10th International Conference on Higher Education Advances (HEAd'24)
Universitat Politècnica de València, València, 2024
DOI: https://doi.org/10.4995/HEAd24.2024.17104

The impact of using formative assessment in introductory programming on teaching and learning

Jagadeeswaran Thangaraj[1], **Monica Ward**[1], **Fiona O'Riordan**[2]
[1]School of Computing, Dublin City University, Ireland, [2]CCT College, Dublin, Ireland.

How to cite: Thangaraj, J.; Ward M.; O'Riordan F. 2024. The impact of using formative assessment in introductory programming on teaching and learning. In: 10th International Conference on Higher Education Advances (HEAd'24). Valencia, 18-21 June 2024.

Abstract
This study offers an efficient formative assessment strategy that may be used to encourage learning and improve students programming comprehension. In addition to traditional teaching sessions, the strategy offers a series of formative assessment quizzes on fundamental programming, utilising an adaptive model for program comprehension as a learning technique. A non-adaptive and an adaptive assessment, both based on multiple-choice questions, are provided to students enrolled in the 2023 introductory programming course. This study investigated how effectively these assessments assisted students in understanding, learning, and developing a sense of their proficiency in computer programming by introducing common programming errors. We gathered data from a survey that 218 students completed at the end of each quiz. Findings from student surveys and observational techniques show that employing adaptive technique was more likely to motivate students and increase their self-confidence. The results also show that formative assessment can be used to support learning programming in addition to classroom instruction to help students grasp key concepts.

Keywords: *Assessment and feedback; Computer programming; Formative assessment; Introductory programming; Learning method; Novice students.*

1. Introduction

A crucial component of learning is assessment, and a good assessment can make sure that students profit from and like taking tests. A programming assessment will, from the viewpoint of the instructor, promote the acquisition and implementation of all required programming skills. However, some students may view the same assignment as requiring them to write code that provides the "correct" response. Students must simultaneously learn more frequent errors and approaches to solve them. Formative assessment is a useful methodology for enhancing learning outcomes and providing learning motivation (Louhab et al., 2018). The student's program submissions are evaluated and delivered immediate feedback by an automated formative assessment system or manual assessor. Two characteristics were deemed crucial while intending to develop formative assessments. For error messages can be properly understood, feedback needs to be quick and detailed (Louhab et al., 2018) . Second, students must be given the chance to discover their mistakes after making several attempts on different questions. Due to the nature of these elements, it is necessary to create a quiz large enough so that students may repeat assessments without encountering the same questions twice. This study presents how effectively these assessments assisted students in understanding, learning, and developing a sense of their proficiency in computer programming.

This study investigates the formative assessment for introductory programming in higher education and proposes a framework for its customization and enhancement to fulfill this purpose. Our goal with this framework is to help novices who want to develop a variety of advanced applied programming skills by offering them helpful feedback and resources. While considering the limitations of automatic assessment, this framework emphasizes the importance of achieving comparable difficulty for questions and maximizes the potential for randomization for exercise tasks. This helps to create meaningful feedback for students and supports their learning process.

2. Development of formative assessment for introductory programming

A formative assessment tool might help students feel less anxious about giving incorrect answers by providing feedback on how to improve their work. An answer key and a few choices make up the two components of a multiple-choice question (MCQ) (Henriques Abreu et al., 2018). A query or a remark is typically made by the stem. A stem states the questions with a few potential solutions, including a key that provides the best response, and a few distractions that offer logical but incorrect responses. To proceed with the stem, the learner must choose the best or most accurate option. MCQs are insufficient for evaluating a student’s coding proficiency in programming modules since they do not encourage learners to write their own code, even when they are at an advanced level. The ability to retain programming concepts and increase engagement, however, can be useful. This study has led us to create MCQ quizzes that are an excellent method to increase student engagement and help them remember the programming content (Ross et al., 2018). We have created formative assessment quizzes to introduce common programming errors. We have a list of questions with various answers in these quizzes. As they offer feedback for each selection, these quizzes assist in fostering their learning. While the wrong answer feedback helps them locate the appropriate response, the right answer feedback acknowledges their responses. Students can learn from their incorrect responses and determine the correct response. As a result, it is a system that progresses and aids in their ability to learn from mistakes. This study examines if the formative assessment increases participants’ confidence in their capacity to understand the fundamental ideas behind programming. They assist students in comprehending the common code errors they make as well as compiler error messages.

2.1. Quiz implementation

'Google Forms' is utilized to implement the quizzes as it can be an effective tool for formative assessment and for promoting active learning (Djenno et al., 2015). Each quiz aims to educate students about common code errors they made when studying the assigned topics (Ahadi et al., 2018). Feedback will be given to students for each potential response, which will help them better comprehend the errors and help them to understand easily. Feedback is customized messages that are like enhanced error messages (Becker et al., 2016). Every incorrect reaction offers advice on how to respond. Students can make the best alternative based on the feedback. These assessments are presently being examined in two distinct models: non-adaptive and adaptive.

2.1.1. Non-adaptive model

In a non-adaptive model, students receive feedback for each response regardless of whether it is correct or incorrect. This feedback allows students to learn from their mistakes and help them comprehend what the correct response is. However, they cannot return to correctly answering the same or similar questions.

2.1.2. Adaptive model

With adaptive model, the questions are redirected to match the student's present proficiency level and ongoing advancement (Ross et al., 2018). Repetition of questions until the answer is correct or the level of knowledge is reached is known as adaptive assessment. The knowledge level of the learners increased by varying the order of the assessment questions in adaptive approach (Heitmann et al., 2018). Difficulty levels have been added to learning objects in the next model. These things could be topics, questions, and a variety of errors. The goals relate to questions with varying degrees of difficulty. Difficulties in programming are classified as Bloom's taxonomy of programming (Thompson et al., 2008). In this model, we classified a list of questions in three cognitive levels based on the complexity (like easy, moderate, and difficult) (Louhab et al., 2018). Here easy questions assess the basic concepts, moderate questions assess comprehensive knowledge and difficult questions do the applications of the knowledge (Vie et al., 2017). If a student successfully responds to a moderate question on this assessment, the subsequent question is hard. If not, the easy questions will be. It goes on until the system forecasts the competency level of the students (Simon-Campbell & Phelan, 2018). A sample classification of a question is described as Table 1.

Table 1. Summary of ‘print’ statement question in adaptive model

Difficulty low	**Difficulty moderate**	**Difficulty high**	**Summary**

var = ’Amazon’
print(var[4])
	var = ’computer’
	print(var[5 :: 1])
		var = ’Ireland’
		print(var[4 :: -1])
			var = ’James Bond’
			print(var)
			print(var[3])
			print(var[5 :: 1])
			print(var[5 :: -1])

2.2. Research questions

Using the formative assessment quizzes, this study will particularly investigate the following research questions.

> RQ-1: Does formative assessment help to build self-confidence in novice programmers in learning basic concepts of programming?
>
> RQ-2: Does formative assessment support the ability of novices to understand and correct errors and encourage them to improve their programming skills?
>
> RQ-3: Does formative assessment help novices effectively learn the modular parts of programming concepts?

3. Research method and data collection

3.1. Methodological paradigm

This research is using a mixed method approach (Mertens, 2019). An online survey was used in this study to gather data that was both quantitative and qualitative. This approach works well with a combination of qualitative and quantitative techniques. In this study, we use quantitative intervention to evaluate validity and reliability of formative assessment quizzes. A brief, optional, anonymous survey was employed to gain more insight into how students perceived and experienced formative self-assessment. It also includes qualitative elements and makes use of a range of data sources and data collection methods.

3.2. Research design

To answer the research questions, this research developed a set of quizzes for basic topics of introduction to programming. Due to its convenience and syntactical simplicity, Python is a popular programming language used in introductory programming classes (Johnson et al., 2020). Python is the language used for instruction, and topics covered include variables, operators, conditionals, loops, and a few concepts related to functions. We conducted these quizzes periodically during teaching sessions to build novice's confidence as well as to capture their barriers in programming. At the end of each quiz, we conducted a survey about how it effectively helped them to learn programming. The respondents were questioned about how they felt about formative assessment quizzes of each programming topic. Open-ended questions for qualitative data and closed-ended 'Likert' scale questions for quantitative data were both used in the survey form.

3.3. Data collection strategies

Data collection is the methodical process of gathering information from relevant sources in order to address research questions, test hypotheses, and achieve the project's objectives (Kabir, 2016). Regular quizzes were offered during the study periods as an option. This facilitated students considering what they learned through taking the quizzes. The data includes 218 students' programming quiz attempts that they turned in at the end of each quiz session. In proportion to the number of quiz attempts, some students attempted numerous surveys. The data presented here is both quantitative and qualitative in traits, covering a two-semester span (2022–2023) (n=115, n=103) from each model. Student surveys provide quantitative data. The student questionnaires and their reflective writing assignments provide qualitative data. Every piece of qualitative data is anonymous.

4. Results

4.1. RQ1 – Increasing self-confidence.

This study asked, "Does this quiz increase your self-confidence in learning programming?", at the end of the quizzes to answer RQ-1. The responses ranged from 'Strongly disagree' to 'Strongly agree'. Figure 1 offers a thorough understanding of the students' feelings regarding their level of self-confidence in handling these quizzes. Responses for 'Strongly agree' and 'Agree' were higher than 'Disagree'. Consequently, this study discovered that these formative assessment quizzes helped them increase their self-confidence.

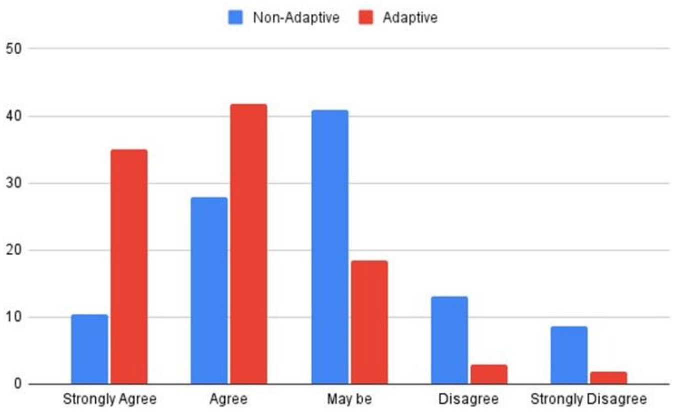

Figure 1. Students' feedback on self-confidence

4.2. RQ2 – Understand and correct the errors.

To answer the RQ-2, it included another 'Likert' question, "Do these quizzes help to understand and correct the errors?". The responses ranged from 'Strongly disagree' to 'Strongly agree'. Using a chart, the outcome is shown in Figure 2. High responses were submitted as 'Strongly agree' and 'Agree'. It highlights how these quizzes make it easier to learn common programming errors. These outcomes show that the self-assessment quizzes aided in their understanding of the frequent errors of programming.

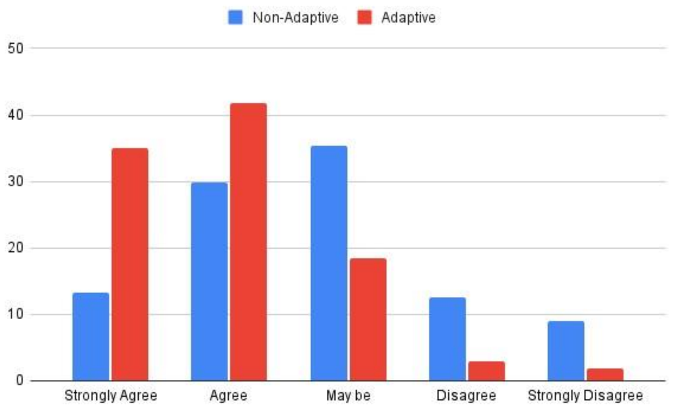

Figure 2. Students' feedback on understanding errors

4.3. RQ3 – Learning tool

To answer the RQ-3, it included another ’Likert’ question, "Do these questions help to understand basic concepts of Python language?", at the end of the quizzes. The responses were ’Yes’, ’May be’ and ’No’. Figure 3 offers a thorough understanding of the students’ feelings regarding these quizzes, help in understanding basic concepts. Overall, there were more responses for ’Yes’ than ’No’. Quantitative data alone does not provide the full picture of the learning experience. Finding out what students think and feel about formative assessment as a computer programming learning activity is critical. They delighted in gaining knowledge by taking quizzes in various models. As stated in the comments below, they also valued these quizzes as a learning tool for various reasons.

> *...It helped to recall...introduced me to new elements of python...made me realize what I didn’t know...was good to refresh my brain...very helpful exercises...I think they are much better than the way the lectures are being taught...Maybe do the quizzes in the lectures to fully understand what is being taught…*

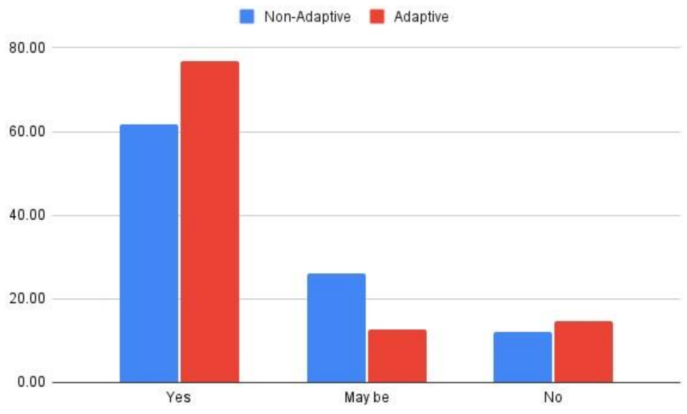

Figure 3. Students’ responses on learning the Python basics.

5. Discussion

We plot the outcome charts in Figure 1, 2 & 3 that indicate the influence on self-confidence, comprehension of errors and supporting learning to analyze the impact of the adaptive technique. Based on the surveys’ responses, it demonstrates that the students firmly believe the quizzes assisted in grasping fundamental programming principles. Figure 1 depicts a comparison of levels of self-confidence. Students’ confidence in learning programming is reportedly much increased by the adaptive model quizzes. The adaptive model was far more helpful than the others as shown in Figure 2. It also demonstrates that adaptive assessments helped them understand and correct errors better than non-adaptive ones. Additionally, students firmly felt that adaptive assessments increased their confidence relative to non-adaptive. Adaptive model maintains its lead in each quiz. In a conclusion, we argue that formative assessment quizzes motivate students to evaluate and learn from their mistakes, which in turn encourages them to learn computer programming. As they can effectively aid in the learning of programming, as a result, learning opportunities have expanded, increasing students' confidence, and understanding the frequent errors. This research demonstrates that adaptive quizzes help engage and motivate novice programming students, thus improving their programming comprehension. Future work involves a further iteration of adaptive model quiz to include closer alignment with the curriculum teaching. We will also concentrate efforts on deeper analysis of qualitative data.

References

Ahadi, A., Lister, R., Lal, S., & Hellas, A. (2018). *Learning programming, syntax errors and institution-specific factors*. https://doi.org/10.1145/3160489.3160490

Becker, B., Glanville, G., Iwashima, R., McDonnell, C., Goslin, K., & Mooney, C. (2016). Effective compiler error message enhancement for novice programming students. *Computer Science Education*, 1-28. https://doi.org/10.1080/08993408.2016.1225464

Djenno, M., Insua, G., & Pho, A. (2015). From paper to pixels: Using Google Forms for collaboration and assessment. *Library Hi Tech News*, *32*, 9-13. https://doi.org/10.1108/LHTN-12-2014-0105

Heitmann, S., Grund, A., Berthold, K., Fries, S., & Roelle, J. (2018). Testing Is More Desirable When It Is Adaptive and Still Desirable When Compared to Note-Taking. *Frontiers in Psychology*, *9*.

Henriques Abreu, P., Silva, D., & Gomes, A. (2018). Multiple-Choice Questions in Programming Courses: Can We Use Them and Are Students Motivated by Them? *ACM Transactions on Computing Education*, *19*, 1-16. https://doi.org/10.1145/3243137

Johnson, F., McQuistin, S., & O'Donnell, J. (2020). *Analysis of Student Misconceptions using Python as an Introductory Programming Language*. https://doi.org/10.1145/3372356.3372360

Kabir, S. M. (2016). Methods of data collection. In (pp. 201-275).

Louhab, F. E., Bahnasse, A., & Talea, M. (2018). Towards an Adaptive Formative Assessment in Context-Aware Mobile Learning. *Procedia Computer Science*, *135*, 441-448. https://doi.org/10.1016/j.procs.2018.08.195

Mertens, D. (2019). *Research and Evaluation in Education and Psychology: Integrating Diversity with Quantitative, Qualitative, and Mixed Methods 5th edition*.

Ross, B., Chase, A.-M., Robbie, D., Oates, G., & Absalom, Y. (2018). Adaptive quizzes to increase motivation, engagement and learning outcomes in a first year accounting unit. *International Journal of Educational Technology in Higher Education*, *15*. https://doi.org/10.1186/s41239-018-0113-2

Simon-Campbell, E. l., & Phelan, J. (2018). Effectiveness of an Adaptive Quizzing System as a Self-Regulated Study Tool to Improve Nursing Students' Learning. *International Journal of Nursing & Clinical Practices*, *5*. https://doi.org/10.15344/2394-4978/2018/290

Thompson, E., Luxton-Reilly, A., Whalley, J., Hu, M., & Robbins, P. (2008). Bloom's taxonomy for CS assessment. *78*, 155-161.

Vie, J.-J., Popineau, F., Bruillard, É., & Bourda, Y. (2017). A Review of Recent Advances in Adaptive Assessment. In (Vol. 94, pp. 113-142). https://doi.org/10.1007/978-3-319-52977-6_4