ORIGINAL ARTICLE

The Effect of Block Coding (Scratch) Activities Integrated into the 5E Learning Model in Science Teaching on Students' Computational Thinking Skills and Programming Self-Efficacy

Abdullah Koray,[1] Eda Bilgin[2]

1 *Zonguldak Bülent Ecevit University, Kdz. Ereğli, Zonguldak, Turkey*
2 *Cumhurbaşkanı Süleyman Demirel, Imam Hatip Middle School, Zonguldak, Turkey*

***Abstract**: This study was carried out to determine the effect of Scratch-based coding applications integrated into the 5E learning model used in science teaching on students' computational thinking skills and self-efficacy towards block-based programming. In addition, students' perceptions of the activity were measured after each Scratch activity, which was applied at different stages of the course and with different difficulty. The study employed the pretest-posttest control group less design, one of the quasi-experimental methods. The study sample consist of 22 6th grade students attending a public school in Turkey located in a district center in the Eastern Black Sea region. The study was carried out in a five-week period in the 2022-2023 academic years. Computational thinking scale and robotics attitude scale, self-efficacy perception scale related to block-based programming and activity perception scale were used as data collection tools. The data were analyzed using the dependent samples t-test. The findings suggest that computational thinking skills level of students and their self-efficacy perception related to block-based programming increased significantly with the Scratch-based activities integrated into 5E learning model applied in science subjects. In addition, students have positive attitudes towards these activities. Thus, it is recommended to apply Scratch-based coding applications in teaching science subjects.*

Science Insights Education Frontiers 2023; 18(1):2825-2845.
Doi: 10.15354/sief.23.or410

How to Cite: Koray, A., & Bilgin, E. (2023). The effect of block coding (Scratch) activities integrated into the 5E learning model in science teaching on students' computational thinking skills and programming self-efficacy. Science Insights Education Frontiers, 18(1):2825-2845.

Keywords: *Science Teaching, Scratch, 5E Learning Model, Computational Thinking, Self-Efficacy*

About the Authors: *Abdullah Koray, Ereğli Education Faculty, Zonguldak Bülent Ecevit University, Kdz. Ereğli, Zonguldak, Turkey, E-mail: korayabdullah@gmail.com, ORCID: https://orcid.org/0000-0002-2972-1317*

Eda Bilgin, Cumhurbaşkanı Süleyman Demirel, Imam Hatip Middle School, Zonguldak, Turkey, E-mail: edaozturk6767@gmail.com, ORCID: https://orcid.org/0000-0002-6154-0778

Correspondence to: *Abdullah Koray at Zonguldak Bülent Ecevit University of Turkey.*

Conflict of Interests: *None*

Introduction

FOR science education to be more effective, it is a very common approach to create teaching environments that allow students to learn by doing and experiencing. Constructivist approach that focuses on active learning and the creation of knowledge by connecting prior knowledge with newly encountered knowledge (Appleton, 1997; Cakir, 2008; Copley, 1992; Hand & Treagust, 1991) becomes an essential tool used in science education. It also helps to facilitate the learning process by using learning cycle models. One such model is the 5E learning model. Based on the relevant literature, the 5E learning model can be described as one of the most useful models of constructivist learning theory in the teaching process (Çoruhlu, 2013). The 5E model, developed by Bybee et al. (2006), is named after the English initials of the model (Engage, Explore, Explain, Elaborate, and Evaluate). In these stages, all the steps of a learning-teaching process that is based on research and inquiry are covered, and the roles of teacher and student are expressed in a rich way structurally and pedagogically throughout the process. When the relevant literature is examined, it is seen that there are very few studies in which science education is designed by integrating technology into the 5E learning model. However, there are studies in which technologies such as augmented reality (Abdusselam et al., 2018), robotics (Cakir & Guven, 2019), interactive simulation (Lye et al., 2014) and mobile learning (Celik et al., 2020; Lai et al., 2015) are adapted to the science teaching process through the 5E learning model.

Technology and Education

Technology can take many forms in science education, from online simulations to interactive whiteboards, or from virtual labs to educational software (Fraser, 2023; Özbek & Uslu, 2021). The most important reason for the widespread use of technology in science education is that it can make abstract or complex concepts more concrete and understandable for students. For example, simulations and visualizations can be used to explore scientific phenomena that may be difficult to observe in the real world or to model complex systems and processes that are difficult to replicate in a classroom setting (Oliveira et al., 2019). Students can design and conduct experiments, collect and analyze data using technology. Thus, they experience applied, inquiry-based learning environments using digital tools (Unlu & Dokme, 2020). Using technology, students not only learn more deeply about science-related topics, but also develop skills such as scientific inquiry, problem solving and critical thinking (Kim & Hannafin, 2011).

Although the use of technology in educational processes has a certain cost and there are difficulties such as the educational needs of the educators who will use it, it can be said that technology is one of the most powerful tools students must benefit for life. (Hew & Brush, 2007; Hsu, 2016). The use of technology in science education has become increasingly prevalent in recent years, with computers being one of the most widely used tools. Computers have become an important tool in modern science education, considering their easily accessible and increasing usage areas (Jong et al., 2013). As a result, educators are increasingly using computers to enhance their teaching and students' learning experiences in science.

Coding and Education

In recent years, there has been a movement led by Code.org, a non-profit organization aimed at promoting the teaching of programming in schools around the world. This movement primarily focuses on filling the shortage of IT professionals that exist today and is expected to increase in the coming years (Moreno-León & Robles, 2016). Academics from both education and science circles state that data on the potential benefits of children learning to code is important, regardless of their future professional field. It aims not only to teach coding itself, but also to serve as a tool to develop other skills, improve learning outcomes, and enhances student motivation (Resnick, 2013). The idea of coding for learning was first introduced by Seymour Papert in the 1970s. Logo programming language has been developed for children to create games, music, and repetitive drawings on computers (Papert & Solomon, 1971). While programming was taught in many schools in the 1980s, it vanished from the educational landscape in the 1990s because it was not integrated into subjects beyond mathematics and physics, and classroom activities didn't appeal to students' interests (Kafai & Bruke, 2013).

Another theory that supports teaching programming by integrating it into other courses is related to computational thinking. Computational Thinking (CT) refers to the thought processes involved in formulating problems so their solutions can be represented as computational steps and algorithms (Aho, 2012). Although the concept of computational thinking dates to the 1950s (Tedre & Denning, 2016), it was introduced to education world by Jeannette Wing's article published in Communications of the ACM in 2006 (Wing, 2006; Grover & Pea, 2013). The article suggested that computational thinking is a fundamental skill for everyone, not just computer scientists, and argued for the importance of integrating computational ideas into other subjects in school. The trial also said that by learning to think computationally, children would do better at many everyday tasks.

Visually Based Programming Languages and Scratch

Although it is a very effective process for students to learn a subject by coding, it requires students and branch teachers to have coding knowledge to be applied in the classroom environment. Learning text-based programming languages is a very challenging process. Block coding, on the other hand, is a visual programming language that uses pre-written blocks or stacks of code that can be dragged and dropped to create programs. It is an excellent tool for coding beginners as it offers several advantages over traditional text-based scripting languages (Resnick et al., 2009).

In recent years, visual-based programming languages such as Alice, Code, and Scratch have rekindled interest in programming among educators. Scratch permits the creation of many different types of projects, so learners with varying interests and learning styles can express themselves through programming. As a result, teaching children and teens extracurricular activities, summer camps, and classroom programming has become more common. However, for this interest in coding in schools to continue, evidence of the educational impact of programming is needed (Kafai & Bruke, 2013).

One of the most widely used block-based programming languages is Scratch. Scratch is a free block-based programming language developed by MIT Media Lab that allows users to create interactive stories, animations, and games. In educational processes, it is widely used in teaching computer science and computational thinking concepts to students of all ages (Resnick et al., 2009). In recent years, Scratch has been recognized as an effective tool for teaching science concepts as it allows students to model scientific phenomena, simulate experiments, and explore complex systems in an interactive and engaging way. By using Scratch to create interactive simulations, students can develop a deeper understanding of scientific concepts and processes, and gain valuable experience in experimental design, data analysis, and scientific reasoning (Blikstein et al., 2013). There are many resources available for educators who are interested in using Scratch in science education, including online tutorials, lesson plans, and project ideas. One great place to start is the ScratchEd website, which offers a variety of resources and support for educators who are using Scratch in their teaching practice (Ogegbo & Ramnarain, 2022).

Using Scratch in Education

There have been many studies reporting both positive and non-significant results regarding the use of Scratch in education (Talan, 2020). Shamir, Kocherovsky and Chung (2019) used Scratch applications in mathematics and computer teaching in their study with 7th grade students. According to the results of the study, the mathematics and computer ability of the students in the group using Scratch applications improved significantly compared to those who do not. Another result obtained from this study, which was applied in STEM classrooms, shows that students' interest in STEM increased sfter using this application. In another study on STEM education, which aims to gather science, technology, engineering, and mathematics applications under one roof, it has been suggested that Scrath can be an alternative to very expensive robot sets (Yamamori, 2019). The researcher suggested that the time spent on assembling the small parts of the robot can be used more efficiently with Scratch for the same purpose, especially when the lesson hours are limited. Aiming to investigate the effectiveness of scratch programming in teaching science to 5th grade students, Lai and Lai (2012) asked students to program while teaching the "Observation of the Sun" and "The Weather has Changed" units. The outcomes of this research revealed that students' performance in logical thinking and problem-solving improved after using Scratch Programming.

In some studies where Scratch is used in the teaching process, there are also cases where significant differences do not occur. For example, Momcilovic (2020) used Scratch applications to teach geometry subjects in mathematics class and obtained positive results. The results of the study show that the academic performance of the students, who use both three-dimensional modules and scratch applications, has increased. There was no statistically significant difference between the two groups. In another study Jiang and Li (2021) aimed to analyze the effects of Scratch language learning on the computational thinking skills (creativity, algorithmic thinking, cooperativity, critical thinking, and problem solving) of primary school students. While the research findings show that there is a significant difference in creativity, collaboration, and critical thinking skills, it is seen that learning with Scratch does not cause a significant difference in the problem solving and algorithmic thinking skills of primary school students.

With the inclusion of block-based programming (BBP) education in the curriculum in Turkey simultaneously with many countries, studies are still being carried out to make programming education more effective. When the relevant literature is examined, studies that emphasize the relationship between individuals' self-efficacy perception and programming performance draw attention (Aşkar & Davenport, 2009; Yükseltürk & Altıok, 2016; Akar & Altun, 2017). Albert Bandura (1977) defined self-efficacy perceptions as individuals' judgments about how well they could perform the actions necessary to cope with possible situations. Self-efficacy perceptions related to BBP refer to students' confidence in their ability to move the Scratch's puppets with code, and this perception is important in teaching programming (Altun & Kasalak, 2018).

As a result, when the studies in the field are examined, the use of both the 5E teaching model and Scratch in the education process has positive effects in general. However, there are few studies where both (5E teaching model and Scratch) are used together. In addition, the related literature lacks data on the reactions of students to coding applications with different difficulties during these applications. Thus, the aim of this study is to examine the effects of Scratch applications with different difficulties on students by applying them to different steps of the 5E teaching model. The effect of Scratch-based activities integrated into the 5E teaching model in science teaching on students' computational thinking skills and self-efficacy perception regarding block-based programming was examined. In addition, students' perceptions of the activity were measured after each Scratch activity, which was applied at different stages of the course and with different difficulty.

Accordingly, the following research questions were examined.

1. *What is the effect of Scratch-based activities on students' computational thinking skills levels?*
2. *What is the effect of Scratch-based activities on students' levels of self-efficacy perception regarding block-based programming?*
3. *What is the experience of students regarding Scratch-based activities?*

Method

Experimental research method, one of the quantitative research methods, was employed in the study. The pretest-posttest control groupless design, a quasi-experimental research method, was utilized. Experimental studies aim to test the effect of the independent variables on the dependent variable. The main purpose of these studies is to reveal the cause-and-effect relationship between the variables. For this purpose, the independent variable is manipulated and the variables that may affect the dependent variable are controlled (Büyüköztürk et al., 2018; Fraenkel et al., 2012). In the study, the independent variable whose effect on the dependent variables of "computational thinking skills" and "self-efficacy perception related to block-based programming" was examined is the "Scratch-based activities integrated into the 5E teaching model". The design of the research was given in **Table 1**.

Study Group

The study was carried out with 6th grade students who were attending a public school in Turkey located in a district center in the Eastern Black Sea region in the 2022-2023 academic years. Two classrooms were included in the study. The reason for choosing the study group students is that the relevant school has a computer laboratory and before the application, the students get basic coding and Scratch training in the "Information technologies and software" course. In this direction, the study group was determined by purposive sampling method and the study was carried out with the participation of 22 6th grade students (12 females and 10 males).

Table 1. Standard Notation of Study Design.

Pretest	← APS (After Each Activity) → Application	Posttest
CTS1 SPSRBP 1	X1	CTS2 SPSRBP 2

X1: Scratch-based activity applications.
CTS: Computational Thinking Scale
SPSRBP: Self-Efficacy Perception Scale Related to Block-Based Programming
APS: Activity Perception Scale

Data collection Tools

Computational Thinking Scale (CTS)

The "Computational Thinking Scale", which was developed by Korkmaz, Çakır, and Özden (2015) first for university students and then adapted to the secondary school level, was used to measure students' computational thinking skills. The scale is a five-point Likert type scale consisting of 22 items with 5 factors: creativity (4 items), problem solving (6 items), algorithmic thinking (4 items), collaboration (4item) and critical thinking (4 items). As a result of the confirmatory factor analysis of the scale using the maximum likelihood technique, the regression values of the items varied between 0.507 and 0.872. Item test correlation coefficients ranged from 0.655 to 0.862. To calculate the reliability of CTS, internal reliability analyzes were performed on the data and the Cronbach Alpha reliability coefficient of the scale was determined as 0.809.

Self-Efficacy Perception Scale Related to Block-Based Programming (SPSRBP)

The "Self-Efficacy Perception Scale Related to Block-Based Programming" (SPSRBP) developed by (Altun & Kasalak, 2018) was utilised to measure the students' self-efficacy perceptions related to Block-Based Programming. The scale consists of 12 items and 5-point Likert type. The scale has two sub-dimensions: "simple block-based programming tasks" (5 items) and "complex block-based programming tasks" (7 items). The overall Cronbach's Alpha reliability coefficient of the scale was calculated as 0.893.

Activity Perception Scale (APS)

The original version of the Activity Perception Scale, which was used to determine student experiences regarding coding activities, was developed by Deci et al. (1994) for uninteresting (boring) computer tasks. The adaptation of the perception of effectiveness scale into Turkish was carried out by Kasalak (2017). As a result of the evaluations, it was predicted that the translation of only 11 items of the 25-item scale would be understood correctly by the students in Turkey due to the differences in cultures and education systems, and the scale was finalized. Student experiences are evaluated in terms of finding activities enjoyable, contribution of activities to personal development, willingness to do activities and finding activities interesting.

Research Process

Before the application, the students were given basic Scratch training within the scope of the “Program Solving and Programming” unit of the “Information Technologies and Software” course. Before starting the applications in the science course, the Computational Thinking Scale (CTS) and Self-Efficacy Perception Scale Related to Block-Based Programming (SPSRBP) were administered to the students as a pre-test.

In the present study, the unit “Force and Motion” was selected as it was compatible with the Scratch activities to be done. Before starting the applications for the learning objectives, teacher instructions and student worksheets were developed. At one stage of the lesson plans developed in accordance with the 5E learning model (Engage, Explore, Explain, Elaborate or Evaluate), the Scratch activity was included. The developed Scratch activities were related to the learning objectives of the 6th grade science lesson “Force and Motion” unit.

In the five-week practice, the Scratch activities according to the topics and the phase of the lesson to be used in these activities are as follows.

Week 1 (Engage stage): The first Scratch activity was used in the introduction to the topic “Force and its properties”. Students, who used the Scratch application only in the “Information Technologies and Software” class before, used it for the first time in the science class. The students were introduced to the concepts of “application point”, “direction”, “direction and magnitude” of the force by running the previously prepared program. In the interactive Scratch application, firstly, the scenes with rotating or moving objects with the effect of the applied force were shown to the students. Afterwards, they were asked to predict the direction of motion of these objects under force (**Figure 1**). The answers given are discussed and explained in the exploration and explain stages of the course.

Figure 1. First Week Scratch App.

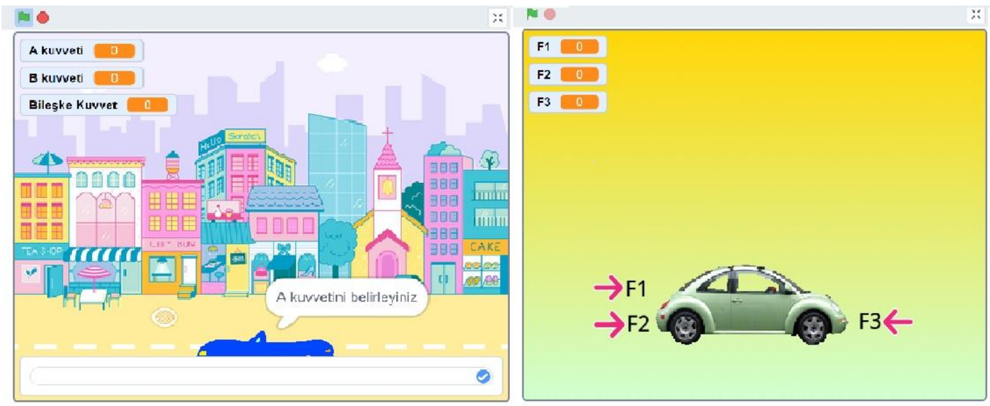

Figure 2. Second Week Scratch App.

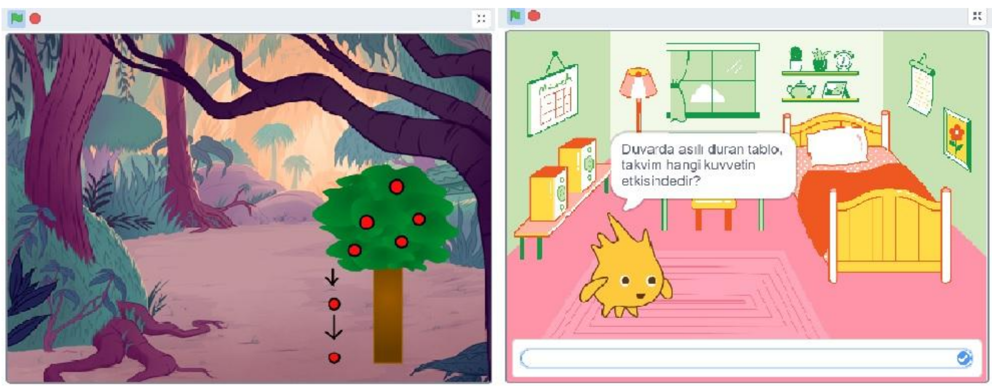

Figure 3. Third Week Scratch App.

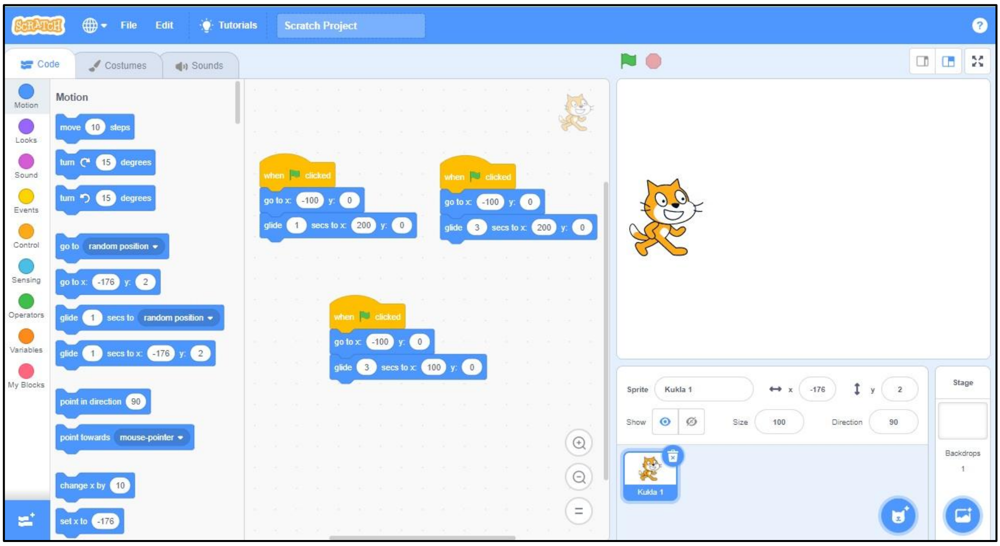

Figure 4. Fourth Week Scratch App.

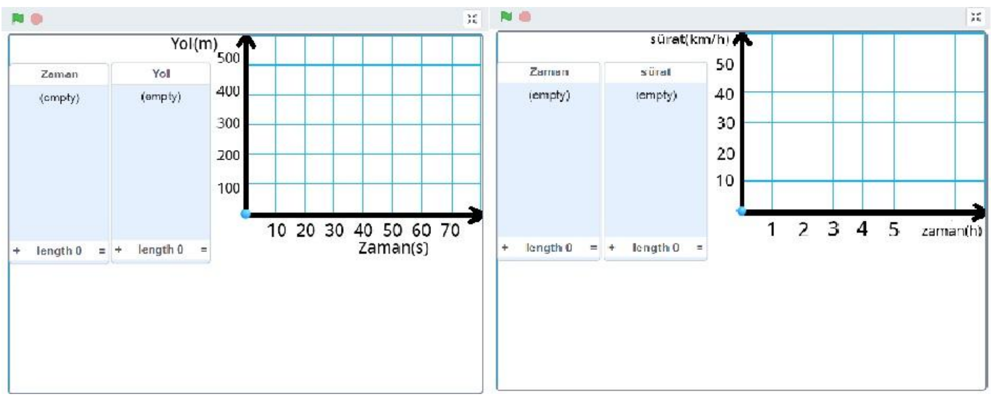

Figure 5. Fifth Week Scratch App.

Week 2 (Explore phase): The second Scratch activity was used in the explore phase of the "Result force" topic. The interactive Scratch activity was implemented in two stages. In the first stage, two forces acting on an object were given in opposite directions, and students were asked to enter magnitude values for both forces and estimate the resultant force. In the second stage, the number of forces was increased to three, and similarly, students were asked to enter force values and estimate the resultant force (**Figure 2**). The answers given were discussed in the explain phase of the lesson.

Week 3 (Explain stage): The third Scratch activity was used in the explain stage of the “Balanced and unbalanced forces” topic, which is a special case of the “Result force” topic. In the engage phase of the third week, students were given a rope planting activity to make them realize the balance of forces. In the Explore step, the balance of forces is explained, and in the explain step, the concepts of “balanced and unbalanced force” are defined by adapting them to some examples around us by using the Scratch activity. In the Scratch application, first some examples were given, such as an apple standing on a tree and falling, and then students were asked to adapt the concept to daily life (for example, a painting hanging on the wall) (**Figure 3**).

Week 4 (Elaborate stage): The fourth Scratch activity was used in the elaborate stage of “Constant velocity movement”. In the first three stages of the fourth week, students were taught the concept of speed. In the elaborate step, students were asked to investigate how speed depends on the distance travelled and time. For this purpose, students were asked to use the motion commands of the Scratch platform. The selected sprite was asked to travel a specified path at different times, and then at the same time, different paths (**Figure 4**). For both cases, students were asked to record their observations with different values.

Week 5 (Evaluate phase): The fifth Scratch activity was used in the evaluate phase of “Path-time and velocity-time graphs of constant velocity motion”. In the first four phases of the fifth week, students were taught how to draw and interpret graphs of constant velocity motion. In the Evaluate stage, the students were given a part of the path-time and velocity-time table values of a motion and were asked to complete the missing data and draw graphs with the Scratch program, which were prepared before, if the motion would continue with a constant velocity **(Figure 5**). In the prepared program, it was requested to enter the path-time and speed-time data pairs to be used for drawing. For the graphs to be created with a total of 5 data pairs, the first data pair was given completely, while one of the next data pairs was given incomplete and the students were asked to guess. After each data pair is entered, the relevant point is marked in the graphic area and finally the graph is completed.

The applications designed for the unit learning objectives continued for five weeks (a total of 20 periods, four periods a week, as stipulated by the curriculum. One of the researchers was the class teacher and carried out the application, while the other researcher participated as an observer. In the week following the completion of the activities, the CTS and the SPSRBP were administered to students as a posttest. APS was applied to the students after each activity for five weeks. The study continued for a total of seven weeks, with the administration of the pretest and posttest (excluding the basic Scartch training).

Data Analysis

A statistical software program SPSS 20 was used in the analysis of the data. Appropriate statistical methods were tried to be determined by investigating the suitability of the data to the normal distribution. Since the number of participants in each group was less than 50, Shapiro-Wilk test was used for the assumption of normality (Mishra et al., 2019). As a result of the analysis, it was decided to use parametric tests because the data sets had a normal distribution ($p > 0.05$). In this context, to test whether there was a significant difference between the variables, the collected data were analyzed using the t-test for dependent groups. For interpretation, the significance level for the hypothesis tests was set to 0.05.

Findings

In this section, the findings are presented, and the data are explained in tables. Findings, interpretations, and tables are organized in order of the study research questions.

The results of the dependent samples t-test analysis regarding the first research question are given in **Table 2**.

When **Table 2** is examined, it is seen that the students' scores from the Computational Thinking scale in Creativity, Algorithmic thinking, Collaboration, Critical thinking and Problem solving dimensions and Computational Thinking total scores increased after they practiced Scratch supported activities:

$$\Delta\bar{X}_{(\text{Creativity})} = +0.91$$
$$\Delta\bar{X}_{(\text{Alg. thinking})} = +0.69$$
$$\Delta\bar{X}_{(\text{Collaboration})} = +0.78$$
$$\Delta\bar{X}_{(\text{Crit. thinking})} = +0.54$$
$$\Delta\bar{X}_{(\text{Prob. solving})} = +0.91$$
$$\Delta\bar{X}_{(\text{Total CTS})} = +3.81\,)$$

While this increase is statistically significant for Creativity ($t(21) = -2.83$, $p < 0.05$) and Collaboration ($t(21) = -3.04$, $p < 0.05$) dimensions, it is not significant for Algorithmic thinking ($t(21) = -1.88$, $p > 0.05$), Critical thinking ($t(21) = -1.63$, $p > 0.05$) and Problem solving ($t(21) = -1.39$, $p > 0.05$). However, the increase in Computational Thinking total scores is statistically significant ($t(21) = -4.16$, $p < 0.05$). The Cohen d effect sizes of these determined significant differences were found to be medium for Creativity ($d = 0.60$) and Collaboration ($d = 0.65$) and large for total Computational Thinking ($d = 0.89$).

The results of the dependent samples t-test analysis regarding the second research question are given in **Table 3**.

Table 2. Dependent Samples T-Test Results Related to Computational Thinking Scale (CTS).

Dimensions of CTS	N	$\bar{X}$	SD	df	t	p
Creativity (Pre) Creativity (Post)	22	14.14 15.05	2.10 2.06	21	-2.83	0.010
Algorithmic thinking (Pre) Algorithmic thinking (Post)	22	11.36 12.05	2.11 2.08	21	-1.88	0.074
Collaboration (Pre) Collaboration (Post)	22	14.86 15.64	3.26 2.63	21	-3.04	0.006
Critical thinking (Pre) Critical thinking (Post)	22	12.05 12.59	2.48 2.21	21	-1.63	0.117
Problem solving (Pre) Problem solving (Post)	22	16.55 17.46	2.43 2.44	21	-1.39	0.179
Computational Thinking (Pre) Computational Thinking (Post) (Total score of the scale)	22	68.96 72.77	7.25 6.00	21	-4.16	0.000

$p < 0.05$.

Table 3. Dependent Samples T-Test Results Related to Self-Efficacy Perception Scale Related to Block-Based Programming (SPSRBP).

Dimensions of SPSRBP	N	$\bar{X}$	SD	df	t	p
Simple tasks (Pre) Simple tasks (Post)	22	15.36 16.73	1.84 2.05	21	-8.80	0.000
Complex tasks (Pre) Complex tasks (Post)	22	15.36 16.73	2.38 2.96	21	-4.57	0.000
SPSRBP (Pre) SPSRBP (Post) (Total score of the scale)	22	30.73 33.45	3.93 4.89	21	-7.09	0.000

$p < 0.05$

When **Table 3** is examined, the scores of the students in the simple and complex tasks dimensions from SPSRBP and the total scores of the scale are higher after the Scratch-supported activity applications than before, and they are statistically significant ($t(21)_{(\text{Simple tasks})} = -8.80$, $t(21)_{(\text{Complex tasks})} = -4.57$, $t(21)_{(\text{Total SPSRBP})} = -7.09$, $p < 0.05$). The Cohen d effect sizes of these determined differences were found to be large for Complex tasks (d = 0.97) and very large for Simple tasks (d = 1.88) and total SPSRBP (d = 1.51).

The results of the analysis regarding the tirtd research question are given in **Table 4**.

When the average values for the 1st and 3rd items related to “Finding activities is enjoyable” in **Table 4** are examined, it is seen that the average scores given are the lowest 4.00 and the highest 4.76. The average scores given to item 6, which contains a negative statement, ranged from 1.18 to 1.59. These values show that student perception levels of finding Scratch

Table 4. Average Values of Students' Activity Perception Scores Related to Scratch Activities.

		Act. 1	Act. 2	Act. 3	Act. 4	Act. 5
Finding activities enjoyable	It was fun doing this activity. (Item 1)	4.23	4.56	4.64	4.46	4.59
	I had a lot of fun doing this activity. (Item 3)	4.00	4.18	4.76	4.27	4.59
	I think it was such a boring activity. (Item 6)	1.46	1.46	1.27	1.59	1.18
Contribution of activities to personal development	I believe this activity is important for my development. (Item 2)	3.73	4.14	4.09	4.05	4.14
	I think this was a really important activity. (Item 4)	4.00	3.91	4.41	3.86	4.09
	I would like to do this activity again Because I think it is useful. (Item 7)	4.05	4.23	4.09	3.77	3.91
	I believe that doing this activity can be beneficial for me. (Item 8)	4.18	4.18	4.46	4.05	4.09
	I believe this activity can help me get better at school. (Item 9)	3.46	4.05	4.09	3.96	3.86
	I would like to do it again because there are some things that this event contributed to me. (Item 11)	3.73	4.10	4.27	4.09	4.05
Willingness to do activities and finding activities interesting	I did this activity because I wanted to do it. (Item 5)	3.41	3.64	3.50	3.50	3.96
	I thought this was a very interesting event. (Item 10)	3.68	3.68	4.00	3.91	4.00
	Total	43.02	45.22	46.99	44.19	46.10

activities fun are quite high, and their perception levels of finding them boring are quite low. In the same table, it is seen that the average score given by the Scratch activities to the items related to the activity perceptions of the students regarding their personal development is between 3.73 and 4.46. Finally, when the table values are examined, it is seen that the average score given by the students to the item prepared to determine their perceptions of doing activities willingly varies between 4.41 and 4.00. From this, it is understood that the students do not have a perception that they do the activities because they want to do it, or that they must do the activity even if they do not want to.

Results and Discussion

The first result of the research is that Scratch-based activities increase computational thinking skills of students. The increase in the total score of the scale and the increase in the creativity and cooperation factors are statistically significant. Even though the scores on algorithmic thinking, critical thinking and problem-solving factors increased, it was not statistically significant. When we look at the literature, it is possible to come across studies that use Scratch activities as course material. Rodríguez-Martínez, González-Calero and Sáez-López (2020) examined the effect of Scratch activities used in mathematics lessons on gaining mathematical concepts and developing computational thinking skills. Their results were in line with Scratch's development of both. The adaptation of Scratch applications to the mathematics lesson was applied not only at the k12 level (Benton et al., 2018; Foerster, 2016; Shamir et al., 2019; Vinayakumar et al., 2018) but also at the higher education level (Molina-Ayuso et al., 2022), and generally positive results were obtained. Adaptation studies of Scratch applications to the science curriculum are much less than that in mathematics (Hacıoğlu & Dönmez Usta, 2020; Silva et al., 2020; Yamamori, 2019).

In a study conducted with 5th grade students, Jiang and Li (2021) taught Scratch within the scope of "Basics of Information Technologies" course and its effect on computational thinking skills was examined. The research findings indicate that there was a significant difference in the skills of creativity, cooperatives, and critical thinking. However, in this study, Scratch learning did not cause any significant differences in the problem-solving and algorithmic thinking skills of students. These results are quite consistent with our findings. In both studies, while Scratch applications were effective on students' creativity and cooperation skills, they did not have a significant effect on problem-solving and algorithmic thinking skills.

The second result of the research is that Scratch-based activities increase self-efficacy perception of students related to block-based programming. The increase in all sub-dimensions (simple and complex tasks) together with the increase in the scale total score is statistically significant. Looking at the literature, no research has been found that examines the effect of Scratch activities adapted to science lessons on students' self-efficacy perception towards Block-based programming. However, there are studies examining the effect of Scratch or Scratch-based robotic coding training on self-efficacy perception towards block-based programming (Buyukkarci & Taslidere, 2021; Durak et al., 2019; Güleryüz, 2022). The related literature suggests that training increased the perception of self-efficacy towards block-based programming. Coşkunserçe (2023), in his study with sixth grade students, compared Scratch and Scratch-based robotic coding (mBlock) trainings in terms of increasing self-efficacy perception towards block-based programming. According to the findings, Scratch-based robotic coding (mBlock) training is more effective. Another study examining how the size of the study groups affects self-efficacy versus coding was conducted by Arslan and İşbulan (2021). According to the results of this research, in which the Scratch programming platform was used, there is no effect on self-efficacy related to block-based programming if the studies are done individually or as a group.

The findings of the study suggest that students have positive attitudes towards these activities. When total scores for each activity in the data in **Table 4** are examined, it is seen that the scores gradually increase in the first three activities, and there is a decrease after the fourth activity. In the fourth activity, instead of using a previously prepared program, students were asked to write the codes themselves. This situation caused the students to have difficulties. Thus, this may have negatively affected their perceptions of the activities. After the fifth activity, an increase was detected in the total scores again. Although challenging tasks negatively affect perceptions of students for a short time, they are thought to have positive effects in the long run.

As a result, it is known that block-based programming education is a very effective method for primary and secondary school students who have no previous programming experience (Resnick et al., 2009). As mentioned in the introduction, considering the contributions of coding education to students, it has been included in educational environments since the 1970s, but the desired success could not be achieved because it did not attract the attention of students. It is thought that coding education can be more successful by integrating it into courses such as mathematics and science. Considering the results of this study we conducted, the use of coding activities in science lessons can be a very effective method for overcoming the problems encountered in the 1970s.

Recommendations

Assuming that basic education about block-based programming is given in courses such as information technologies, it is thought that it would be appropriate to use coding activities in science classes to reinforce and improve students' coding skills. For this purpose, the 5E teaching model, which is used effectively by many teachers, can be used. Since the 5E teaching model allows Scratch activities to be used in different ways, it offers a variety of usage. First, we recommend that students use ready-made Scratch activities in their science lessons. It will be beneficial to complicate the activities by encouraging students to intervene in the codes over time in accordance with the level of students' programming skills. In case of positive results, it will be appropriate for students to code their own programs.

In this study, data on computational thinking and self-efficacy were collected. It would be useful to examine the effect of including coding activities in science lessons on different skills and achievements. In addition, this adaptation can be tried with different teaching models such as problem-based, project-based, or cooperative learning.

Limitations

This study is limited to the "Force and Motion" unit and was carried out over a five-week period excluding the pre-test and post-test applications. The study was carried out with 22 students and the students encountered Scratch-based applications for the first time in the science lesson.

References

Abdusselam, M. S., Kilis, S., Sahin Cakir, C., & Abdusselam, Z. (2018). Examining microscopic organisms under augmented reality microscope: A 5E learning model lesson. *Science Activities*, 55(1-2):68-74. DOI: https://doi.org/10.1080/00368121.2018.1517717

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal*, 55(7):832-835. DOI: https://doi.org/10.1093/comjnl/bxs074

Akar, S. G. M., & Altun, A. (2017). Individual differences in learning computer programming: A social cognitive approach. Contemporary *Educational Technology*, 8(3):195-213. Available at: https://dergipark.org.tr/en/pub/cet/issue/30468/329156

Appleton, K. (1997). Analysis and description of students' learning during science classes using a constructivist‐based model. *Journal of Research in Science Teaching: the Official Journal of the National Association for Research in Science Teaching*, 34(3):303-318. DOI: https://doi.org/10.1002/(SICI)1098-2736(199703)34:3<303::AID-TEA6>3.0.CO;2-W

Arslan, E., & Isbulan, O. (2021). The Effect of Individual and Group Learning on Block-Based Programming Self-Efficacy and Robotic Programming Attitudes of Secondary School Students. *Malaysian Online Journal of Educational Technology*, 9(1):108-121. DOI: http://dx.doi.org/10.17220/mojet.2021.9.1.249

Askar, P., & Davenport, D. (2009). An investigation of factors related to self-efficacy for Java programming among engineering students. *TOJET: The Turkish Online Journal of Educational Technology*, 8(1):26-32. Available at: http://files.eric.ed.gov/fulltext/ED503900.pdf

Bandura, A. (1977). Self-efficacy: Toward a unifying theory of behavioral change. *Psychological Review*, 84(2):191-215. DOI: https://doi.org/10.1037/0033-295X.84.2.191

Benton, L., Saunders, P., Kalas, I., Hoyles, C., & Noss, R. (2018). Designing for learning mathematics through programming: A case study of pupils engaging with place value. *International Journal of Child-Computer Interaction*, 16(2018):68-76. DOI: https://doi.org/10.1016/j.ijcci.2017.12.004

Blikstein, P., Walter-Herrmann, J., & Büching, C. (2013). FabLabs: Of machines, makers and inventors. Fab Lab-Of Machines, Makers and Inventors; Julia Walter-Herrmann & Corinne Büching, Bielefeld, ISBN: 978-3837623826. pp. 203-233.

Buyukkarci, A., and Taslidere, E. (2021). the effect of coding education on students' efficiency and scratch achievements. *i-manager's Journal of Educational Technology*, 18(2):63-74. DOI: https://doi.org/10.26634/jet.18.2.17970

Büyüköztürk, Ş., Kılıç Çakmak, E., Akgün, Ö. E., Karadeniz, Ş., & Demirel, F. (2018). Eğitimde bilimsel araştırma yöntemleri. Ankara: Pegem Akademi. ISBN: 978-6052419649. p:202.

Bybee, R. W., Taylor, J. A., Gardner, A., Van

Scotter, P., Powell, J. C., Westbrook, A., & Landes, N. (2006). The BSCS 5E instructional model: Origins and effectiveness. Colorado Springs, Co: BSCS, 5, 88-98. Available at: https://fremonths.org/ourpages/auto/2008/5/11/1210522036057/bscs5efullreport2006.pdf

Cakir, M. (2008). Constructivist approaches to learning in science and their implications for science pedagogy: A literature review. *International Journal of Environmental and Science Education*, 3(4):193-206.

Celik, C., Guven, G., & Kozcu Cakir, N. (2020). Integration of mobile augmented reality (MAR) applications into biology laboratory: Anatomic structure of the heart. *Research in Learning Technology*, 28(2020):1-11. DOI: https://doi.org/10.25304/rlt.v28.2355

Copley, J. (1992). The integration of teacher education and technology; a constructivist model. In D. Carey., R. Carey., D. Willis & S. Willis (Eds.), Technology and teacher education. AACE. pp. 681.

Coşkunserçe, O. (2023). Comparing the use of block‐based and robotprogramming in introductory programmingeducation: Effects on perceptions of programmingself‐efficacy, Computer Applications in Engineering Education. (Early View). DOI: https://doi.org/10.1002/cae.2263722

Çoruhlu, T. Ş. (2013). Determining the effectiveness of guided materials based on enriched 5e instructional model related to solar system and beyond: Space puzzle unit. Ph. D. Diss, Karadeniz Technical University, Council of Higher Education Thesis Center, Turkey.

De Jong, T., Linn, M. C., & Zacharia, Z. C. (2013). Physical and virtual laboratories in science and engineering education. *Science*, 340(6130):305-308. DOI: https://doi.org/10.1126/science.1230579

Deci, E. L., Eghrari, H., Patrick, B. C., & Leone, D. (1994). Facilitating internalization: The selfdetermination theory perspective. *Journal of Personality*, 62(1):119-142.

Durak, H. Y., Yilmaz, F. G. K., & Yilmaz, R. (2019). Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. *Contemporary Educational Technology*, 10(2):173-197. DOI: https://doi.org/10.30935/cet.554493

Foerster, K. T. (2016). Integrating programming into the Mathematics curriculum: Combining Scratch and Geometry in grades 6 and 7. In Proceedings of the 17th annual conference on information technology education (pp. 91-96). September 28. DOI: https://doi.org/10.1145/2978192.2978222

Fraenkel, J. R., Wallen, N. E., & Hyun, H. H. (2012). How to design and evaluate research in education (Eight Edition), p:264 . New York: McGraw-Hill. ISBN: 978-0078097850.

Fraser, B. J. (2023). The evolution of the field of learning environments research. *Education Sciences*, 13(3):257. https://doi.org/10.3390/educsci13030257

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

Güleryüz, H. (2022). The effect of robotic coding (mblock-arduino) activities on students' self-efficacy and attitudes. *Acta Scientific Computer Sciences*, 4(8):2-9.

Hacıoğlu, Y., & Dönmez Usta, N. (2020). Digital game design-based STEM activity: Biodiversity example. *Science Activities*, 57(1):1-15. DOI: https://doi.org/10.1080/00368121.2020.1764468

Hand, B., & Treagust, D. F. (1991). Student achievement and science curriculum development using a constructivist framework. *School Science and Mathematics*, 91(4):172–176. DOI: https://doi.org/10.1111/j.1949-8594.1991.tb12073.x

Hew, K.F., Brush, T. (2007). Integrating technology into K-12 teaching and learning: current knowledge gaps and recommendations for future research. *Education Tech Research Dev*elopment, 55(3):223-252. DOI: https://doi.org/10.1007/s11423-006-9022-5

Hsu, PS. (2016). Examining current beliefs, practices and barriers about technology integration: A case study. *TechTrends* 60(1):30-40. DOI: https://doi.org/10.1007/s11528-015-0014-3

Iskrenovic-Momcilovic, O. (2020). Improving geometry teaching with scratch. *International Electronic Journal of Mathematics Education*, 15(2):em0582. DOI: https://doi.org/10.29333/iejme/7807

Jiang, B., & Li, Z. (2021). Effect of Scratch on computational thinking skills of Chinese primary school students. *Journal of Computers in Education*, 8(4):505-525. DOI: https://doi.org/10.1007/s40692-021-00190-z

Kafai, Y. B., & Burke, Q. (2013). Computer programming goes back to school. *Phi Delta Kappan*, 95(1):61-65. DOI: https://doi.org/10.1177/003172171309500111

Kasalak, I. (2017). Effects of robotic coding activities on the effectiveness of secondary school students' self-efficacy and student experience about activities. Unpublished master's thesis. Hacettepe University, Ankara.

Kasalak, İ. & Altun, A. (2018). Percelved self-efficacy scale development study related to blockbased programmıng: Scratch case. *Educatıonal Technology Theory And Practıce*, 8 (1):209-225. DOI: https://doi.org/10.17943/Etku.335916

Kim, C., & Hannafin, M. J. (2011). Scaffolding problem-solving in technology-enhanced learning environments (TELEs): Bridging research and theory with practice. *Computers & Education*, 56(2):403-417. DOI: https://doi.org/10.1016/j.compedu.2010.08.024

Korkmaz, Ö., Çakır, R., & Özden, M. Y. (2015). Computatıonal thınkıng levels scale (ctls) adaptatıon for secondary school level. *Gazi Journal of Educational Science*, 1(2):143-162.

Koyunlu Unlu, Z. & Dokme, I. (2020). The effect of technology-supported inquiry-based learning in science education: Action research. *Journal of Education in Science, Environment and Health (JESEH)*, 6(2):120-133. DOI: https://doi.org/10.21891/jeseh.632375

Kozcu Cakir, N., & Guven, G. (2019). Arduino-assisted robotic and coding applications in science teaching: Pulsimeter activity in compliance with the 5E learning model. *Science Activities*, 56(2):42-51. DOI: https://doi.org/10.1080/00368121.2019.1675574

Lai, A. F., Lai, H. Y., Chuang, W. H., & Wu, Z. H. (2015). Developing a mobile learning management system for outdoors nature science activities based on 5e learning cycle. Paper Presented at the International Association for Development of the Information Society (IADIS) International Conference on e-Learning, Spain.

Lai, C. S., & Lai, M. H. (2012, June). Using computer programming to enhance science learning for 5th graders in Taipei. In 2012 International Symposium on Computer, Consumer and Control, Taiwan, June 4-6, pp. 146-148. IEEE. DOI: https://doi.org/10.1109/IS3C.2012.45

Lye, S. Y., Wee, L. K., Kwek, Y. C., Abas, S., & Tay, L. Y. (2014). Design, customization and implementation of energy simulation with 5E model in elementary classroom. *Journal of Educational Technology & Society*, 17(3):121-137. Available at: https://www.jstor.org/stable/10.2307/jeductechsoci.17.3.121

Mishra, P., Pandey, C. M., Singh, U., Gupta, A., Sahu, C., & Keshri, A. (2019). Descriptive statistics and normality tests for statistical data. *Annals of Cardiac Anaesthesia*, 22(1):67. DOI: https://doi.org/10.4103/aca.ACA_157_18

Molina-Ayuso, Á., Adamuz-Povedano, N., Bracho-López, R., & Torralbo-Rodríguez, M. (2022). Introduction to computational thinking with scratch for teacher training for spanish primary school teachers in mathematics. *Education Sciences*, 12(12):899. DOI: https://doi.org/10.3390/educsci12120899

Moreno-León, J., & Robles, G. (2016). Code to learn with Scratch? A systematic literature review. In 2016 IEEE Global Engineering Education Conference (EDUCON), United Arab Emirates, April 10-13, pp. 150-156. IEEE. DOI: https://doi.org/10.1109/EDUCON.2016.7474546

Ogegbo, A. A., & Ramnarain, U. (2022). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, 58(2):203-230. DOI: https://doi.org/10.1080/03057267.2021.1963580

Oliveira, A., Feyzi Behnagh, R., Ni, L., Mohsinah, A. A., Burgess, K. J., & Guo, L. (2019). Emerging technologies as pedagogical tools for teaching and learning science: A literature review. *Human Behavior and Emerging Technologies*, 1(2):149-160. DOI: https://doi.org/10.1002/hbe2.141

Özbek, Z. T., & Uslu, N. A. (2021). Technology integration into science education: Systematic review and mapping of postgraduate theses in Turkey. *Başkent University Journal of Education*, 8(2):427-440. Available at: http://buje.baskent.edu.tr/index.php/buje/a rticle/view/404

Papert, S., & Solomon, C. (1971). Twenty things to dowith a computer (Artificial Intelligence Memo Number248).Cambridge, MA: Massachusetts Institute of Technology. Available at: https://files.eric.ed.gov/fulltext/ED077240.pdf

Resnick, M. (2013). Learn to code, code to learn. how programming prepares kids for more than math. EdSurge (May 8, 2013). Available at: https://www.edsurge.com/news/2013-05-08-learn-to-code-code-to-learn

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM*, 52(11):60-67. DOI: https://doi.org/10.1145/1592761.1592779

Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020). Computational thinking and mathematics using Scratch: an experiment with sixth-grade students. *Interactive Learning Environments*, 28(3):316-327. DOI: https://doi.org/10.1080/10494820.2019.1612448

Shamir, M., Kocherovsky, M., & Chung, C. (2019). A paradigm for teaching math and computer science concepts in k-12 learning environment by integrating coding, animation, dance, music and art. In 2019 IEEE Integrated STEM Education Conference (ISEC), Princeton, March 16, pp. 62-68. IEEE. DOI: https://doi.org/10.1109/ISECon.2019.8882015

Silva, A., Silva, J., Gouveia, C., Silva, E., Rodrigues, P., Barbot, A.,. & Coelho, D. (2020). Science education and computational thinking–adapting two projects from classroom learning to emergency distance learning. *International Journal on Lifelong Education and Leadership*, 6(2):31-38. DOI: https://doi.org/10.25233/ijlel.803552

Talan, T. (2020). Investigation of the Studies on the Use of Scratch Software in Education. *Journal of Education and Future*, 2020(18): 95-111. DOI: https://doi.org/10.30786/jef.556701

Tedre, M., & Denning, P. J. (2016). The long quest for computational thinking. In Proceedings of the 16th Koli Calling international conference on computing education research, Finland, November 24-27, pp. 120-129. DOI: https://doi.org/10.1145/2999541.2999542

Vinayakumar R., Soman K. & Menon, P. (2018). Fractal Geometry: Enhancing Computational Thinking with MIT Scratch, 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, July 10-12. pp. 1-6, DOI: https://doi.org/10.1109/ICCCNT.2018.8494172

Wing, J. (2006). Computational thinking. *Communications of the ACM*, 49(3):33–36. DOI: https://doi.org/10.1145/1118178.1118215

Yamamori, K. (2019). Classroom practices of low-cost STEM education using scratch. *Journal of Advanced Research in Social Sciences and Humanities*, 4(6):192-198. DOI: http://dx.doi.org/10.2139/ssrn.3791166

Yukselturk, E. & Altıok, S. (2016). Pre-Service Information Technology Teachers` Perceptions about Using Scratch Tool in Teaching Programming. *Mersin University Journal of the Faculty of Education*, 12(1):39-52. DOI: https://doi.org/10.17860/efd.94270

Received: 21 June 2023
Revised: 28 July 2023
Accepted: 07 August 2023