

ITiCSE 2021, June 26–July 1, 2021, Virtual Event, Germany

ACM ISBN 978-1-4503-8214-4/21/06…$15.00
https://doi.org/10.1145/3430665.3456333

1 INTRODUCTION

The average number of students majoring in computer science per university more than tripled between 2006 and 2015 [15]. Larger class sizes make it difficult for instructors to identify and diagnose the pockets of misunderstandings that may exist among students. We developed the CS1 Reviewer App [3] to address this issue by gathering data on student responses to multiple-choice questions and estimating student knowledge. There is a public-facing version of the website that is available for readers to use.

The CS1 Reviewer App allows students to create custom problem sets from multiple topics and with any number of questions. Novice programmers rate code tracing [26], or predicting the expected output of running a block of code, as one of the most helpful learning strategies [14]. To help students master code tracing, our tool presents Python code and asks students to identify the code's output or determine if an error exists in the code.

The tool provides individualized problem sets based on a student's mastery across concepts, which is a key component of intelligent tutoring systems (ITSs) [21]. An analysis by Akyuz [1] on the role of ITSs in providing personalized learning found that they can effectively deliver material to address individual needs. A study investigating the impacts of a personalized learning experience by Hwang et al. [8] found that a personalized game improves motivation to play whereas a "one-size-fits-all" approach only engages students for a limited time. Similarly, personalized learning scenarios have been linked to an improvement in retention [11].

Our goals in developing the CS1 Reviewer App are:

- Provide quality questions that challenge students to understand concepts that span the CS1 curriculum.
- Identify concepts that a student is struggling with by using response history to trace student mastery across concepts.

2 RELATED WORK

We divided our related work into two groups that share similarities with our Reviewer App. First are tools that emphasize code tracing skills and visualize program execution. Second are ITSs, which provide individualized learning experiences.

2.1 Online Textbooks

Online textbook platforms provide content to learn a computer science topic and include built-in quizzes with automatic feedback on correctness. Examples of such platforms include Runestone Academy [23], Zybooks [27], and OpenDSA [5; 6; 19]. For example, our university's CS1 course uses an online Python programming textbook on Runestone Academy that includes intermittent quizzes [9]. The quizzes in these online textbooks are typically fixed questions, so everyone gets the same questions. If one goes back to review a section, the quizzes are the same and they may be marked as already completed with the correct answers still displayed. In contrast, the breadth and depth of the Reviewer App's content ensures that students will almost never see the same question twice.

2.2 Program Visualization Tools

Another kind of tool visualizes code execution to help students read and understand code, which is also the CS1 Reviewer App's goal. Projects such as Kumar's animations for Java and C++ [12; 13], Guo's Python Tutor visualizations [7], and Nelson's PLTutor for code reading skills [16] include visual representations of program execution and step-by-step explanations after each command. Typically, the questions on these tools include fixed concepts, such as pointers in Kumar's C++ tool [12] or do not allow the same customization of concepts as in the CS1 Reviewer App. For example, Python Tutor [7] requires users to input their own code to see the visualization of its execution rather than providing code for students to interpret.

While the tools mentioned above provide students practice with code evaluation, they do not provide the flexibility and specificity in our tool. Our tool comes with thousands of randomly-generated sample programs categorized by topic to promote exposure to a variety of questions. We also choose not to focus on code visualization in our tool because the CS1 instructors at our university make students aware of Guo's Python Tutor website [7] early in the semester. Instead, we prioritize assessing students to identify the concepts each student struggles with.

2.3 Intelligent Tutoring Systems (ITS)

Many online tutors also exist as ITSs that can provide individualized learning experiences. Nesbit et al. [17] define an ITS as a tool that "performs teaching or tutoring functions (e.g., selecting assignments, asking questions, giving hints, evaluating responses, providing feedback, prompting reflection, providing comments that boost student interest) and adapts or personalizes those functions by modeling students' cognitive, motivational or emotional states".

Research on ITSs has increased tremendously over the past decade [17]. Nesbit's analysis [17] confirms that ITSs result in a statistically significant positive impact on learning outcomes for students compared to non-ITS computer-based instruction across most domains and all education levels (elementary, secondary, post-secondary). Soh's [24] experiments show that intelligent tutoring systems not only improve student learning outcomes but also become more effective at tutoring students over time as it accumulates student data.

The CS1 Reviewer App functions as an ITS because the site models student mastery of a concept and uses such modelling to automatically generate quizzes for students.

3 QUESTION GENERATION

3.1 Question Set

The quality of our tool depends on the quantity, variety, and difficulty of the questions. Currently, there are 2,049 questions available to students. These questions are grouped by higher-order *concepts*, and then categorized into more specific *topics*. Each question falls under only one topic; we never categorize a question into two different topics. In total, there are 9 concepts and 22 topics available to answer on the tool, allowing a great diversity of question combinations on a given quiz. Figure 1 shows all the concepts and topics available to the user. For example, the concept "Sorting" has the topics "Sort by Key", "Tuples", and "Basic Sort".

Over the past year, we added questions to topics that are historically more popular among students. The five question topics that have the most questions are Dictionary-Items, Lists-Pointers, Sorting-Tuples, Tuple-Reassignment, and Other-While Loops. Topics within the Math Operators concept have about half the number of questions as more popular topics because students do not practice Math Operator questions often and the simplicity of the concept does not allow for much question diversity.

3.2 Question Templates

We wrote question templates in Python to generate many variations of the same question. Almost all are code-tracing questions that ask what Python will print or the final value of a variable. The wrong answers are mainly inspired by Stephens-Martinez's work [26], which inspected the wrong answers students submitted to open-text response, code-tracing questions. This work identified common ways students misunderstand how Python works.

The questions have a question body, one correct answer, and three wrong answers. We generate variations of the same question using two randomization methods. One variation alters the question body by changing variable values, code line order, or boolean statements (e.g., Using > versus <=). The other variation is with the wrong answer options. If more than three potential wrong answer options are available, the code randomly chooses a subset.

4 KEY FEATURES

4.1 Custom Quizzes

Figure 1 shows the student view on opening the website. On this page, students can select the concepts and topics to appear on the quiz. The settings in Figure 1 show that the student has selected "Lists-Pointers" and "Sorting-Tuples." Students can include as many topics as they like.

The practice of combining a variety of concepts and question types, known as *interleaving*, has empirically shown to be beneficial in student retention and learning [10; 20]. Specifically, Rorher [20] finds that interleaved quizzes help students distinguish between similar topics and Dunlosky et al. [4] found that interleaving could improve long-term memory retention and promote organizational processing when approaching a question. Our tool promotes interleaving by allowing a mix of concepts on quizzes. Moreover, the questions within a topic could cover multiple subtopics (e.g. sorting topics could include ties between objects or not) so interleaving is often present regardless.

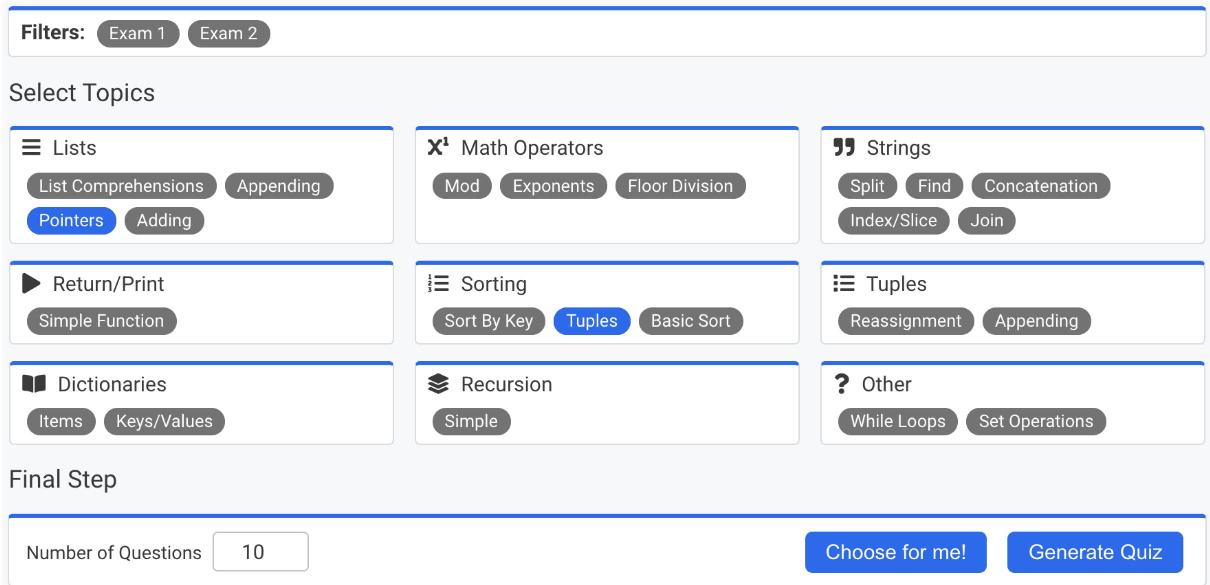

Figure 1: Student view of the CS1 Reviewer App

Students can also select the total number of questions on the quiz. The tool aims to create the most evenly distributed distribution of questions per concept. In Figure 1, a student has selected to answer 10 questions, which is the default. When the student presses the “Generate Quiz” button, a quiz will be generated with five questions on Lists-Pointers and five questions on Sorting-Tuples.

Finally, students can also use filters to list concepts for a specific use case. For example, when a student selects the filter for “Exam 1,” only topics covered in Exam 1 can be selected.

Figure 2 shows the student view of a quiz that a student generated with three questions. Question 1 is an example of a student identifying an error in the code, whereas questions 2 and 3 ask the student to respond with the code’s output. Note that all the questions in a quiz are shown at once rather than one at a time, so students must give a response to each question before receiving feedback on the quiz.

4.2 Mastery Score

For each topic a student encounters, the tool tracks the student’s response history and estimates the mastery level for that topic.

Figure 3 shows the student dashboard. Note that the mastery level is grouped into Beginner, Intermediate, Advanced, and Master. Students can see their “Mastery at a Glance” at the top, which shows one square for each topic. When a student hovers over a square, the app displays the topic that the square represents. Students can also scroll through the page to see a more detailed display of mastery per topic.

The calculation of mastery is currently based on a weighted average of a student’s most recent quiz and the percentage of correct answers on the previous questions. We chose the weights after looking at how mastery scores changed over time based on a student’s quiz correctness and selecting reasonable values to represent knowledge gain. In the future, we hope to conduct experiments to identify more empirical ways to estimate mastery. Equation (1) shows the current calculation for a student’s mastery level.

$$\mathbf{M} = 0.6 \cdot Q_n + 0.4 \cdot Q_{1\ldots n-1} \quad (1)$$

In the equation above, M represents the mastery level for a student on a specific topic, Q_n represents a student’s most recent quiz score on the topic, and $Q_{1\ldots n-1}$ represents the percent of correct answers on all previous quizzes for the student-topic pair. Students are not assigned a mastery level for a topic until they have answered at least five questions from the topic. A gray square under “Mastery at a Glance” in Figure 3 indicates that a student has not answered enough questions to receive a mastery score.

4.3 Auto-generated Quizzes

Rather than choosing a quiz’s topics by hand, the student can use the "Choose for Me!" button. The Reviewer App then picks topics for the student, prioritizing topics with a low mastery score. It also focuses on selecting questions from a variety of topics in order to prevent the quiz from being repetitive or discouragingly challenging.

The tool first generates a distribution where each topic’s weight is inversely proportional to its mastery level. Each topic’s weight is also lower-bounded by 0.1 to allow students to occasionally get review questions from mastered topics. This ensures that the distribution is not entirely dominated by difficult questions. To summarize, the topics are weighted with Equation (2), where t indicates a question topic.

$$\mathbf{Weight}_t = \max(0.1, 1 - \mathbf{Mastery}_t) \quad (2)$$

Topic weights may also be manually overridden by the course instructor. For example, all “Math Operations” questions were given lowered weights near the course’s final exam because they were less relevant to the exam.

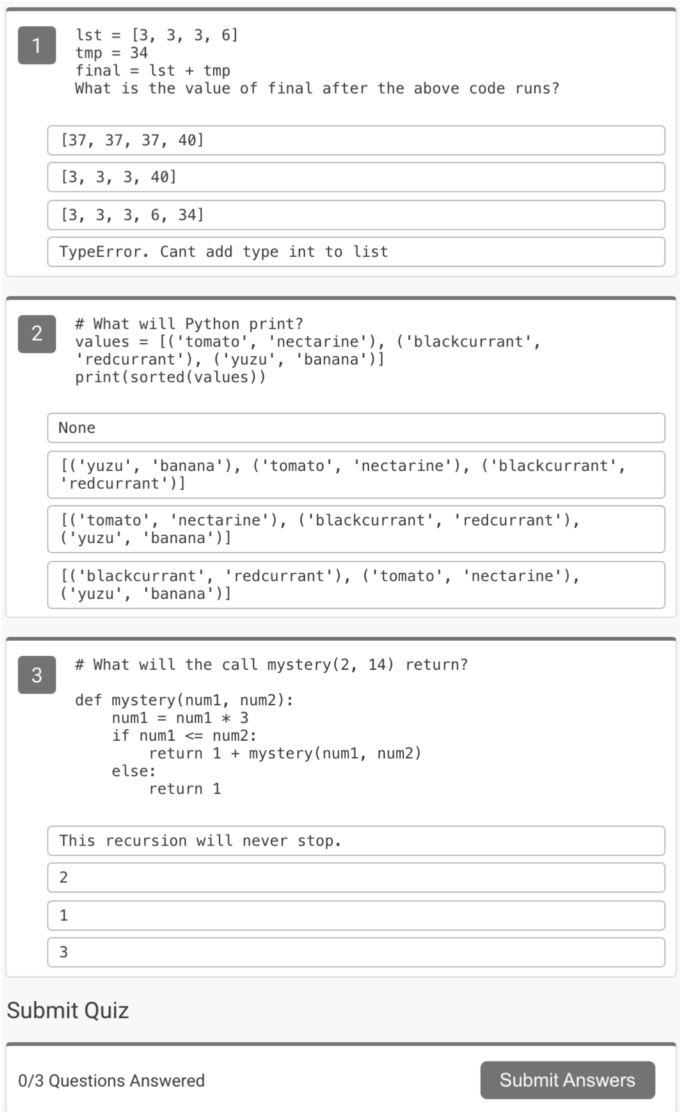

Figure 2: Sample quiz view

Once these adjustments are made, the tool takes an independent sample from this distribution for each question, and generates the quiz accordingly. The quiz length is still specified by the student.

4.4 Correctness Feedback

Our tool provides results to students upon completion of a quiz. Figure 4 shows the view after submitting a quiz. The quiz had a total of five questions - two on List Comprehensions and three on While Loops. We display the total percentage correct (80% in this case), as well as a breakdown of the performance per topic to provide a more specific evaluation of student skills. In this case, the student incorrectly answered question one, which was about list comprehensions. An incorrect student response is highlighted blue, and the correct answer is then colored green. Note that we currently do not have any written feedback for answers, only whether the student

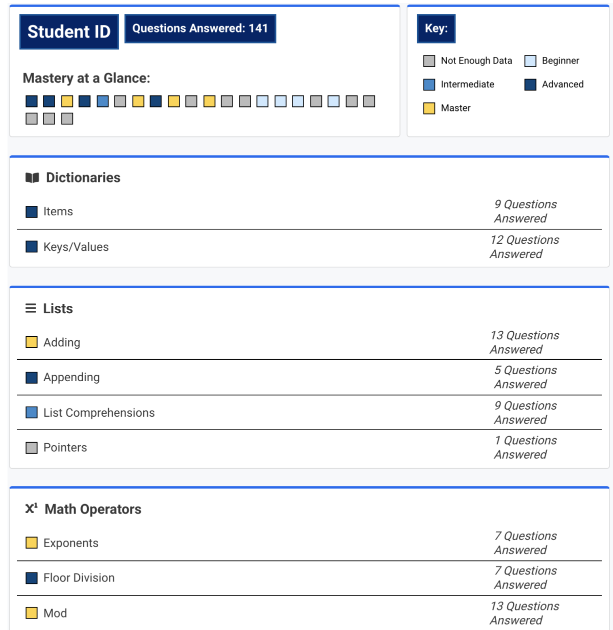

Figure 3: Student dashboard of mastery

got it right or wrong. The student answered the remaining four questions correctly. When a student correctly answers a question, the question box and correct answer are colored green, with a check mark appearing next to the correct answer. Question 2 in Figure 4 is an example of how correct answers are displayed.

We also provide students with easy options for taking a new quiz from the quiz results page. They can either select the “Use Same Settings” button, which creates a new quiz with the same parameters, or the "Choose New Settings" button, which takes the user back to the display in Figure 1.

5 STUDENT USAGE

In the Fall 2020 semester, 188 students enrolled in the CS1 course at Duke University. Of these, 152 students (80.8%) consented to having their classroom data used for research. Due to COVID-19, our university mandated [18] that large classes such as CS1 should be conducted entirely online, with students attending labs, lectures, and office hours completely virtually .

Students were introduced to the tool during week three of the class. We present the Reviewer App to students in three different ways from week to week:

- *An optional supplement*
  In weeks 4, 6, and 15, we reminded students that the Reviewer App is an optional study tool to reinforce the concepts covered by the lab and lecture that week. Specifically, in week 4 we first introduced students to the app as a supplemental tool. During exam weeks, we told students that the app includes questions related to topics on the upcoming exam.
- *A required activity*
  In weeks 5, 10, and 13, students were required to log into the Reviewer App and complete 10 questions on a fixed set

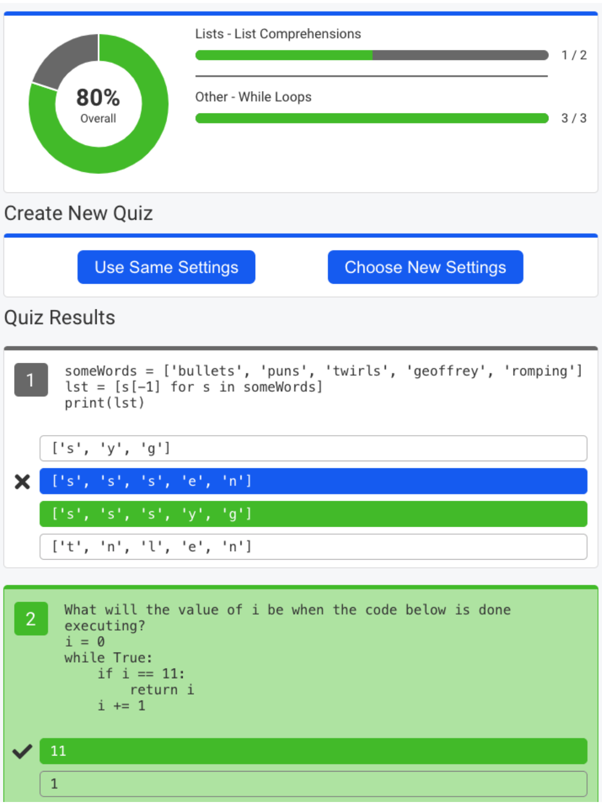

Figure 4: Sample results view

of topics prior to the start of lab. We awarded students with one point towards their lab grade for completing the quiz.

- *No mention of the Reviewer App*
  In the remaining weeks, students were neither reminded of nor required to complete any questions from the Reviewer App. Therefore, we observe much lower usage in these weeks, but students who used the application did so without any prodding from course staff.

In total, students answered 12,672 questions during the 15-week Fall 2020 semester. On average, each student answered 87.3 questions throughout the semester. Figure 5 displays the pattern of usage over time. Note that usage spikes when an exam is upcoming and dips in between periods of preparing for exams.

Students created 1,265 quizzes throughout the semester with varying parameters. The mean, median, and mode of the lengths of quizzes were all 10 questions, likely due to the default setting. The standard deviation of the quiz lengths was 6 questions, indicating a high amount of variability among quiz lengths.

A total of 739 out of 1,265 quizzes (58.4%) covered more than one high-level concept (Sorting, Recursion, Lists, Strings, etc). On average, students included 2.9 unique concepts per quiz. This indicates a high level of interleaving among the top-order concepts, but a significant amount of interleaving within the more specific topics was also present. A total of 988 of the 1,265 quizzes (78.1%) combined multiple topics into one quiz, and the average number of topics on all quizzes was 6.1. The student feedback confirms the popularity of the ability to interleave as students were receptive to the idea of customizing the topics they encountered.

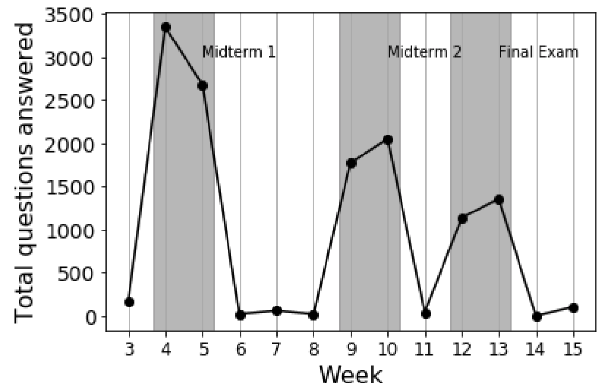

Figure 5: Number of questions answered per week

Table 1: Five most answered concepts

Top 5 Concepts	
Concept	Number of Questions
Strings	3,246
Sorting	2,503
Lists	1,590
Math Operators	1,267
Other (Sets and While Loops)	1,160

Table 2: Five most answered topics

Top 5 Topics	
Topic	Number of Questions
Sorting-Basic Sort	1,103
Sorting-Sort by key	1,098
Strings-Join	913
Sorting- Tuples	869
Strings-Indexing/Slicing	868

Finally, several concepts stood out as more popular studying material. As Table 1 indicates, students answered questions on Strings and Sorting more than any other concepts, likely because these concepts are large components of Exam 1 and Exam 2 material.

We added the “Choose for me!” button in week 13 of the semester, after the second exam. As a result, students could only use the button for the final exam, but they still answered 1,359 questions from 137 auto-generated quizzes. These values represent 10.7% of all questions and 10.9% of all quizzes overall. The most popular question concepts automatically selected through auto-generated quizzes were Dictionaries, Tuples, and Lists. Since the algorithm for selecting question topics allows a higher probability for a topic that a student has not mastered, these concepts represent the questions that were more difficult for students ahead of the final exam.

6 STUDENT FEEDBACK

In week 13 of the course, we asked students for feedback on the Reviewer App. We gathered 160 responses offering comments on the tool and suggestions for improvements.

6.1 Strengths

The three most common strengths that students mention are the filters to group concepts according to lab section or exam, custom quizzes to guide more focused studying, and the ability to auto-generate quizzes. Students seem to be receptive to both the ability to create their own quizzes and our tool's ability to generate personalized quizzes. Students report that the tool serves as effective preparation for exams, and helps them pinpoint topics to practice.

We hope to build upon these strengths by improving our knowledge tracing algorithm to create more effective auto-generated quizzes and adding more question topics to the website.

6.2 Weaknesses

Among the tool's major weaknesses are the relative ease of some questions and repetitive question templates within some topics.

Based on student responses, many questions are simple to solve. For example, the group of Math Operator questions has a high correctness rate of around 95 percent. However, we are constantly adding questions to the tool and will seek to increase the difficulty of these additional problems.

Finally, within a topic (e.g. Adding Lists Together), the questions indeed have a similar format with only slight changes to operators and variables. As a result, students feel that many questions are similar because the general problem-solving pattern is nearly identical between questions within the same topic.

6.3 Suggestions

By far, the most popular suggestion among students is to present an explanation of the correct answer when a student gets a question wrong. Currently, our tool only *identifies* the correct answer. While the effectiveness of delivering hints or explanations is difficult to measure, we hope to utilize the same method for auto-generating hints that Stephens-Martinez used with another automatic quiz tool [25]. By tracing incorrect answers, we can generate tags to trigger corresponding explanations to provide accurate feedback to students.

7 FUTURE WORK

We plan to incorporate several features to the CS1 Reviewer App to help execute the goals of the tool outlined in the introduction.

To improve the tool's ability to serve as an effective study tool with challenging questions, we are constantly adding new questions. We also aim for these questions to add to the amount of topics available, so students have more options when creating quizzes.

We also hope to amend the format in which questions are delivered. As depicted in Figure 2, we show all the questions in a quiz at once. In the future, we hope to create a flashcard-style quiz template, where students are shown one question at a time and see each question's result before the next question.

Using this flashcard-style template, we can implement an endless, adaptive quiz, where the results of previous questions dictate the next question. Recent work has shown that adaptive, multiple-choice questions show slight improvements in scores among students and can increase motivation and engagement [22]. Allowing students to answer unlimited questions in this one-by-one manner can serve as an effective study tool for students since the Reviewer App will prioritize topics that are more difficult for a student.

The flashcard-style quiz will also enable a more effective knowledge tracing algorithm to be implemented. Specifically, the Bayesian Knowledge Tracing model assumes that students see their results for a question before answering the next question [2]. Changing the quiz format will allow us to use such a learner model to estimate mastery and select the next question in our adaptive quiz.

8 CONCLUSION

We built the CS1 Reviewer App, an intelligent tutoring system designed for introductory computer science courses. The tool emphasizes code tracing skills and allows students to create customized quizzes that assess multiple concepts. The auto-generated questions on the app are grouped into 9 concepts and 22 topics that span the CS1 curriculum at our university. We also provide students an option of answering auto-generated quizzes that are personalized for students and take into account students' concept mastery.

The tool has been available for two semesters and was integrated into the CS1 course as a required activity in Fall 2020. During the semester, 152 students answered 12,672 questions, with 10.7% of all questions provided to students through the auto-generated quizzing feature. We also observe that in 78.1% of all quizzes, students interleaved multiple topics within a single quiz.

Feedback from students indicates they are highly receptive to the flexibility of customizing quizzes for a personalized learning experience and the auto-generated quizzes that target difficult concepts. Students generally use the tool to prepare for exams since we see a spike in the number of questions answered on the days before an exam. Students also offered suggestions to improve the tool, such as incorporating hints or explanations for correct answers. We hope to continue developing the CS1 Reviewer App to further meet the goals of the tool.

9 ACKNOWLEDGEMENTS

We would like to thank Alex Bendeck, Varun Nukala, Richard Chen, and Will Herbst for contributing to the initial codebase. We also want to thank Nitya Raviprakash, Ben Stewart, Eric Young, and Frank Tang for improving the front-end design and Joe Shamblin for his assistance in setting up the website. Finally, this work was funded in part by NSF Grant 1934965.

REFERENCES

[1] Akyuz, Y. Effects of intelligent tutoring systems (its) on personalized learning (pl). *Creative Education 11* (01 2020), 953–978.

[2] Corbett, A., and Anderson, J. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction 4* (2005), 253–278.

[3] CS reviewer app. https://cs-reviewer.cs.duke.edu/, 2021.

[4] Dunlosky, J., Rawson, K., Marsh, E., Nathan, M., and Willingham, D. Improving students' learning with effective learning techniques. *Psychological Science in the Public Interest 14* (2013), 4 – 58.

[5] Fouh, E., Breakiron, D., Hamouda, S., Farghally, M., and Shaffer, C. Exploring students learning behavior with an interactive etextbook in computer science courses. *Computers in Human Behavior* (December 2014), 478–485.

[6] Fouh, E., Karavirta, V., Breakiron, D. A., Hamouda, S., Hall, S., Naps, T. L., and Shaffer, C. A. Design and Architecture of an Interactive ETextbook–The OpenDSA System. *Science of Computer Programming 88* (2014), 22–40.
[7] Guo, P. J. Online python tutor: Embeddable web-based program visualization for cs education. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (New York, NY, USA, 2013), SIGCSE '13, ACM, pp. 579–584.
[8] Hwang, G.-J., Sung, H.-Y., Hung, C.-M., Huang, I., and Tsai, C.-C. Development of a personalized educational computer game based on students' learning styles. *Educational Technology Research and Development 60* (08 2012).
[9] Jeffrey Elkner, Allen B. Downey, C. M. B. M., and Ranum, D. How to think like a computer scientist: Interactive edition, 2021.
[10] Kang, S. H. K. The benefits of interleaved practice for learning.
[11] Kozierkiewicz-Hetmańska, A. Evaluating the effectiveness of intelligent tutoring system offering personalized learning scenario. In *Intelligent Information and Database Systems* (Berlin, Heidelberg, 2012), J.-S. Pan, S.-M. Chen, and N. T. Nguyen, Eds., Springer Berlin Heidelberg, pp. 310–319.
[12] Kumar, A. Data space animation for learning the semantics of c++ pointers. vol. 41, pp. 499–503.
[13] Kumar, A. N. Explanation of step-by-step execution as feedback for problems on program analysis, and its generation in model-based problem-solving tutors. *Technology, Instruction, Cognition and Learning (TICL) Journal 4*, 1 (2006).
[14] Lahtinen, E., Ala-Mutka, K., and Järvinen, H.-M. A study of the difficulties of novice programmers. vol. 37, pp. 14–18.
[15] National Academies of Sciences, E., and Medicine. *Assessing and Responding to the Growth of Computer Science Undergraduate Enrollments*. The National Academies Press, Washington, DC, 2018.
[16] Nelson, G. L., Xie, B., and Ko, A. J. Comprehension first: Evaluating a novel pedagogy and tutoring system for program tracing in cs1. In *Proceedings of the 2017 ACM Conference on International Computing Education Research* (New York, NY, USA, 2017), ICER '17, ACM, pp. 2–11.
[17] Nesbit, J. C., Adesope, O. O., Liu, Q., and Ma, W. How effective are intelligent tutoring systems in computer science education? In *2014 IEEE 14th International Conference on Advanced Learning Technologies* (2014), pp. 99–103.
[18] Office of the Provost. Covid-19 response from trinity college of arts amp; sciences, Sep 2020.
[19] https://opendsa.cs.vt.edu/, 2021.
[20] Rohrer, D. Interleaving helps students distinguish among similar concepts. *Educational Psychology Review 24* (09 2012), 355–367.
[21] Rosic, M., Glavinic, V., and Stankov, S. *Intelligent Tutoring Systems for the New Learning Infrastructure.* 01 2006.
[22] Ross, B., Chase, A., Robbie, D., Oates, G., and Absalom, Y. Adaptive quizzes to increase motivation, engagement and learning outcomes in a first year accounting unit. *International Journal of Educational Technology in Higher Education 15* (2018), 1–14.
[23] https://runestone.academy, 2021.
[24] Soh, L.-K. Incorporating an intelligent tutoring system into cs1. In *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education* (New York, NY, USA, 2006), SIGCSE '06, Association for Computing Machinery, p. 486–490.
[25] Stephens-Martinez, K., and Fox, A. Giving hints is complicated: Understanding the challenges of an automated hint system based on frequent wrong answers. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education* (New York, NY, USA, 2018), ITiCSE 2018, Association for Computing Machinery, p. 45–50.
[26] Stephens-Martinez, K., Ju, A., Parashar, K., Ongowarsito, R., Jain, N., Venkat, S., and Fox, A. Taking advantage of scale by analyzing frequent constructed-response, code tracing wrong answers. In *Proceedings of the 2017 ACM Conference on International Computing Education Research* (New York, NY, USA, 2017), ICER '17, Association for Computing Machinery, p. 56–64.
[27] https://www.zybooks.com/, 2021.