CONTEMPORARY EDUCATIONAL TECHNOLOGY, 2019, 10(2), 173-197
DOI: https://doi.org/10.30935/cet.554493 - TYPE: Research Article

Computational Thinking, Programming Self-Efficacy, Problem Solving and Experiences in the Programming Process Conducted with Robotic Activities

Hatice Yildiz Durak
Bartin University, Turkey
ORCID: 0000-0002-5689-1805

Fatma Gizem Karaoglan Yilmaz
Bartin University, Turkey
ORCID: 0000-0003-4963-8083

Ramazan Yilmaz
Bartin University, Turkey
ORCID: 0000-0002-2041-1750

Received: 23.10.2018 Accepted: 21.03.2019 Published: 16.04.2019

Abstract

The purpose of this study was to determine the skill levels of secondary school students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving and examine their experiences in the programming training process on robotic activities. Toward this purpose, a 10-week application was conducted with 55 students from 6th and 7th grades who received education at a secondary school in Western Black Sea region of Turkey during the school year of 2017-2018. The study was conducted using the mixed model and various scales in the quantitative dimension. On the other hand, a semi-structured interview form developed by the researchers was applied in the qualitative dimension. As a result, it was found out that students' computational thinking skills, programming self-efficacy and reflective thinking aimed at problem solving were moderate. Students' levels of computational thinking and programming self-efficacy were observed to differ depending on their grade levels. In addition, a positive and moderate relationship was found among the levels of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving.

Keywords: *Robotics; Computational thinking; Programming self-efficacy; Reflective thinking; Problem solving; Programming*

Introduction

Intended population in programming training generally consists of students receiving higher education. On the other hand, the acquisition of 21st century competences in education and acceptance of today's learners as digital natives have caused some changes in certain skills which must be acquired by the students. Thus, along with various applications, programming training has started to be considered important especially for K-12 students (Burke, 2012; Fessakis, Gouli, & Mavroudi, 2013; Gulbahar & Kalelioglu, 2014; Kazakoff, 2014). Programming training is considered functional particularly for earning skills like “creativity, critical thinking and problem solving, communication and cooperation, social and intercultural skills, productivity and responsibility, leadership and responsibility”, which are called 21st century skills (Einhorn, 2011; Grover & Pea, 2013; Lau & Yuen, 2011; Yen, Wu, & Lin, 2012).

As programming processes include different thinking skills and information fields, these processes help children develop crucial skills like “communication skills, creativity, intellectual curiosity, critical and systematic thinking, interpersonal and cooperation skills, problem identification/formulization/solution and self-orientation” (Partnership for 21st Century Skills, 2007), which exist in the nature of programming processes (Ismail, Ngah, & Umar, 2010; Lau & Yuen, 2011). In addition, programming skill which is considered among the most important skills of the 21st century is also considered a basic strategy for developing computational thinking (CT) skills (Fields, Searle, Kafai, & Min, 2012; García-Peñalvo, 2016; Grover & Pea, 2013; Lawanto, Close, Ames, & Brasiel, 2017; Lye & Koh, 2014; Meerbaum-Salant, Armoni, & Ben-Ari, 2013; Moreno, 2012; Saritepeci & Durak, 2017; Werner, Denner, Campe, & Kawamoto, 2012; Wing, 2014).

CT includes different top-level thinking skills and information fields which exist in the nature of programming process. The effect of programming processes on developing these skills and information helps students develop skills underlying CT. On the other hand, human interaction with computers is increasing each passing day in our age (Manovich, 2013). In order to understand and solve the problems that are encountered in a world surrounded by computers and various programs, it has become an obligation to act like a “computer” (Wing, 2006). Individuals need to know the computer language and have code literacy in order to participate in daily life activities (Rushkoff, 2010). Code literacy is achieved by reading/writing in the computer language and computational thinking (Román-González, 2014). Thus, programming training may support CT skills and develop the computer-based problem solving process.

Current studies basically focus on variables like problem solving, programming self-efficacy, attitude toward programming, programming and cooperation (Adleberg, 2013; Aslan, 2014; Bers, Flannery, Kazakoff, & Sullivan, 2014; Brennan, 2013; Burke, 2012; Ceylan, 2015; Cetin, 2012; Dogan, 2015; Durak, 2016; Gregg, 2014; Gulmez, 2009; Kayabasi, 2016; Noble, 2013; Olgun, 2014; Ozturk, 2016; Patan, 2016; Sáez-López, Román-González, & Vázquez-Cano, 2016; Yildiz- Durak & Guyer, 2018; Yildiz-Durak & Guyer, 2019; Yildiz-Durak, 2019). Even though there are some studies embracing robotic activities in programming training (Atmatzidou & Demetriadis 2012; Atmatzidou, Demetriadis, & Nika, 2018; Bers 2010; Castledine & Chalmers 201), it is possible to state that there are insufficient number of studies which focus on CT, programming self-efficacy and reflective thinking aimed at problem solving and learners experiences by using ER activities in programming training and offer an integrative perspective. This study aims to close this gap in the literature.

Purpose of the Study

The study aims to determine the skill levels of secondary school students regarding CT, programming self-efficacy and reflective thinking aimed at problem solving in the programming training process conducted with robotic activities and examine their experiences in this process. In line with this purpose, the following questions were tried to be answered:

1. How are the skill levels of students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving?

2. Do the skill levels of students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving differentiate by gender?
3. Do the skill levels of students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving differentiate by their grades?
4. Do the skill levels of students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving differentiate according to their state of having pre-knowledge about programming?
5. Is there a statistically significant relationship between the skill levels of students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving?
6. What are the experiences and views of students regarding the programming training process with robotic activities?

Conceptual Framework

Relationships between Variables

CT includes problem solving based on the concepts in computer sciences, system design and understanding human behavior reflected (Wing, 2006). CT skills can be regarded as thinking processes on formulation of the problems. Thus, these processes can be represented by problem-solving stages and algorithms (Aho, 2012). For this reason, CT skills and problem solving skills are combined and considered highly related variables. These skills include several stages of thinking and taking a role which may offer a solution for certain real world issues lying beyond programming (Faber, Wierdsma, Doornbos, van der Ven, & Vette, 2017). Reflective thinking is defined as thinking process which ensures clarity and consistency for a situation of perplexity (Dewey, 1933). The students not only address various problems in creative and demonstrative manner but also synthesize various research and observation activities. In robotic programming, reflective thinking skills based on problem solving have a significant importance as they materialize the reflections of an individual’s learning in a physical environment following the process of abstract programming (Yildiz-Durak, 2018c). On the other hand, when learners create solutions for their robotic programming problems, they need programming self-efficacy to overcome these problems through critical perspective, CT, problem solving and reflective thinking skills (Yildiz-Durak, 2018a).

What is CT?

CT is a skill that is included in computer science and considered important (Yildiz-Durak & Saritepeci, 2018). The skill that used to be expressed as an algorithmic thinking in the beginning has expanded in the course of time and become a basic skill where various top-level skills are used together (Hsu, Chang, & Hung, 2018) and which needs to be acquired by everyone (Wing, 2006). As this view has been accepted by many, The International Society for Technology in Education [ISTE] (2016), which has developed the standards for teachers and students to use the technology in learning-teaching processes, has included CT in basic skills to be acquired by students.

The scope of CT is not limited to brain's problem solving processes or processes administered by the computers for processing information. CT involves the whole of computing processes (Hsu, Chang, & Hung, 2018). According to Wing (2006), the children should be taught not only reading, writing and arithmetical ability but also the contribution of the information to problem solving process and how to apply CT skills and make logical analysis during early childhood education. For comprehension of CT skills by children, transformation process of a problem to a solvable one by describing it with his/her own expression is required as well as the use of basic concepts of computer sciences in this process. CT involves the skills for solving CT-related problems through a method which is similar to the method used by computer scientists and for transformation of the problem to a system, solution process of which becomes understandable by the individuals (Wing, 2006; Yildiz-Durak, 2018c). In brief, CT is a problem solving process of people. CT not only copies thinking method of the computer in problem solving process but also supports logical and creative solutions of people with ICT (Yildiz Durak & Saritepeci, 2018).

CT in Robotic Programming Training

It is possible to state that computational thinking is a problem solving process and a way of thinking which generates designs with technological tools for solving problems (ISTE, 2016; Wing, 2014). In broad terms, computational thinking is expressed as a reflection of various skills of the 21st century, such as algorithmic thinking, problem solving, abstract thinking, creative thinking and critical thinking (Basogain, Olabe, Olabe, Maiz, & Castaño, 2012). Wing (2006, 2008), on the other hand, describes computational thinking as a form of analytical thinking displayed for solving a problem, revealing system designs to develop a solution for the problem and understand the behaviors of a person who presents a pattern concerning basic concepts for data processing. Bundy (2007) suggests that CT is used via problem solving processes in other disciplines and this skill is indispensable for every discipline.

Considering the advantages of computational thinking in the learning-teaching process and daily life; the importance of providing these skills for the individuals increases more than ever. It is believed that programming training plays an important role in enabling individuals to gain computational thinking skills (Boechler, Artym, Dejong, Carbonaro & Stroulia, 2014; Pellas & Peroutseas, 2016). At this point, examining curriculum change in order to ensure CT at the K12 level, it is seen that programming training comes into prominence (Bocconi, Chioccariello, Dettori, Ferrari, & Engelhardt, 2016). This shows a parallelism with a determination in the literature suggesting that activities for the use of production-based technology, especially activities aimed at programming training support the development of CT skills (Boechler et al., 2014; Lee, Martin & Apone, 2014). In addition, even though programming is an important part of CT, it is possible to state that CT has a broader scope (Bocconi et al., 2016). On the other hand, programming is an efficient tool for concretizing and teaching the concepts regarding CT in the process of acquiring CT skills (Bocconi et al., 2016; Sarıtepeci & Durak, 2017).

Programming Self-Efficacy

According to Bandura (1997), self-efficacy is related to student's beliefs and motivation research interest. Self-efficacy refers to the belief in one's ability to perform actions that are identified before or required to achieve a specific goal or to solve a problem (Yang & Cheng, 2009). Schunk, Meece and Pintrich (2014) point out that self-efficacy is a significant variable for perception of one's task selection, efforts and success. Research refers to the significance of self-efficacy in transactions conducted with the help of digital technologies (Cassidy & Eachus, 2002) In fact, Bandura (1997) puts emphasis on the presence and importance of field-specific self-efficacy beliefs. In other words, one's perception of skills and expectations about his/her performance in a certain field as ICT may differ from his/her perception of any field beyond the scope of ICT. For this reason, analysis of programming self-efficacy is significant. Programming self-efficacy is the level of the efforts shown for programming issues and embracing the effect of learning motivation and attitude towards programming. Also, it is seen that self-efficacy perception of the students for solving programming based problems assures increase in programming success (Yagci, 2016)

Programming Self-Efficacy in Robotic Programming Training

One of the prominent elements encountered in programming training for children is programming self-efficacy. Self-efficacy affects an individual's selection of activity to accomplish a mission, level of effort she or he makes, resistance in coping with difficulties and performance (Bandura, 1977). Programming self-efficacy is believed to be a key variable for learning in the programming process, which is considered a complex and difficult process. In the study conducted by Hongwarittorrn and Krairit (2010), negative attitudes of students toward programming training and lower self-efficacy were described as an obstacle in programming training.

Ramalingam, LaBelle and Wiedenbeck (2004) examined the effect of past experiences, self-efficacy and mental models on programming performance and concluded that past experiences and mental models of individuals were directly associated with their self-efficacy, which significantly predicted programming performance. In programming training, it is important for students to take an active part in the application and sustain information during transfers and activities. Considering that there is a close relationship between programming self-efficacy and programming training (Davidson, Larzon, & Ljunggren, 2010); programming self-efficacy was included in this study as it was considered important for achieving programming activities in programming education.

Reflective Thinking Aimed at Problem Solving

Kizilkaya and Askar (2009) suggest that the best way to show reflective thinking skills is problem solving process. Problem solving skills should be improved to learn programming processes (Antonakos, 2016). In addition, reflective thinking comes out when a problem is perceived (Shermis, 1992)

According to Dewey (1933), reflective thinking is defined as active, constant and careful thinking of an issue. Rodgers (2002) emphasizes that reflection is an alternate process from practice to theory and vice versa. In robotic programming teaching process, new opportunities are provided for students to reflect their own learning. Thus, students can identify their learning targets and feel responsibility for learning, correct their mistakes, motivate themselves by noticing positive behavior and explain their opinion straight out (Yildiz-Durak, 2018c). The reflection is defined as expression of students with various methods on how they structured their own learning process as a result of the experiences gained.

Reflective Thinking Aimed at Problem Solving in Robotic Programming Training

In programming processes, it is not enough to know programming codes while solving the problems. It is required to develop problem solving skills in to learn programming processes, (Altun & Mazman, 2012) when programming training is evaluated as a process comprising problem solving stages (Gomes & Mendes, 2007). On the other hand, problem solving is among the most important skills of individuals in the 21st century and reflective thinking emerges only when a certain problem is perceived. Thus, it is possible to state that reflection will be observed in the problem solving process the best (Kizilkaya & Askar, 2009). The study embraces reflective thinking aimed at problem solving as it is considered an important skill in the programming training process.

Use of Robotic Coding Activities in Programming Training

Robotic is a strong and flexible education tool which enables students to conduct both robotic programming and control activities by using special programming tools (Alimisis, 2013). In robotic activities, students design to handle their complex problems and receive immediate feedback about the outputs of the programs they write by testing their solutions (Atmatzidou, Demetriadis, & Nika, 2018). By this way, students learn how to cope with difficult situations within the context of the real world.

Today, robotic codings and robotic kits are increasingly getting popular in all stages of K12 (Eguchi, 2014; Rogers, Wendell, & Foster, 2010; Saritepeci & Durak, 2017). Robotic coding enables learners to learn sensors, motors, programming and digital area (Bers, 2010). In this study, the robotic concept expresses the development of interactive automatic systems using coding tools (scratch) of sensors with a physical programming platform (Arduino) rather than robotic kits. Learners can develop systems that perceive the world around them and move accordingly by using the aforementioned sensors, physical programming platforms and programming cycles (Bers, 2010). By this way, it is possible to provide learners a more flexible learning process that supports their creativity. In addition, it is believed that using such a structure will remove limitations in robotic kits and contribute to the diversification of real life problems even further in the learning process. Accordingly, it is possible to state that including elements like block-based programming and robotic coding in programming training conducted at the K12 level will contribute to the learners to acquire CT skills or develop the skills and programming self-efficacy, and acquire reflective thinking skills aimed at problem solving (Brennan & Resnick, 2012; Lye & Koh, 2014).

Method

Mixed model was used in the study aiming to determine the skill levels of secondary school students regarding CT, programming self-efficacy and reflective thinking aimed at problem solving in the programming training process conducted with robotic activities and examine their experiences in this process.

Current mixed research was designed in line with “explanatory sequential mixed methods design” which is one of the research strategies. Creswell (2012) points out that using quantitative and qualitative method together may help better understanding of research problem when compared to the use of both approaches separately. In this study, quantitative data were collected from the students. Being one of the qualitative research methods, interview technique was applied for a detailed observation on application process. A total of 40 students were interviewed within the scope of this method. Various scales were applied in the quantitative dimension. On the other hand, a semi-structured interview form, which was developed by the researchers, was applied in the qualitative dimension.

Study Group

The study participants consisted of 55 students from 6th and 7th grades receiving education at a secondary school in one of the provinces located in Western Black Sea region in Turkey during the school year of 2017–2018. Of the total, 54.5% of the participants were male while 45.5% of the participants were female. Participation rate for 6th grade students was 49.1% and 50.9% for 7th grade students; 36.4% of the participants received programming training before.

Data Collection Tools

In the present study, five different data collection tools were used. Data collection tools were employed on the 10th week of the application during the fall semester of the 2017-2018 school year.

Personal Information Form: The form was developed by the researchers. Personal information data of the participants were collected by using this data collection tool consisting of five items. Questionnaire items differentiated according to types of questions and were generally in the Likert structure.

Scale for Reflective Thinking Skills Aimed at Problem Solving: The scale was developed by Kizilkaya and Askar (2009). This five point Likert scale consists of a total of 14 items and 3 factors. In this three-factor scale, subdimension of “Questioning” consists of 5 items; subdimension of “Evaluation” 5 items and subdimension of “Causation” 4 items. The 5-point Likert type scale is based on gradation of answers as “Always”, “”Usually”, “Occasionally”, “”Rarely”, “Never”. In the study, factor loadings of scaling tool range between 0.49 and 0.77. Internal consistency of the scale, namely Cronbach alfa coefficient was found as 0.85 and the results of split half method was found as 0.94. These results show that internal consistency value of scaling tool is high. Exploratory factor analyses were conducted for identification of structural validity of the scale and a single scale structure was developed.

Computational Thinking Scale: The scale was developed by Korkmaz, Cakir and Ozden (2016). It consists of a total of 22 items and five factors. In this five-factor scale, subdimension of “Creativity” consists of 4 items; subdimension of “Algorithmic Thinking” 4 items; subdimension of “Collaboration” 4 items; subdimension of “Critical Thinking” 4 items and subdimension of “Problem Solving” 6 items. In the study, the Cronbach alpha reliability coefficient was calculated as .89 for the scale. The 5-point Likert type scale is based on gradation of answers as “Strongly Disagree (1)” … “Strongly Agree (5)”. It is seen that factor-total correlation of all factors in the scale range between .48-.73 and t values are significant ($p < .001$) There results are interpreted as follows: Validity level of the factors in the scale is higher and these factors are aimed to measure the same behavior. On the other hand, the Cronbach alpha reliability coefficient varied between .78 - .94 for the subscales.

Programming Self-Efficacy Scale for Secondary School Students: The scale was developed by Kukul, Gokceearslan and Gunbatar (2017). It consists of a total of 31 items and single factor. The scale aims to measure the programming self-efficacy levels of students during the training of programs like Scratch, Logo, Alice, which are used by teachers and researchers for introducing programming skills for children. This five point Likert scale includes no reverse items and is graded as "Strongly Disagree (1)" and "Strongly Agree (5)". In the study, factor loadings of scaling tool range between 0.49 and 0.77. Internal consistency of the scale, namely Cronbach alfa coefficient was found as 0.85 and the results of split half method was found as 0.94. These results show that internal consistency value of scaling tool is high. Exploratory factor analyses were conducted for identification of structural validity of the scale and a single scale structure was developed.

Semi-Structured Interview Form: In this data collection tool developed by the researchers, 11 questions were addressed to the participants regarding "the effect of the process on skill development, interaction, opinion on the programming process, favored and disfavored aspects of the process, opinions of students on process experiences". In order to ensure validity and reliability during the development process of this data collection tool, opinions were received from two domain experts and the data collection tool was arranged. Field experts are the persons who gained their undergraduate, master and doctoral degrees in the department of Computer and Instructional Technologies. Field experts made suggestions on the content of the questions and way of questioning. It was suggested that 5 questions in the form which has 15 questions in total are combined in 1 question and the form was rearranged accordingly.

Application Process

The method was applied in a secondary school in one of the provinces located in the Western Black Sea region of Turkey during the fall semester of the 2017-2018 school year. Data collection tools were implemented in the 10th week of the operation.

The application was carried out during "information technologies and software" course which includes teaching programming. The application took 2 hours weekly. The application was carried out 3 days in a week with 3 groups. Each group involved 15-20 students. The students were divided into small cooperative groups composed of 2-3 students for the application. The teacher who carried out the application established the groups. The teacher pointed out that he/she formed the groups considering fellowship of the students. Also, there is a robotic programming workshop in the school where the application was conducted. This occasion prevented occurrence of some problems such as a mess of equipment and vacancy in the course of application. Teaching programming concepts effectively was aimed within the scope of the course. In this context, science-related course subjects such as electricity, electric circuits, and voltage, current or electrical elements were not addressed in detail during programming activities with robotic.

Scratch, a programming environment, was introduced in the first week of application process. The students were instructed to carry out the first coding activities. Scratch for Arduino program was introduced afterwards. Scratch for Arduino program is a Scratch modification which offers new blocks for managing sensors and actuators connected to Arduino. In the second week, the structure of Arduino was introduced and the studies were conducted to connect Arduino to the computer. Various activities were performed by using Arduino from the 3rd week of application to the 10th week.

The activity of the tenth week took 1 hour. Data collection tools were applied following the activities. Application of data collection tools took more than 1 hour.

Data Analysis

Quantitative and qualitative data were analyzed by using the SPSS and Nvivo programs. Quantitative data were analyzed using the Mann Whitney U, Spearman Rho correlation test and descriptive statistics. Before the appropriate analysis, the data were examined in terms of normality, kurtosis, skewness coefficients and homogeneity.

Kurtotis values ranged between -3.5 and 7.0; skewness values ranged between -4.0 and 6.0. Homogeneity of variances of scores relating to dependent variable for each group was tested by Levene's test. In this technique, if p value is higher than 0.05 ($p>0.05$), the variances are regarded as homogenous (Field, 2009). The observed p value was found to be significant ($p< 0.05$).

As normality values required for parametric tests could not be attained, non-parametric tests were used. Qualitative data, on the other hand, were analyzed using the content analysis method. The data collected through interview forms were examined under themes and codes. A part of the data was coded by two coders in order to ensure reliability between the coders. In the coding that was performed by the two researchers for reliability, the consistency between the coders was calculated as 90.2% (reliability=(consensus number)/(total consensus+dissensus number)) (Miles & Huberman, 1994). When evaluating the remarks of the students, abbreviation S1…S40 was used to refer to the "students".

Findings

Table 1 demonstrates descriptive findings concerning the skill levels of students regarding CT, programming self-efficacy and reflective thinking aimed at problem solving.

Table 1. Descriptive Statistics

Scales	Items	Min. Score	Max. Score	Mean	Mean /k*	Sd
Computational Thinking	22	51.00	110.00	77.20	3.51	12.26
Creativity	4	5.00	20.00	15.08	3.77	.79
Algorithmic Thinking	4	7.00	20.00	14.24	3.56	.81
Collaboration	4	7.00	20.00	15.16	3.79	.89
Critical Thinking	4	7.00	20.00	14.32	3.58	.86
Problem Solving	6	6.00	30.00	18.30	3.05	1.02

Programming Self-Efficacy	31	89.00	154.00	110.53	3.57	16.15
Reflective Thinking Aimed at Problem Solving	14	32.00	70.00	50.02	3.57	9.11
Questioning	5	11.00	25.00	18.15	3.63	.72
Evaluation	5	11.00	25.00	18.35	3.67	.75
Causation	4	6.00	25.00	13.52	3.38	.83

**k: number of items*

According to Table 1, the mean score for computational thinking skills of the students is 77.2. Considering the subscales of the overall CT scale, it is found out that the mean scores of the cooperation subscale are higher than other scales. The mean scores obtained from the problem solving subscale are the lowest. The mean score for programming self-efficacy of the students is 110.53. The mean scores of students regarding programming self-efficacy and reflective thinking aimed at problem solving are 50.02. When subdimensions of this scale are examined, it is seen that the mean scores obtained by students from the subscale of evaluation are higher than other subscales; the mean scores obtained from the subscale of causation are lower than other subscales.

Table 2 shows the results of the Mann Whitney U test concerning the skill levels of students in terms of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving by gender variance.

Table 2. The results of Mann Whitney U Test Relating to Gender Variance

Variables	**Gender**	**f**	**Mean Rank**	**Sum of Ranks**	**U**	**Sig.**
Computational Thinking	Female	25	31.28	782.00	293.00	.165
	Male	30	25.27	758.00		
Programming Self-Efficacy	Female	25	30.18	754.50	320.50	.357
	Male	30	26.18	785.50		
Reflective Thinking Aimed at Problem Solving	Female	25	33.78	844.50	230.50	.014
	Male	30	23.18	695.50		

According to Table 2, it is seen that there is no statistically significant difference between the skill levels of students regarding computational thinking and programming self-efficacy by gender (U_{CT}= 293.00, $p_{CT} \geq .05$; U_{PSE}= 320.50, $p_{PSE} \geq .05$). Although significant difference doesn’t come out, computational thinking skill levels and programming self-efficacy of female are found to be relatively higher when compared to male, considering the mean rank of the groups’ reflective thinking aimed at problem solving skill differs significantly by gender (U_{RTPS}= 230.50, $p_{RTPS} < .05$) Considering mean rank of the groups, reflective thinking aimed at problem solving skills of female are higher when compared to male.

Table 3 illustrates the results of the Mann Whitney U test concerning the skill levels of students in terms of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving by class level variance.

Table 3. The Results of Mann Whitney U Test Related to Class Level Variance

Variables	**Grade**	**f**	**Mean Rank**	**Sum of Ranks**	**U**	**Sig.**
Computational Thinking	6th grades	27	23.19	626.00	218.00	.002
	7th grades	28	32.64	914.00		
Programming Self-Efficacy	6th grades	27	20.11	543.00	165.00	.000
	7th grades	28	35.61	997.00		
Reflective Thinking Aimed at Problem Solving	6th grades	27	25.11	678.00	300.00	.189
	7th grades	28	30.79	862.00		

According to Table 3, there is a statistically significant difference between the skill levels of students regarding computational thinking and programming self-efficacy by class level variance; on the other hand, there is no statistically significant difference in the skill level of reflective thinking aimed at problem solving by class level variance (U_{CT}= 218.00, p_{CT}< .05; U_{PSE}= 165.00, p_{PSE}< .05; U_{RTPS}= 300.00, p_{RTPS}≥ .05). Considering the ranking of the mean scores of the groups; it is seen that 7th grade students have higher skill levels of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving than 6th grade students.

Table 4 shows the results of the Mann Whitney U test concerning the skill levels of students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving according to their state of having pre-knowledge about programming.

Table 4. The Results of Mann Whitney U Test Related to Pre-Knowledge on Programming

Variables	**Having Pre-knowledge about Programming**	**f**	**Mean Rank**	**Sum of Ranks**	**U**	**Sig.**
Computational Thinking	There is Pre-knowledge	20	24.68	493.50	283.50	.244
	There is no Pre-knowledge	35	29.90	1046.50		
Programming Self-Efficacy	There is Pre-knowledge	20	26.08	521.50	311.50	.500
	There is no Pre-knowledge	35	29.10	1018.50		
Reflective Thinking Aimed at Problem Solving	There is Pre-knowledge	20	24.48	489.50	279.50	.217
	There is no Pre-knowledge	35	30.01	1050.50		

According to Table 5, it is seen that there is no statistically significant difference between the skill levels of students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving according to their state of having pre-knowledge about programming (U_{CT}= 283.50, p_{CT}≥ .05; U_{PSE}= 311.50, p_{PSE}≥ .05; U_{RTPS}= 279.50, p_{RTPS}≥ .05). Even though there is no significant difference, considering the ranking of the mean scores of the groups; it is seen that students without pre-knowledge about programming have higher skill levels of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving than students with pre-knowledge. It is an interesting finding. The reason of this situation might be associated with the fact that students who see programming for the first time have a higher effort to learn programming.

Table 5 demonstrates the results of the correlation analysis that was conducted for the purpose of determining whether or not there was a statistically significant relationship between the skill levels of students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving.

Table 5. Results of the Correlation Analysis

		1	2	3
Computational Thinking	Spearman's rho	1.00	.474**	.542**
	Sig. (2-tailed)	-	.000	.000
Programming Self-Efficacy	Spearman's rho	.474**	1.00	.463**
	Sig. (2-tailed)	.000	-	.000
Reflective Thinking Aimed at Problem Solving	Spearman's rho	.542**	.463**	1.00
	Sig. (2-tailed)	.000	.000	-

According to Table 5, it is seen that there is a positive, moderate and significant relationship between computational thinking, programming self-efficacy and reflective thinking aimed at problem solving ($r_{ct\text{-}pse}$ = .474, $p_{ct\text{-}pse}$ <.01; $r_{ct\text{-}rtps}$ = .542, p $_{ct\text{-}rtps}$<.01). On the other hand, it is seen that there is a positive, moderate and significant relationship between programming self-efficacy and reflective thinking aimed at problem solving ($r_{pse\text{-}rtps}$ = .463, p $_{pse\text{-}rtps}$ <.01). From this point of view, the increase of programming self-efficacy will affect reflective thinking aimed at problem solving positively.

Robotic Activities of Students and Their Experiences and Opinions about the Programming Training Process

Opinions of students about the programming training process conducted with robotic activities were examined in line with the study question. Table 6, Table 7 and Table 8 show the themes examining the opinions of students about robotic activities being designed, as well as codes under the themes and coding frequencies of coding these codes. Additionally, as students expressed their opinions on more than one subject during interviews, the number of frequencies does not coincide with the number of students.

Table 6. Opinions of Students about the Educational Contributions of Robotic Activities Conducted in Programming Training

Themes	Codes	Subcodes	f
Educational contributions of robotic activities in the programming process	Contributions to learning basic programming concepts	Cycles	15
		Movement instructions	7
		Variables	5
		Perceiving	3
		Logic of programming	3
		Term structures	2
		Repetition instructions	2
	Contributions to using information technologies	Technical skills	4
		Technological literacy (Goal-oriented hardware and software utilization)	3

	Operating logic of a computer	3
	Making a search on the web	1
Contributions to acquiring and developing different information and skills	Problem solving process	5
	Logical thinking	5
	Algorithmic thinking	5
	Creative thinking	4
	Communication	4
	Making a search / inquiring	3
	Sharing / working with a group	3
	Mechanical	3
	Mathematical position / direction knowledge	2
	Use of the Scratch program	2
Contributions to forming cooperative groups	Increase of interaction with friends / cooperation	4
	Receiving help in difficult subjects	3
	Trying / daring to try to use different solutions	2
	Increasing the problem solving skill	2
	Generating / sharing opinions	2
	No contribution	1

*For giving a better reflection, subcodes were assigned depending on the codes *

Examining the student opinions in Table 6; it is seen that students generally consider robotic activities conducted in programming training a process that enables them to learn programming concepts and develops multiple skills. Examining the codes and subcodes; it is seen that students mainly express contributions for learning basic programming concepts (f=37) while explaining the contributions of programming activities conducted with robotic. They express contributions for acquiring and developing different information and skills (f=36), contributions for forming cooperative groups (f=12) and contributions for using information technologies (f=11), respectively. Remarks of the participants are as follows:

Learning with robotic helped me learn movement, programming, mechanics, perception, moving to different directions and angles, variables, loops, software and hardware and file creation process. S11

I learned programming, game design, some functions of the computer and how they operate.

I learned some information on Scratch and programming. I learned Scratch Arduino, wiring, movements, locations, perception (e.g. move when it touches) making a character move, sensors etc.) S5

I learned programming, game design and some functions of a computer. It helped me develop problem solving and research skills. S18

I learned problem solving process and logical thinking. S14

I learned creative thinking, problem solving process and communication. It helped me develop logical and creative thinking as well as sharing and cooperation. S22

Table 7. Opinions of Learners about Motivational Elements of Robotic Activities Conducted in Programming Training

Themes	Codes	Subcodes	f
Motivational Elements of Robotic Activities	Elements Regarding Content	Learning new things	5
		Learning computer	3
		Learning coding	3
	Elements Regarding Learning Environment	Entertaining	7
		Cooperative work	2
		Intriguing	2
	Elements Regarding the Method Being Used	Opportunity of learning from peers	2
		Practicing	2
	Elements Regarding Learning Instruments Being Used	Using computer	3
		Using robots	2
		Using the Scratch	2

It is seen in Table 7 that robotic activities conducted in programming training have motivational elements regarding all aspects of their educational status. When the codes and subcodes are analyzed; students mainly express elements regarding content (f=11) and learning environment (f=11), while explaining the motivational elements of programming activities conducted with robotics. They also express elements regarding learning instruments used (f=7) and elements regarding the method employed (f=4), respectively. Remarks of the participants are as follows:

We solved problems and gained new skills. Thus, the lesson was enjoyable. S2

It was enjoyable and good because I used Arduino for robotics. I learned how Arduino functions. S9

This lesson helped me understand the computer more precisely. Thanks to teamwork and team spirit, I like the lesson ever more.S10

Enjoyable activities, opportunity to use technology, working with classmates in groups and building a robot with Arduino…S40

Table 8. Opinions of Learners about the Lesson Where Programming Training is Conducted

Themes	Codes	Subcodes	f
Opinions about the Lesson Where Programming Training is Conducted	Positive/Favored Aspects of the Lesson	Moving robots	5
		Practicing	5
		Learning new things	5
		Using computer	2
		Scratch program	2

Negative/Disfavored Aspects of the Lesson	Having a difficulty in some subjects	3
	Tiring activities	2
	Obligation of making a great effort for learning	2
	Fewer lesson hours	1
Difficulties Faced in the Lesson	Failing to understand subjects / Difficulty of coding	5
	Failing to generate an appropriate code	4
	Failing to determine errors in coding	2
	I have had no difficulty	20
State of Demanding the Use of Robotic Activities in Other Lessons	Yes	44
	No	4

When the students' opinions are examined in Table 8, it is seen that students like programming activities conducted with robotics mainly due to moving robots (f=5), practicing (f=5) and learning new things (f=5). Then it is found out that students like these activities due to using computer (f=2) and the Scratch program (f=2), respectively. When the students' opinions regarding the process are examined; it is seen that students dislike the activities due to having a difficulty in some subjects (f=3), tiring activities (f=2), obligation of making a great effort for learning (f=2) and fewer lesson hours (f=1).

Even though majority of students reported that they had no difficulty in the lesson (f=20), some of them faced difficulties in understanding subjects/coding (f=5), generating an appropriate code (f=4) and determining errors in coding (f=2). In addition, majority of the students (f=44) demanded the use of robotic activities in other courses. Remarks of the participants are as follows:

> *Moving something with commands. However; I was so tired at the end of the course.S18*
>
> *We used Arduino with my classmates. S31*
>
> *We didn't perform many activities. Course hours were short. S21*
>
> *The negative side of the courses was its complexity. At the beginning, I had difficulty in understanding the course. Also, there were so many item names which I had difficulty in learning. S35*
>
> *The course was so enjoyable. I wish I could attend the course the next year. This course helped me learn using computer. It is also necessary for my success. S38*

Conclusions and Discussion

The purpose of this study is to determine the skill levels of secondary school students regarding computational thinking, programming self-efficacy and reflective thinking aimed at problem solving in the programming training process conducted with robotic activities and to examine their experiences and opinions about the process.

As a result, female students were observed to have higher skill levels of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving than male students. One of the reasons of this situation is the change of students' perceptions of programming self-efficacy relating to gender variance. On the other hand, it is pointed out that during the activities conducted by Yildiz-Durak (2018b) by using Scratch, secondary school students' problem solving, algorithmic thinking and programming self-efficacy is related to readiness of the students for designed activities. Rusk, Resnick, Berg, and Pezalla-Granlund (2008) state that perception of activities and its effects are related to gender as a result of findings on determination of students' living. When qualitative data are analyzed with a holistic approach, it is seen that female students expressed their opinion mostly on qualifications of the process; however, male students emphasized the points which relate to the content.

In the literature, there are studies emphasizing that demographic characteristics of individuals are associated with performances displayed in computer-based environments (Boechler et al., 2014; Lee, Martin, & Apone, 2014). Especially the variable of gender is associated with many variables like the frequency of using ICT, experience of using ICT, attitude toward ICT, programming self-efficacy, academic achievement, computational thinking and problem solving and the results mostly show that females have higher scores than males (Askar & Davenport, 2009; Byrne & Lyons, 2001; Crews & Butterfield, 2003; Gurer & Camp, 2002; Nourbakhsh, Hamner, Crowley & Wilkinson, 2004; Saritepeci & Durak, 2017; Román-González, Pérez-González & Jiménez-Fernández, 2017; Werner, Denner, Campe, & Kawamoto, 2012; Yildiz-Durak & Saritepeci, 2018). According to one of the findings of the study, females have higher skill levels of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving than males, which shows a parallelism with the findings of many studies in the literature. On the other hand, some studies suggest that the underlying reason is differentiation of programming thinking in relation to gender variance, self-belief, motivation and anxiety levels (Cegielski & Hall, 2006; Wiedenbeck, 2005).

Computational thinking and programming self-efficacy levels of students differentiate according to their grades. It was found out that 7th grade students had higher skill levels of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving when compared to the related skill levels of 6th grade students. As educational level differentiates especially many cognitive skills depending on the variation in using ICT, it is thought that the difference will change the levels of programming performance, problem solving and CT skills, either directly or indirectly (Askar & Davenport, 2009; Yildiz- Durak & Saritepeci, 2018). As a consequence, educational level will be an effective variable on performance in programming processes that contain many skills and tasks that require using CT skills.

It is observed that students who have no preliminary information about programming have higher levels of computational thinking, programming self-efficacy and reflective thinking aimed at problem solving than students who have preliminary information. According to Maloney, Resnick, Rusk, Silverman, and Eastmond (2010) main objective of block-based programming environment is to provide single screen single user interface and simple command set having language option by which amateurs take design decisions on their own (e.g. selection of virtual blocks).Thus, Scratch programming environment and virtual design helps students express themselves without difficulty and discover the potential of easy learning. In addition to this, in the course of the study in which robotic programming activities conducted by Kasalak (2017) with the participation of secondary school students, simple block-based programming self-efficacy perception of the students differs from complex block-based programming self-efficacy perception, depending upon programming achievement of the students or taking Scratch programming course before. The reason of this situation is explained as follows: Simple block-based programming tasks enable students to comprehend programming concepts. Similarly, the study conducted by Resnick et al. (2009) suggests that problem solving process may be learned easily without the need for special education or support in block-based programming. Thus, it can be inferred that students' programming self-efficacy perception and the level of reflective thinking aimed at problem solving differ depending on the difficulty perceived by the students on programming tasks. The current study refers that perception of learning tasks as difficult and complex may regarded as an advantage and supportive process for the students in terms of learning; however, some studies suggest the vice versa.

There is a positive and moderate relationship between computational thinking, programming self-efficacy and reflective thinking aimed at problem solving, which can be explained with the relationship that is asserted to be between programming training, problem solving and computational thinking in the literature (Bocconi et al., 2016; ISTE, 2016; Wing, 2014). This relationship can be explained with the fact that students who train robotic coding interact with unstructured problems related with real life in robotic coding, show an interest to robotic activities and make a greater mental effort with creative and cooperative studies. Regarding this assessment; Bers, Flannery, Kazakoff, and Sullivan (2014) suggest that students who learn with robotic coding are actively involved in the problem solving process and display skills related with the basic concepts of computational thinking throughout this process. In addition, the fact that learners' problems that are expected to be solved in the robotic coding process require a more complex and interdisciplinary study (Calder, 2010; Gulbahar & Kalelioglu, 2014; Lai & Yang, 2011; Liu, Cheng & Huang, 2011) might have also caused the difference.

When the opinions of the students are analyzed; it is seen that students generally consider robotic activities conducted in programming training as a process that enables them to learn programming concepts and develops multiple skills. It is also obvious that robotic activities conducted in programming training have motivational elements regarding all elements of their educational status. It is indicated that students like programming activities conducted with robotics mainly due to moving robots, practicing and learning new things and dislike due to having a difficulty in some subjects, challenging activities, spending a great effort for learning and fewer lesson hours.

Limitations and Recommendations

Programming teaching activities were conducted within the scope of this study. This study has some limitations. One of the limitations of the study is that robotic programming activities planned within the scope of information technologies and software course are oriented towards solely programming achievements. As it is understood from the remarks of the students, they had difficulty in comprehending some subjects such as installing electric circuit and electric current etc. From this point of view, new research may be conducted for achievements regarding the electricity in the future. Thus, robotic programming activities may be associated to different disciplines. Another limitation of the study is that some students understand same process designed in line with qualitative data better than the other students. Further and varying course contents may be investigated in future studies for the students displaying different level of perception. On the other hand, it can be suggested that when establishing cooperative groups, the students who have difficulty in performing activities may be paired with the students performing the activities with relative ease following a certain observation period

In this study, it is found out that the students who didn't have prior information on programming displayed higher levels on programming computational thinking, programming self-efficacy and reflective thinking aimed problem solving when compared to the students who had prior information on programming. The underlying cause of this situation is explained in the literature as programming tasks which may be perceived as simple or complex. Some further research can be conducted on the experiences of the students who have/don't have prior information on programming in the course of programming training which covers simple and complex tasks and it can be suggested that qualitative research is carried out to find out how prior information affect learning robotics programming concepts. Experimental studies can be conducted to examine the effects of robotics programming activities on students' computational thinking, self-efficacy perception and reflective thinking aimed at problem solving depending upon students' experiences.

Acknowledgements

This study was funded by Bartin University (2017-SOS-A-013).

References

Adleberg, B. M. (2013). *Scratch programming and remix culture: Gender differences in interaction and motivation for pre-adolescents* (Unpublished master's thesis). Georgetown University, Washington, D.C.

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal*, *55*(7), 832-835.

Alimisis, D. (2013). Educational robotics: Open questions and new challenges. *Themes in Science and Technology Education*, *6*(1), 63-71.

Altun, A. & Mazman, S. G. (2012). Developing computer programming self-efficacy scale. *Journal of Measurement and Evaluation in Education and Psychology*, *3*(2), 297-308.

Antonakos, J. L. (Ed.). (2016). *Computer technology and computer programming: Research and strategies*. Boca Raton, Florida: CRC Press.

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). Retrieved on 22 October 2018 from http://files.eric.ed.gov/fulltext/ED503900.pdf

Aslan, U. (2014). *Fostering students' learning of probability through video game programming* (Unpublished master's thesis). Bogazici University, Istanbul.

Atmatzidou, S. & Demetriadis, S. N. (2012, July). Evaluating the role of collaboration scripts as group guiding tools in activities of educational robotics: Conclusions from three case studies. In *Advanced Learning Technologies (ICALT), 2012 IEEE 12th International Conference on* (pp. 298-302). IEEE.

Atmatzidou, S., Demetriadis, S., & Nika, P. (2018). How does the degree of guidance support students' metacognitive and problem solving skills in educational robotics? *Journal of Science Education and Technology*, *27*(1), 70-85.

Bandura, A. & Wessels, S. (1997). *Self-efficacy*. New York: W.H. Freeman & Company.

Basogain, X., Olabe, M. A., Olabe, J. C., Maiz, I., & Castaño, C. (2012). Mathematics education through programming languages. In *21st annual world congress on learning disabilities* (pp. 553-559). Oviedo, Spain: agapea.com.

Bers, M. U. (2010). The TangibleK robotics program: Applied computational thinking for young children. *Early Childhood Research & Practice*, *12*(2), 1-20.

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education*, *72*, 145-157.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). *Developing computational thinking in compulsory education - European Commission JRC science for policy report*. Luxembourg: European Union.

Boechler, P., Dragon, K., & Wasniewski, E. (2014). Digital literacy concepts and definitions: Implications for educational assessment and practice. *International Journal of Digital Literacy and Digital Competence (IJDLDC)*, *5*(4), 1-18.

Brennan, K. A. (2013). *Best of both worlds: Issues of structure and agency in computational creation, in and out of school* (Unpublished doctoral dissertation). Massachusetts Institute of Technology, Cambridge, MA.

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 (pp.1-25).* Vancouver, Canada: AERA.

Bundy, A. (2007). Computational thinking is pervasive. *Journal of Scientific and Practical Computing*, *1*(2), 67-69.

Burke, W. Q. (2012). Coding and composition: *Youth storytelling with Scratch programming (Unpublished d*octoral dissertation). Available from ProQuest Dissertations and Theses database. (UMI No. 3510989).

Buyukozturk, S., Cakmak, E. K., Akgun, O. E., Karadeniz, S. & Demirel, F. (2013). *Scientific research methods*. Ankara: Pegem Academy.

Byrne, P., & Lyons, G. (2001, June). The effect of student attributes on success in programming. *ACM SIGCSE Bulletin, 33(3),* 49-52).

Calder, N. (2010). Using Scratch: an integrated problem-solving approach to mathematical thinking. *APMC 15* (4), 9-14.

Cassidy, S. & Eachus, P. (2002). Developing the computer user self-efficacy (CUSE) scale: Investigating the relationship between computer self-efficacy, gender and experience with computers. *Journal of Educational Computing Research*, *26*(2), 133-153.

Castledine, A. R. & Chalmers, C. (2011). LEGO robotics: An authentic problem solving tool? *Design and Technology Education: An International Journal*, *16*(3), 19-27.

Cegielski, C. G. & Hall, D. J. (2006). What makes a good programmer? *Communications of the ACM*, *49*(10), 73-75.

Ceylan, V., K. (2015). *Effect of blended learning to academic achievement* (Unpublished master's thesis). Adnan Menderes University, Aydin, Turkey.

Creswell, J. W. (2012). *Educational research: Planning, conducting, and evaluating quantitative and qualitative research* (4th ed.). Boston, MA: Pearson.

Crews, T. & Butterfield, J. (2003). Improving the learning environment in beginning programming classes: An experiment in gender equity. *Journal of Information Systems Education*, *14*(1), 69-76.

CSTA, (2010). *Running on empty: The Failure to teach K–12 computer science in the digital age*. Retrieved on 22 October 2018 from http://runningonempty.acm.org/fullreport2.pdf

Cetin, E. (2012). *The effect of computer programming training on children's problem-solving skills* (Unpublished master's thesis). Gazi University, Ankara.

Davidson, K., Larzon, L., & Ljunggren, K. (2010). *Self-efficacy in programming among STS students*. Retrieved on 22 October 2018 from http://www.it.uu.se/edu/course/ homepage/datadidaktik/ht10/reports/Self-Efficacy.pdf.

Dewey, J. (1933). How we think: A restatement of the relation of reflective thinking to the educative process 8(31), 360-361.

DiSessa, A. A. (2001). *Changing minds: Computers, learning, and literacy*. Cambridge, MA: MIT Press.

Dogan, V., K. (2015). *The effects of computer games development process on primary school students' critical thinking skills and algorithm achievements* (Unpublished master's thesis). Yıldız Technical University, Istanbul.

Durak, H. (2016). *Design and development of an instructional program for teaching programming process to gifted students* (Unpublished doctoral dissertation). Gazi University, Ankara.

Eguchi, A. (2010). What is educational robotics? Theories behind it and practical implementation. In D. Gibson & B. Dodge (eds.), Proceedings of Society for Information Technology & Teacher Education International Conference 2010 (pp. 4006-4014). Chesapeake, VA: AACE.

Einhorn, S. (2011). *Microworlds, computational thinking, and 21st century learning*. Logo Computer System Inc., White paper. Retrieved from on 22 October 2018 from http://www.microworlds.com/.

Faber, H. H., Wierdsma, M. D., Doornbos, R. P., van der Ven, J. S., & de Vette, K. (2017). Teaching computational thinking to primary school students via unplugged programming lessons. *Journal of the European Teacher Education Network*, *12*, 13-24.

Fairchild, A. J., Horst, S. J., Finney, S. J., & Barron, K. E. (2005). Evaluating existing and new validity evidence for the academic motivation scale. *Contemporary Educational Psychology*, *30*(3), 331-358.

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education, 63*, 87-97.

Fields, D. A., Searle, K. A., Kafai, Y. B., & Min, H. S. (2012, February). Debuggems to assess student learning in e-textiles. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 699-699). ACM.

García-Peñalvo, F.J., 2016. What computational thinking is. *Journal of Information Technology Research 9*(3), v--viii.

Gomes, A. & Mendes, A. J. (2007, September). Learning to program-difficulties and solutions. In *International Conference on Engineering Education–ICEE*. Retrieved on 22 October 2018 from https://www.researchgate.net/profile/Anabela_Gomes2/publication/228328491_Learning_to_program_-_difficulties_and_solutions/links/02e7e52389017b9984000000.pdf.

Gregg, E. A. (2014). *Teaching critical media literacy through videogame creation in scratch programming* (Unpublished doctoral dissertation). Loyola Marymount University.

Grover, S. & Pea, R. (2013). Computational thinking in K-12: a review of the state of the field. *Educational Researcher, 42*(1), 38-43.

Gulbahar, Y. & Kalelioglu, F. (2014). The effects of teaching programming via Scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education-An International Journal*, *13*(1), 33-50.

Gulmez, I. (2009). *Effects of using visualization tools in programming instruction on student success and motivation* (Unpublished master's thesis). Marmara University, Istanbul.

Gurer, D. & Camp, T. (2002). An ACM-W literature review on women in computing. *ACM SIGCSE Bulletin*, *34*(2), 121-127.

Hansen, J. B. & Toso, S. J. (2007). Gifted dropouts: Personality, family, social, and school factors. *Gifted Child Today*, *30*(4), 30-41.

Hongwarittorrn, N. & Krairit, D. (2010, April). Effects of program visualization (jeliot3) on students' performance and attitudes towards java programming. In *The spring 8th International conference on Computing, Communication and Control Technologies* (pp. 6-9). Orlando, Florida.

Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education*, *126*, 296-310.

International Society for Technology in Education (ISTE) (2016). *CT Leadership toolkit.* Retrieved on 22 October 2018 from http://www.iste.org/docs/ct-documents/ctleadershipt-toolkit.pdf?sfvrsn=4.

Ismail, M. N., Ngah, N. A., & Umar, I. N. (2010). Instructional strategy in the teaching of computer programming: A need assessment analyses. *TOJET: The Turkish Online Journal of Educational Technology*, *9*(2), 125-131.

Kasalak, I. (2017). E*ffects 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.

Kayabasi, E. (2016). *Prospective teachers' experiences on Alice: Programming in 3D environment* (Unpublished master's thesis). Uludag University, Bursa.

Kazakoff, E. R. (2015). Technology-based literacies for young children: digital literacy through learning to code. Retrieved on 22 October 2018 from http://link.springer.com/chapter/10.1007/978-94-017-9184-7_3#page-1.

Kizilkaya, G. & Askar, P. (2009). The development of a reflective thinking skill scale towards problem solving. *Education and Science*, *34*(154), 82-92.

Korkmaz, O., Cakir, R., & Ozden, M. Y. (2016). Computational thinking levels scale (CTLS) adaptation for secondary school level. *Gazi Journal of Educational Science*, *1*(2), 143-162.

Kukul, V., Gokcearslan, S., & Gunbatar, M. S. (2017). Computer programming self-efficacy scale (CPSES) for secondary school students: Development, validation and reliability. *Egitim Teknolojisi Kuram ve Uygulama*, *7*(1), 158-179.

Lai, A. F. & Yang, S. M. (2011, September). The learning effect of visualized programming learning on 6th graders' problem solving and logical reasoning abilities. In *Electrical and Control Engineering (ICECE), 2011 International Conference on* (pp. 6940-6944). IEEE.

Lau, W. W. & Yuen, A. H. (2011). Modelling programming performance: Beyond the influence of learner characteristics. *Computers & Education, 57(1),* 1202-1213.

Law, K. M., Lee, V. C., & Yu, Y. T. (2010). Learning motivation in e-learning facilitated computer programming courses. *Computers & Education*, *55*(1), 218-228.

Lawanto, K., Close, K., Ames, C., & Brasiel, S. (2017). Exploring Strengths and Weaknesses in Middle School Students' Computational Thinking in Scratch. In *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 307-326). Springer, Cham.

Lee, I., Martin, F., & Apone, K. (2014). Integrating computational thinking across the K--8 curriculum. *Acm Inroads*, *5*(4), 64-71.

Lee, Y. J. (2011). Scratch: Multimedia programming environment for young gifted learners. *Gifted Child Today*, *34*(2), 26-31.

Liu, C. C., Cheng, Y. B., & Huang, C. W. (2011). The effect of simulation games on the learning of computational problem solving. *Computers & Education*, *57*(3), 1907-1918.

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.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, *10*(4), 16.

Manovich, L. (2013). Media after software. *Journal of Visual Culture*, *12*(1), 30-37.

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with scratch. *Computer Science Education*, *23*(3), 239-264.

MIT Media Lab, (2015). *Scratch About.* Retrieved on 22 October 2018 from https://scratch.mit.edu/about.

Microsoft, (2014). *Hour of code*. Retrieved on 22 October 2018 from http://www.microsoft. com/about/corporatecitizenship/en-us/youthspark/youthsparkhub/hourofcode/.

Middleton, J. & Spanish, P. (1999). Motivation for achievement in mathematics: Findings, generalizations and criticism of the research. *IRME Online*, *30*(1), 65-88.

Miles, M. B. & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook*. London: Sage.

Moreno, J. (2012). Digital competition game to improve programming skills. *Journal of Educational Technology & Society*, *15*(3), 288.

Noble, J. (2013). *Building a LEGO-based robotics platform for a 3rd grade classroom.* (Unpublished doctoral dissertation). Tufts University.

Nourbakhsh, I. R., Hamner, E., Crowley, K., & Wilkinson, K. (2004, April). Formal measures of learning in a secondary school mobile robotics course. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on* (Vol. 2, pp. 1831-1836). IEEE.

Olgun, K., B. (2014). *The effect of programming on middle school students' thinking styles* (Unpublished master's thesis). Istanbul University.

Ozturk, S. (2016). *The effect of flip learning method on the students' academic achievement, computer attitudes and self-directed learning skills in programming language teaching* (Unpublished master's thesis). Ankara University, Ankara.

Partnership for 21st Century Skills (P21), (2007). *Partnership for 21st century skills*. Retrieved on 22 October 2018 from http://www.p21.org/about-us/p21-framework/60.

Patan, B. (2016). *Development of coding curriculum for kindergarten.* (Unpublished master's thesis). Bahcesehir University, Istanbul.

Pellas, N. & Peroutseas, E. (2016). Gaming in Second Life via Scratch4SL: Engaging high school students in programming courses. *Journal of Educational Computing Research*, *54*(1), 108-143.

Ramalingam, V., LaBelle, D., & Wiedenbeck, S. (2004, June). Self-efficacy and mental models in learning to program. *ACM SIGCSE Bulletin, 36*(3), 171-175.

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.

Rodgers, C. (2002). Defining reflection: Another look at John Dewey and reflective thinking. *Teachers college record*, *104*(4), 842-866.

Rogers, C. B., Wendell, K., & Foster, J. (2010). A review of the NAE report, engineering in K-12 education. *Journal of Engineering Education*, *99*(2), 179-181

Román-González, M. (2014). Aprender a programar 'apps' como enriquecimiento curricular en alumnado de alta capacidad. *Bordón. Revista de Pedagogía, 66*(4), 135-155.

Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*, *72*, 678-691.

Rusk, N., Resnick, M., Berg, R., & Pezalla-Granlund, M. (2008). New pathways into robotics: Strategies for broadening participation. *Journal of Science Education and Technology*, *17*(1), 59-69.

Ryan, R. M., & Deci, E. L. (2000). Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary Educational Psychology, 25*(1), 54-67.

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools. *Computers & Education, 97,* 129-141.

Saritepeci, M. & Durak, H. (2017). Analyzing the effect of block and robotic coding activities on computational thinking in programming training. In I. Koleva & G. Duman (Eds.). *Educational research and practice*, (pp. 490-501). Sofia, Bulgaria: St. Kliment Ohridski University Press.

Schunk, D.H., Meece, J.R., Pintrich, P.R. (2014). *Motivation in education: Theory, research, and applications* (4th ed.). Boston, MA: Pearson.

Shermis, S. S. (1992). *Critical thinking: Helping students learn reflectively*. ERIC Clearinghouse on Reading and Communication Skills, Indiana University, Bloomington, IN.

Tomlinson, C. A., Kaplan, S. N., Renzulli, J. S., Purcell, J. H., Leppien, J. H., Burns, D. E., & Imbeau, M. B. (2008). *The parallel curriculum: A design to develop learner potential and challenge advanced learners*. London: Sage.

Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012). The fairy performance assessment: measuring computational thinking in middle school. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (pp. 215-220). ACM.

Wing, J. (2014). Computational thinking benefits society. *40th Anniversary Blog of Social Issues in Computing*. New York: Academic Press.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical transactions of the Royal Society of London: A Mathematical, Physical and Engineering Sciences*, *366*(1881), 3717-3725. http://dx.doi.org/10.1098/rsta.2008.0118.

Yagci, M. (2016). Effect of attitudes of information technologies (IT) preservice teachers and computer programming (CP) students toward programming on their perception regarding their self-sufficiency for programming. *International Journal of Human Sciences*, *13*(1), 1418-1432.

Yang, H. L. & Cheng, H. H. (2009). Creative self-efficacy and its factors: An empirical study of information system analysts and programmers. *Computers in Human Behavior*, *25*(2), 429-438.

Yen, C.-Z., Wu, P.-H., & Lin, C.-F. (2012). Analysis of expert's and novice's thinking process. Engaging Learners through Emerging Technologies, *Communication in Computer and Information Science, 302,* 122-134.

Yildiz Durak, H. & Guyer, T. (2018). Design and development of an instructional program for teaching programming processes to gifted students using Scratch. In *Curriculum Development for Gifted Education Programs* (pp. 61-99). Hershey, PA: IGI Global.

Yildiz-Durak, H. & Guyer, T. (2019). An investigation of the opinions of gifted primary school students' in the programming training processes [Programlama ogretim surecinde ustun yetenekli ilkokul ogrencilerinin goruslerinin incelenmesi]. *Ankara University Journal of Faculty of Educational Sciences,* 52(1), 107-137. DOI: 10.30964/auebfd.466922

Yildiz- Durak, H. & Saritepeci, M. (2018). Analysis of the relation between computational thinking skills and various variables with the structural equation model. *Computers & Education*, *116*, 191-202.

Yildiz Durak, H. (2019). Modelling different variables in learning basic concepts of programming in flipped classrooms. *Journal of Educational Computing Research*. doi: https://doi.org/ 10.1177/0735633119827956

Yildiz Durak, H. (2018a). Digital story design activities used for teaching programming effect on learning of programming concepts, programming self-efficacy, and participation and analysis of student experiences. *Journal of Computer Assisted Learning*. *34*(6), 740-752.

Yildiz Durak, H. (2018b). Flipped learning readiness in teaching programming in middle schools: Modelling its relation to various variables. *Journal of Computer Assisted Learning*. *34*(6), 939-959.

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

Correspondence: Hatice Yildiz Durak, Assistant Professor, Department of Computer Education and Instructional Technology, Faculty of Education, Bartin University, Bartin, Turkey