AJIT-e Academic Journal of Information Technology
10.5824/ajite.2023.04.001.x

2023 • Fall - Güz • Volume – Cilt: 14 • Issue – Sayı: 55

The Effect of Project Based Learning Approach on Computational Thinking Skills and Programming Self-Efficacy Beliefs*

Hayrünnisa ERGİN, *Izmir Provincial Directorate of National Education, Teacher, nisaergin@gmail.com,* *0000-0003-2625-0043*

Yüksel Deniz ARIKAN[**]**,** *Ege University, Faculty of Education, Department of Computer and Instructional Technologies Education, Asst. Prof. Dr., y.deniz.arikan@gmail.com,* *0000-0002-7151-5381*

ABSTRACT

The aim of this study is to reveal the effect of project use on students' self-efficacy beliefs towards programming and their computational thinking skills. A one-group pretest-posttest experimental design was used in the study. The research was conducted in 2018 with 14 12th-grade students in a Vocational and Technical High School in Izmir. In the research, the application of project use in programming teaching lasted 18 weeks. The research data were collected with the Self-Efficacy Scale for Programming (SESP) developed by Altun and Mazman (2012) and the Computer Thinking Skill Levels Scale (CTSLS) developed by Korkmaz, Çakır, Özden, Oluk, Sarıoğlu (2015). Wilcoxon Signed Ranks Test, one of the nonparametric tests, was used to analyze the research data. As a result of the study, it was observed that the use of projects in programming instruction had a positive effect on students' self-efficacy in programming, while it did not have a significant effect on their computational thinking skills. Based on the results of the research, it is recommended to teach block-based programming before text-based programming, to include game-themed activities, to ensure active participation of students, and to use multidimensional and alternative measurement tools to measure computational thinking skills to comprehend algorithm stages in programming instruction.

Keywords **:** Computational Thinking, Programming Instruction, Programming Self-Efficacy Beliefs, Project-Based Learning, Purdue Model

---

[*]*This article is derived from Hayrünnisa Ergin's master's thesis entitled "The Effect of Project Use in Computer Programming on Students' Computational Thinking Skills and Programming Self-Efficacy Belief", conducted under the supervision of Asst. Prof. Dr. Yüksel Deniz Arıkan.*

[**] *Corresponding Author*

Araştırma Makalesi / Research Article
Date Received: 12.09.2023
Publishing Date: 22.11.2023

AJIT-e Academic Journal of Information Technology
2023 Fall/Güz – Cilt/Vol: 14 - Sayı/Issue: 55
10.5824/ajite.2023.04.001.x

Proje Tabanlı Öğrenme Yaklaşımının Bilgi İşlemsel Düşünme Becerilerine ve Programlama Öz Yeterlilik İnancına Etkisi

ÖZ

Bu araştırmanın amacı proje kullanımının, öğrencilerin programlamaya yönelik öz yeterlilik inançlarına ve bilgi işlemsel düşünme becerilerine etkisini ortaya koymaktır. Araştırmada tek grup ön test- son test deneysel desen kullanılmıştır. Araştırma 2018 yılında, İzmir ilinde bir Mesleki ve Teknik Lisesi'nde 12. sınıfta öğrenim gören 14 öğrenci ile gerçekleştirilmiştir. Araştırmada programlama öğretiminde proje kullanımı uygulaması 18 hafta sürmüştür. Araştırma verileri Altun ve Mazman (2012) tarafından geliştirilen Programlamaya İlişkin Öz Yeterlilik Ölçeği ile Korkmaz, Çakır, Özden, Oluk, Sarıoğlu (2015) tarafından geliştirilen Bilgisayarca Düşünme Beceri Düzeyleri Ölçeği ile toplanmıştır. Araştırma verilerinin analizinde parametrik olmayan testlerden Wilcoxon İşaretli Sıralar Testi kullanılmıştır. Araştırma sonucunda, programlama öğretiminde proje kullanımının öğrencilerin programlamaya ilişkin öz yeterliliklerini olumlu etkilediği görülürken, bilgi işlemsel düşünme becerilerinde anlamlı bir etkisinin olmadığı görülmüştür. Araştırma sonuçlarına dayalı olarak programlama öğretiminde algoritma aşamalarının kavranabilmesi için metin tabanlı programlamaya geçmeden önce blok tabanlı programlama öğretimin yapılması, oyun temalı etkinliklere yer verilmesi, öğrencilerin aktif katılımlarının sağlanması ve bilgi işlemsel düşünme becerilerinin ölçülmesinde çok boyutlu ve alternatif ölçme araçlarının kullanılması önerilmektedir.

Anahtar Kelimeler **:** Bilgi İşlemsel Düşünme, Programlama Öğretimi, Programlama Öz Yeterlilik İnancı, Proje Tabanlı Öğrenme, Purdue Modeli

INTRODUCTION

The information and communication technologies sector are consistently developing in our country as well as in the world. The needs of society for information technologies and the need for people with competence in the field are similarly increasing. For this reason, the importance of programming education is increasing day by day (Akpınar & Altun; Barut & Kuzu, 2017) and learning methods and tools are being investigated in education and training in this field.

Today, programming education is considered very important for the cognitive development of students for reasons such as developing questioning and thinking skills and enabling them to see the connections between events (Akçay & Çoklar, 2016). Learning programming is a difficult process because it requires cognitive activities such as analysis and synthesis, and conceptual knowledge must be acquired before it can be put into practice (Porter & Calder, 2004). According to Churches (2008), programming skill is included in the creativity stage according to Bloom's taxonomy, which shows that learning takes place in stages, as learners create their own applications to suit their needs and goals.

When we look at the factors that make programming education difficult, we see those negative prejudices against programming, programming languages being in English, and trying to teach programming logic with traditional teaching methods (Arabacıoğlu et al., 2007). In addition to these, rote learning, lack of abstract thinking ability, inadequacy of arithmetic, mathematical and analytical thinking, and inability to determine the usage areas of the applications made in programming courses in real life are among the difficulties encountered by students in programming teaching (Cevahir & Özdemir, 2017). It is stated that the three main factors affecting students' learning of computer programming are teaching approach, choice of computer programming language and programming development environment, and activities based on the constructivist approach should be emphasized in the formation of these learning outcomes (Ali, Tumian, & Seman, 2017). Different methods and strategies based on the principles of the constructivist approach, in which learners are active in their own learning processes and responsible for their own learning, and in which what is learned is associated with real-life problems, are gaining importance. Ülküdür and Bacanak (2016) state that project-based learning is one of these applications. Project-based learning environments provide systematic structuring of what is learned and offer individuals with different learning styles, intelligence, and abilities the opportunity to work individually or collaboratively (Saraçoğlu, Akamca, & Yeşildere, 2006). It is thought that these methods and strategies make students active and affect their thinking skills (Güneş, 2010).

In the process of learning programming, students not only develop their coding knowledge but also their mathematical skills and computational thinking. Thus, students are able to produce projects and share their ideas. These earnings are considered necessary for individuals from every professional group (Sayın & Seferoğlu, 2016). One of the most important 21st-century skills thought to be acquired in the process of learning programming is computational thinking skill. Computational thinking skills indicate creative thinking, algorithmic thinking, critical thinking, problem solving, collaborative learning and communication skills. The computational thinking skill includes decomposition, generalization, algorithmic thinking, evaluation, and abstraction. These steps teach students the basics of how to approach a problem and solve it in a computational context. Programming is seen as a part of this systematic approach (Nash, 2017). Computational thinking refers to the cognitive process used in basic problem-solving steps. Therefore, learning programming is a fundamental factor for developing computational thinking (Lye & Koh, 2014).

Although programming knowledge is accepted as one of the most important requirements for becoming computer literate today, it is seen as a difficult process to learn. For this reason, it is important to develop self-efficacy beliefs towards programming, which is seen as one of the most important factors that facilitate learning programming. Self-efficacy is an individual's belief in his/her own potential that his/her knowledge and skill level is sufficient to perform a function (Bandura, 1988; as cited in Azar, 2012). Many studies show that students who have developed programs before taking the programming course have high achievement and self-efficacy (Gezgin & Adnan, 2016).

Different applications need to be developed to make programming teaching more effective and to facilitate overcoming the difficulties encountered. For this reason, within the scope of the research, it is thought that it would be useful to use the project in programming teaching, which is based on the principles of the constructivist approach, responds to the interests, and needs of learners, and enables learning to take place through real-life problems. In this study, the effect of project use in programming instruction on students' self-efficacy beliefs towards programming and computational thinking skills was examined. In this context, the following questions were searched for answers.

1. What is the effect of project use on students' self-efficacy beliefs about computer programming?
2. What is the effect of project use on students' computational thinking skills?

1. METHOD

This section includes the research design, participants, experimental process, data collection tools and data analysis.

1.1. Research Design

In this study, one group pretest-posttest experimental design method was used. The experimental design used in the study is shown in Table 1.

Table 1: Research Design

Group	Pre-test	Process	Post-test
Experiment	SESP-CTSLS	PT	SESP-CTSLS

SESP: Self-Efficacy Scale for Programming, *CTSLS:* Computer Thinking Skill Levels Scale, *PT:* Project-oriented teaching.

As seen in Table 1, the independent variable of the study was the learning environment in which the project was used, and the dependent variables were students' self-efficacy beliefs about programming and computational thinking skills.

1.2. Participants

The research was conducted in 2018 with the participation of 12th-grade students of a Vocational and Technical High School in Izmir. Purposive sampling was used to determine the participants. This school is where one of the researchers works. Participants were included in the study because they took a programming course. All 14 students were male in the study.

1.3. Experimental Process

Within the scope of the research, firstly, the curriculum was prepared in accordance with the three-stage Purdue model by considering the existing course curriculum. At the beginning of the experimental implementation process, SESP and CTSLS were applied as a pretest. Then, the prepared curriculum was implemented. Within the scope of the application, students worked individually or collaboratively. Students were allowed to choose project topics based on different ideas. In this process, the teacher-researcher guided the students in the process of resource searching and implementation. In the application, the students presented their projects, and their feedback was received to improve the projects. In the last stage of the research, SESP and CTSLS were applied as a post-test.

1.3.1. Needs Analysis

First, it was decided to use the project method to transform the knowledge acquired in the programming teaching process into learning products. The project method is a teaching method applied in developed countries to prevent the curriculum content from being taught in small pieces of information that are not associated with each other (Çakallıoğlu, 2008). In the research process, a curriculum was needed for students to develop their prior knowledge to comprehend the algorithm logic required by programming and to create independent project studies. Considering that prior knowledge of the subject to be learned has a significant effect on academic success according to the research (Coşar, 2013), it was decided to prepare a curriculum according to the three-stage Purdue model, which enriches project-based learning and emphasizes the importance of prior learning, created by Feldhusen and Kolloff (1988). The content of the curriculum is based on Bloom's taxonomy and the cognitive skill sequence required by the curriculum. Before the implementation, it is expected that conceptual knowledge is acquired, and the synthesis step is expected to be realized by bringing these basic concepts together on a different problem situation. Accordingly, the implementation process of the curriculum was carried out in accordance with the three-stage Purdue model. Table 2 shows the steps corresponding to the three-stage Purdue model and the implementation process in the curriculum of the learning outcomes to be acquired according to the conceptual framework of the components of programming knowledge.

Table 2: Comparison of the Components of Programming Knowledge and Curriculum with the Three-Stage Purdue Model

	Stages of the Purdue model	Components of programming knowledge	Curriculum
Phase I Knowledge and Comprehension Level	Development of scientific process skills. Practices for creative and critical thinking skills	Syntactic (Spelling) Information	Variables, Arithmetic Operators, Loops, Arrays, Flowcharts, Creating Algorithms
Phase II Application and Analysis	Facing a problem situation. Discussion, brainstorming, etc. with the teacher and small groups, active student participation	Conceptual (Procedural) Knowledge	Two-person dice throwing game, creating a quiz, making a stopwatch, Button catching game

Phase III Synthesis and Evaluation	Independent project work. Problem-solving. Acquired in the first two stages, and the use of high-level cognitive skills	Strategic Knowledge	Creation, development, and evaluation of independent project work

While creating the curriculum content, attention was paid to ensure that the activities selected were of a quality that could attract students' interest and were appropriate for the achievements of the course. The realization of learning through the theme of games makes the lessons interesting, provides a better understanding of abstract concepts and the creation of connections between concepts that are effective in establishing the algorithmic structure, and thus increases the motivation to learn (Çatlak, Tekdal, & Baz, 2015). It is stated that learners learn better and enjoy the learning process while working on meaningful projects in line with their interests and needs. While developing the Scratch programming interface, two design criteria were given importance. The first one is that it includes many different story situations and game designs, and the second one is that the activities can be shaped according to personal interests and needs (Resnick et all., 2009). For these reasons, game-themed activities were included in the curriculum, allowing students to work on different problem situations.

1.3.2. Curriculum

The three-stage Purdue model created by Feldhusen and Kolloff (1988) was used in the preparation of the curriculum. Kutlu and Gökdere (2013) stated that although this model was developed for gifted students at the primary education level, it can also be applied in regular education institutions by making necessary arrangements since it enriches project-based learning and provides appropriate learning opportunities for each student. The stages of the curriculum are presented in Table 3, Table 4, and Table 5.

Table 3: *Curriculum Stage 1*

	Activity 1	Activity 2	Activity 3	Learning out comes	Time
1. Week	Ability to print sample value according to the data type selected in Combo box			Will be able to use variables in an object-oriented programming environment.	9 class hours

2. Week	To be able to create the change table according to the order of operation of variables			Will be able to use arithmetic operators.	9 class hours
3. Week	Area and perimeter calculations of different geometric objects	Favorite team voting using Progress Bar object		Will be able to create the algorithmic structure.	9 class hours
4. Week	Calculation of the grade range of the grade point average in the 5-point system	Listing the properties of the element selected from the combo box with Switch Case structure		Will be able to use logical operators and conditional statements.	9 class hours
5. Week	Creating an array of numbers and initializing the array,	Accessing array elements, for each expression array copy	Guide application created by entering the array elements by the user	Will be able to use arrays.	9 class hours
6. Week	Transferring array elements to list box with for loop	Addition of array elements and use of nested for loops with two-dimensional arrays	Factorial calculation	Will be able to use loop expressions.	9 class hours

Table 4: *Curriculum Stage 2*

	Activity	Learning out comes	Time
7. Week	Two-person dice rolling game	will be able to use the Random class and logical operators.	9 class hours
8. Week	Creation of a knowledge competition	Problem Solving and Application of Creative Problem-Solving Models	9 class hours
9. Week	Stopwatch making		9 class hours
10. Week	Button capture game		9 class hours

Table 5: *Curriculum Stage 3*

	Activity		Learning out comes	Time
11. Week	Independent project work		It will be carried out under student control and teacher guidance.	9 class hours
12. Week	Independent project work			9 class hours
13. Week	Presentation of independent project work			9 class hours
The process of developing projects in the light of new acquisitions				
	Activity 1	Activity 2	Learning out comes	Time
14. Week	Creating a database, Creating a connection to a database,	Querying data, closing the connection	Will be able to query information in a database.	9 class hours
15. Week	Data update,	Adding and deleting data	Will be able to perform operations on data.	9 class hours
16. Week	Development of Independent Project Studies		It will be carried out under student control and teacher guidance.	9 class hours
17. Week	Development of Independent Project Studies			9 class hours
18. Week	Presentation of Independent Project Work			9 class hours

As seen in Table 3, Table 4, and Table 5, the curriculum prepared based on the Purdue model consists of three stages. In the first stage of the curriculum, knowledge and skills were acquired, in the second stage, sample applications were developed based on problem-solving models, and in the third stage, independent project studies were carried out under the guidance of the teacher.

In the first 10 weeks of the study, the first and second stages of the three-stage Purdue model were carried out. In the third stage, independent project studies were conducted. Throughout the research, the students studied in a computer laboratory consisting of 15 student computers and 1 teacher computer. Each student had a computer that they could use regularly in their studies. Students who wanted to continue their studies brought their personal computers. The course teacher, who was also one of the researchers, helped the students comprehend the programming logic and develop sample applications in the first and second stages of the model and guided the students in the independent project development process. Students carried out the project development process in the school environment. At the end of the independent project studies, which was the third stage of the research, the SESP and CTSLS were applied as a post-test.

1.3.3. The Role of Researchers

One of the researchers is the teacher of the course and one is the advisor. The researchers prepared a curriculum in line with the needs of the participants and carried out the necessary work to enable programming instruction to be carried out using projects. The teacher-researcher guided the students in the process of creating and developing project ideas. The researchers collected the data by providing the necessary information about the scales used as pre-test and post-test. The researchers analyzed and reported the data in an unbiased manner.

1.4. Data Collection Tools

1.4.1. SESP

The SESP, the validity, and reliability study of which was conducted by Altun and Mazman (2012), was applied to the students. The Turkish version of the SESP consists of 9 items and 2 factors (Simple programming tasks and complex programing tasks). The Cronbach Alpha coefficient of the SESP is 0.928.

1.4.2. CTSLS

The CTSLS was developed to measure the computational thinking skills of individuals who can be defined as adult learners. The internal consistency coefficient of the CTSLS is 0.822. CTSLS consists of 29 five-point Likert-type items and 5 sub-factors. CTSLS sub-factors are creativity, algorithmic thinking, collaborative work, critical thinking and problem-solving.

1.5. Analyzing the Data

For the sub-problems of the study, the Wilcoxon signed-rank test and dependent groups T test were used to analyze whether there was a difference between the pretest and posttest scores of the SESP and CTSLS.

2. FINDINGS

2.1. The Effect of Project Use on Students' Self-Efficacy in Computer Programming

The results of the Wilcoxon Signed Ranks Test conducted for the analysis of the first sub-problem of the study, "How is the effect of using projects on students' self-efficacy beliefs about computer programming?" are presented in Table 6.

Table 6: *Wilcoxon Signed-Ranks Test Results of SESP Pre-Test-Post-Test Application Scores*

		n	Rank Mean	Row Total	z	p
Programming Self-efficacy	Negative Sequence	3	5,50	16,50	-2,030	0,042
	Positive Sequence	10	7,45	74,50		
	Equal	1				

When Table 6 is examined, it is seen that the difference between students' self-efficacy in programming before and after programming instruction is significant (z=-2,030, p= 0,042). These data were also analyzed with paired samples t-test (t= -2,44, p= 0,029), and a significant difference was found. Accordingly, it is seen that the use of project-based learning method in programming instruction has a positive effect on self-efficacy beliefs about programming (p<0.05). The results of the analysis of the sub-factors of the scale of students' self-efficacy beliefs about computer programming are presented in Table 7.

Table 7: *Wilcoxon Signed-Ranks Test Results of SESP Pre-Test-Post-Test Application Scores According to Sub-Factors*

		n	Rank Mean	Row Total	z	p
Simple Programming Tasks	Negative Sequence	1	3,5	3,5	-2,46	0,014
	Positive Sequence	9	5,72	51,5		
	Equal	4				
Complex Programming Tasks	Negative Sequence	3	7,5	22,5	-1,61	0,107
	Positive Sequence	10	6,85	68,5		
	Equal	1				

Table 7 shows the results of the analysis of students' self-efficacy beliefs about computer programming on the sub-factors of simple programming tasks and complex programming tasks. Accordingly, it is seen that the difference in the sub-factor of simple programming tasks is significant (z= -2,46, p= 0,014), while the difference in the sub-factor of complex programming tasks is not significant (z= -1,61, p= 0,107).

2.2. The Effect of Project Use in on Students' Computational Thinking Skills

The results of the Wilcoxon Signed Ranks Test conducted for the second sub-problem of the study, "How is the effect of project use on students' computational thinking skills?" are presented in Table 8.

Table 8: *Wilcoxon Signed-Ranks Test Results of CTSLS Pre-Test-Post-Test Application Scores*

		n	Rank Mean	Row Total	z	p
Computational Thinking	Negative Sequence	7	6,50	45,50	-0,440	0,66
	Positive Sequence	7	8,50	59,50		
	Equal	0				

When Table 8 is examined, it is seen that there is no significant difference between the pre-test and post-test scores of the students' computational thinking skills (z= -0,440, p= 0,66). These data were also analyzed with the Paired Samples T Test (t= -0,372, p= 0,716) and no significant difference was found. Accordingly, it can be said that the use of projects has no significant effect on students' computational thinking skills (p>0.05). When the analysis results related to the sub-factors of computational thinking skills were examined, it was seen that the difference between the pre-test and post-test scores of creativity (z= -0,47, p= 0,63), algorithmic thinking (z= -1,47, p= 0,14), collaboration (z= -0,25, p= 0,79), critical thinking (z= -0,42, p= 0,67), problem solving (z= -0,63, p= 0,52) was not statistically significant.

3. DISCUSSION AND CONCLUSION

In this study, the effect of using projects on students' self-efficacy beliefs towards programming and computational thinking skills was examined. As a result of the research, it was concluded that the project-based learning method had a significant positive effect on students' self-efficacy beliefs towards programming. There are similar studies in the literature (Wiedenbeck, 2005; Jegede, 2009; Aşkar & Davenport, 2009; Davidson, Larzon, & Ljunggren 2010; Mazman & Altun, 2013). Wiedenbeck (2005), in a study with 120 university students who took C++ programming course for five academic semesters, stated that previous experiences affect perceived self-efficacy and that self-efficacy towards programming also affects success in programming courses. According to a study conducted with 190 engineering students randomly selected from six different engineering departments at the University of Nigeria, it was revealed that the number of programming courses taken by students and their success based on their scores in programming courses significantly predicted their Java programming self-efficacy (Jegede, 2009). Aşkar and Davenport (2009), in their study with engineering students, focused on gender, choice of major, previous computer skills and frequency of computer use as factors determining self-efficacy beliefs. It was found that students who used computers every day had significantly higher self-efficacy scores than those who used computers several times a week and computer engineering students had significantly higher self-efficacy scores than students in other engineering departments. Davidson, Larzon, and Ljunggren (2010) examined how self-efficacy beliefs towards programming changed after a one-year introduction to a programming course. As a result of the study, although students' self-efficacy scores did not show a significant difference, a significant increase was observed in self-regulation and in many of the skills related to the course objectives. Mazman and Altun (2013) examined the self-efficacy beliefs of the students of the department of ITTE according to whether they had prior experience after the programming course they took. At the end of the study, they found that the programming course provided a significant increase in self-efficacy beliefs about programming in both groups with and without prior experience, and this increase was higher for the group without prior experience. It was also observed that the difference between the self-efficacy beliefs of the groups with and without prior experience decreased after the programming course.

Secondly, in this study, it was observed that the use of projects in programming instruction did not have a significant effect on students' computational thinking skills. In the studies conducted in the literature, it is seen that different application-based methods have been studied to measure the development of computational thinking skills (Denner & Werner, 2011; Brennan & Resnick, 2012; Grover, 2015; Kert, Yeni, & Şahiner, 2017). Denner and Werner (2011) developed a method called Fairy assessment by suggesting not to use scales in the assessment of computational thinking, but rather to make measurements based on qualitative analysis. This method is designed to measure whether middle school students understand the stages of programming, as well as whether they have gained skills such as abstraction, modeling, and whether they can apply algorithmic thinking to solve a problem. In this study, each student was assigned three tasks to be performed in the Alice programming environment and a scoring system ranging from 0 to 10 was developed to measure student performance in each task. They stated that each task given in the assessment process should be as independent as possible from other assessment tasks to measure a different dimension of computational thinking. Like this result, Brennan, and Resnick (2012) stated that applications for computational thinking should focus on the learning process and how it is learned rather than what is learned. Brennan and Resnick (2012), who analyzed the practices shared in online Scratch workshops, developed three basic dimensions for the assessment of computational thinking. These dimensions consist of computational concepts (conditional statements, loops, triggers, arithmetic, and logical operators, etc.), computational practices (algorithm creation, abstraction), and perspectives on practices (communication, inquiry). After identifying these dimensions, they defined three approaches to assess the development of computational thinking in students who design programs with Scratch. As a result of the study, it was thought that a single approach was not sufficient, and that it was appropriate to use a combination of approaches in environments suitable for assessment. Similarly, Grover (2015) stated that assessment systems that are more comprehensively structured than the traditional methods used in the assessment of computational thinking, that include multiple assessment tools such as formative assessments, open-ended programming assignments, that are based on applied learning, that can transfer what is learned to different situations and that can measure algorithmic thinking skills are necessary. Kert, Yeni, and Şahiner (2017) stated that for computational thinking to be measurable, the relationship between the sub-skills it contains should be revealed. In this context, they put forward a model that includes sub-skills such as formulation, abstraction, dividing the problem into small parts, algorithmic thinking, and indirectly associates collaborative learning and communication skills. As a result of the study, they stated that to monitor the development of computational thinking, the sub-skills it covers should be measured. In line with these results in the literature, it is seen that it would be more appropriate to use multidimensional and alternative measurement tools to measure computational thinking skills.

During the research process, it was observed that students had difficulty in creating algorithms and synthesizing the information they learned in different problem situations. For this reason, it is thought that block-based programming instruction may be useful to comprehend the algorithm stages before moving on to text-based programming in the process of learning programming. It is thought that instead of giving the students the algorithmic process that takes place during the coding phase of the programs, it would facilitate learning if the interrelated particles were given in order and these stages are done simultaneously with the students. In the research process, it was observed that teaching the concepts learned in programming education through game-themed programs attracted students' attention and increased their motivation. For this reason, it is thought that including game-themed activities in programming teaching will make learning interesting and thus enable students to actively participate in the learning process. In addition, it is thought that it is important for teachers to choose different methods in which they can make students active and to provide the necessary help as a guide to overcome the difficulties in the process of writing a program and not to decrease motivation. In the research process, it was seen that determining the boundaries in the planning of the project applications and the development of the projects contributed to the smoothing of the process and not overburdening the teacher. In cases where similar teaching methods are used, proper task sharing in collaborative working groups will help the process to work. Since the high level of students' self-efficacy beliefs about programming is one of the most important factors facilitating learning, it is recommended to include more experiences that can improve self-efficacy. Although project-based programming makes learning meaningful and interesting, limiting the number of students will enable the teacher to perform the guidance task effectively.

As a result, researchers can examine the effects of project-based instructional practices in programming instruction at different educational levels in terms of the dependent variables of the study. In addition, researchers can examine the effects of different learning methods to overcome the difficulties encountered in programming teaching. In future studies, measurement tools that are structured in a process-oriented way and organized to include the subcomponents of computational thinking skills can be developed and applied to measure computational thinking skills. The researchers expect this study to contribute to future research.

Note:

This research was carried out with the permission of the Izmir Directorate of National Education Research Evaluation Commission dated 09.02.2018 and the permission of the İzmir Directorate of National Education dated 12.02.2018 and numbered 12018877-604.01.02-E.2938219.

REFERENCES

Akçay, A., Çoklar, A. N. (2016). A proposal for the development of cognitive skills: Programming education. In: A. İşman, H. F. Odabaşı, B. Akkoyunlu (Eds.), *Educational technology readings* (pp.121-140). TOJET- The Turkish Online Journal of Educational Technology.

Akpınar, Y., Altun, A. (2014). The need for programming education in information society schools. *İlköğretim Online, 13(1)*, 1-4.

Ali, A. M., Tumian, A., Seman, M. S. A. (2017, July). A conceptual approach for understanding computer programming skills development. *Proceeding of Research and Innovation in Information Systems (ICRIIS) 2017*, Malaysia, pp. 1-5.

https://doi.org/10.1109/ICRIIS.2017.8002526

Altun, A., Mazman, S. G. (2012). Validity and reliability study of the Turkish form of the perception of self-efficacy for programming scale. *Journal of Measurement and Evaluation in Education and Psychology, 3(2),* 297-308.

Arabacıoğlu, C., Bülbül, H., & Filiz, A. (2007, January-February). A new approach in computer programming teaching. *Academic İnformation'07 - IX. Proceedings of Academic İnformation Conference*, Turkey- Kütahya, pp.193-197.

Aşkar, P., Davenport, D. (2009). An investigation of factors related to self-efficacy for Java programming among engineering students. *The Turkish Online Journal of Educational Technology, 8(1).*

Azar, A. (2012). Self-efficacy beliefs of prospective secondary science and mathematics teachers. *International Journal of Management Economics and Business, 6(12*), 235-252.

Barut, E., Kuzu, A. (2017). Comparison of Turkey and England information technologies curricula in terms of objectives, learning outcomes, activities, measurement, and evaluation processes. *Trakya University Journal of Faculty of Education, 7(2),* 721-745.

Brennan, K., Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. *In Proceedings of the 2012 Annual Meeting of The American Educational Research Association*, Vancouver, Canada, Vol. 1, p. 25.

Cevahir, H., Özdemir, M. (2017). Teacher opinions and solution suggestions for the difficulties encountered in programming teaching. *11th International Computer and Instructional Technology Symposium Proceedings*, Turkey- Malatya, pp. 24-26.

Churches, A. (2008). Bloom's taxonomy blooms digitally. *Tech & Learning*, *1*, 1-6.

Coşar, M. (2013). *The effects of computer programming studies in a problem-based learning environment on academic achievement, critical thinking tendency and attitude towards computers.* [Unpublished Doctoral Dissertation]. Gazi University.

Çakallıoğlu, S. N. (2008). *The effect of science teaching based on project-based learning approach on academic achievement and attitude.* [Unpublished master's thesis]. Çukurova University.

Çatlak, Ş., Tekdal, M., Baz, F. Ç. (2015). The status of teaching programming with Scratch software: A document review study. *Journal of Instructional Technologies & Teacher Education, 4(3),* 13-25

Davidson, K., Larzon, L., & Ljunggren, K. (2010). *Self-efficacy in programming among STS students*. Retrieved from http://www.it.uu.se/edu/course/homepage/datadidaktik/ht10/reports/Self-Efficacy.pdf.

Denner, J., Werner, L. (2011, April 8-12). *Measuring computational thinking in middle school using game programming.* Annual Meeting of the American Educational Research Association (AERA), New Orleans, Louisiana. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=f629baf5301d843c9f1d448edca901d0632dd278

Feldhusen, J. F., & Kolloff, M. B. (1988). A three-stage model for gifted education. *Gifted Child Today Magazine, 11(1),* 14-20. https://doi.org/10.1177/107621758801100104

Gezgin, D. M., Adnan, M. (2016). Investigation of mechanical engineering and econometrics students' self-efficacy perceptions about programming. *Journal of Ahi Evran University Kırşehir Faculty of Education, 17(2),* 509-525.

Grover, S. (2015, April 15-20). Systems of assessments" for deeper learning of computational thinking in K-12. *In Proceedings of the 2015 Annual Meeting of The American Educational Research Association (AERA), Chicago, Illinois.* https://www.sri.com/wp-content/uploads/2022/04/aera2015-_systems_of_assessments_for_deeper_learning_of_computational_thinking_in_k-12.pdf

Güneş, F. (2010). Innovations coming with constructivist approach in education. *Journal of Education Review. 6(16), 3-10*

Jegede, P. O. (2009, August). Predictors of java programming self-efficacy among engineering students in a Nigerian university. *International Journal of Computer Science and Information Security, IJCSIS*, USA. Vol. 4, No. 1&2, https://doi.org/10.48550/arXiv.0909.0074

Kert, S. B., Yeni, S., Şahiner, A. (2017, May 24-26). Investigation of sub-skills associated with computational thinking. *International Computer and Instructional Technologies Symposium Full Textbook*, Malatya, Turkey. 726-738.

Korkmaz, Ö., Çakır, R., Özden, M. Y., Oluk, A. & Sarıoğlu, S. (2015). Investigation of individuals' computer thinking skills in terms of different variables. *Ondokuz Mayıs University Journal of Education Faculty, 34 (2),* 68-87. Retrieved from https://dergipark.org.tr/tr/pub/omuefd/issue/20284/215276.

Kutlu, N., Gökdere, M. (2013). Enriching project-based learning: The three-stage Purdue model. *Journal of Dicle University Ziya Gökalp Faculty of Education, 20(2013*), 293-311.

Lye, S. Y., Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41,* 51-61.

Mazman, S. G., Altun, A. (2013). The effect of the Programming-I courses on the self-efficacy perceptions of the students of the Department of ITIT on programming. *Journal of Instructional Technologies & Teacher Education, 2(3),* 24-29

Nash, J. (2017). *Turn coders into computational thinkers | ISTE*. ISTE. https://www.iste.org/explore/Innovator-solutions/Turn-coders-into- computational thinkers

Porter, R., Calder, P. (2004, January). Patterns in learning to program: an experiment? *In Proceedings of the Sixth Australasian Conference on Computing Education*, Australian Computer Society, Inc., 30, 241-246.

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.

Saracaloğlu, A. S., Akamca, G. Ö., Yeşildere, S. (2006). The place of project-based learning in primary education. *Turkish Journal of Educational Sciences, 4(3),* 241-260.

Sayın, Z., Seferoğlu, S. S. (2016, February 3-5). Coding education as a new 21st century skill and the impact of coding on education policies. *Academic İnformation Conference,* Aydın, Turkey. https://yunus.hacettepe.edu.tr/~sadi/yayayin/AB16_Sayin-Seferoglu_Kodlama.pdf

Ülküdür, M. A., Bacanak, A. (2016). Comparison of project-based learning activities and game-based learning activities in preparation (development) dimension. *Bayburt Education Faculty Journal, 8(1),* 21-43.

Wiedenbeck, S. (2005, October). Factors affecting the success of non-majors in learning to program. *In Proceedings of The First International Workshop on Computing Education Research (ICER'05)*, 13-24. https://doi.org/10.1145/1089786.1089788