International Journal of Science Education and Teaching
Vol. 3, No. 3, pp. 97 - 112, Sep. – Dec. 2024
http://doi.org/10.14456/ijset.2024.9

Incorporating Computational Thinking in Mathematics through Block-Based Programming: Effects on Students' Problem-Solving Skills

Ian Cesar P. Balacuit[1*], Amelia T. Buan [1], Joneil B. Medina[1], Alexis Michael B. Oledan[1]
[1]*Mindanao State University – Iligan Institute of Technology, 9200 Iligan City, Mindanao, Philippines*
*Email: iancesar.balacuit@g.msuiit.edu.ph

Received: 15 October 2024; Revised: 11 November 2024; Accepted: 12 November 2024 2024

Abstract. Computational thinking is considered a fundamental skill in the 21st century. It is a vital skill for students to empower their problem-solving skills through the growing presence of computer technology. As a result, this study utilized the Research and Development method with a Backward Curriculum design to develop Block-Based Programming-Integrated Mathematics Problem-Solving activities that aimed to incorporate computational thinking in block-based programming with mathematics learning competencies. The activities were implemented on Grade 7 students to investigate their problem-solving and computational thinking skills. Results suggest that the students generally exhibited the characteristics of problem-solving and computational thinking skills during the implementation. In addition, they also acquired some improvements. For students' problem-solving skills, the improvements were marked by their acquired ability to 1) establish goals in problem understanding, 2) provide mathematical reasoning in solution planning, 3) draw on the application of mathematics concepts in solution execution, and 4) debug program errors in monitoring and evaluation. On the other hand, students' computational thinking improvements were highlighted by their learned ability to 1) apply a rules-based and systematic approach in problem-solving for algorithmic thinking and 2) utilize patterns in generalizing solutions for pattern recognition. Based on the results, it can be concluded that the activities helped the students develop their problem-solving and computational thinking skills.

Keywords: Block-based Programming, Block-based Programming - Integrated Mathematics Problem-Solving, Computational Thinking Skill, Problem-Solving Skill

INTRODUCTION

Problem-solving (PS) is considered one of the central activities in most educational curricula in the modern era (Ozpinar & Arslan, 2023). Researchers have seen this skill as significant to students because of its applicability to real life. Consequently, developing PS skills has been a goal of many educational programs, including the Philippine K-12 Mathematics curriculum (DepEd, 2012).

One strategy for improving the PS skills of students is to introduce Computational Thinking (CT) in the curriculum (Prsala, 2024). CT, which gained prominence in 21stcentury education, was first suggested by Wing (2006) to involve "solving problems, designing systems, and understanding human behavior by drawing on the concept fundamental to computer science." This skill enables students to practice PS by employing computer-related concepts and approaches, i.e., through representation, abstraction, decomposition, simulation, verification, and innovation (Sengupta et al., 2018). These components also cater to the PS framework in mathematics proposed by Fraillon et al. (2019), who argue that PS includes analyzing the problem, developing an algorithm, and testing. With this, studies have demonstrated a strong link between CT and PS (e.g., Bati et al., 2018; Israel-Fishelson & Hershkovitz, 2022), and a high CT skill is shown to add confidence to students' PS skills (Wei et al., 2021).

Researchers agree that CT development should focus on guiding students to obtain an appropriate understanding of problems and their solutions with the help of computing and computers (Hurt et al., 2023). The advent of computers implies that CT is essential in fostering students' PS skills (Lin et al., 2021). In an era where technology is an integral part of learning, Jocius et al. (2021) assert that every student should learn CT skills and that the need to introduce it to the K-12 mathematics curriculum arises (Ye et al., 2023).

Meanwhile, programming has been agreed to explicitly teach CT, including its core concepts (Basogain et al., 2018). Programming includes writing, testing, debugging, and maintaining codes, a necessary skill set to foster CT among students (Harimurti, 2019). Computer programming is a mutual strategy for developing CT and other essential skills such as PS and communication (Yusoff et al., 2020). With this, researchers support pedagogies allowing all learners, regardless of grade level, to learn programming (Zeng et al., 2023).

A crucial issue of learning CT through programming, especially among students outside computer science, is its complexity and the burden of following the correct syntax in writing codes (Bala, 2021). As a result, researchers suggest the use of Block-based Programming (BBP) language in engaging K-12 students with programming activities to facilitate CT (e.g., Brennan & Resnick, 2012; Yu et al., 2024). Lye and Koh (2014) claim that BBP language is suitable for incorporating CT in programming contexts in K-12 education since it better facilitates the three dimensions, including CT concepts, practices, and perspectives.

Several studies have already investigated the effects of engaging students in BBP language programming activities. For instance, recent studies reveal that introducing BBP activities to the K-12 context improves students' CT skills (e.g., Totan & Korucu, 2023; Kastner-Hauler et al., 2022). Additionally, studies conducted by Durak (2018) and Kwon & Cheon (2019) show empirical evidence of how BBP fosters students' PS skills.

For these reasons, there is an evident need to incorporate CT through BBP with K-12 learning competencies to foster PS skill development. However, the search for pedagogies that would introduce CT for programming through BBP in Philippine K-12 mathematics education continues. There is still limited attempt to introduce CT concepts formally within the context of the mathematics curriculum. This current state is even though PS is one of the twin goals set in the Department of Education's (DepEd) K-12 mathematics education framework. Therefore, this study was conducted to develop PS activities that incorporate CT through BBP within K-12 mathematics education.

RESEARCH OBJECTIVES

CT as a PS approach allows students to create or use technology to solve problems (McClelland & Grata, 2018). Teaching CT empowers students' PS skills by allowing them to maximize the presence of computer technology. Researchers agree that integrating BBP in K-12 education fosters students' CT and PS skills development (e.g., Hickmott et al., 2018; De Chenne & Lockwood, 2022).

However, in the Philippine setting, introducing BBP in K-12 education is still in its infancy, particularly in mathematics. Therefore, this study developed PS activities incorporating CT in BBP into mathematics learning competencies and used these activities to investigate students' PS and CT skills. Specifically, it sought to:

a) Develop Block-based programming–integrated Mathematics problem-solving (BBP-IMPS) activities; and
b) Investigate students' PS and CT skills as they do the BBP–IMPS activities.

METHODOLOGY

This study adopted the research and development (R&D) method of Borg & Gall (1983) in developing BBP-IMPS activities. The R&D approach was utilized to support a systematic and iterative process of analyzing, designing, testing, and refining activities that improved the reliability of the developed intervention. The reasoning is anchored on Gay et al.'s (2012) claim that the purpose of R&D is to develop effective products for use in schools.

Alongside R&D, the principles of Backward curriculum design were followed to ensure that each component of the BBP-IMPS activities is aligned with the desired learning outcomes (Wiggins & McTighe, 2011). Defining the learning outcomes right at the beginning of the process enhanced the coherence of activities. The principle facilitated the alignment of objectives, strategies, and assessments with the competencies in Grade 7 mathematics and concepts in CT.

Participants and Data Collection

This study was conducted online with six Grade 7 students from one of the schools in Dumingag, Zamboanga del Sur, Philippines. The participants were purposively selected based on their capacity to participate in online learning. None of the participants had any experience in programming activities. This criterion ensured that the BBP-IMPS activities fostered PS and CT skills without relying on prior knowledge.

This study lasted two weeks and was implemented using a synchronous learning modality with Google Meet as the online platform. Throughout the implementation, the researcher recorded all events, including student-to-student interactions and student-to-teacher interactions, using Open Broadcaster Software (OBS). Writing prompts were prepared so that students could write down their thought processes for every activity. The student participants were divided into three pairs for the entire implementation. In addition, three experts evaluated students' PS and CT using observation checklists.

Data Analysis

The study's sample size is a notable limitation that could influence the generalizability of its findings. To address this, the study employed data triangulation from multiple sources to include both quantitative and qualitative data. Triangulation was done to validate patterns across different data sources. The quantitative data was obtained through experts' ratings on students' PS and CT using the checklists reported using mode.

On the other hand, qualitative data included recordings of students' conversations and interactions, researchers' observations, and writing prompts that elicited students' thought processes as they did every activity. An in-depth narrative analysis supported the quantitative findings for students' PS and CT skills. Moreover, the analysis was reviewed by an expert to ensure reliability.

Process of the Development of the BBP-IMPS Activities

The development of BBP-IMPS activities followed Borg and Gall's research and development method as recommended by Daulay and Zaman (2012). The R&D method suggested six steps: research and information collection, planning, development of a preliminary product, preliminary testing, revising, and field testing. In the development process, backward curriculum design was embedded in the R&D method. Figure 1 presents a diagram of the development process.

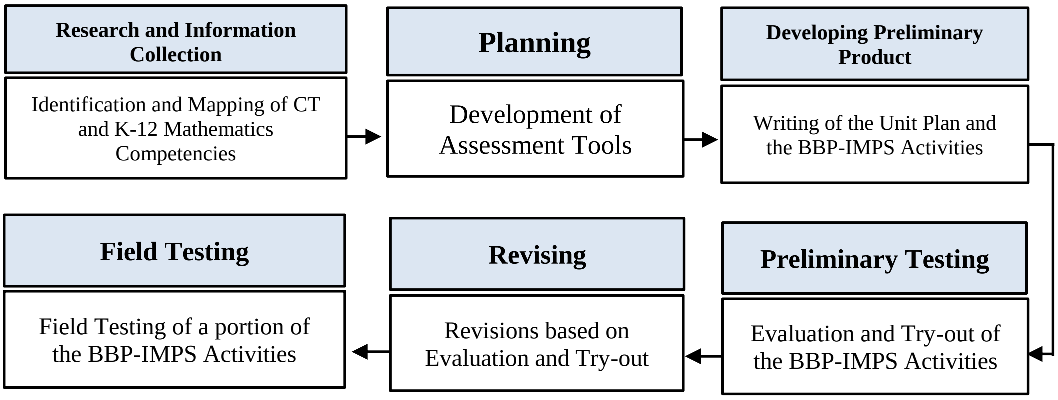

Figure 1: Process of the Development of the BBP-IMPS Activities

Identification and Mapping of K-12 Mathematics Competencies CT Concept and Skills

Identifying learning competencies in Mathematics and CT concepts was an essential step in ensuring an efficient integration of programming activities in Mathematics. As a result of the literature review, the researchers identified competencies from the Philippine Mathematics K-12 curriculum guide under the content area of Geometry. The chosen competencies were: (44) classify the different kinds of angles and (51) construct triangles, squares, rectangles, regular pentagons, and hexagons. The researchers also identified CT concepts based on the framework of Brennan and Resnick (2012), namely sequence, events, loops, variables, and operators. The CT concepts were mapped with Mathematics learning competencies. In addition, cornerstones of CT including decomposition, algorithmic thinking, and pattern recognition, were also considered by the researchers in the mapping. Table 1 presents the mapping of competencies for the developed BBP-IMPS activities.

TABLE 1. Mapping of Mathematics and CT Concepts and Skills

Topic	Act #	Mathematics Learning Competency	CT Concept	CT Skill
Basic Geometric Concept	**1**	classify the different kinds of angles	Sequence; Events	Decomposition; Algorithmic Thinking
	2	construct triangles and squares	Loops	Algorithmic Thinking; Pattern Recognition
Polygons	**3**	construct regular pentagon, hexagons, etc.	Variables; Operators	Decomposition; Pattern Recognition

Development of Assessment Tool

In determining assessment tools for assessing students' output, the researchers determined characteristics from related literature to be included in the PS Skill Observation Checklist and CT Skill Observation Checklist. The checklists' contents were validated based on the results of the field testing conducted. The two checklists have an inter-rater reliability of 87% and 82%, respectively.

Writing of the Unit Plan and the BBP-IMPS Activities

A significant consideration in writing the unit plan and the BBP-IMPS activities was the basic information of Grade 7 students. In designing the unit plan, the researchers adopted an established unit planning template from Intel Education's Designing Effective Project resource (2011). The researchers determined the targeted standards of the learning unit and the essential questions that guided the entire unit. The unit plan design allowed the students to construct a regular program generator with game-like elements as their final output. The design was rooted in the principle of game-design-based learning.

On the other hand, in writing the BBP-IMPS activities, the researchers adapted a framework from the Programme for International Student Assessments (PISA) for PS, which are composed of five components: namely, 1) problem recognition; 2) problem understanding and establishment; 3) solution planning; 4) execution of solution; and 5) monitoring and evaluation (2013). Three (3) BBP-IMPS activities were written by the researchers, with each activity composing three main parts: Exploration, Challenge, and Generalization. Exploration includes activities that allow learners to explore the language of Scratch and Mathematics concepts anchored on Lye & Koh's suggestion (2014). The Challenge part presents the PS task related to Mathematics that the learners must solve by applying the CT and mathematics concepts they have explored in the previous part.

Evaluation of the Unit Plan Design and the Developed BBP-IMPS Activities

The panel of evaluators rated the unit plan using the Unit Planning Rubric (Intel Education, 2011). The unit plan design was rated according to its Targeted Standard, Curriculum Framing Questions, Assessment Plan, and Procedures with *excellent*, *satisfactory*, *below satisfactory*, and *poor* scales. Based on the mean rating, the unit plan was rated "Excellent" in terms of its adherence to the four categories, namely targeted standards ($\bar{X} = 3.67, SD = 0.33$), curriculum framing questions ($\bar{X} = 3.89, SD = 0.19$), assessment plan ($\bar{X} = 3.67, SD = 0.58$), and procedures ($\bar{X} = 3.78, SD = 0.38$). The grand mean rating ($\bar{X} = 3.75, SD = 0.35$) of the unit plan evaluation implies that the unit plan only needs minor revisions based on the evaluators' suggestions.

On the other hand, the panel of evaluators rated the developed BBP-IMPS activities based on their potential to support the characteristics of five skills: Problem Recognition, Problem Understanding, Solution Planning, Execution of Solution, and Monitoring and Evaluation. The evaluation suggests that all BBP-IMPS activities had the potential to support the development of the five skills as seen in the rating for the BBP-IMPS Activity 1 ($\bar{X} = 1.95, SD = 0.12$), Activity 2 ($\bar{X} = 1.94, SD = 0.14$), and Activity 3 ($\bar{X} = 1.94, SD = 0.12$). Furthermore, the grand mean rating of the evaluation $\bar{X} = 1.94$ ($SD = 0.04$) implies that the characteristics of PS activity are observable in all the developed BBP-IMPS activities.

Try-out and Revisions of the BBP-IMPS Activities

The developed BBP-IMPS activities were subjected to a try-out to determine their appropriateness and acceptability concerning the target respondents. It was also conducted to find the aspects of the activities that need possible improvement. During the try-out, the performances of the student participants were carefully observed. The difficulties they encountered throughout the try-out were noted and became the basis of improvement. The researchers, with two mathematics field teachers, observed students' performance during the try-out.

Meanwhile, the BBP-IMPS activities were revised based on the tryout results. Students' responses and interactions, observers' observations, and researchers' observations during the tryout were processed and became the basis for revising the BBP-IMPS activities.

RESULTS AND DISCUSSION

Students' PS Process During the Field Testing of BBP-IMPS activities

The developed activities were subjected to field testing to determine the actual PS process of students doing the BBP-IMPS activities. The field testing significantly validated the developed PS and CT Skills Observation Checklists. It was conducted with a different set of Grade 7 students. Figure 2 shows a diagram of the actual PS process of students during the activity try-out.

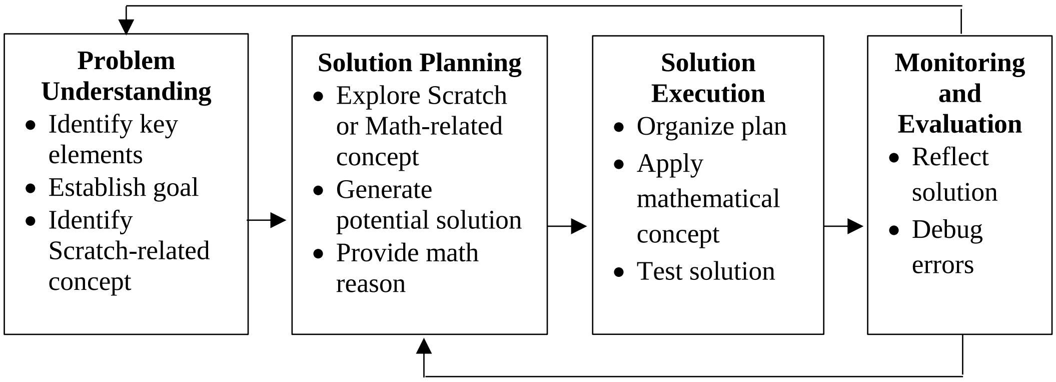

FIGURE 2. Actual PS Process of Students in the Field Testing

In the activities, students display their skill in understanding the problem by identifying its essential elements. Their ability to identify the material elements in solving the problem led them to establish goals that guided them in their solution planning. In planning, students started to generate potential solutions by determining possible code block arrangements in their coding screens (Çakiroglu & Mumcu, 2020). In this stage, they explored the Scratch concept to deepen their understanding of the functions of each code block. The understanding they acquired in their exploration was used to support the rationale of their plans. Hence, how the students provided mathematical reasoning in their plans was based on their exploration (Bouck & Yadav, 2020).

In organizing their plans, students drew on the application of geometric concepts. Students attempted to develop algorithms by following mathematical rules, a characterization of algorithmic thinking (Dagiene et al., 2017). Moreover, testing solution plans to validate the correctness of their solutions was a significant aspect of their PS process. It was also done to determine errors in their program codes.

Meanwhile, students evaluated and monitored program solutions after running their respective programs. At this stage, reflection on their solution was observable during the process. This is a crucial stage in their PS process as it led them to reflect on the possible reasons for their errors so that they could proceed to iterate their process (Kim et al., 2018).

Students' PS Skill During the Implementation of the BBP-IMPS Activities

After analyzing students' processes and program codes, the researchers identified students who displayed variations in constructing their program solutions across activities. Hence, this study's findings intend to present two cases of PS and CT development. Case 1 and Case 2 represent the pairs of students whose produced program codes display consistent variations across implemented BBP-IMPS activities.

Problem Understanding Skill

Table 2 summarizes the ratings obtained for students' problem understanding skills in all implemented activities. It can be observed that identifying key elements of the problem and identifying Scratch-related concepts are the two characteristics consistently observed from students in both cases across all activities. This finding is consistent with Oliveri et al.'s (2017) proposition that in solving problems, students start by defining the problem through identifying its central elements. An example of students' interactions reflecting these characteristics in solving problem understanding says, "[*S_1*] *We can use go to [block] here since it can create a slanted line without the need to turn its [sprite's] direction*."

TABLE 2. Summary of Ratings for Problem Understanding Skill

Problem Understanding	Act 1	Act 2	Act 3
	Mode	Mode	Mode
CASE 1			
Identify key elements	1	1	1
Establish goal	0	0	1
Identify Scratch related concept	1	1	1
CASE 2			
Identify key elements	1	1	1
Establish goal	1	0	1
Identify Scratch related concept	1	1	1

Note: *The data in the table follows the description: 0- Not Observed and 1- Observed.*

In Case 1, students did not engage in goal establishment, at least for the first and second activities; however, they learned to display the skill in the final activity. The improvement was evident when S_1 suggested, "*Let's improve this one; let us use this slider-slider [referring to variable] and operator. But let us put base from our output [in Activity 2].*" This suggestion reflected S_1's desire to enhance their previous code in solving Activity 3 by integrating the CT concept variable.

On the other hand, students in Case 2 engaged in goal establishment in their initial and final activity; however, they did not exhibit the subskill in the second activity. This instance where Case 2 tends to disregard goal establishment in the process of understanding the problem is a potential consequence of their developed familiarity with the problem, as evident in S_4's statement, *"It seems to resemble [the problem] in the previous activity. Let us try solving this one."* Congruent to Priemer et al.'s (2019) claim, developing familiarity with the problem situation directs students' solutions despite being unable to set goals explicitly.

Hence, Case 2 sustained their problem-understanding skills, while Case 1 improved them, manifesting in their learned ability to establish clear PS goals. Problem understanding is essential to their PS skill development as it helps them formulate their plans more effectively (Wei et al., 2021). As a result, both cases displayed more fluency in planning Activity 3 compared to the previous activities, given their established goal. This finding highlights the significance of establishing goals to pursue efficient planning within programming-related problems (Chao, 2016).

Solution Planning Skill

Table 3 summarizes the ratings obtained for the students' solution planning skills in all three activities. Students generally generated potential solutions by planning the arrangement of code blocks. In doing so, they were focused on determining the sequence of their program. An example of how students generate their program solution is exemplified by S_3 saying, "*Just place [turn after turn] lines, only two of them. That's good. Then, place them inside repeat [block]. Then insert [wait time]*."

TABLE 3. Summary of Rating for Solution Planning Skill

Solution Planning	Act 1	Act 2	Act 3
	Mode	Mode	Mode
CASE 1			
Explore Math or Scratch Concept	1	1	1
Generate potential solution	1	1	1
Provide mathematical reasoning	0	1	1
CASE 2			
Explore Math or Scratch Concept	1	1	1
Generate potential solution	1	1	1
Provide mathematical reasoning	0	0	1

Note: *The data in the table follows the description: 0- Not Observed and 1- Observed.*

However, it can be noticed that the students failed to provide mathematical reasoning when planning the first activity. This indicates they lack understanding of the mathematical foundations of their plans during this activity. This further led them to use a trial-and-error strategy to build their program solution. Nevertheless, both cases are already engaged in mathematical reasoning in their final activity. As a result, students already generated their plans based on mathematical rules. It eventually increased the efficiency of their solution process since it lessened their reliance on the trial-and-error method. The finding implies that solutions are more systematic when plans are supported with math-informed reasoning (Cui et al., 2023).

The significance of providing mathematical reasons in planning can be emphasized in the second activity, where this characteristic was observed by students in Case 1 but not in Case 2. In this activity, Case 1 planned fluently and developed a systematic approach to solving the problem more than Case 2. This claim was evident in how well students in Case 1 organized their solution in creating a program that draws a square.

In general, both cases show improvement in solution planning skills. The improvement is marked by students' learned ability to provide mathematical reasons for their plans, particularly in the final activity. The ability allowed them to construct solid plans and helped them increase the efficiency of their solution processes (Çakiroglu & Mumcu, 2020). This conclusion parallels the claim that engaging in mathematical reasoning increases success in solving math-related problems (Ayal et al., 2016).

Solution Execution Skill

Table 4 displays the summary of ratings obtained for the students in terms of solution execution skills across activities. Students tend to execute their solutions by organizing what has been discussed among their pairs during the planning stage. They organized the arrangement of the code blocks they had planned and applied the things they had learned during their explorations.

Testing every solution was also integral to the students' PS process, as this was consistently observed from the students' processes across all activities. An example of this claim is reflected when S_4 said, "*What was it [discussed during planning]? Reduce the size of [sprite]. Then, place the [sprite] here and then "go to" here. Next is pen, erase all, repeat [block], and move. Then wait [block] and turn left [block]. Let us see!*" The statement reflects how students test the solutions they have conceived in the planning stage.

Meanwhile, both cases vary their solution execution skills regarding how they apply mathematical concepts in their solutions. While Case 1 was consistent in drawing the application of math concepts in their solutions, Case 2 did not apply them in their solution for the first two activities. This instance can be traced back to the latter failing to support their plans with mathematical reasoning during these activities. As a result, their solution process tends to characterize a trial-and-error strategy rather than an application of mathematical rules (Sumartini, 2018). This further affected the success of their solution, particularly in the first activity, where their final program did not draw a perfect match with the figure they were tasked to construct.

TABLE 4. Summary of Rating for Solution Execution Skill

Solution Execution	Act 1	Act 2	Act 3
	Mode	Mode	Mode
CASE 1			
Organize argument or plans	1	1	1
Apply mathematical concept	1	1	1
Test solution plan	1	1	1
CASE 2			
Organize argument or plans	1	1	1
Apply mathematical concept	0	0	1
Test solution plan	1	1	1

Note: *The data in the table follows the description: 0- Not Observed and 1- Observed.*

Therefore, data supports that for Case 1, students sustained their skill in solution execution across activities. On the other hand, students in Case 2 improved the said skill, as revealed by how they improved their solution process from the initial to their final activity. The improvement is highlighted by how Case 2 learned to apply mathematical concepts in executing their solution plan. This helped students enhance the fluency and accuracy of their program code (Ran, et al., 2020). It is said that PS skills in the integrated math and programming domain must include the ability to apply mathematical concepts efficiently (Lu & Fletcher, 2009).

Monitoring and Evaluation Skill

Table 5 summarizes the ratings obtained for students' monitoring and evaluation skills. Regarding reflecting program solutions, both cases consistently exhibited the said characteristic across all activities. Reflection allowed the students to evaluate the correctness of their solutions. It also helped them identify errors in their programs so that they could be susceptible to fixing (Angeli et al., 2016).

TABLE 5. Summary of Rating for Monitoring and Evaluation Skill

Monitoring and Evaluation	Act 1	Act 2	Act 3
	Mode	Mode	Mode
CASE 1			
Reflect solution	1	1	1
Test and debug program	1	1	1
CASE 2			
Reflect solution	1	1	1
Test and debug program	0	1	1

Note: *The data in the table follows the description: 0- Not Observed and 1- Observed.*

It can be noticed that the subskill testing and debugging programs were not observable in the first activity for students in Case 2. The inefficiency of Case 2 in debugging the errors of their programs can be attributed to the fact that their planning and execution of solutions need to be anchored on mathematical knowledge. As evidence, their output in the first activity did not accurately match the figure presented in their worksheets. Nevertheless, debugging was already observable in Case 2 during the second and third activities, as evident in the statement "[S_3] *We need only to adjust this [variable slider] S_4 so that we do not need to deal with other blocks. We'll find ways on how to deal with it.*" This is an essential aspect of students' monitoring and evaluation skills since it allows them to enhance accuracy and maximize their program solutions (Bocconi et al., 2016).

Meanwhile, for Case 1, the students sustain all characteristics of monitoring and evaluating skills across activities. This is essential to their PS skill development, enabling them to revisit program codes to construct a more refined solution (NRC, 2013).

Students' CT Skill During the Implementation of the BBP-IMPS Activities

In this study, students' CT skills were measured in terms of their decomposition, algorithmic thinking, and pattern recognition skills, which are the three cornerstones addressed in mapping CT and mathematics learning competencies. Hence, the investigation of CT skills was limited to these three cornerstones.

Decomposition Skill

Table 6 summarizes the ratings obtained for students' decomposition skills across activities. In both cases, students sustained their decomposition skills across activities.

TABLE 6. Summary of Rating for Decomposition Skill

Decomposition	Act 1	Act 3
	Mode	Mode
CASE 1		
Make decisions about dividing a task into subtasks with integration in mind,	1	1
Sort subtasks into categories and place them in structured order	1	1
Think solution of problem in terms of its components	1	1
CASE 2		
Make decisions about dividing a task into subtasks with integration in mind	1	1
Sort subtasks into categories and place them in structured order	1	1
Think solution of the problem in terms of its components	1	1

Note: *The data in the table follows the description: 0- Not Observed and 1- Observed.*

Students focused their PS processes on determining parts of the figure tasks and constructed program codes for each identified part. As evidence, some interactions are presented below:

> "[S_1] *I've already made one point, then move 10 steps. Now, I have already two points. You can help me in solving the last part…we can address this angle.*"
> "[S_3] *Let's address this one first. Let's put base code first [referring to previous work] before we proceed to integrate slider-slider [referring to variable block]. Then, let's deal with this one [referring to operator].*"

From the statements above, it can be observed how students determined subparts of the problem, which they addressed separately to solve the whole problem. It can also be observed that students sorted the determined subparts in terms of priorities. According to Andrian & Hikmawan (2021), this technique is an essential aspect of decomposition, which helps students abstract a whole problem.

Thus, in both cases, students managed to exhibit all characteristics essential in problem decomposition. This implies that they banked on the strategy of addressing an entire problem task in terms of its material subparts (Durak, 2018). Such a strategy allowed them to create program solutions more manageably, which is a crucial cognitive strategy in constructing program codes (Kwon & Cheon, 2019).

Algorithmic Thinking Skill

Table 7 summarizes ratings obtained for students' algorithmic thinking skills across activities. In solving Activities 1 and 2, students arranged the code blocks needed in the program following the sequence displayed by the sprite's movement in the problem tasks. Thus, the fourth characteristic of algorithmic thinking was observable in both cases across activities. Sentance & Czismadia (2016) considered this aspect of algorithmic thinking crucial in enhancing students' PS skills.

TABLE 7. Summary of Rating for Algorithmic Thinking Skill

Algorithmic Thinking	Act 1	Act 2
	Mode	Mode
CASE 1		
Think in terms of sequence	1	1
Think and solve problems in terms of rules	0	1
Develop a systematic solution to the problem	0	1
Identify steps that can be communicated as instructions, codes or programs to other people or to computing devices	1	1
CASE 2		
Think in terms of sequence	1	1
Think and solve problems in terms of rules	0	0
Develop a systematic solution to the problem	0	1
Identify steps that can be communicated as instructions, codes or programs to other people or to computing devices	1	1

Note: *The data in the table follows the description: 0- Not Observed and 1- Observed.*

In both cases, students did not devise their program solution based on mathematical rules in Activity 1. In effect, students' solutions were not systematic. It means that they resorted more to exploring different possible arrangements of code blocks. However, in Activity 2, Case 1 already exhibited the subskill thinking solutions based on rules. As a result, they exhibited systematic thinking when planning the solution. This finding can be confirmed by how S_1 explained their solution:

> "*First is we arrange the code blocks in right sequence. For example, we determine first point [referring to the starting point of the sprite]. Second, we determine direction of the sprite so that it moves in that specific direction. Then, we insert erase all and then pen down for drawing purposes. Then, we put wait [time] before moving 100 steps. The 100-step movement will determine the length of the sides of our square. Then we put 90 degrees for turn [block] so that it forms a specific angle for square. Then, we utilize repeat block to repeat the process four times, and eventually make a square.*"

On the other hand, students in Case 2 were able to develop systematic solutions for the problem in Activity 2; however, rules-based thinking is still not observable. This can be attributed to the fact that they did not apply mathematical concepts in executing their solution (Zeng et al., 2023).

Students in both cases still showed improvement in terms of algorithmic thinking skills across activities. For Case 1, the improvement is emphasized by how students learned to apply rules-based thinking and develop systematic solutions to the problem. Case 2's improvement is marked by how students learned to develop systematic solutions to a problem. Such improvement plays a very crucial role in students' algorithmic thinking skills as it reflects systematic thinking in their solutions (Dogan, 2020).

Pattern Recognition Skill

Table 8 presents the ratings for students' pattern recognition skills. Students from both cases exhibited the first and third characteristics essential to pattern recognition across activities. It means that students identified patterns and similarities and used this technique to represent a sequence of codes in the process (Taylor, 2018). For instance, both cases determine the significance of using repeat blocks in their program codes. This recognition of patterns allowed the students to predict the next set of code blocks needed to continue the correct movement of the sprite.

TABLE 8. Summary of Rating for Pattern Recognition Skill

Pattern Recognition	Act 2	Act 3
	Mode	Mode
CASE 1		
Identify patterns and similarities between problems or sub-problems	1	1
Utilize general solution	1	1
Use patterns to describe and represent sequences in data or process	1	1
Make predictions based on arrangement and relationship between parts	1	1
CASE 2		
Identify patterns and similarities between problems or sub-problems	1	1
Utilize general solution	0	1
Use patterns to describe and represent sequences in data or process	1	1
Make predictions based on arrangement and relationship between parts	0	1

Note: *The data in the table follows the description: 0- Not Observed and 1- Observed.*

In doing Activity 2, students in Case 2 failed to recognize the patterns existing in their program solution. This prevented them from generalizing their solution using the repeat block, resulting in a lengthy code (NRC, 2013). On the other hand, students in Case 1 immediately identified patterns in their code as reflected in S1's statement, "*Didn't we repeat the same blocks four times in creating square? Then we repeat the same set of blocks three times for triangle. Can we use slider (variable block) so that the motion of the sprite be just repeated?*" The statement reflects an intent to utilize the repeat block in generalizing the patterns existing in their solution, resulting in the utilization of a shorter code.

In Activity 3, students from both cases exhibited all characteristics essential to pattern recognition. However, what is notable is Case 2's development, which finally utilizes patterns in generalizing their solution using the repeat block. This eventually allowed them to predict the sprite's movement whenever they changed the value of the repeat block (Kalelioglu & Gülbahar, 2014).

The findings suggest that students in Case 1 sustained their pattern recognition skills across activities. The findings also reveal that students in Case 2 improved pattern recognition skills, manifested in how they learned to utilize patterns to build generalized solutions. The improvement shown in Case 2 highlights the significance of pattern recognition in building quick solutions to problems (Dazgupta & Purzer, 2016).

CONCLUSION AND RECOMMENDATIONS

This study demonstrates the potential of BBP-IMPS activities in incorporating CT effectively within the Grade 7 mathematics education curriculum. Following the research and development model and guided by backward curriculum design principles, the developed activities align block-based programming with mathematics competencies, providing students with opportunities to foster PS and CT skills through hands-on practice. Field experts confirmed that the activities are suitable for meeting needs and standards in mathematics education and are appropriate for classroom implementation. Observations during the study revealed improvement in students' PS and CT skills over the three activities. This finding supports the conclusion that the BBP-IMPS framework contributes to achieving the goals of Philippine K-12 mathematics education.

Additionally, this study contributes to CT instruction, mathematics education, and pedagogy. It proposes a concrete model for integrating CT into mathematics education using BBP that can be adapted for K-12 contexts and is accessible to students without prior programming experience. It also offers an outline of curriculum design that developers can use as a foundation to infuse CT skills within the standards of the country's mathematics education framework. It demonstrates how the integration of CT and mathematics fosters PS while highlighting two relevant activities, i.e., (i) constructing CT outputs while drawing on the concepts of mathematics and (ii) understanding mathematics concepts while participating in CT activities.

Furthermore, this study recommends (1) investigating the effectiveness of BBP-IMPS activities across different grade levels considering large sample size; (2) implementing BBP-IMPS framework using other BBP language to establish its acceptability across languages; (3) conducting longitudinal studies to examine the long-term impact of the sustained use of BBP-IMPS activities on students' PS and CT skills; (4) explore alternative assessments to obtain more comprehensive understanding of students' PS and CT skills; and (5) explore on professional development programs focused on integrating BBP into DepEd's K-12 mathematics curriculum.

ACKNOWLEDGEMENTS

The authors would like to express their immense gratitude to the Capacity Building Program in Science and Mathematics Education (CBPSME) of the Philippines' Department of Science and Technology – Science Education Institute (DOST-SEI) for the financial funding that made this study possible.

REFERENCES

Andrian, R., & Hikmawan, R. (2021). The importance of computational thinking to train structured thinking in problem solving. *Jurnal Online Informatika*, *6*(1), 113. https://doi.org/10.15575/join.v6i1.677

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 Computational Thinking Curriculum Framework: Implications for Teacher Knowledge. *Educational Technology & Society, 19 (3), 47–57.*

Ayal C., Kusuma, Y., Sabandar, J., & Dahlan, J. (2016). *The enhancement of mathematical reasoning ability of junior high school students by applying mind mapping strategy.* JEP 7, 25-31

Bala, R. B. (2021). Scratch in Teaching Programming: Effect on Problem Solving Skill and Attitude. *International Journal of Quality in Education, 5(2), 63-81*. https://dergipark.org.tr/tr/pub/ijqe

Basogain, X., Olabe, M. Á., Olabe, J. C., & Rico, M. J. (2018). Computational thinking in pre-university blended learning classrooms. *Computers in Human Behavior*, *80*, 412-419. https://doi.org/10.1016/j.chb.2017.04.058

Bati K., Yetişir M. I., Çalişkan I., Güneş G., Saçan E. G. (2018). Teaching the concept of time: A steam-based program on computational thinking in science education. *Cogent Education*, 5(1), 1–16. https://doi.org/10.1080/2331186X.2018.1507306

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Developing Computational Thinking in Compulsory Education- Implications for policy and practice. *Publications Office of the European Union*. http://dx.doi.org/10.2791/792158

Borg, W., & Gall, M. (1983). *Educational research: An introduction*. Longman, New York.

Bouck, E. C., & Yadav, A. (2020). Providing access and opportunity for computational thinking and computer science to support mathematics for students with disabilities. *Journal of Special Education Technology, 37(1), 151–160*. https://doi.org/10.1177/0162643420978564

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking* [Conference session]. Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada.

Çakiroglu, Ü., & Mumcu, S. (2020). Focus-fight-finalize (3F): Problem-solving steps extracted from behavioral patterns in block based programming. *Journal of Educational Computing Research, 58(7), 1279–1310*. https://doi.org/10.1177/0735633120930673.

Chao, P. Y. (2016). Exploring students' computational practice, design and performance of problem solving through a visual programming environment. *Computers & Education, 95, 202–215*.

Cui, Z., Ng, O., & Jong, M. S.-Y. (2023). Integration of Computational Thinking with Mathematical Problem-based Learning: Insights on Affordances for Learning. *Educational Technology & Society, 26(2), 131-146*. https://doi.org/10.30191/ETS.202304_26(2).0010

Dagienė, V., Sentance, S., & Stupurienė, G. (2017). Developing a two-dimensional categorization system for educational tasks in informatics. *Informatica*, *28*(1), 23-44. https://doi.org/10.15388/informatica.2017.119

Dasgupta, A., & Purzer, S. (2016). No patterns in pattern recognition: A systematic literature review. *EEE Frontiers in Education Conference (FIE),* 1–3. https://doi.org/10.1109/FIE.2016.7757676

Daulay, P., & Zaman, B. (2012). PENGEMBANGAN MODEL PENELUSURAN DISKUSI TUTORIAL ONLINE MELALUI APLIKASI FAQ (FREQUENTLY ASK QUESTION). *Jurnal Pendidikan Terbuka Dan Jarak Jauh, 13*(2), 80-93. http://jurnal.ut.ac.id/index.php/jptjj/article/view/413

De Chenne, A., & Lockwood, E. (2022). A Task to connect counting processes to lists of outcomes in combinatorics. *The Journal of Mathematical Behavior, 65, 100932*. https://doi.org/10.1016/j.jmathb.2021.100932

DepEd. (2012). *K to 12 Mathematics Curriculum Guide*. K to 12 BASIC EDUCATION CURRICULUM. http://lrmds.deped.gov.ph/

Doğan, A. (2020). Algorithmic thinking in primary education. *International Journal of Progressive Education*, *16*(4), 286–301. https://doi.org/10.29329/ijpe.2020.268.18

Durak, H. (2018). The effects of using different tools in programming teaching of secondary school students on engagement, computational thinking and reflective thinking skills for problem solving. *Technology, Knowledge and Learning*, *25*(1), 179-195. https://doi.org/10.1007/s10758-018-9391-y

Fraillon, J., Ainley, J., Schulz, W., Duckworth, D., & Friedman, T. (2019). IEA international computer and information literacy study 2018 assessment framework. *Springer*.

Gay, L. R., Mills, G. E., & Airasian, P. W. (2012). Educational research: Competencies for analysis and applications. *Pearson College Division*.

Harimurti, R., Ekohariadi, E., Munoto, M., Asto B, I. G., & Winanti, E. T. (2019). Analysis of programming skills concept in developing problem solving skills. https://doi.org/10.21831/jptk.v25i1.22638

Hickmott, D., Prieto-Rodriguez, E., & Holmes, K. (2018). A Scoping review of studies on computational thinking in K–12 mathematics classrooms. *Digital Experiences in Mathematics Education, 4(1), 48–69*. https://doi.org/10.1007/s40751-017 0038-8

Hurt, T., Greenwald, E., Allan, S., Cannady, M. A., Krakowski, A., Brodsky, L., Collins, M. A., Montgomery, R., & Dorph, R. (2023). The computational thinking for science (CT-S) framework: operationalizing CT-S for K–12 science education researchers and educators. *International Journal of STEM Education*, *10*(1). https://doi.org/10.1186/s40594-022-00391-7

Intel Education. (2011). *Designing Effective Projects: Intel Teach Development Programs*. Intel Corporation.

Israel-Fishelson., R. & Hershkovitz, A. (2022). Studying interrelations of computational thinking and creativity: A scoping review (2011–2020). *Computers and Education*, 176, 104353. https://doi.org/10.1016/j.compedu.2021.104353

Jocius, R., Ian O'byrne, W., Albert, J., Joshi, D., Robinson, R., & Andrews, A. (2021). Infusing computational thinking into STEM teaching. *Technology & Society, 24*(4), 166–179. https://www.jstor.org/stable/48629253

Kalelioglu, F., & Gülbahar, Y. (2014). The effects of teaching programming via Scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education, 13(1)*, 33–50. https://doi.org/10.15388/infedu.2014.03

Kastner-Hauler, O., Tengler, K., Sabitzer, B., & Lavicza, Z. (2022). Combined effects of Block-Based Programming and Physical Computing on primary students' computational thinking skills. *Frontiers in Psychology, 13*. https://doi.org/10.3389/fpsyg.2022.875382

Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science*, *46*(5), 767–787. https://doi.org/10.1007/s11251-018-9453-5

Kwon, K., & Cheon, J. (2019). Exploring problem decomposition and program development through block-based programs. *International Journal of Computer Science Education in Schools*, *3*(1), 3-16. https://doi.org/10.21585/ijcses.v3i1.54

Lin, Y., Chen, S., Tsai, C., & Lai, Y. (2021). Exploring computational thinking skills training through augmented reality and AIoT learning. *Frontiers in Psychology*, *12*. https://doi.org/10.3389/fpsyg.2021.640115

Lu, J. & Fletcher, G.H. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin*, *41*(1), 260-264. https://doi.org/10.1145/1539024.1508959

Lye, S. Y., & Koh, J. H. (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

McCain, T. (2005). Teaching for tomorrow: Teaching content and problem-solving skills. *Corwin Press*.

McClelland, K., & Grata, L. (2018). A Review of the Importance of Computa tional Thinking in K-12. *The Tenth International Conference on Mobile, Hybrid, and On-line Learning, 32-34*.

National Research Council, Division on Engineering and Physical Sciences, Computer Science and Telecommunications Board, & Committee for the Workshops on Computational Thinking. (2011). *Report of a workshop on the pedagogical aspects of computational thinking*. National Academies Press.

National Research Council. (2013). Education for life and work: Developing transferable knowledge and skills in the 21st century. Division of Behavioral and Social Sciences and Education, Board on Science Education, Board on Testing and Assessment, Committee on Defining Deeper Learning and 21st Century Skills. *National Academies Press*.

OECD. (2013). *PISA 2012 assessment and analytical framework mathematics, reading, science, problem solving and financial literacy: Mathematics, reading, science, problem solving and financial literacy*. OECD Publishing.

Oliveri, M. E., Lawless, R., & Molloy, H. (2017). A literature review on collaborative problem solving for college and workforce readiness. *ETS Research Report Series, 2017(1), 1-27*. https://doi.org/10.1002/ets2.12133

Özpınar, İ., & Arslan, S. (2023). Teacher-based evaluation of students' problem solving skills. *International Journal of Psychology and Educational Studies*, *10*(2), 543–560. https://doi.org/10.52380/ijpes.2023.10.2.1160

Priemer, B., Eilerts, K., Filler, A., Pinkwart, N., R¨ osken-Winter, B., Tiemann, R., & Zu Belzen, A. U. (2019). A framework to foster problem-solving in STEM and computing education.

Research in Science & Technological Education, 38(1), 105-130. https://doi.org/10.1080/02635143.2019.1600490
Pršala, J. (2024). The effect of block-based programming activities on the computational thinking skills of pre-service primary school teachers. *Journal of Pedagogical Sociology and Psychology*. https://doi.org/10.33902/jpsp.202426267
Ran, H., Kasli, M., & Secada, W. (2020). A meta-analysis on computer technology intervention effects on mathematics achievement for low-performing students in K–12 classrooms. *Journal of Educational Computing Research*, *59*(1), 119–153.
Sengupta, P., Dickes, A., & Farris, A. (2018). Toward a phenomenology of computational thinking in STEM education. In *Springer eBooks* (pp. 49–72). https://doi.org/10.1007/978-3-319-93566-9_4
Sentance, S., & Csizmadia, A. (2016). 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
Sumartini, T. S. (2018). Peningkatan Kemampuan Pemecahan Masalah Matematis Siswa melalui Pembelajaran Berbasis Masalah. Mosharafa: *Jurnal Pendidikan Matematika, 5(2), 148–158*. https://doi.org/10.31980/mosharafa.v5i2.270
Taylor, K. (2018). Computational Thinking Framework. Let's Talk Science. Lon don, Ontario, Canada.
Totan, H. N., & Korucu, A. T. (2023). The Effect of Block Based Coding Education on the Students' Attitudes about the Secondary School Students' Computational Learning Skills and Coding Learning: Blocky Sample. *Participatory Educational Research*, *10*(1), 443–461. https://doi.org/10.17275/per.23.24.10.1
Wei X., Lin L., Meng N., Tan W., Kong Kinshuk S. C. (2021). The effectiveness of partial pair programming on elementary school students' computational thinking skills and self-efficacy. *Computers and Education,* 160, 104023. https://doi.org/10.1016/j.compedu.2020.104023
Wiggins, G. P., & McTighe, J. (2011). *The understanding by design guide to creating high-quality units*. ASCD.
Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49(3), 33-35*. https://doi.org/10.1145/1118178.1118215
Wu, T., Asmara, A., Huang, Y., & Hapsari, I. P. (2024). Identification of Problem-Solving Techniques in Computational Thinking Studies: Systematic Literature review. *SAGE Open*, *14*(2). https://doi.org/10.1177/21582440241249897
Ye, H., Liang, B., Ng, O.-L., & Chai, C. S. (2023). Integration of computational thinking in K-12 mathematics education: A systematic review on CT-based mathematics instruction and student learning. *International Journal of STEM Education, 10(1)*. https://doi.org/10.1186/s40594-023-00396-w
Yu, Q., Yu, K., & Li, B. (2024). Effects of Block-Based Visual Programming on K-12 students' learning outcomes. *Journal of Educational Computing Research*. https://doi.org/10.1177/07356331241293163
Yusoff, K. M., Sahari, N., Siti, T., & Mohd, N. (2020). Analysis on the requirements of computational thinking skills to overcome the difficulties in learning programming. *International Journal of Advanced Computer Science and Applications*, *11*(3). https://doi.org/10.14569/ijacsa.2020.0110329
Zeng, Y., Yang, W., & Bautista, A. (2023). Teaching programming and computational thinking in early childhood education: a case study of content knowledge and pedagogical knowledge. *Frontiers in Psychology, 14*. https://doi.org/10.3389/fpsyg.2023.1252718