Integration of Computational Thinking in Learning Computer Programming with Gamification Elements to Foster Student Programming Skill and Student Performance

Intan Nazira Azmi[1*], Noor Azean Atan[1], Lokman Tahir[1], Nur Azmina Paslan[1], Abdulmumini Inda[1]

[1] Universiti Teknologi Malaysia, Malaysia

*intannazira@graduate.utm.my, azean@utm.my, p-lokman@utm.my, nur.azmina@utm.my, inda.abdulmumini@utm.my

Received: 13 March 2024
Received in revised form: 17 April 2024
Accepted: 27 April 2024
Published: 25 July 2024

ABSTRACT

The integration of computational thinking with gamification elements has emerged as a promising approach to enhancing students' programming skills in computer programming education. However, the current state of learning computer programming with computational thinking remains unclear and confusing, leading to a lack of understanding and a decrease in performance among students. Therefore, this study aims to address this issue by investigating the integration of computational thinking in learning computer programming with gamification elements to foster students' programming skills. This research assesses the effects of these learning activities on students' programming skills and performance. Using a quasi-experimental design, the study focuses on designing computer programming learning activities based on computational thinking strategies with gamification elements. The research methodology includes expert validation, pre-activity and post-activity assessments, as well as pre-tests and post-tests. The expert validation process yielded positive feedback, indicating a good understanding of the learning activities. The pre-activity and post-activity assessments demonstrated a significant improvement in students' programming skills, with post-activity scores being higher than pre-activity scores, as confirmed by paired sample t-test analysis. Furthermore, the pre-test and post-test results showed a significant difference, indicating that the learning activities had a significant impact on students' performance. This research study provides valuable insights into the integration of computational thinking and gamification in learning computer programming. The findings support the effectiveness of this approach in fostering students' programming skills and improving the performance of the learning experience.

Keywords
Computational Thinking; Gamification; Computer programming education; programming skills.

Introduction

Educators strive diligently to incorporate technology into educational lessons, integrating student enthusiasm with education in an ever-evolving, technology-driven world. Educators are constantly searching for advanced technologies that will enhance their learners' understanding (Cahyati *et al.,* 2022). Integrating easily accessible human and non-human resources in an effective manner, educational technology serves as an important tool in all teaching and learning activities. On a deeper level of logic, it involves the integration of pedagogical, technological, and conceptual ideas throughout the processes of education and learning (Carstens *et al.,* 2021). All of these features must also be implemented in computer science education, including fields like telecommunications, computer networking, and computer programming (Rouhani *et al.,* 2022). As a field inherently intertwined with technology, learning computer science necessitates the integration of technology into the learning process, alongside appropriate pedagogy and methods.

Therefore, when it comes to learning computer programming, perhaps one of the most crucial skills, mastering programming can be challenging due to the need to grasp knowledge specific to the structure, develop novel ways of thinking about this information, and create programs (Rouhani *et al.,* 2022). In recent times, programming education has received significant attention within school-based curricula, evolving into a major global educational initiative (Holo *et al.,* 2022). However, there remains an issue concerning the high number of dropouts among university students enrolled in computer programming courses and other computer science subjects (Chorfi *et al.,*2020).

Computer science had the highest dropout rates, reaching 9.8% during the 2017-2018 academic year, as revealed by the 2020 UK Debut Career report. Previous investigations have highlighted several factors contributing to this high dropout rate, with challenging and complex subjects being prominent contributors. Subjects such as Data Structures and Algorithms, which demand advanced thinking and skills, are notable examples. The difficulties posed by such demanding subjects, often leading to elevated dropout rates, underscore the significance of nurturing computational thinking skills in students for more effective problem-solving. This viewpoint is supported by research (Sukirman & Ibharim, 2022), which emphasizes the foundational role of computational thinking across various disciplines.

It is believed that students with strong computational thinking skills are better equipped to address intricate issues compared to their peers who lack such skills, especially when confronted with problems that are not only apparent but also straightforward to solve. Furthermore, as highlighted by research (Sukirman & Ibharim, 2022), computational thinking serves as the cornerstone of disciplines encompassing computers, programming, coding, and problem-solving. Individuals who have grasped computational thinking may excel in addressing challenges in programming or related fields (Belmar, 2022). Therefore, in the realm of learning computer programming, the endeavour goes beyond mere technology integration. It necessitates an appropriate pedagogical approach to enable students to grasp the subject with a well-structured thought process.

Literature Review

Due to its stimulating nature and ease of engagement, numerous studies have suggested the use of gaming strategies to enhance students' learning in computer science programs and to develop computational thinking abilities (Chiu *et al.,* 2022). Gamification elements have gained wider acceptance in academic settings over recent years, in part due to evidence that it fosters student engagement and interest. Depending on the specific courses and group characteristics, elements such as points, scoreboards, and badges have been incorporated in gamification implementations (Zahedi *et al.,* 2021). However, as successful gamification in these contexts relies on careful planning to avoid unintended outcomes, several key components need to be considered by professionals implementing gamified learning approaches (Fiş *et al.,*2022).

Incorporating gaming elements into computer science lessons holds the potential to provide participants with valuable learning experiences. This implies that gamification can enhance students' programming skills (Talib *et al.,* 2021). The learners may become more independent due to the incorporation of gaming aspects in the classroom, which can be attributed to the integration of gamification elements in the educational experience. This is because guidelines for each gaming development are provided, and gaming elements allow learners to have more control over their personal learning process (Weurlander & Von Hausswolff, 2021). Thus, this aspect of gamification should align with the integration of the computational thinking phase in learning computer programming.

The integration of gamification, which empowers learners to take charge of their education (Weurlander & Von Hausswolff, 2021), aligns with the incorporation of computational thinking in computer programming learning, a crucial aspect highlighted by the significance of nurturing computational skills for students' success in the modern technological era (Öztürk, 2022). The advancement of pupils' computational thinking skills through programming has become essential for their potential achievements in this modern technological era. Higher education institutions should bear the responsibility for every graduate program they offer and provide students with the kind of coursework and extracurricular activities that will best assist them in achieving this goal (Öztürk, 2022). Higher education institutions can effectively equip students with the problem-solving skills and logical reasoning required to tackle the challenges outlined in computational thinking (Belmar, 2022), thereby better preparing them for their graduate programs and future endeavours. The elements of computational thinking include conceptualizing, devising methods, and creating abstractions in which individuals link challenges with logical reasoning (Belmar, 2022). The ability to handle complex situations, intricate problems, interact with others, and collaborate to achieve a shared objective is all enhanced and developed through computational thinking in programming (Erümit *et al.,*2022).

Since its emergence in the 21st century, most nations have incorporated it into primary education (Holo *et al.,* 2022) . Computational thinking aids in mastering programming skills, aligning with the problem's context. Enhancing students' computational thinking skills could improve their coding efficiency (Öztürk, 2022). As students become proficient in various computational thinking techniques, their problem-solving abilities increase, better equipping them to handle a range of programming issues (Erümit *et al.,* 2022). Therefore, this study integrates computational thinking strategies into programming education by applying gamification elements to foster students' programming skills and performance.

Research Objectives

The objectives of the research are:

1. To design computer programming learning activities based on computational thinking strategy with gamification elements to foster student programming skill.
2. To identify the effect of computer programming learning activities based on computational thinking strategy with gamification elements towards student's programming skill and performances.

Conceptual Framework

According to Carstens *et al.,* (2021), Computational thinking is based on the power and limitations of computer operations, whether performed by a life form or a robot. Computational tools and models offer us the confidence to solve problems and develop systems that none of us could handle unassisted. Computational thinking entails using core computer science principles to solve issues, develop systems, and comprehend human behavior. Computational thinking encompasses a variety of mental skills that mirror the scope of computer science. The majority of scholars agree on four elements: the generalization of patterns and abstraction, the decomposition of the problem, the recognition of patterns, and algorithmic design.

Zahedi *et al.,* (2021) describe the uses of using gamification in an educational environment. In the gamification implementation, components such as points, leaderboards, rules, and badges were employed in line with the course and group characteristics. The various learning techniques certainly have much to offer on many levels, especially when it comes to developing edutainment that can be used in educational settings.

According to Carstens *et al.,* (2021), computational thinking includes a number of crucial elements that aid in understanding and problem-solving in the field of computer science. These elements include issue decomposition, pattern recognition, generalization of patterns and abstraction, and algorithm design.

Finding parallels and commonalities between several examples or circumstances is necessary for the generalization of patterns and abstraction. It enables people to take the core ideas or principles from certain examples and apply them to fresh situations. Individuals are able to create problem-solving techniques that are more effective and adaptable by identifying patterns and segregating them. Decomposing issues into simpler, easier-to-handle components is referred to as problem decomposition. When a difficulty is broken down into smaller difficulties, people may concentrate on fixing each one independently, which frequently results in a better understanding of the bigger issue. Decomposition enables systematic thinking and aids in a more planned and organized approach to difficult issues.

Finding recurrent features or consistency in information or activities is necessary for pattern recognition. People can form hypotheses, reach conclusions, and gain insights into a variety of events by seeing patterns. In many fields, such as analyzing data, the processing of images, and natural language comprehension, pattern recognition is essential. The production of detailed instructions or sets of steps to solve a problem or complete a job is referred to as algorithmic design. Algorithms offer a methodical approach to problem-solving by leading people through a set of clear processes. Considerations for good algorithm development include efficiency, accuracy, and scalability.

According to Zahedi *et al.,* (2021), gamification elements include points, leaderboards, regulations, and badges. These components are frequently employed in gamification to engage and inspire people in a variety of scenarios. A key component of gamification, points serve as a score or kind of payment given to users in relation to their accomplishments or advancement. Points give a sense of accomplishment and act as a success indicator. In a game or gamified system, they can be acquired, compared, and utilized to open prizes or advance to higher levels.

Leaderboards are graphic representations of a person's position or performance in comparison to others. It encourages rivalry and a sense of social comparison, which inspires players to aim for better leaderboard places. Leaderboards may foster a feeling of community and encourage participation as users vie for attention and prestige. The limits, restrictions, and limitations of a gamified experience are defined by rules. It lays down the rules and game mechanics, defining what is permitted and what is not. To ensure that players understand how to use the gamified system and accomplish desired outcomes, rules offer transparency and fairness. A balanced and interesting experience is enhanced by well-designed regulations. In a gamified system, successes or milestones are represented by virtual badges. It acts as tangible markers of success and can be obtained by carrying out duties, achieving objectives, or exhibiting particular abilities. Additionally, It can be distributed to others, encouraging social acceptance and a sense of pride.

As stated by Zahedi *et al.,* (2021), the gamification components of points, leaderboards, rules, and badges all play a significant part in producing interactive and motivating experiences. These components make use of psychological concepts like competition, accomplishment, and incentives to boost engagement, promote behavior change, and heighten enjoyment in a variety of settings, from marketing and employee engagement to education and training.

Methodology

Sampling and Population

This study uses purposive sampling from one class of Diploma in Digital Marketing students at one of private college in Johor Bahru. The population is 30 who take the course which involve two clasesses. However, for this research, only one class is allowed with the agreement of the college with the total number of 12. Thus, these students were chosen and involved in this research for six weeks. The participants were informed about their right to privacy and their ability to withdraw from the study at any time throughout the six-week period. This is done to guarantee that no force is used throughout the research. Gamification features were used to successfully assist students in understanding computer programming and to sustain students' learning abilities in acquiring a better learning experience. Points, timers, medals, and leaderboards are all common gamification aspects with Wordwall and Kahoot. The incorporated learning activities with Wordwall and Kahoot conducted are based on the learning syllabus prepared by the institute. Computational thinking spans a wide range of mental abilities that correspond to the extent of computer science. The majority of researchers agree on four elements which are pattern generalization and abstraction, issue decomposition, pattern recognition, and algorithmic design.

On week one, students participate in the pre-activity, which was a computer programming session delivered in a traditional setting with no gamification elements based on students' knowledge grades. The computer programming skills rubric score was recorded to compare the efficacy of computer programming learning activities based on computational thinking approach. The next week, the lecturer addressed students with computer programming learning exercises based on a computational thinking technique with gamification aspects. During week six, a post-activity programming skills rubric score was calculated to investigate the relationship between the efficacy of learning activities and the development of students' programming abilities.

Students take the pre-test in week one to assess their level of performance at the start of the research depending on their knowledge. The computer programming grading scores of pre-tests and post-tests were recorded to examine the effectiveness of computational thinking-based computer programming learning activities. As the research progressed to the last week, the lecturer addressed students and a post-test performance grading score was computed to evaluate the difference in student performance between the first and last weeks after the learning activities. The test contain theoretical questions of 5 true or false questions and 2 subjective coding questions which revolve around C++, HTML and PHP.

Result

Based on the analysis conducted, the data analysis below demonstrates the effect of computer programming learning activities that incorporate computational thinking strategies and gamification elements on students' programming skills.

The development of the proceeding learning activities content and suitability of instruments in the lesson was evaluated and validated by a selected programming lecturer by the expert of educational technology who has more than 5 years' experience and expert in computational thinking implmentation.

The evaluation of the integration of computational thinking with gamification elements to enhance student programming skills was conducted through a survey questionnaire consisting of two sections. The first section included four items related to the learning activities with computational thinking elements, while the second section comprised several items regarding the gamification elements, as detailed in Table 1. In addition, two open-ended survey questions were included at the end of the questionnaire to capture the lecturer's general viewpoint and comments.

Table 1. Survey of evaluation and validation by Educational Technology Lecturer

Section A: Learning Activities with CT Method			
No	**Item**	**Agree**	**Disagree**
1	The learning activities with computational thinking methods can be dependable to enhance the ability to discern relevant information and prioritize crucial aspects.	/	
2	The learning activities with computational thinking methods involve breaking down complex problems into smaller segments and analysing ideas and issues in a disassembled manner.	/	
3	The learning activities with computational thinking methods involve recognizing patterns within the structures of various segments and identifying similarities and trends among them.	/	
4	The learning activities with computational thinking method includes writing instructions step by step to create a programme consisting of a succession of ordered stages that can be performed by an instrument.	/	
Section B: Learning Activities with Gamification Elements			
No	**Item**	**Agree**	**Disagree**
1	The learning activities with gamification elements benefit students by providing positive reinforcement and foster motivation towards learning.	/	
2	The learning activities with gamification elements track progress and personalize the learning experience.	/	
3	The learning activities with gamification elements provide positive reinforcement, offer personalization, and provide targeted feedback to students.	/	
4	The learning activities with gamification elements can provide clear guidelines and expectations for students, encourage good behaviour and engagement, and create a sense of structure and order in the learning experience.	/	

The survey responses gathered from the lecturer provided a noteworthy outcome, as it unanimously expressed a complete agreement of 100% with the statements. This indicates a strong consensus which, highlighting her shared perspective and alignment with the content or ideas presented in the survey statements. Table 2 shows open ended questions for validation.

Table 2. Open-Ended Questionnaire of Educational Technology Expert

No	Questions	Comments
1	What is your opinion about using gamification elements to support computer programming courses based on computational thinking?	The use of gamification elements in this programming course has supported students' motivation to continuously engage in their learning and fostered a more active involvement, as well as enhanced their thinking in elaborating each component in the development of coding programming that is more structured based on computational thinking elements
2	What is your opinion about learning activities with computational thinking method?	Computational thinking in this course involves breaking down complex problems into smaller, manageable parts, identifying patterns, and developing step-by-step solutions. By incorporating computational thinking into this course, students learn effective problem-solving techniques more systematically and with greater clarity.

Table 2 displays the feedback on the use of gamification elements to support computer programming courses based on computational thinking indicates that it positively impacted student motivation and active engagement while fostering structured coding development. Additionally, the lecturer noted that incorporating computational thinking into learning activities led to more systematic problem-solving techniques and improved clarity for students.

Effect on Students Programming Skills

The study aimed to assess students' programming skills by conducting pre activities and post activities over a six-week period. The research design incorporated four key elements of computational thinking and gamification elements. The progress of the 12 students' programming skills was evaluated using a set of rubric scores, which were recorded for both the pre-activity and post-activity. The table 3 below illustrates the rubric scores obtained.

Table 3. Pre activity and post activity programming rubric score

Student	Pre-Activity (/30)	Post-Activity (/30)	Differences	Substitute
S1	16	28	12	⇧
S2	22	28	6	⇧
S3	17	30	13	⇧
S4	20	30	10	⇧
S5	19	30	11	⇧
S6	14	30	16	⇧
S7	18	30	12	⇧
S8	21	30	9	⇧
S9	12	30	18	⇧
S10	16	30	14	⇧
S11	17	28	11	⇧
S12	18	28	10	⇧

Remark: increase ⇧, decrease ⇩, equal ⇔

Table 3 displays the rubric scores for the pre-activity and post-activity, revealing that all students exhibited an increase in their scores. The highest score recorded for the pre-activity was 22, indicating strong initial programming skills, while the lowest pre-activity score observed was 12, representing a relatively weaker starting point. On the other hand, for the post-activity, the highest score achieved was 30, demonstrating significant improvement in programming skills, while the lowest post-activity score obtained was 28. Table 4 summarizes result of student's programming skills score.

Table 4. Result of student's programming skills score

Grade	Score	Total Student (pre-activity)	Total Student (post-activity)
Distinction	26-30	0	12
Merit	21-25	2	0
Passed	15-20	8	0
Fail	0-14	2	0

Based on table 4, in the pre-activity, none of the students achieved the grade of Distinction (0% of total students), while 16.7% (2 out of 12) of the students received the grade of Merit. The majority of students, 66.7% (8 out of 12), obtained the grade of Passed, and the remaining 16.7% (2 out of 12) fell into the Fail category. However, in the post-activity, all students (100% of total students) were able to achieve the grade of Distinction, resulting in a significant improvement in performance. There were no students who received the grades of Merit, Passed, or Fail in the post-activity. These figures demonstrate the tremendous improvement made by students following the incorporation of computational thinking and gamification aspects, with a significant rise in the percentage of students achieving the highest grade level.

As a result, to analyse the impact of learning activities on students' programming skills, the collected data was subjected to statistical analysis using SPSS. Firstly, a normality test which is Shapiro-Wilk test needs to be conducted to evaluate the distribution of the data. This test is crucial to determine whether the data follows a normal distribution or non normal distribution. The result shows in Table 5.

Table 5. Normality Test for pre activity and post activity

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
PreActivity	.132	12	.200*	.980	12	.984
PostActivity	.417	12	.000	.608	12	.000

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

The results of the normality test indicate that the data distribution for the pre-activity is 0.984, suggesting that it follows a relatively normal distribution. Contrarily, the post-activity normality test results in a p-value of less than 0.001, suggesting that the post-activity data does not correspond to a normal distribution. These results indicate that while the post-activity data greatly deviates from normalcy, the pre-activity data may be thought of as being nearly normally distributed. This suggests that non-parametric methods, such as the Wilcoxon Signed Ranks Test, may be better suitable for assessing the statistical significance of the observed differences when comparing the pre- acticity and post-activity scores for investigating the effect of the learning activities on students' programming skills. The result of the test shows in Table 5 and 6.

Table 5. Rank for pre-activity and post-activity

Ranks				
		N	Mean Rank	Sum of Ranks
PostActivity – PreActivity	Negative Ranks	0[a]	.00	.00
	Positive Ranks	12[b]	6.50	78.00
	Ties	0[c]		
	Total	12		

a. PostActivity < PreActivity
b. PostActivity > PreActivity
c. PostActivity = PreActivity

Table 6. Wilcoxon Signed Rank Test for pre-activity and post-activity

Test Statistics[a]	
	PostActivity - PreActivity
Z	-3.063[b]
Asymp. Sig. (2-tailed)	.002

a. Wilcoxon Signed Ranks Test
b. Based on negative ranks.

Table 6 displays the results of the Wilcoxon signed rank test, indicating a Z-value of -3.063 and an asymp (2 tailed) value of 0.002. These results offer statistical support for a substantial difference between the pre- activity and post-activity ratings. The negative Z-value implies that the post-activity results were a lot higher than the pre-activity levels, demonstrating a development in the students' programming skills. The modest asymp (2 tailed) value of 0.002 shows that this distinction is statistically significant at a significance level of p less than 0.05, further demonstrating the effectiveness of the learning activities to foster students' programming abilities.

Effect of Learning Activities Toward Students Performance

As part of the research project, a pre- test and post-test was given over the course of four weeks in order to thoroughly examine student performance. The results of these assessments were based on the students' grade system, which offered a uniform measurement of their academic success. The result shows student's performance for the pre- test and post-tests in Table 7.

Table 7. Pre-test and post-test grade score

Student	Pre-Test (/100)	Grade	Post-Test (/100)	Grade	Differences	Substitute
S1	65	B-	100	A+	35	⇧
S2	80	A-	95	A+	15	⇧
S3	60	C+	100	A+	40	⇧
S4	60	C+	100	A+	40	⇧
S5	55	C+	100	A+	45	⇧
S6	65	B-	95	A+	30	⇧
S7	60	C+	100	A+	40	⇧
S8	55	C+	85	A	30	⇧
S9	65	B-	95	A+	30	⇧
S10	60	C+	95	A+	35	⇧
S11	60	C+	95	A+	35	⇧
S12	60	C+	95	A+	35	⇧

Remark: increase ⇧, decrease ⇩, equal ⇔

By comparing the scores in the table 7, it becomes evident that the assessments were effective in capturing the students' progress over time. Remarkably, all students demonstrated a 100% increase in their performance between the pre-test and post-tests. The highest pre-test grade achieved was an A-, indicating a strong initial level of understanding, while the lowest pre-test grade obtained was a C+. However, in the post-test, the highest grade attained was an A+, showcasing a substantial improvement in students' comprehension and skills. Intriguingly, the lowest grade in the post-test was an A, indicating that even students who initially struggled managed to reach an exemplary level of achievement. Table 8 summarizes result of student's performance grade score.

Table 8. Result of student's performance grade score

Grade	Score	Total Student (pre-test)	Total Student (post-test)
A+	90 - 100	0	11
A	85 - 89	0	1
A-	80 - 84	1	0
B+	75 - 79	0	0
B	70 - 74	0	0
B-	65 - 69	3	0
C+	60 - 64	8	0
C	55 - 59	0	0
C-	50 - 54	0	0
D+	45 - 49	0	0
D	40 - 44	0	0
F	0 - 39	0	0

Based on table 8, the grade distribution, presented in terms of percentages, reveals that in the pre-test, no students achieved an A+ grade, accounting for 0% of the total students. However, in the post-test, 11 students, or 100% of the total students, attained an A+ grade. Additionally, in the pre-test, no students received an A grade, representing 0% of the total students. During the post-test, a single student, comprising 9.1% of the total student population, earned an A grade. Furthermore, one student, or 9.1% of the total students, received an A- grade in the pre-test, while during the post-test, there was a lack of students obtained this grade, accounting for 0% of the total students. No students achieved a B+ or B grade in pre-test and post-test, indicating 0% for both categories. Finally, in the pre-test, three students, or 27.3% of the total students, attained a B- grade, while in the post-test, none of the students received this grade, representing 0% of the total students. Similarly, in the pre-test, eight students, or 72.7% of the total students, achieved a C+ grade, and in the post-test, no students obtained this grade, accounting for 0% of the total students.

In the next step of the analysis, the data measured using SPSS, a statistical software. Before proceeding with further analysis, it is essential to conduct a normality test. Shapiro-Wilk's normality test is used to determine the likelihood that the data has a normal distribution. By examining the distribution of the data, researchers can determine the appropriateness of applying certain statistical tests. Table 9 shows normality test of Shapiro-Wilk test result.

Table 9. Normality Test for pre-test and post-test

Tests of Normality						
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
PreTest	.291	12	.006	.762	12	.004
PostTest	.303	12	.003	.734	12	.002

a. Lilliefors Significance Correction

The p-value for the pre-test is 0.004, and for the post-test, it is 0.002, according to the findings of the normality test. These p-values indicate that the data distribution for both the pre-test and post-test substantially deviates from a normal distribution and are below the standard significance level of 0.05. This shows that the normality presumption is invalidated. Consequently, the Wilcoxon signed-rank test, a non-parametric test, used to determine the significance of

any changes between the pre-test and post-test scores. The Wilcoxon signed-rank test does not rely on the assumption of normality and is appropriate for comparing matched data. The result of the test shows in Table 10 and Table 11.

Table 10. Rank Test for pre-test and post-test

Ranks				
		N	Mean Rank	Sum of Ranks
PostTest - PreTest	Negative Ranks	0[a]	.00	.00
	Positive Ranks	12[b]	6.50	78.00
	Ties	0[c]		
	Total	12		

a. PostTest < PreTest
b. PostTest > PreTest
c. PostTest = PreTest

Table 11. Wilcoxon Signed Rank Test for pre-test and post-test

Test Statistics[a]	
	PostTest - PreTest
Z	-3.081[b]
Asymp. Sig. (2-tailed)	.002

a. Wilcoxon Signed Ranks Test
b. Based on negative ranks.

Table 10 displays the results of the Wilcoxon rank test, stating that the asympt (2-tailed) p-value is 0.002 and that the estimated Z statistic is -3.081. These findings show that the paired observations from the pre-test and post-test differ significantly. The low Z value indicates that post-test scores frequently exceed pre-test levels. The p-value of 0.002 is less than 0.005 shows that there is possible to draw the conclusion that there is significant difference in statistics concerning the pre-test and post-test scores, indicating that the study's learning activities had a major impact on the students' performance.

Summary on The Design Learning Activities

The design and development of this study have successfully integrated computational thinking in learning computer programming with gamification elements. The positive impact of this approach was evident in terms of increased student programming skills and performance. These findings align with the results of the two selected studies, further supporting the notion that integrating computational thinking with gamification elements in programming education can be highly beneficial.

The learning activities emphasized computational thinking elements which are the generalization of patterns and abstraction, decomposition of problems, recognition of patterns, and algorithmic design. It enabled students to break down complex programming challenges into smaller components. This approach encouraged systematic thinking and problem-solving skills (Belmar, 2022). Additionally, the incorporation of gamification elements, particularly the implementation of rules, points, badges created a sense of challenge and competition among students, increasing performance and engagement (Zahedi *et al.,* 2021).

Aligned with research by Sukirman & Ibharim, (2022), who conducted a study investigating the influence of programming robot learning mode based on conceptual mapping on college students' computational thinking abilities. The findings of their research indicated that using programmable robots to teach computational thinking abilities was beneficial regardless of age class. Moreover, the study highlighted that pupils showed a high inclination to utilize conceptual mapping to learn computational thinking. This aligns with this research, as both studies emphasize the positive impact of integrating computational thinking in learning computer programming.

In a similar vein, Erümit *et al.,* (2022) explored the balance between computational thinking and learning motivation in elementary programming education through an empirical study with the integration of gamification elements. The findings of the study revealed that employing gamification elements in education as a teaching method significantly improved students' learning and computational thinking, regardless of the programming environment used. This aligns with the research as well, as both studies emphasize the positive influence of incorporating gamification elements in learning computer programming.

It is evident that all three investigations highlight the benefits of integrating computational thinking with gamification elements in learning computer programming. It focused on the influence of programming robot learning mode, and Erümit *et al.,* (2022) explored the use of game-based learning. Despite the differences in the specific approaches, all studies observed positive outcomes in terms of improved computational thinking skills, enhanced learning motivation, and better performance in programming tasks.

In conclusion, the alignment of the integration of computational thinking in learning computer programming with gamification elements to foster student programming skill's finding with the studies conducted by Sukirman & Ibharim, (2022) and Erümit *et al.,* (2022) confirms the effectiveness of integrating computational thinking with gamification elements in learning computer programming. The positive outcomes observed across all studies highlight the potential of this approach to enhance student programming skills and performance. These insights contribute to the existing body of knowledge and emphasize the significance of incorporating computational thinking and gamification in programming education to create engaging and effective learning experiences for students.

Overall, the research findings support the effect of incorporating computational thinking and gamification elements in computer programming learning activities. The positive feedback from the lecturer and the consensus among survey respondents demonstrate the potential of this approach to enhance skill development in programming.

Summary on Student's Programming Skills

The integration of computational thinking in learning computer programming with gamification elements to foster student programming skill conducted pre-activity and post-activity assessments to evaluate the impact of learning activities on students' programming skills. The findings indicate that it positively influenced students' programming skills, as evidenced by the improvement in their scores.

As such, study aligned by Chorfi *et al.,* (2020) conducted research on the use of online coding platforms as additional distance tools in programming education with computational thinking and gamification elements. The study emphasized that while these platforms provide enhancements for practice and skill enhancement, they are not standalone tools for learning programming languages. The findings highlighted the importance of students having a solid understanding of core programming concepts, techniques, and logic, in addition to engaging in practical exercises. This aligns with the notion that integrating computational thinking and gamification elements should be accompanied by a strong foundation in programming principles to enhance students' skills effectively.

In contrast, the study by Holo *et al.,* (2022), investigated the impact of gamification on student programming skills. The objective of the study was to utilize gamification elements to increase student engagement and proficiency in programming. However, the research findings indicated that despite the incorporation of gamification, there was no significant improvement in students' programming skills. The study highlighted potential limitations in the design or implementation of the gamified activities, suggesting that not all gamification approaches may achieve the desired outcomes in enhancing programming skills.

Overall, the integration of computational thinking in learning computer programming with gamification elements to foster student programming skill aligns with the study by Chorfi *et al.,* (2020) in emphasizing the significance of understanding core programming concepts and integrating computational thinking and gamification elements. However, the study by Holo *et al.,* (2022) introduces a contrasting perspective, highlighting that not all gamification approaches may effectively enhance programming skills. These findings underscore the importance of careful design and implementation of gamified activities in programming education to ensure alignment with learning objectives and maximize student outcomes.

Summary on Student's Performance

The study conducted assessed the effect of the learning activities for student's performance using pre-test and post-test. The findings of the study indicate a significant improvement in student performance after participating in learning activities that incorporated computational thinking with gamification elements. Computational thinking refers to the ability to solve problems using computational strategies such as patterns and abstraction, decomposition of problems, recognition of patterns, and algorithmic design (Belmar, 2022).

The integration of computational thinking in learning activities helps students develop essential skills necessary for problem-solving and critical thinking in various domains (Belmar, 2022). The findings of the study affirm the positive impact of computational thinking on student performance; by incorporating elements such as rewards, challenges, and competition, gamification can create a more immersive and interactive learning experience (Zahedi *et al.,* 2021). The study suggests that the inclusion of gamification elements contributed to the observed improvement in student performance.

The combination of computational thinking and gamification seems to have a synergistic effect on student performance. The study findings indicate that the integration of computational thinking concepts within a gamified learning environment positively influences student learning outcome. This suggests that the interactive and engaging nature of gamification can enhance the acquisition and application of computational thinking skills.

A contrasting finding of a study by Rouhani *et al.,* (2022) indicated that the gamification approach employed in their study did not lead to a significant improvement in student performance. Despite the incorporation of game elements and incentives, the results did not align with the intended objective of enhancing student performance. This disparity suggests that not all gamification strategies may effectively translate into improved performance outcomes in programming education.
Similarly, the study by Holo *et al.,* (2022) focused on investigating the effectiveness of gamification in improving student performance in programming education. The research findings revealed that the gamification approach implemented in their study did not result in a significant enhancement in student performance. Despite the initial expectations of increased motivation through gamification, the observed results did not align with the desired outcome.

Comparing these studies with the integration of computational thinking in learning computer programming with gamification elements to foster student programming skill, it is evident that the project achieved a different outcome. The integration of computational thinking with gamification elements in your learning activities positively impacted student performance, leading to a significant improvement in their scores. This highlights the effectiveness of the approach in fostering student performance.

Overall, the integration of computational thinking strategies with gamification elements in computer programming learning activities improved students' performance and led to notable advancements in academic performance.

Conclusion

In conclusion, this research study examined the integration of computational thinking and gamification in learning computer programming and its impact on student programming skills and performance. The findings provide valuable insights for educators and researchers interested in enhancing programming education through innovative approaches. The study involved pre-activity and post-activity assessments to evaluate the effectiveness of the learning activities, as well as pre-tests and post-tests to assess the impact on students' performance. The results of pre-activity and post-activity indicated a notable improvement in students' programming skills from the pre-activity to the post-activity assessments. The results of pre-test and post-test showed a significant improvement in students' programming skills, with all students demonstrating an increase in their scores. These insights are significant for educators and researchers interested in incorporating computational thinking and gamification into programming education, allowing them to refine instructional materials and delivery methods based on student characteristics and preferences.

Acknowledgment

The authors would like to acknowledge the financial support from Universiti Teknologi Malaysia under Matching Grant UTM-UiTM (Cost Center No.: R.J130000.7353.4B771).

References

Belmar, H. (2022). Review on the teaching of programming and computational thinking in the world. *Frontiers in Computer Science*, *4*. https://doi.org/10.3389/fcomp.2022.997222

Cahyati, S. S., Tukiyo, T., Saputra, N., Julyanthry, J., & Herman, H. (2022). How to Improve the Quality of Learning for Early Childhood? An Implementation of Education Management in the Industrial Revolution Era 4.0. *Jurnal Obsesi : Jurnal Pendidikan Anak Usia Dini*, *6*(5), 5437–5446. https://doi.org/10.31004/obsesi.v6i5.2979

Carstens, K. J., Mallon, J. M., & Al-bataineh, A. (2021). *Effects of Technology on Student Learning*. *20*(1), 105–113.

Chiu, M. M., Lei, H., & Cui, Y. (2022). *Effects of Game-Based Learning on Students ' Effects of Game-Based Learning on Students ' Computational Thinking : A Meta-Analysis*. (June). https://doi.org/10.1177/07356331221100740

Chorfi, A., Hedjazi, D., Aouag, S., & Boubiche, D. (2020). Problem-based collaborative learning groupware to improve computer programming skills. *Behaviour and Information Technology*, *0*(0), 1–20. https://doi.org/10.1080/0144929X.2020.1795263

Fiş Erümit, S., & Karakuş Yılmaz, T. (2022). Gamification Design in Education: What Might Give a Sense of Play and Learning? *Technology, Knowledge and Learning*, 1039–1061. https://doi.org/10.1007/s10758-022-09604-y

Holo, O. E., Kveim, E. N., Lysne, M. S., Taraldsen, L. H., & Haara, F. O. (2022). A review of research on teaching of computer programming in primary school mathematics: moving towards sustainable classroom action. *Education Inquiry*, *00*(00), 1–16. https://doi.org/10.1080/20004508.2022.2072575

Öztürk, M. (2022). *The effect of self-regulated programming learning on undergraduate students ' academic performance and motivation*. (August 2021). https://doi.org/10.1108/ITSE-04-2021-0074

Rouhani, M., Lillebo, M., Farshchian, V., & Divitini, M. (2022). *Learning to Program : an In-service Teachers ' Perspective*. 123–132.

Sukirman, S., & Ibharim, L. F. (2022). *A Strategy of Learning Computational Thinking through Game Based in Virtual Reality : Systematic Review and Conceptual Framework*. *21*(1), 179–200. https://doi.org/10.15388/infedu.2022.07

Talib, N., Yassin, S. F. M., & Nasir, M. K. M. (2021). Teaching and Learning Computer Programming Using Gamification and Observation through Action Research. *International Journal of Academic Research in Progressive Education and Development*, *6*(3), 1–11. https://doi.org/10.6007/ijarped/v6-i3/3045

Weurlander, M., & Von Hausswolff, K. (2021). Engineering students' strategies to learn programming correlate with motivation and gender. *Proceedings - Frontiers in Education Conference, FIE*, *2021-Octob*. https://doi.org/10.1109/FIE49875.2021.9637375

Zahedi, L., Batten, J., Ross, M., Potvin, G., Damas, S., Clarke, P., & Davis, D. (2021). Gamification in education: a mixed-methods study of gender on computer science students' academic performance and identity development. In *Journal of Computing in Higher Education* (Vol. 33). https://doi.org/10.1007/s12528-021-09271-5