PAEDAGOGIA
Jurnal Penelitian Pendidikan
http://jurnal.uns.ac.id/paedagogia

DOI : 10.20961/Paedagogia.v29i3.125769
Vol. 29, No. 3, 2026, pp. 346-360
p-ISSN 0126-4109
e-ISSN 2549-6670

Students' Perceived Learning Outcomes from a Web-Based Programming Learning Resource in Resource-Constrained Secondary Schools

Ian L. Tiao[1*], Renald Jay O. Fio[2], Renjun A. Orain[3], Mark Anthony P. Cezar[3], Merlin A. Marfa[3], Pepito R. Guerrero Jr.[3], Milagros V. Ortega[3]

[1] College of Computer Studies, Bestlink College of the Philippines, Quezon City, Metro Manila, Philippines.
[2] Philippine Merchant Marine School, Inc., Las Piñas City, Metro Manila, Philippines.
[3] College of Computer Studies, Far Eastern University Alabang, Muntinlupa City, Metro Manila, Philippines.

Keywords: *Educational Technology, Perceived Learning Outcomes; Programming Education; Student Engagement; Web-Based Learning*

Article history
Received: 12 August 2026
Revised: 4 October 2026
Accepted: 20 October 2026
Published: 31 October 2026

*Corresponding Author Email:
illtiao@bcp.edu.ph

DOI: 10.20961/paedagogia.v29i3.125769

© 2026 The Authors. This open-access article is distributed under a CC BY-SA 4.0 DEED License

Abstract: Programming education still faces teaching challenges in secondary schools that have limited resources. In those resource-constrained schools, students do not have much access to teaching tools and opportunities to practice programming for a long time, which can affect how students learn. This study examined how students believe they have learned when using a web-based programming learning resource in resource-constrained schools. The study focused on students' views of performance and student engagement. The study also examined how students' opinions about the web-based programming learning resource relate to learning outcomes that students think they have achieved. A quantitative correlational research design was employed among Senior High School Information and Communications Technology students from selected resource-constrained secondary schools. Data were collected through a validated, structured questionnaire regarding students' use of the learning resource and were analyzed using descriptive statistics and Pearson Product–Moment Correlation. Students generally reported positive perceptions of academic performance and student engagement. The analysis also showed positive and statistically significant associations between students' evaluation of the learning resource and both dimensions of perceived learning outcomes. These findings suggest that web-based programming learning resources may support classroom instruction by providing students opportunities to practice programming and helping to keep them engaged, especially in settings where resources are limited. The study offers evidence to help teachers, curriculum developers, and researchers consider how to include online learning tools in programming classes. Since the study relied on self-reported data and used an approach, the results show connections and what learners felt they learned, but they don't prove cause-and-effect or that students improved academically.

How to cite: Tiao, I. L., Fio, R. J. O., Orain, R. A., Cezar, M. A. P., Marfa, M. A., Guerrero Jr., P. R., & Ortega, M. V. (2026). Students' Perceived Learning Outcomes from a Web-Based Programming Learning Resource in Resource-Constrained Secondary Schools. *PAEDAGOGIA, 29*(3), 346-360. DOI: 10.20961/paedagogia.v29i3.125769

INTRODUCTION

Programming has become part of high school learning because students are increasingly asked to work in settings that use digital technology. Learning programming does not stop at knowing the names of languages or memorizing commands. Programming also helps students develop thinking, logical reasoning, creativity, and problem-solving skills that can be used outside the programming classroom. These skills matter not only for students who want to study computing but also for learners who will face digital tools in further study, jobs, and daily life. Because of this, programming is linked to aims of digital literacy and learning for the twenty-first century (Grover & Pea 2013; Lye & Koh 2014).

Despite how important programming has become, it is still one of the parts of school for many students. Becoming good at programming is not just about knowing the rules or remembering what to type. Students learn programming by trying things over, testing ideas, fixing mistakes, and thinking about what they did. They build their understanding as they work on problems. So, the best way to teach

programming is to create places where students can keep exploring, get help when needed, and keep practicing all the time (Lye & Koh 2014; Sentence & Csizmadia 2017).

These instructional demands pose difficulties for resource-constrained secondary schools. Access to computer laboratories, instructional resources, reliable internet, and enough classroom time is limited, which stops students from taking part in programming activities. I feel that these constraints show how important it is to find ways for students to learn outside scheduled classes while remaining feasible with the resources already available in schools (Barikzai et al., 2024; UNESCO, 2023).

One instructional approach gaining attention is the use of web-based resources for learning programming. These web-based resources give students access to lessons, coding tasks, and extra learning activities. Web-based resources can help students practice coding outside of the classroom. Web-based resources do not replace teacher-led lessons; instead, they add to classroom teaching. Web-based resources support learning, reinforce concepts that students have already learned, and keep students engaged with coding tasks (Anthony et al., 2022; Martin et al., 2020)

Although educational technology has been studied extensively in programming education, there are few studies that examine how students feel they are learning when using web-based programming tools in schools with limited resources. Current research often talks about using technology, how good the software is, how effective the teaching is, or how good the system is. There is not as much focus on what students think about their learning in real classroom settings. Knowing what students think helps us understand how they experience learning with technology and how these tools can support education in places where learning opportunities are limited (Alghamdi, 2025; Fodouop Kouam, 2025).

This study looks at how students feel about what they learn when they use a programming learning tool in schools that don't have much money or resources. It specifically examines how students perceive their grades and how involved they feel in their learning. It also looks at how students feel about the learning tool and how that connects to what they think they have learned. By looking at what students go through while learning, rather than just how good the software is, the study aims to provide information that helps with teaching programming using technology in education and how teachers teach in schools that don't have much money or resources.

Research Questions

The study was designed to make sure that the goals of the research, the questions in the survey, and the way the data was analyzed were all clearly connected. To do this, the study followed four research questions.

1. What do students think about their academic performance after using the web-based programming learning resource?
2. How do students feel about their level of engagement after using the web-based programming learning resource?
3. Is there a connection between how students rate the web-based programming learning resource and how they see their academic performance?
4. Is there a connection between how students rate the web-based programming learning resource and how engaged they feel in their learning?

These questions shape learning outcomes as having parts. Academic performance looks at how students feel about their growing skills and their ability to finish work that involves programming, while engagement looks at how students focus on feeling curious, getting motivated, and feeling happy while learning. This difference matters because learning to program requires both cognitive work and sustained engagement. A resource may be easy to use but still not push students to keep practicing; on the other hand, an engaging resource may still not be enough if its activities do not help students learn real programming. Looking at both performance and engagement together gives a clearer view of how students experience the resource.

Focusing on schools with limited resources also puts the study into a real-world teaching setting. When usual lab equipment or extra in-person practice is hard to get, a web-based resource can serve as a learning opportunity instead of a full substitute for classroom teaching. This view aligns with research that emphasizes student activity, teacher presence, and effective lesson planning in mixed learning settings (Martin et al., 2020; Rapanta et al., 2020). The web-based resource also shows that technology

alone does not ensure learning. The worth of a web-based resource depends on how the resource's activities are set up, how students interact with the material, and how the web-based resource is woven into the teaching.

Accordingly, the present study does not treat the learning resource as a technological intervention. Instead, it examines how students evaluated their learning experience with the learning resource and whether favorable evaluations were associated with more favorable views of academic performance and engagement. This approach is especially suitable for an investigation in schools with limited resources because it records students' real experiences with learning resources in a genuine classroom setting. The findings can therefore give evidence to teachers and curriculum developers while also pointing out issues that future studies should examine using objective achievement measures or experimental designs.

METHOD

Research Design

A quantitative correlational research design was employed to examine the relationship between students' evaluation of the web-based programming learning resource and their perceived learning outcomes. This design was appropriate because the study sought to investigate naturally occurring relationships among variables without manipulating the instructional environment or introducing an experimental treatment.

Participants

The participants consisted of Senior High School students enrolled in the Information and Communications Technology strand at selected resource-constrained secondary schools. All participants had experienced the use of the web-based programming learning resource during programming instruction and were therefore able to provide informed evaluations of their learning experiences. Purposive sampling was employed to select students who met the inclusion criteria.

Research Instrument

Data were gathered using a questionnaire designed to examine the study's focus. The tool included information about participants' thoughts on the online programming tool and on their learning outcomes, performance in school, and level of involvement. The answers were given on a four-point scale, and a higher number indicated a positive opinion.

Validity and Reliability of the Instrument

The tool used for the research was checked by experts in technology, programming, and research methods. Their thoughts and ideas were considered and used to improve the tool. The tool was tested with people who were not part of the study. The tool's reliability was checked using Cronbach's alpha. The part about learning results had a reliability score.

Data Collection Procedure

Prior to data collection, the necessary permissions were obtained from the appropriate school authorities. Participants were informed of the objectives of the study and the voluntary nature of participation. After providing informed consent, respondents completed the questionnaire after using the web-based programming learning resource. Completed questionnaires were reviewed for completeness and prepared for statistical analysis. Figure 1 presents the sequence of the study from participant selection through instrument validation, data collection, statistical analysis, and interpretation.

Statistical Treatment of Data

Descriptive statistics, including frequencies, percentages, and means, were used to summarize respondents' demographic characteristics and students' perceptions of academic performance and student engagement. Pearson Product–Moment Correlation was employed to determine the relationship between students' evaluation of the web-based programming learning resource and their perceived learning outcomes.

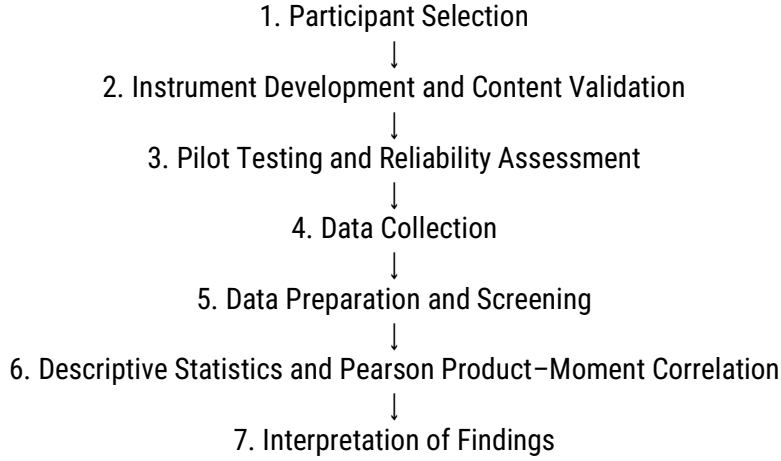


Figure 1. Research Methodology Flowchart

Ethical Considerations

Ethical principles were followed during the study. People taking part were told what the research was about. They chose to take part on their own. They gave their permission before any information was collected. Names or any personal details were not included in the data. Results were shared as a group, not about any one person. This was done to keep the people in the study safe and to protect their information.

RESULTS AND DISCUSSION

The findings focus on students' perceived learning outcomes associated with the use of the web-based programming learning resource. Descriptive statistics in Table 1 summarize students' perceptions of academic performance and student engagement, while Pearson Product–Moment Correlation examines the relationship between students' evaluation of the learning resource and their perceived learning outcomes.

Taken together, the results address the study's four research questions. The descriptive findings indicate generally favorable perceptions of both academic performance and engagement, while the correlation analysis indicates that more favorable evaluations of the learning resource were associated with more favorable perceptions of both outcomes. Because the design is correlational and the measures are self-reported, these results should be understood as evidence about students' perceived learning experiences and associations among variables, rather than as evidence that the resource caused changes in achievement or engagement.

Table 1. Descriptive Statistics for Students' Perceptions of Academic Performance

Perception Indicator	Mean
Demonstrated mastery of fundamental programming concepts	3.80
Demonstrated improvement in programming-related assessments	3.72
Completed practical programming activities according to expected standards	3.78
Completed programming tasks within the expected time frame	3.75
Demonstrated logical reasoning and solution accuracy	3.77
Overall Mean	3.76

Note. Responses were measured using a four-point Likert scale ranging from 1 (Strongly Disagree) to 4 (Strongly Agree). Higher mean scores indicate more positive perceptions of academic performance.

Students gave feedback on every measure of academic performance. The high average scores show that students saw the learning resource as helping them grasp programming concepts, boosting

their confidence when doing programming tasks, and supporting their learning in class. These results show how students feel about their learning experiences, not the academic achievement.

The overall mean of 3.76 stands out because it reflects responses across many aspects of programming performance, not just a single area. The rated part was mastery of fundamental programming concepts with a mean of 3.80. This was followed by completing programming activities according to expected standards, which had a mean of 3.78. Reasoning and solution accuracy came next with a mean of 3.77. Completing programming tasks within the expected time frame had a mean of 3.75. The lowest rating was for improvement in programming-related assessments, at 3.72.

The values are very close to each other, which means there wasn't a difference between how students rated each part. It suggests that students saw the resource as helpful in supporting connected areas of learning. They did not think one aspect was much better or worse than the others. Instead, they felt the support helped them in a way across most parts of programming performance.

The highest mean for mastery of programming concepts may suggest that the resource was especially helpful in reinforcing ideas students learned during instruction. In programming education, basic concepts form the foundation for skills such as problem decomposition, logical reasoning, syntax, and program development. When students can access examples and practice activities outside of class, they have opportunities to go over the material at their own pace. This supports the idea that learning programming involves repeated use and solving problems, not understanding (Lye & Koh 2014; Grover & Pea 2013). Still, this result reflects how students feel about their mastery, not an actual test score or direct measure of their programming ability.

The mean of 3.78 for completing practical programming activities is also important in a resource-constrained context. Programming competence cannot be developed through conceptual explanation alone; students need opportunities to write, inspect, revise, and test programs. A web-based learning resource can extend these opportunities when students have access to a suitable device and an internet connection, allowing practice to continue beyond the limited time available during classroom activities. The result therefore supports the resource's role as a complement to classroom instruction. It does not, however, establish that students completed more activities than they would have without the resources, because the study did not include a comparison group or objective usage records.

The average score for reasoning and solution accuracy was 3.77. This finding is important because programming tasks ask students to turn problems into steps of action and to find mistakes in their work. The feeling that students had support for their reasoning suggests that they saw the activities as requiring them to solve problems actively, not just remember things. This idea aligns with what is known about how programming helps develop thinking, especially in areas such as breaking things down, considering the big picture, and creating steps to solve problems (Grover & Pea 2013; Lye & Koh 2014). Again, the right takeaway is that students felt they improved or got help in this area. The study did not actually check if their ability to think computationally changed.

The lower-yet-still-good average score of 3.72 for improvement in programming-related assessments should not be ignored. It suggests that students liked the resource for learning and practice but were less sure about using that experience on formal tests. This difference can help shape teaching plans. When a resource is mainly used for examples and practice, teachers might need to add activities that mirror the classroom tests in structure and cognitive demands. Practice should then move from guided examples to problem-solving, providing opportunities for formative feedback before students take formal tests. This approach aligns with the ideas of alignment and purposeful learning activity (Biggs & Tang, 2011).

Similarly, the average of 3.75 for finishing tasks within the expected time frame shows that students think efficiency is good, but not perfect. Programming problems often need fixing. Students might take time to figure out why a program doesn't work as it should. If teachers see completion as just a problem, they can use this result to create step-by-step tasks, provide debugging hints, and gradually remove support. In settings like this, such choices can be very helpful because they allow students to use the time given for instruction.

Overall, the performance results show that the resource was seen as helpful across basic knowledge, doing practical activities, thinking logically, finishing tasks, and doing assessments. The

similar average scores make it clear that students viewed the resource as a learning aid, not just for one kind of activity. This aligns with research indicating that learning environments can help students feel they are learning when they experience good interaction, clear structure, and active participation (Kuo et al., 2014; Martin et al., 2020). For this study, though, the strongest idea is that students thought they learned well after using the resource.

Table 2. Descriptive Statistics for Students' Perceptions of Student Engagement

Perception Indicator	Mean
Learning activities were more interesting	3.83
Multimedia elements maintained attention during learning	3.88
Motivated to explore programming topics beyond classroom requirements	3.75
Participated more actively in programming learning activities	3.80
Learning experience was enjoyable and satisfying	3.85
Overall Mean	3.82

Note. Responses were measured using a four-point Likert scale ranging from 1 (Strongly Disagree) to 4 (Strongly Agree). Higher mean scores indicate more positive perceptions of student engagement.

Students likewise reported positive perceptions of engagement shown in Table 2. The consistently favorable ratings indicate that respondents viewed the learning resource as encouraging active participation, sustaining attention during programming activities, and increasing interest in learning programming concepts.

The average level of engagement was 3.82, which was a bit higher than the average level of academic performance, which was 3.76. Even though the difference should not be taken seriously, it shows that students might have felt the resource was especially good at making programming activities more interesting and more engaging. Engagement is a result, in programming education because students are supposed to keep going through trial and error fixing mistakes and revising their work. If students stop paying attention when a program does not work, the chances of building programming skills are lessened. On the other hand, staying focused and wanting to take part can lead to more chances to practice and get help.

Among the engagement indicators, multimedia elements maintaining attention received the highest mean (M = 3.88), followed by the learning experience being enjoyable and satisfying (M = 3.85), learning activities being more interesting (M = 3.83), active participation in programming learning activities (M = 3.80), and motivation to explore programming topics beyond classroom requirements (M = 3.75). The ordering suggests that the resource's immediate presentation and activity characteristics were especially well received. Students appeared to value how content was presented and experienced, while the somewhat lower mean for extending learning beyond classroom requirements suggests that sustained voluntary exploration may require additional instructional supports.

The score of 3.88 for multimedia elements and attention is in line with the principle that well-designed multimedia can aid learning by combining ways to present information while preventing overload (Mayer, 2021; Clark & Mayer, 2016). In programming education, chosen visual explanations, examples, demonstrations, and interactive parts can make abstract steps more concrete. The high score does not mean that more multimedia is automatically better. Rather, this result shows that students found the multimedia elements in this resource useful for keeping students focused. Therefore, learning design should emphasize relevance, clarity, pacing, and alignment between media and the programming concept being taught.

The average score of 3.85 for enjoyment and satisfaction shows that students' experiences were mostly positive. Enjoyment can help students want to keep doing an activity. It should not replace learning. A programming resource should be enjoyable because it makes meaningful practice easier to approach, not because it takes away the challenge of programming. The mix of engagement and good academic performance views in this study is therefore encouraging; students seemed to find the learning activities engaging while also seeing benefits for programming performance.

The average score of 3.75 for exploring programming topics beyond what was required in class was the level of engagement, even though it was still positive. This could mean that the resource helped

students stay engaged while doing their assigned work but didn't do well in encouraging them to learn on their own. Teachers might respond by adding challenges, extra problems, small projects, or questions that get students to use what they learned in ways that matter to them. These kinds of activities can help move engagement from doing what is required to learning on their own. This aligns with the idea that school engagement has components such as what students do, how they feel, and how they think (Fredricks et al., 2004).

The 3.80 mean for active participation is particularly relevant to resource-constrained schools because the instructional value of a web-based resource depends on students' interaction with it. Passive exposure to pages or videos may be insufficient for programming learning. Resources should therefore require students to predict outputs, modify code, solve short problems, diagnose errors, and explain their reasoning. Active learning design can make limited access time more productive by giving each interaction a clear learning purpose.

Overall, the engagement results show that students found the resource interesting, supportive of attention, motivating, engaging, and satisfying. I think these findings strengthen the idea that engagement should be considered alongside perceived performance when evaluating technology-enhanced programming instruction. A resource that students are willing to use and interact with helps build a foundation for repeated programming practice. This is more effective than a resource that's technically available but difficult to navigate or not engaging. Still, the results in this study reflect perceived engagement rather than actual behavior. Future studies could make the evidence stronger by looking at the frequency of logins, time spent, on-task completion rates, or classroom observations.

Table 3. Correlation Between Students' Evaluation of the Web-Based Programming Learning Resource and Their Perceived Learning Outcomes

Variables	Pearson's r	Strength of Association	p
Learning Resource × Perceived Academic Performance	0.81	High Positive	< .05
Learning Resource × Perceived Student Engagement	0.79	High Positive	< .05

Note. Pearson Product–Moment Correlation was used to examine the relationship between students' evaluation of the learning resource and their perceived learning outcomes.

The correlation analysis in Table 3 revealed positive and statistically significant associations between students' evaluation of the web-based programming learning resource and both dimensions of perceived learning outcomes. Students who reported more favorable evaluations of the learning resource also tended to report more positive perceptions of academic performance and student engagement. Given the correlational nature of the research design, these findings indicate an association between the variables rather than a causal relationship.

The correlation, r = .81, between how students evaluate the learning resource and how they perceive their performance is strong and positive. In terms of students, those who rate the learning resource more favorably also tend to rate their academic performance more favorably. The strength of this association matters because it ties students' appraisal of the resource to a learning outcome, not just to satisfaction. It suggests that students' experiences of clarity, usefulness, accessibility, and instructional value may be closely related to how students perceive their programming performance.

The correlation of r = .79 between evaluation of the learning resource and perceived student engagement is likewise high and positive. This indicates that favorable evaluations of the resource tended to occur alongside stronger perceptions of participation, attention, interest, motivation, and satisfaction. The magnitude is slightly lower than the academic-performance correlation, but the difference is small. Both coefficients therefore point to a consistent pattern: students who viewed the resource positively tended to report positive learning experiences across both performance and engagement dimensions.

These relationships can be seen through how a web-based learning resource teaches. When learning materials are organized clearly, when activities are at the right level of challenge, and when students can interact with content in many ways, the learning resource becomes more useful and easier

to use. Students who find the learning resource are then more likely to put effort into programming practice and see larger progress. Earlier studies on learning have also highlighted how interaction, learning activity, and the presence of an instructor shape learning experiences (Kuo et al. 2014; Martin et al. 2020; Rapanta et al. 2020). This study takes that idea further by looking at programming learning in schools that do not have many resources.

Do not read the correlations as proof that the learning resource caused the reported outcomes. A correlation of .81 does not mean that the learning resource gave an 81 percent boost in performance, nor does it prove a causal link. The variables may be linked because students who are already more motivated or confident also rate the learning resource favorably. Other factors, such as teachers' support, prior programming experience, access to devices, internet reliability, peer support, and personal learning habits, may also add to the pattern we see. Because the study used a correlational design and self-reported data, claiming causation would go beyond the evidence we have.

The two important relationships still hold value. They show that students' views of learning resources are not separate from the goals schools aim to achieve. If teachers and schools want students to take an active part and feel they are making progress in programming, the design and ease of use of learning materials must be considered. The findings support a method in which online materials are carefully integrated with classroom teaching, structured practice, feedback, and assessment. These materials should not just be offered as a storage of content.

The results also match what the study was focused on, which is places with resources. In these kinds of situations, the main question is not whether online learning is always better than classes, but whether a well-designed digital tool can offer helpful additional opportunities when regular materials are not available. The good links found here show that students saw the tool as something that added value to their lessons. This idea goes along with research into using technology in countries that don't have a lot of money or tools. That research says that how technology is used and how the lessons are set up are just as important as having the technology itself (Barikzai et al., 2024; Barrot et al., 2021).

Students' Perceptions of Academic Performance

The positive perceptions of performance show that students saw the web-based programming learning resource as a useful addition to classroom teaching. Programming is a skill-based subject where learners become proficient by practicing ideas, conducting experiments, and solving problems. In schools with limited supplies, tools that let students review lessons, strengthen classroom work, and practice on their own can extend learning opportunities. This view is consistent with studies highlighting the importance of technologically supported learning environments in providing opportunities for continued learning (Anthony et al., 2022; Martin et al., 2020).

The way people respond also suggests that the resource might have reduced one teaching limitation: the time students have to practice repeatedly. Learning to program often follows a cycle where students try a solution to find an error, check the result, fix the code and try again. A website-based tool can help with this cycle by providing examples and tasks students can revisit. This kind of flexibility is especially important when class time is short or when lab access is shared. But the tool needs to be designed so that practicing over and over again has a real purpose. Just adding exercises might not help if the tasks are the same each time, poorly arranged, or not tied to what the course is trying to teach.

From the perspective of how learning is structured, the results suggest that activities should start with basic ideas and then progress to more real-world programming challenges. Early tasks can include solved examples to support practice, while later tasks can ask students to solve problems on their own and fix errors. Feedback needs to do more than say if an answer is right or wrong. It should also explain why an error happened and how a student can fix it. This kind of support can help students move from recognizing programming ideas to using them independently, which aligns with the general idea of ensuring that what is taught and what is learned fit together well (Biggs & Tang 2011).

Students' Perceptions of Student Engagement

The positive perceptions of student engagement show that students considered the web-based programming learning resource supportive of participation and involvement during programming activities. Engagement is especially important in programming education because learning to program

requires sustained attention, persistence, and repeated interaction with complex tasks. The findings are in line with the idea that learner engagement is a multi-dimensional construct that involves behavioral, cognitive, and emotional participation (Fredricks et al., 2004).

The engagement findings also have implications for how teachers introduce and use the resource. Engagement is unlikely to be sustained if digital activities are assigned without explanation, monitoring, or feedback. Teacher presence remains important in blended environments because students may need clarification, encouragement, and help interpreting errors. Rapanta et al. (2020) emphasized the role of learning activity and teacher presence in teaching, while Martin et al. (2020) highlighted the diversity of design and implementation factors that shape online learning. In this context, the resource is therefore best understood as part of a teaching system in which teachers guide students before, during, and after digital activities.

The lower rating for motivation to explore beyond classroom requirements provides a design signal. The resource appears capable of engaging students during learning, but independent extension may require additional features or pedagogical strategies. Teachers can build on the existing engagement by introducing optional challenge levels, project-based tasks, programming puzzles, and opportunities for students to create simple applications connected to real-life contexts. These approaches may help transform short-term interest into cognitive engagement and self-directed practice.

Relationship Between the Learning Resource and Students' Perceived Learning Outcomes

The study found that students who evaluated a web-based learning resource also tended to report better learning outcomes and greater engagement. This suggests that when students have experience with a learning tool, they are more likely to feel they are learning well and staying involved. These results show that examining how learners feel about technology is part of understanding how it works in education.

Limitation of the Study

However, because the study used a particular design and relied on self-reported data, we cannot say for sure that the learning resource led to better performance. The connections seen may be due to other factors. For example, students who already feel confident might rate the resource highly. Also do well. Those who put in more effort might both use the tool more and improve at it. So, while there is a link, we can't claim causation based on this research.

One idea is that usefulness and experience feed each other. If a student finds a resource that is easy to understand and manage, they may use it often. As they practice more, their success could make them believe the tool is helpful. This back-and-forth relationship is logical. The current study does not prove it. We only know there is a connection, not whether one thing causes the other or if something else links them.

Recommendations

Future studies could explore patterns. For instance, does student engagement act as a step between the quality of the resource and actual achievement? Could prior knowledge affect how students respond to the material? Answering these questions would help us better understand what makes a learning tool effective.

The strong correlations also remind us that evaluating technology should include the learner's point of view. Just because a tool works technically loading and having correct content—does not mean it helps students learn. If explanations are unclear, activities seem pointless, or feedback is unhelpful, even the best tech fails. On the other hand, if students find a tool engaging and useful, they are more likely to use it regularly. That means student perception must be part of any evaluation along with real test scores, usability metrics, and teacher input.

This is especially true for schools with resources. When schools invest in technology, they need to match it with teaching needs. A web-based resource can be very useful if it fills a gap—for example, by offering practice or structured lessons that are hard to deliver otherwise. The positive findings in this study support that idea. They don't mean every online tool will work the same way. The results suggest that choosing and designing tools carefully matters. It's not enough to have technology; it has to fit the learning goals and meet students where they are.

Comparison with previous literature

The results match studies that show designed technology-based learning spaces can provide good opportunities for learning and student interest. The average academic performance score of 3.76 indicates that students found the online programming tool helpful for learning programming. This result is consistent with the findings of Anthony et al. (2022) on the positive contribution of blended learning to students' learning experiences and outcomes. Martin et al. (2020) also found that blended and online learning spaces can offer learning opportunities when lessons are well planned. In the area of programming, Lye and Koh (2014) and Grover and Pea (2013) also found that repeated practice, problem solving, and applying what you learn help build programming and thinking skills. This study adds to that conversation by looking at students in schools with few resources.

The engagement results also match research. The average engagement score of 3.82, along with a score of 3.88 for multimedia keeping attention, shows that students found the tool interesting and fun to use. This aligns with learning research indicating that learner involvement, interaction, and teacher support are important for good learning experiences (Kuo et al., 2014; Martin et al., 2020; Rapanta et al., 2020). The results also align with the concept of engagement in Fredricks et al. (2004), where behavior, feelings, and thinking all matter in student participation. However, the lower score for motivation to explore programming of class (3.75) adds something new: a tool can help with classwork but not always push students to keep learning on their own.

The correlation results agree with studies but also give new details. The strong link between how students rate the learning tool and how they see their learning ($r =$, $p < .05$) matches research showing that students think the tool is useful, interactive, and has good lessons that help them learn (Kuo et al., 2014; Ashill, 2016). The link with engagement ($r = .79$, $p < .05$) also aligns with studies showing that online learning tasks and teacher support keep students engaged (Martin et al., 2020; Rapanta et al., 2020). What this study adds is information from resource-rich schools, where students may have fewer opportunities to practice than those in better-resourced places.

Still, the results should be looked at differently than in other studies. This study used self-reported feelings, so it can't prove the learning tool caused the results. The results add to what's already known, but only at the level of how students feel about learning. They don't show the tool works. This difference is important when comparing to studies that use tests, experiments, or follow students over time.

Implications for learning design

The findings point to principles for designing web-based programming learning resources for secondary schools with limited resources. First, the content should be built around programming skills and the goals of the course. Students liked how well they understood concepts, showing that clear explanations and organized practice are useful. However, the lower scores on assessment-based questions indicate that the activities must also align with the tests that will be administered. Thus, learning goals, examples, practice problems, and tests must flow together, not stand alone.

Second, the resource must focus on programming practice. Students should be asked to guess what a program will do to change code, spot mistakes, build solutions, and explain why they did what they did. Short activities are especially helpful when devices or internet are not always available because they let students learn in windows. When possible, the activities should give feedback, so mistakes become chances to learn, not just signs of failure.

Third, multimedia must have a purpose. The highest engagement scores came when multimedia kept attention. That means people liked multimedia. Designers should not add media just for looks. Explanations, diagrams, demos, and other media should tie directly to programming ideas. Do not overload the mind. This follows multimedia learning rules that stress clarity, signal breaks in content, and match words with pictures (Mayer, 2021; Clark & Mayer, 2016).

Fourth, the resource must accommodate students' diverse backgrounds and learning paces. Guided examples help beginners; optional extra tasks can challenge those who move faster. The highest, though still lowest, engagement score came from topics outside the class ($M = 3.75$). That shows an opportunity to boost learning with extra tasks, project ideas, and more open programming challenges that grow step by step.

Finally, the design should view the web resource as a helper to teachers, not a replacement. Teachers can explain activities, spot misunderstandings, give feedback, and link work to class projects. In resource-rich places, this mixed role is even more important because students' access can vary. So, the resource should be flexible, with activities that still matter when students have internet or only short online time.

Implications for Curriculum development

Researchers notice that at the curriculum level, the results show that structured digital learning resources should be a part of programming instruction rather than optional extras that are not linked to curriculum outcomes. Curriculum planners can match web-based activities to competencies, performance standards, and assessment criteria. This mapping helps ensure that technology use directly supports the intended learning outcomes of programming courses.

The researchers also see that programming curricula need repeated chances for application. Students gave views across concept mastery, practical activities, and logical reasoning, which shows the importance of learning sequences that go beyond introductory explanations. Curriculum planners can add practice, debugging tasks, small projects, and authentic problem contexts so that students keep applying programming concepts in many situations. This fits with the role of programming in developing thinking.

Equity is another point. A digital curriculum that assumes high-speed connectivity or one device per student may unintentionally leave behind learners in resource-constrained schools. Instead, resources should be checked for accessibility, bandwidth requirements, device compatibility, offline or low-bandwidth options, and the amount of teacher support required. Research on e-learning in emerging economies shows that local conditions shape success. The curriculum must therefore state not which digital resources to use but also how they can work under different resource settings.

Finally, curriculum evaluation should use sources of evidence. Students' perceptions are valuable because they show whether learners find a resource engaging, but curriculum decisions must also consider objective programming assessments, completion data, teacher observations, and long-term learning outcomes. Combining these sources provides evidence to decide whether a resource should be scaled across schools.

Implications for educational policy

The findings matter for policy because access to quality programming education is increasingly about having the right digital learning opportunities. For schools that don't have money, policy should go beyond just giving out devices or internet access without also providing good teaching materials. A lasting digital learning plan should consider how infrastructure, teacher skills, curriculum fit, content quality, technical support, and student access work together.

Policymakers might also consider establishing design and accessibility rules for online programming resources used in public schools or schools with limited resources. These rules involve working with devices that are easy to find, using internet connections efficiently, having clear ways to move around content, making it easy to understand, including activities that help students learn, and making sure the material matches what students are supposed to learn. These rules would help schools choose resources based on how useful they are for learning, not just how new or complicated they are.

Teacher training is another part of policy. For learning to work well, teachers need to know how to include online activities in their lessons, understand student performance data, give feedback online or in a mix of online and in-person ways, and change activities when the internet or devices are not available. Spending on teacher training can help make the most of the technology already in place. Policy should see teacher training as part of digital learning, not as something only considered after everything else.

The study also supports policies that prioritize fairness in learning. UNESCO (2023) says that technology in education should be viewed in terms of access, management, and learning goals. For programming education, fair access means more than being able to open a website. It also means having opportunities to practice coding, get help, and do meaningful tasks. Policies that support low-bandwidth

options, times when students can use school computers, sharing devices, and content that fits the community may help close the gap between schools that have more resources and those that have less.

Limitations

The researchers think these limits show that we need to be careful when reading the results. Several limitations need to be considered when interpreting the results. First, this study used a correlational design. The strong links found between students' ratings of the learning resource and their perceived learning outcomes suggest a connection, not that one caused the other. We cannot say that the resource alone led to better performance or engagement among students.

Second, this study relied on self-reported questionnaire responses. Perceptions give information about students' experiences, but students' perceptions may differ from objective measures of programming competence or observed engagement. Students may also be influenced by students' expectations, experiences or students desire to provide favorable responses. Future studies can strengthen the evidence base by combining questionnaires with programming assessments, activity logs, observations, interviews, or performance-based tasks.

Thirdly, the participants were Senior High School students in the Information and Communications Technology strand from selected resource-constrained schools. Selecting these participants limits the extent to which the findings can be generalized to all secondary students, schools, or educational contexts. Differences in curriculum, teacher preparation, infrastructure, prior programming experience, and student characteristics may yield different results in other settings.

Fourth, this study did not include a comparison group. Consequently, changes in perceived outcomes cannot be attributed uniquely to the web-based resource. Classroom instruction, teacher support, peer interaction, prior knowledge, and other learning experiences may also have influenced responses. This present study should therefore be viewed as an empirical description of students' perceived outcomes and relationships among the variables.

Finally, the available measures focused on perceived performance and engagement. These constructs are important. They do not capture every dimension of programming learning. Future research could examine programming achievement, computational thinking, and the retention of programming concepts and their transfer to new problems and longer-term persistence. Addressing these limitations would provide evidence about the educational effectiveness of web-based programming resources.

Recommendations

Based on the findings and limitations, teachers should use programming resources as complements to classroom instruction. Activities must be directly connected to lesson objectives. Must include opportunities for guided practice, independent problem solving, debugging, and feedback. Teachers can also use programming resources to provide differentiated activities for students with different levels of prior programming knowledge.

Schools should not check whether resources are available from a technical standpoint but must also verify that resources are suitable for teaching, easy to use, accessible to everyone, and aligned with the programming curriculum. School leaders can help by giving teachers a fixed schedule to use devices and the internet, helping teachers when needed, and giving teachers chances to talk about what works. Where connectivity is inconsistent, schools should consider downloadable alternatives when such options are available.

Curriculum developers should align programming activities with competency standards and assessment expectations. The favorable academic performance results suggest that structured practice is useful, while the lower assessment-related rating indicates that practice tasks must be closely aligned with formal assessment demands. Curriculum materials should therefore include exercises, formative assessment, and authentic programming tasks.

Educational policymakers should consider digital-learning equity when supporting programming education. Investments should address the implementation ecosystem, including infrastructure, teacher professional development, accessible learning content and technical support. Policies should encourage resources that can operate under the conditions found in resource-constrained schools, rather than assuming uniform access to high-end devices and reliable high-speed internet.

For researchers, experimental or quasi-experimental studies are recommended to determine whether use of web-based programming resources leads to measurable improvements in programming achievement and engagement. Longitudinal studies could examine whether benefits persist over time. Mixed-methods research could also explore why students perceive activities as useful or engaging, while usage analytics could identify which features of web-based programming resources are most strongly linked to participation and completion. Comparative studies across schools with levels of resources would further clarify the role of context.

Researchers may also investigate whether student characteristics, prior programming experience, access conditions, or teacher support moderate the relationships identified in studies of web-based programming resources. Such analyses could help identify which learners benefit most from web-based programming resources and what implementation conditions are necessary for outcomes.

CONCLUSION

The study examined how students in resource-limited schools perceived their learning outcomes when using a web-based programming resource. The findings are grouped by the four research questions to ensure the results clearly align with the study's goals.

For Research Question 1, students generally expressed views on their academic performance after using the resource. Their average score was 3.76 out of 4. The highest score was for mastering programming concepts with a mean of 3.80. Next was completing programming tasks as expected, with a mean of 3.78. Reasoning and solution accuracy also scored high at 3.77. These results suggest the resource helped students in learning programming. Since performance was based on student self-reports, these scores show how students saw their own progress, not actual results verified by tests or assessments.

For Research Question 2, students also reported levels of engagement with the resource. The average score was 3.82. Multimedia features that kept attention were rated the highest, at 3.88. Students also found the content enjoyable and satisfying (3.85), interesting (3.83), and engaging in participation (3.80). Their motivation to explore programming beyond classwork was slightly lower at 3.75. This pattern shows the resource was engaging. Helped students stay focused. It also suggests there is room to improve students' ability to learn independently and keep going beyond what's assigned.

For Research Question 3, the study found a positive relationship between students' ratings of the learning resource and their views of their academic performance. The correlation was $r = .81$ and statistically significant ($p < .05$). This means students who liked the resource also tended to feel they did better at programming. While this is a link, it does not mean the resource caused the improvement. The study used a design, so it cannot prove cause and effect.

For Research Question 4, there was also a positive relationship between student evaluations of the resource and their perceived engagement. The correlation was $r = .79$, statistically significant ($p < .05$). Students who gave the resource ratings also reported higher levels of attention, participation, interest, motivation, and satisfaction. Like with the academic performance results, this shows an association, not a cause. Other factors, such as teacher support, programming experience, access to devices, and personal study habits, might also play a role.

Overall, the results give evidence that a well-designed web-based programming resource can add value to classroom teaching in schools with limited resources. The study does not claim that the resource causes learning. Still, it does show that students who view the resource positively also tend to feel better about their academic performance and engagement. These findings support focusing on instructional design that aligns resources with the curriculum, provides teacher support, ensures fair access, and uses objective measures to assess outcomes when adding web-based tools to secondary education.

AI USE DISCLOSURE

During the preparation of this manuscript, the authors used AI tools to assist with selected parts of the manuscript, particularly the Abstract, Introduction, Results and Discussion, and Conclusion, mainly

for grammar checking, improving sentence clarity, language editing, and proofreading. The authors reviewed the suggestions, revised the text as necessary, and made the final decisions regarding the study's content and presentation. AI tools were not used to collect or analyze the data, interpret the statistical results, or draw the study's conclusions. The authors take full responsibility for the accuracy, originality, and final content of the manuscript.

REFERENCES

Agbo, F. J., Olivia, C., Oguibe, G., Sanusi, I. T., & Sani, G. (2025). Computing education using generative artificial intelligence tools: A systematic literature review. Computers & Education Open, 9, 100266. https://doi.org/10.1016/j.caeo.2025.100266

Alghamdi, M. Y. (2025). Measuring the impact of web-based educational tools on enhancing student learning indicators in programming skills, computational thinking, and problem-solving. Computer Applications in Engineering Education, 33, e70011. https://doi.org/10.1002/cae.70011

Anthony, B., Kamaludin, A., Romli, A., Raffei, A. F. M., Abdullah, A., Ming, G. L., et al. (2022). Blended learning adoption and implementation in higher education: A theoretical and systematic review. *Technology, Knowledge and Learning, 27*, 531–578. https://doi.org/10.1007/s10758-020-09477-z

Barikzai, S., Bharathi S, V., & Perdana, A. (2024). Challenges and strategies in e-learning adoption in emerging economies: A scoping review. Cogent Education, 11(1), 2400415. https://doi.org/10.1080/2331186X.2024.2400415

Barrot, J. S., Llenares, I. I., & Del Rosario, L. S. (2021). Students' online learning challenges during the pandemic and how they cope with them: The case of the Philippines. Education and Information Technologies, 26(6), 7321–7338. https://doi.org/10.1007/s10639-021-10589-x

Biggs, J., & Tang, C. (2011). Teaching for quality learning at university (4th ed.). Open University Press.

Bond, M., Bedenlier, S., Marín, V. I., & Händel, M. (2020). Emergency remote teaching in higher education: Mapping the first global online semester. International Journal of Educational Technology in Higher Education, 17, Article 50. https://doi.org/10.1186/s41239-020-00282-x

Branch, R. M. (2009). Instructional design: The ADDIE approach. Springer. https://doi.org/10.1007/978-0-387-09506-6

Chen, P.-S. D., Lambert, A. D., & Guidry, K. R. (2010). Engaging online learners: The impact of Web-based learning technology on college student engagement. *Computers & Education, 54*(4), 1222–1232. https://doi.org/10.1016/j.compedu.2009.11.008

Clark, R. C., & Mayer, R. E. (2016). e-Learning and the science of instruction: Proven guidelines for consumers and designers of multimedia learning (4th ed.). Wiley. https://doi.org/10.1002/9781119239086

Eom, S. B., & Ashill, N. (2016). The determinants of students' perceived learning outcomes and satisfaction in university online education: An update. Decision Sciences Journal of Innovative Education, 14(2), 185–215. https://doi.org/10.1111/dsji.12097

Fodouop Kouam, A. W. (2025). Transforming programming education: A comparative study of traditional and tech-enhanced online learning for undergraduates and graduates. Discover Education, 4, Article 575. https://doi.org/10.1007/s44217-025-00981-x

Fredricks, J. A., Blumenfeld, P. C., & Paris, A. H. (2004). School engagement: Potential of the concept, state of the evidence. Review of Educational Research, 74(1), 59–109. https://doi.org/10.3102/00346543074001059

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. Educational Researcher, 42(1), 38–43. https://doi.org/10.3102/0013189X12463051

Hodges, C., Moore, S., Lockee, B., Trust, T., & Bond, A. (2020). The difference between emergency remote teaching and online learning. EDUCAUSE Review.

Katona, J., & Katonane Gyonyoru, K. I. (2025). AI-based adaptive programming education for socially disadvantaged students: Bridging the digital divide. TechTrends, 69, 925–942. https://doi.org/10.1007/s11528-025-01088-8

Kohen-Vacs, D., Usher, M., & Jansen, M. (2025). Integrating generative AI into programming education: Student perceptions and the challenge of correcting AI errors. International Journal of Artificial Intelligence in Education, 35, 3166–3184. https://doi.org/10.1007/s40593-025-00496-4

Kuo, Y.-C., Walker, A. E., Schroder, K. E. E., & Belland, B. R. (2014). Interaction, Internet self-efficacy, and self-regulated learning as predictors of student satisfaction in online education courses. The Internet and Higher Education, 20, 35–50. https://doi.org/10.1016/j.iheduc.2013.10.001

Lin, H.-C. K., Tseng, C.-H., & Chen, N.-S. (2025). Enhancing programming education: The impact of AI-based pedagogical agents on student self-efficacy, engagement, and learning outcomes. Educational Technology & Society, 28(2), 279–294. https://doi.org/10.30191/ETS.202504_28(2).TP02

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K–12? Computers in Human Behavior, 41, 51–61. https://doi.org/10.1016/j.chb.2014.09.012

Martin, F., Sun, T., & Westine, C. D. (2020). A systematic review of research on online teaching and learning from 2009 to 2018. Computers & Education, 159, 104009. https://doi.org/10.1016/j.compedu.2020.104009

Mayer, R. E. (2021). Multimedia learning (3rd ed.). Cambridge University Press. https://doi.org/10.1017/9781316941355

OECD. (2021). The state of school education: One year into the COVID-19 pandemic. OECD Publishing.

Omeh, C. B., Olelewe, C. J., & Ohanu, I. B. (2025). Impact of artificial intelligence technology on students' computational and reflective thinking in a computer programming course. Computer Applications in Engineering Education, 33, e70052. https://doi.org/10.1002/cae.70052

Rapanta, C., Botturi, L., Goodyear, P., Guàrdia, L., & Koole, M. (2020). Online university teaching during and after the COVID-19 crisis: Refocusing teacher presence and learning activity. *Postdigital Science and Education, 2*(3), 923–945. https://doi.org/10.1007/s42438-020-00155-y

Schunk, D. H. (2020). Learning theories: An educational perspective (8th ed.). Pearson.

Sentance, S., & Csizmadia, A. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies, 22*(2), 469–495. https://doi.org/10.1007/s10639-016-9482-0

UNESCO. (2021). Reimagining our futures together: A new social contract for education. UNESCO.

UNESCO. (2023). Global education monitoring report 2023: Technology in education—A tool on whose terms? UNESCO.

United Nations. (2015). Transforming our world: The 2030 agenda for sustainable development. United Nations.

Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User acceptance of information technology: Toward a unified view. MIS Quarterly, 27(3), 425–478. https://doi.org/10.2307/30036540

Yuana, R. A., Sajidan, S., Wiranto, W., & Nizam, M. (2025). Strategies for integrating computational thinking and scientific approaches in programming education: A systematic literature review. Discover Education, 4, Article 371. https://doi.org/10.1007/s44217-025-00834-7

Zhang, S., Jaldi, C. D., Schroeder, N. L., López, A. A., Gladstone, J. R., & Heidig, S. (2024). Pedagogical agent design for K–12 education: A systematic review. Computers & Education, 223, 105165. https://doi.org/10.1016/j.compedu.2024.105165