International Journal of Computer Science Education in Schools, January 2021, Vol. 4, No. 3

ISSN 2513-835

The Effect of Unplugged Coding Education for Special Education Students on Problem-Solving Skills

Ümit DEMİR[1]

[1]Canakkale Onsekiz Mart University

DOI: 10.21585/ijcses.v4i3.95

Abstract

During recent years coding education has been an important issue in many countries. Coding education has been an important topic for these countries. One of the reasons why coding education is being discussed by educators and other partners of the schools is that it is seen as a key competence for students, and workers at developing problem-solving skills. Coding as an academic skill is seen as a part of logical reasoning. Coding is also accepted as one of the skills called "21st-century skills" required from individuals. Special education students are in a disadvantaged situation as in other learning platforms. Thus, this study aims to analyze the place of coding education in developing problem-solving skills of special education students. Within the scope of the study, unplugged coding applications were carried out with the participation of 34 students having mild intellectual disabilities who are continuing their education in a special education vocational school aged between 14 and 18. A question form was used to evaluate problem-solving skills. There was a significant difference between the pre-course and post-course skills of the students. Students' average scores of problem-solving skills in the post-course was higher than their average scores in the pre-course. The analysis of the findings showed that the students' skill scores in using problem-solving steps have increased in all these steps.

Keywords: special education, unplugged coding, problem-solving skill

1. Introduction

21st-century skills include basic skills areas such as critical thinking, problem-solving, and decision making, collaboration and communication, information literacy, technology literacy, flexibility and adaptability, global competencies, and financial literacy (Kereluik et al., 2013). The new 21st century skills are based on reasoning and logical thinking (Durak & Şahin, 2018). Because of this coding skills are important to get 21st-century skills. The ability of coding is called "algorithmic thinking" and "computational thinking" in different research (Durak & Şahin, 2018). In performing the coding process, it is important to follow the steps of comprehending, analyzing, solving the problems, and making the results as algorithms, establishing the correct algorithm, and encoding the algorithm with a program over the language. In this process, coders should use their algorithmic thinking or computational thinking skills (Lee et al., 2011).

In the economy, coding and programming skills have gained importance in many sectors and fields (Arora et al., 2001). The coding and programming skills included in the digital competencies with lifelong learning competence are among the important skills of the 21st century and the future (van Laar et al, 2017). The current workplace needs highly skilled employees that can work in increasingly interactive and tasks. Such employees are expected to effectively choose and use information from the number of accessible information and effectively use them in their works (Ahmad et al., 2013; Carnevale & Smith, 2013). For this reason, coding education should take place in education policies for the development of coding skills. Teachers have great responsibilities for the development of individuals with lifelong learning competencies (Durak & Şahin, 2018).

ISSN 2513-835

Students can gain problem-solving, critical thinking, independent thinking, sharing, ethical behavior, knowledge, and digital literacy skills via programming and coding education. (Durak & Şahin, 2018; Voogt et al., 2013).

Programming education at an early age is becoming increasingly important. In this context, radical changes have been made in the educational curriculums around the world and in Turkey to provide programming education in elementary school (Demirer & Sak, 2016). In this regard, the European countries, and other developed countries as South Korea have begun to provide programming education to children and young people starting from primary school (Boccani et al, 2016; Demirer & Sak, 2016; Salter, 2013). The United Kingdom has integrated computer programming as a main course starting from primary school (Jones, 2013). The European Union (EU) organizes various events like European Code Week which is held in November. In the framework of the European Code Week, activities were held in various European countries, approximately three hundred workshops were organized, and thousands of students participated in these workshops (Demirer & Sak, 2016). In the programming education process, both plugged and unplugged activities are used. While unplugged programming broadly refers to learning computational thinking and computer science concepts without relying on computational devices, plugged programming relies on the usage of computational devices in these learning processes (Aranda & Ferguson, 2018). Unplugged coding can include role-playing, manipulation of real-world objects (eg sticky notes, cards, wooden blocks), and physical actions of the body activity (Aranda & Ferguson, 2018). Special education students are also disadvantaged in this coding education as in other training. Most of the studies and applications for special education are prepared for gifted students (Alkan, 2019; Hagge, 2017; Lee, 2011). In these studies, plugged visual programming tools as Kodu and Scratch are used.

Empirical research on the impact of coding education on special education students is very limited (Adams & Cook, 2013; Miller, 2009; Topal, Budak & Geçer, 2017). Adams and Cook (2013) evaluated the effect of using programming and controlling robots' game for people with dementia about the stimulation of social behavior. As a result, it was found out that social behavior occurred more often than non-social behavior during the sessions. Miller (2009) reported that programming (Logo software) education improved both the programming skills and applying written vocabulary in a purposeful, rule-based manner. Topal, Budak, and Geçer (2017) found out that algorithm teaching via Scratch was effective on the problem-solving skills of deaf-hard hearing students. Coding activities attended by students with intellectual disabilities are also extremely limited (Karna-Lin et al., 2006; Ratcliff & Anderson, 2011; Taylor, 2018; Taylor, Vasquez, & Donehower, 2017; Wainer et. al., 2010). Karna-Lin et al. (2006) worked with children, 8 to 18 years old having learning difficulties and mild cognitive delays. Students used Lego robots in the experimental process. As a result, it was found out that students' social abilities as asking for advice, sharing ideas increased. Students also had an opportunity to practice their problem solving, logical thinking abilities. Ratchlif and Anderson (2011) reported that using programming education software (Logo) captured the students' interest. This programming experience was also described as a viable source of interactive challenge and problem-solving experience that provided a great deal of pride, intrinsic reward, enjoyment, and a sense of ownership of learning by the attending students. Taylor, Vasquez, and Donehower (2017) found out that despite programming education for students with mild disabilities was not effective in enhancing problem-solving abilities, programming in Logo captured students' interest. Taylor (2018) researched the potential effect of learning skills in computer programming for students with intellectual disabilities. Students were assessed through baseline, treatment, and generalization phases. As a result, students were successful at programming the robot. Wainer et. al. (2010) found out that interacting with Lego robots for programming education increased the collaborative behaviors of the students having Autism Spectrum Disorder (ASD). When these empirical researches were analyzed, it was observed that they generally include plugged applications. Coding training or practices involving only plugged coding applications can only address students having computer usage skills. However, coding may provide many opportunities for special education students to support their learning and life-skills (Duff, McPherson, King & Kingsnorth, 2019; Taylor, Vasquez & Donehower, 2017). For this reason, the use of unplugged applications in coding training may enable a wider group of people with having different disabilities to benefit from this training. Consequently, it is expected that this study which investigates the development of special education students in using problem-solving steps with coding education will contribute to the literature. One of the main benefits of coding education is developing problem-solving skills (Chao, 2016; Hooshyar et al., 2016). Problem-solving skills can be improved by computational thinking processes. Through computational thinking, we can explain the problem and use simple methods or formulas to solve the problem by computer computation (Hsu et al., 2018).

ISSN 2513-835

1.1 Computational thinking

Computational thinking (CT) skills have very high importance in obtaining 21st-century learning skills (Barut et al., 2016). Wing (2008) defines computational thinking as "designing systems, solving problems, and understanding human behavior by drawing on the concepts main to computer science". She stated that computational thinking includes some common concepts, such as data representation, problem decomposition, and modeling. She also stated that "computational thinking is a basic skill for everyone, not just for computer scientists. We should add computational thinking to every child's analytical ability like writing, reading, and arithmetic abilities." Thinking like a computer scientist enables children to solve any problem they can face logically. Computational thinking can help in designing solutions to automation-sensitive problems (Allsop, 2019). In addition to thinking like a computer scientist, there seems to be value in teaching critical thinking skills and the problem-solving mentality that comes with traditional writing, reading, and arithmetic. (Kazakoff, 2014).

Computational thinking includes problem-solving processes:

• Formulating problems in a way that let us use a computer and other tools to help solve these problems,

• Analyzing and organizing data logically,

• Representing data through abstractions, such as simulations and models,

• Automating solutions through algorithmic thinking (a series of ordered steps),

• Identifying, analyzing, and implementing possible solutions to ensure that the steps and resources are combined most efficiently and effectively,

• Generalizing and transferring this problem-solving process to various problems (Barr et al., 2011). As a result, computational thinking helps in the development of problem-solving strategies.

Problem-solving strategies as computational thinking and algorithmic thinking give us loads of talents and skills, including:

• Confidence in dealing with complexity,

• Instance on working with difficult problems,

• Tolerance for uncertainty,

• The ability to deal with open-ended problems,

• The ability to communicate and work with others to achieve a common goal or solution (Barr & Stephenson, 2011). Therefore, including problem-solving skills developing activities via computational thinking in the education process can provide loads of talents and skills that are of great importance to the students. In the process of realizing these activities, evaluating, and following the development levels of students is of great importance in terms of structuring the teaching processes. The Computational skill assessment model developed by Allsop (2019) includes three aspects: 'computational concepts', 'metacognitive practices', and 'learning behaviors'. Computer game design processes are generally used in evaluating these dimensions (Allsop, 2019; Brennan & Resnick, 2012; Werner et al., 2012; Werner et al., 2014).

1.2 Algorithmic Thinking

Some difficulties in mathematics are generally associated with weaknesses in algorithmic problem-solving (Plerou & Vlamos, 2016). The algorithm also describes a finite sequence of actions that describe how to solve a given problem (Kotthoff, 2016). The ability to design and use algorithmic forms or schemas in problem-solving processes pre-requires cognitive skills and enforces these skills (Antonia, Panagiotis & Panagiotis, 2014). This means that generally algorithmic techniques can be applied similarly to solve a problem (Atmatzidou & Demetriadis, 2016). Algorithmic thinking is a critical skill in getting problem-solving skills. Algorithmic thinking is defined as the ability to construct new algorithms to solve given problems (Futschek, 2006). In algorithmic thinking, a set of solution rules including devising a step-by-step solution are designed and application instructions for rules are defined by steps (Angeli et al, 2016; Curzon et al., 2014). Algorithmic thinking includes understanding, applying, evaluating, and producing algorithms skills (Kanbul & Uzunboylu,

2017). In these activities putting actions in the correct sequence and using flow control are very important elements. These are also considered important for computational thinking (Angeli et al, 2016).

In the research in which the effect of using algorithmic thinking in programming activities on the development of computational and mathematical thinking was examined, it was found out that coding and programming education is effective for developing problem-solving strategies, teaching mathematics, and for creative thinking. (Taylor et al., 2010; Kalelioglu & Gulbahar, 2014). Algorithmic thinking is not only an important part of computer science, but it is also an important part of our lives especially in decision-making (Kátai, 2015; Mohaghegh & McCauley, 2016). Decision-making processes are very important for individuals with intellectual disabilities. They need supports in decision-making processes (Jamesson et. al., 2015). As a result, algorithmic thinking can be an important supporter and facilitator of problem-solving and decision-making processes.

1.3 Problem-Solving

Problem-solving is a critical learning process in formal education activities for all the education levels from primary to higher education (Jonassen, Howland, Moore, & Marra, 2003; Lazakidou & Retalis, 2010). This advanced cognitive ability can be understood as the ability to use rules and concepts to solve the problem (Wang, Han, Zhan, Xu, Liu & Ren, 2015). Educators expect that their students will graduate from their courses with good problem-solving skills. They use open-ended problems and activities to develop these skills (Woods et al., 1997). Students' participation in problem-solving activities assists them to gain helpful attitudes that are crucial to real life, such as creativity, flexibility, thinking, and efficiency. Besides, all these attitudes have already been linked to life-long learning skills (Goffin & Tull, 1985). So, problems used in the educational process should be real-life scenarios that provide students opportunities to become real-life problem solvers (Yu, Fan, & Lin, 2015).

To date, many problem-solving models have been suggested (Bransford & Stein, 1993; Good & Brophy, 1995; Hohn & Frey, 2002; Ormrod, 2000; Polya, 1973; Sternberg, 2003). A well-known problem-solving model is Sternberg's 7 steps model. Sternberg's 7 steps model can be used to both well-defined (known) and ill-defined problems in which learners engage a different set of epistemic beliefs (Jonassen, 2010; Lazakidou & Retalis, 2010). This model includes the following steps: problem identification, the definition of the problem, constructing a strategy, organizing information, allocation of resources, monitoring, and evaluating problem-solving. (Sternberg, 2003: 360). These steps of the problem-solving process require one to arrange each step and make decisions at the same time (Özsoy & Ataman, 2017). Also, the learner should experience and make strong the problem-solving process continues to complete the task (Wang, Han, Zhan, Xu, Liu & Ren, 2015).

Problem-solving skills have a potential impact on the individuals' having intellectual disabilities independence and academic achievement (Erickson, Noonan, Zheng, & Brussow, 2015; Root, Saunders, Spooner, & Brosh, 2017). By using schema-based instruction in the practice process of teaching problem solving to students with mild intellectual disabilities, these students can benefit from many instructional features (Jidentra et al., 2015). Because of this visual representation are very important for special education students (Root, Saunders, Spooner, & Brosh, 2017). In problem-solving processes, some thinking strategies are used. More common of these are algorithmic thinking and computational thinking.

1.4 Research Questions

During recent years coding education has been an important issue in many countries to support and develop problem-solving skills. One of the reasons why coding education is being discussed by educators and other partners of the schools is that it is seen as a key competence for students and workers. Coding as an academic skill is seen as a part of logical reasoning. It is also accepted as one of the skills called "21st-century skills" required from individuals (van Laar et al, 2017). Learning how to code is equally valuable as learning math, reading, and writing (Horizon, 2015). It is very important to include coding educations with plugged or unplugged activities in the training process. It is thought that especially unplugged coding activities will appeal to a much wider group of students with having different disabilities. Lechelt et al. (2018), in their research, found out that a physical toolkit (magic cube) could be used both supporting comprehensions of computational concepts and enabling students to get excited about learning with fun. Because of these, computer-aided coding

ISSN 2513-835

activities may only address students having computer usage skills. As a result, this study aims to analyze the effect of coding education having unplugged activities in developing problem-solving skills of special education students. Within the aim of this research following questions are identified:

RQ1: What is the effect of unplugged coding education on the problem-solving skills of special education students?

RQ2: What is the effect of unplugged coding education on the steps of the problem-solving skill of special education students?

2. The study

2.1. Method

An experimental method was used in the study. Among the different types of these methods, the full experimental method is the method with the highest scientific value. Thus, the full experimental method is used in this research. Within the scope of this study, both experimental and control groups were formed according to unplugged coding education materials usage status.

2.2. Participants

This study was conducted with 34 students having mild intellectual disabilities attending a Math education in the Çanakkale Special Education Vocational and Technical High School, Turkey. Students with a mild intellectual disability typically defined by having an IQ between 55 and 70 and having impairments in adaptive skills, such as daily living, social skills, and communication (Schalock, et al., 2010). The distribution of the students according to demographic characteristics is shown in Table 1. The number of females (n=14) and males (n=11) students are close to each other and the average age of the students is approximately 15.5. In the sample choosing process, students' parental written permission and their voluntary participation were asked. All the students wanted to participate in the experimental process.

Table 1. Distribution of demographic features of students (n=25)

Gender	**f**	**%**
Female	14	56
Male	11	44
Age	**f**	**%**
14	5	20
15	7	28
16	6	24
17	5	20
18	2	8
Total	25	100

2.3. Methods of Data Collection

The full experimental design was used in this research. There are 4-6 students in the classes in the special education school. At each class level, this school has two classes. The experimental processes were carried out in the mathematics course. Before the experimental processes, each class level (9th, 10th, 11th) categorized randomly as one control and one experimental group. The data collection process consists of 4 basic stages (Figure 1).

ISSN 2513-835

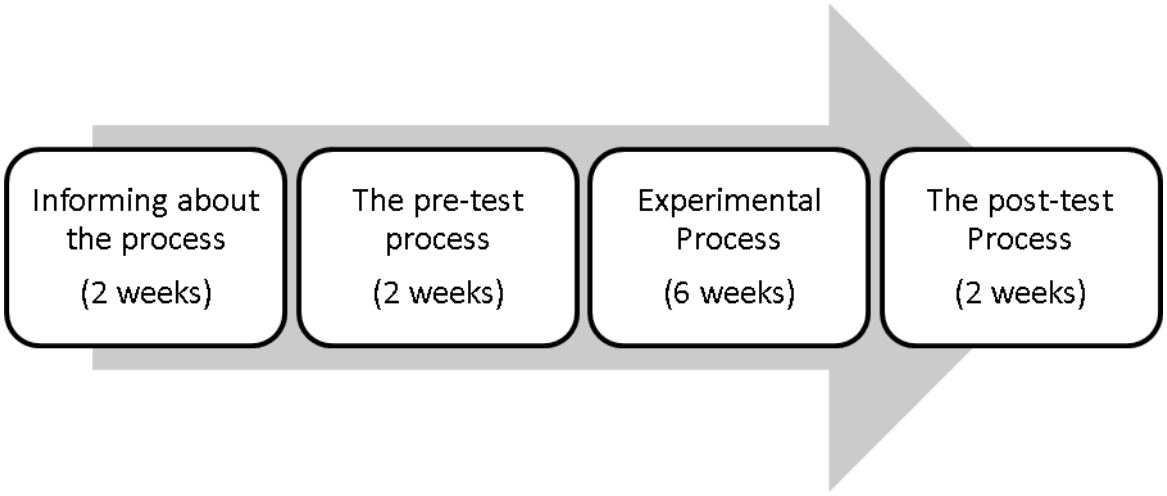

Figure 1. The data collection process steps

2.3.1. Informing About the Process

Before the experimental process, the math teacher was informed about the aim and process of the research including problem-solving steps, materials that will be used by students, and what to pay attention to in the evaluation process.

A performance rating form for determining the status of using problem-solving skills was administered to the students in terms of their skills in using problem-solving. In this form, including checklist items, students' usage of problem-solving steps defined by Sternberg (2003) are questioned. The problem-solving cycle's steps are problem identification, the definition of the problem, constructing a strategy, organizing information, allocation of resources, monitoring, and evaluating the problem-solving stage. The first step "problem identification" was not included in this research. Because the problem situation was given to the student and asked to find a solution to the problem. In the problem identification step, students are wanted to answer the question of "Do we actually have a problem?" (Sternberg & Sternberg, 2012:445). In pre-test and post-test processes, the teacher was wanted to ask the students about each problem-solving stage defined in the question form (Appendix A). In filling the performance rating form, students' written answers to the questions, and their verbal explanations of these written answers on problem-solving steps were evaluated. A blank sample of the student question form (translated to English) is presented in appendix A. In preparing both the question form asked to the students and performance rating form Sternberg & Sternberg (2012:445-446) and Lazakidou & Retalis (2010:5) were used. In the question form, students wanted to identify the problem, define the problem, construct a strategy, organize the information, allocate the resources, and monitor and evaluate their solving strategy. In the performance rating form, students' verbal and written answers to the questions were evaluated. In the performance rating form, problem-solving skills are scored out of 5 for each problem-solving cycle step (1: insufficient, 2: acceptable, 3: intermediate, 4: good, 5: very good). In filling this form, the math teacher assisted in evaluating the verbal and written answers of the special education students because of the communication and writing problems of some students. The main aim of the study was not to give special education students the skill of sorting the algorithm problems correctly. The main aim was to develop special education students' problem-solving skills in a logical framework by using problem-solving steps. For example, the student may find the correct sorting in the pre-test by chance or by memorizing in the post-test. But for the research, without creating a logical framework by using problem-solving steps sorting the algorithm was not valuable. At the beginning of the experimental process, the experimental group students also were informed about the rules of the games and problem-solving steps. After the informing process, the experimental process was started.

2.3.2 The Pre-test Processes

In the pre-test and post-test process (applied following 6 weeks), algorithm cards shown in figure 2-3-4 and the question form (Appendix A) were used. Both the students in the control and experimental groups were asked to

solve “Buying Bread”, “Washing Dishes” and “Making Pasta” problems by using problem-solving steps. At the same time, students were asked to solve these problems by answering the questions on the student question form. In the pre-test students' written answers to the questions and their verbal explanations of these written answers on problem-solving steps were evaluated by the performance rating form. The opinions of the course teacher about student performance were also taken into consideration during the evaluation process.

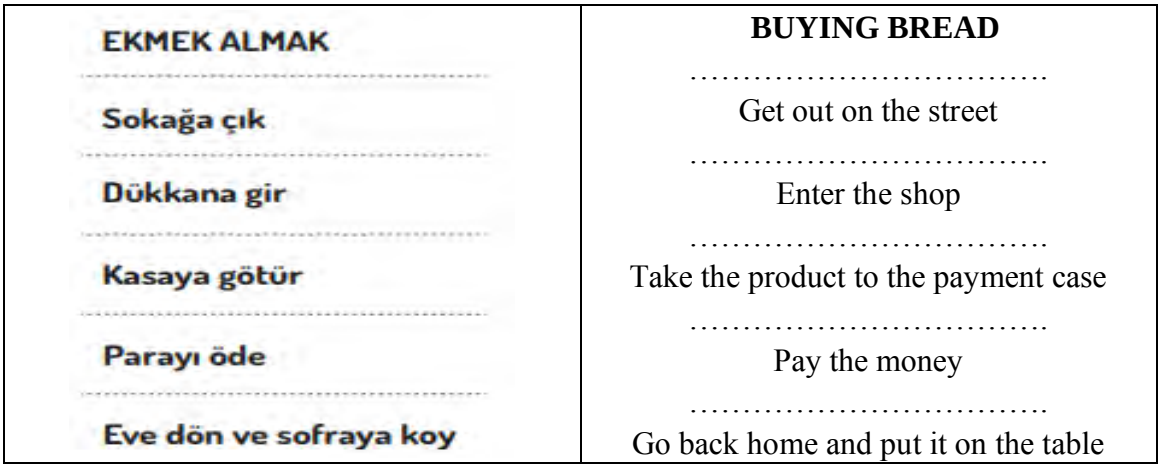

Figure 2. Buying bread and 5 algorithmic steps (Turkish –English)

In this process, the teacher was wanted to ask the students about each problem-solving stage defined in the question form. In figure 2, buying bread step cards (including: get out on the street, enter the shop, take the product to the payment case, pay the money, go back home, and put it to the table) are shown. In figure 3, washing dishes step cards (including cleaning the kitchen midden, rinsing out, soaping, rinsing, drying) are shown. In figure 4, making pasta cards (including boiling, opening the package, putting it into the water, waiting for 15 minutes, pouring to the strainer) is shown.

Figure 3. Washing dishes (title) and 5 algorithmic steps (Turkish –English)

In each game, firstly students were wanted to define the problem with their own words. Then they were wanted to construct a strategy to solve the defined problem and explain the choosing reason for that strategy. After this step, they were wanted to organize having information and explain how to use these in finding a solution. After this problem-solving step, they were wanted to allocate resources (time, effort, etc.) in solving the problem. Following this step, they were wanted to question themselves if they are on true track as they proceed to solve the problem. At the last step, they were wanted to evaluate their problem-solving strategy if the strategy worked or not. If the discovered strategy did not work, they were wanted to explain the reasons. In all problem-solving steps, all students were wanted to write their problem-solving steps and make an oral presentation of their

solving. These written answers were evaluated by the researcher and the teacher.

MAKARNA YAPMAK	**MAKING PASTA**
Suyu kaynat	Boiling
Paketi aç	Opening the package
Suyun içine at	Putting into water
15 dk bekle	Waiting for 15 minutes
Süzgece dök	Pouring to the strainer

Figure 4. Making pasta (title) and 5 algorithmic steps (Turkish –English)

2.3.3 The Pre-test Processes

In the classroom environment, the experimental group students played three games to learn the usage of problem-solving steps to solve problems in the teaching process. They played the games individually and as a member of the game group under the math teacher supervision. In the selection of the games, expert opinions were received from the special education teachers working in the special education school where the application was carried out and the academic staff of Çanakkale Onsekiz Mart University Special Education department. The first game was "The wolf, the lamb, the weed and the farmer activity" (figure 5). This game includes real-life concepts that students can easily fictionalize. Concrete and close-to-life learning environments are crucial for the success of the teaching process (Holt, Segrave & Cybulski, 2013).

Figure 5. The wolf, the lamb, the weed, and the farmer activity

In this game, Uncle Ahmet's farm was just outside of the village, just across the river. Uncle Ahmet took his lamb one day, the wolf descending from the forest to his garden, and a certain amount of grass he had reserved for his lamb and wanted to cross over to the shore. But the only way he could cross over was a small boat and it was impossible to cross them all together. He could take one to her at a time; He can either put the lamb, the wolf, or the weed. If the farmer leaves the lamb and wolf, the wolf eats the lamb. If the farmer leaves the lamb and the weed, the lamb eats the weed. So, the question of the game is "How do you think Uncle Ahmet will get all three

ISSN 2513-835

of them across? The solution of this game consists of 5 steps. It is important for special education students facing problems to have fewer solution steps especially at the beginning of the learning process. Because students with disabilities rely on considerable step by step instructions (Mechling & Ortega-Humdon, 2007). In this game, problem-solving steps are:

(1) Transporting the lamb across the river,

(2) Transporting the wolf across the river,

(3) Transporting the lamb back,

(4) Transporting the weed across the river.

(5) Transporting the lamb across the river.

The second game played by students was “Tower of Hanoi” (Figure 6). The Towers of Hanoi is a puzzle that has been studied by computer scientists and mathematicians for many years. The goal is to recreate the 4-disk tower on the third post. The monks must move the disks according to two rules:

(1) The monks can only move one disk at a time.

(2) The monks can only place smaller disks on top of larger disks.

ISSN 2513-835

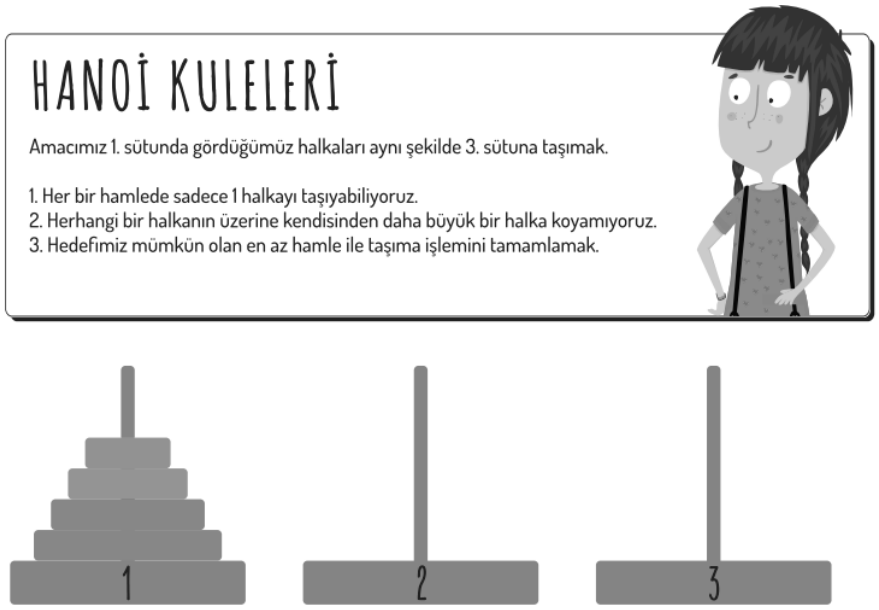

Figure 6. The towers of Hanoi activity

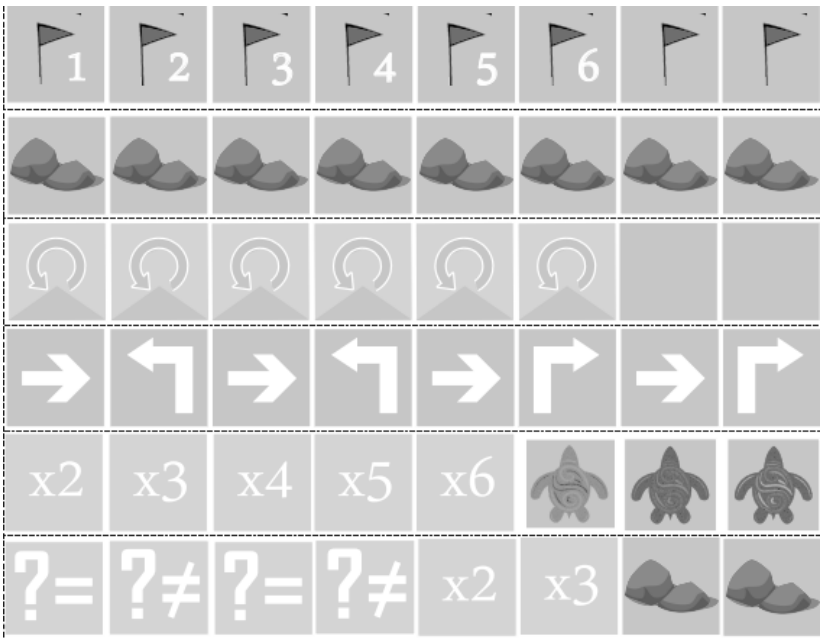

Figure 7. Tospaa unplugged coding game algorithm cards

The third game played by students was the "Tospaa Unplugged Coding Game". The game aims to bring the turtle to the targets without getting stuck. To reach the targets, algorithm cards in figure 7 are used. In this game, one of the most important terms of programming, the loop concept can be taught easily with this game. As a group game, gamers make algorithms to reach their targets. The gamer that uses the least algorithm card wins the game. An example scenario of the Tospaa game is shown in figure 8. At the same time, the control group students were informed about the problem-solving steps by the researcher. In this process, firstly the same math teacher gave the basic information about "why we need to use problem-solving steps?" and the problem-solving steps. In this process, the questions needed to answer in each problem-solving step were explained in detail by the researcher. Then different problems were asked students with worksheets. Students' answers for problem-solving steps were discussed in teacher management. Students were informed about their mistakes.

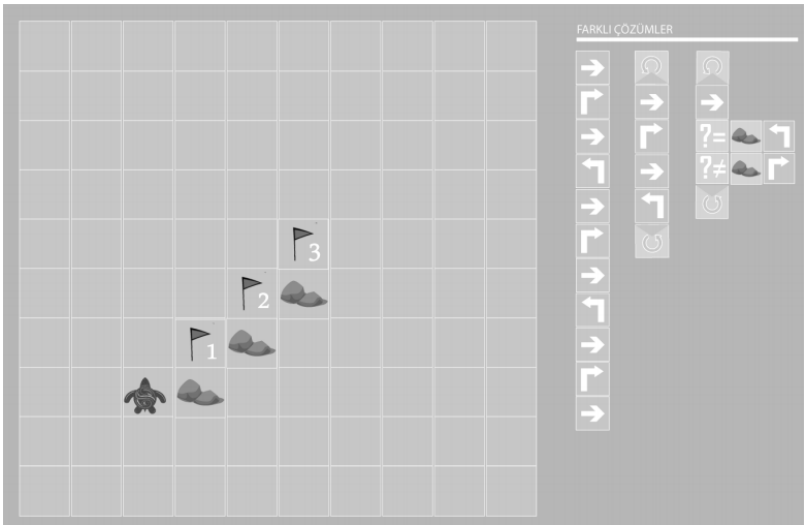

Figure 8. An example scenario of tospaa unplugged coding game

2.4. Data Analysis

Data collection involved by the performance rating form for determining the status of using problem-solving skills was administered to the students in terms of their skills in using problem-solving. The pre-test and post-test scores obtained by the students on this form were used in data analysis. These analysis values are given in the results section.

3. Results

3.1 Question 1: What is the effect of unplugged coding education on the problem-solving skills of special education students?

The pre-course skills of students were determined before the application while their post-course skills were determined just after the application. The results regarding whether a significant difference occurred between the scores in terms of skill change are presented in Table 2. Table 2 demonstrates that there is a significant difference [$t_{(24)}$=-7.19, p<.001] between the pre and post-course skills of the student group. While the average point of the pre-course skill was 10.68, the post-course average point increased to 13.36.

Table 2. Pre-course and post-course comparison of skill scores (n=25)

	N	$\overline{X}$	Sd	t	p
Pre-course skill scores	25	10.68	3.56	-7.19	.000**
Post-course skill scores	25	13.36	4.58		

*p< .01, ** p< .001

3.2 Question 2: What is the effect of unplugged coding education on the steps of the problem-solving skill of special education students?

Problem-solving skills were categorized into 6 steps. The results regarding whether a significant difference occurred between the scores in terms of skill change are presented in Table 3.

Table 3. Pre- and post-course skill comparison of the students (n=25)

Problem-Solving Step Name	Score Type	N	$\overline{X}$	Sd	t	p
Definition of the problem	Pre-course skill score	25	2.40	1.08	-4.10	.000**
	Post-course skill score	25	2.88	1.01		

ISSN 2513-835

Problem-Solving Step Name	Score Type	N	$\overline{X}$	Sd	t	p
Constructing a strategy	Pre-course skill score	25	2.48	1.05	-3.36	.003*
	Post-course skill score	25	2.80	1.04		
Organizing information	Pre-course skill score	25	2.12	1.01	-2.87	.008*
	Post-course skill score	25	2.44	.96		
Allocation of resources	Pre-course skill score	25	1.36	.49	-6.20	.000**
	Post-course skill score	25	2.16	.85		
Monitoring the process	Pre-course skill score	25	1.20	.41	-3.98	.001*
	Post-course skill score	25	1.72	.84		
Evaluating the process	Pre-course skill score	25	1.12	.33	-2.30	.031
	Post-course skill score	25	1.36	.57		

*p< .01, ** p< .001

Table 3 demonstrates that there are significant differences between pre and post-course observation scores of the student group in all the problem-solving skill steps (step 1: [$t_{(24)}$=-4.10, p<.001], step 2: [$t_{(24)}$=-3.36, p<.01], step 3: [$t_{(24)}$=-2.87, p<.01], step 4: [$t_{(24)}$=-6.20, p<.001], step 5: [$t_{(24)}$=-3.98, p<.01], step 6: [$t_{(24)}$=-2.30, p<.05]).

4. Discussion of Findings

This study evaluated the effect of the unplugged algorithm training for special education students. The study demonstrated that using unplugged algorithm training games can considerably improve problem-solving skills within all six steps. When the change in the problem-solving skill steps was examined, it was found that the most increase occurred in step 5 (allocating of resources) and the least increase was in step 7 (evaluating problem-solving stage). In step 5, students' experience of organizing information can be increased more with the experience they have with unplugged coding games. In steps 6 and 7, the monitoring and evaluating the problem-solving stage may require long-term development. Therefore, this step may require longer-term practice and experience. As the pre-test scores in the first 3 steps are not too low, the increase may be limited. As a result, using unplugged algorithm games improve the special education students' problem-solving skill. This result is important to see the importance of giving coding education.

The effect results of algorithm education consisted of the research results conducted by Alotaibi; Allan & Kolesar (1996), Erdem (2018), Fessakis et al. (2013), Howland & Good (2015); Topal, Budak, and Geçer (2017). Allan & Kolear (2011) gave algorithm and coding education in the computer science course (CS1) at Utah State University. In the course, students gained mathematical and problem-solving skills while becoming familiar with the computer as a tool and learning. Erdem (2018) found out that coding and algorithm education developed 5th-grade students' problem-solving skills. Fessakis et al. (2013) found out in their research that 5–6 years old kindergarten children were pleased with the attractive learning activities and had chances to improve mathematical concepts, social and problem-solving skills. Howland and Good (2015), there were significant improvements in 12–13-year-olds teenagers' computational communication and problem–solving skills after using algorithm developing applications for creating games. Topal, Budak, and Geçer (2017) found out that algorithm teaching via Scratch was effective on the problem-solving skills of deaf-hard hearing students. There are also research results pointing out that algorithmic thinking is not effective in learning and problem-solving (Doleck, Bazelais, Saxena & Basnet, 2017; Psycharis & Kallia, 2017).

Doleck, Bazelais, Saxena & Basnet (2017) found out a lack of association between computational thinking skills and academic performance. They emphasized that this result maybe since the curriculum has yet to be adequately associated with 21st-century skills teaching. Pyscharis & Kallia (2017) found out in their research conducted with 66 high school students that programming education including algorithmic procedures enhanced students' reasoning skills but did not enhance their problem-solving skills. As a result, When the studies using algorithmic structures in coding education are examined it is seen that it generally affects problem-solving skills positively. It is especially important to make the curriculum supportive of problem-solving skills within 21 st-century skills teaching (Doleck, Bazelais, Saxena & Basnet, 2017; Israel, Wherfel, Pearson, Shehab & Tapia, 2015; Psycharis & Kallia, 2017). There are several instructional benefits for students that can get from the inclusion of problem-solving skills within K-12 programs. They can benefit from this skill in building

higher-order thinking skills and increasing collaborative problem-solving (Kafai & Burke, 2014). For this reason, it is of great importance to add the skills for the acquisition of 21st-century skills to the curriculum and to evaluate them.

In Turkey, it is observed that there is not still sufficient importance given to robotic applications and coding education within 21st-century skills. As a result of institutional studies by universities, there are also effective robotic studies in almost all universities. However, there is not enough initiative to integrate coding education into the programs of different education levels from pre-school to university. But robotic coding education also can create lots of benefits for students. It can facilitate advanced hands-on programming, increase the rate of two-directional communication between the students and the robot (Virnes, Sutinen & Kärnä-Lin, 2008). The course of information technologies and software development (ITSD) course in the education program of Turkey is included as an elective course in 5th, 6th, 7th, and 8th grades based on the curriculum published in 2012 (Kanbul & Uzunboylu, 2017). In the last curriculum of ITSD course developed in 2017, not only setting integrity between informatics technology units but also with other subjects and real life is important. so that it is important to realize the discourses in real life (students benefit from informatics technologies and software development courses) and to learn how to use technology appropriately (Karaman & Karaman, 2019). We cannot say that this change in planning is sufficient in practice. Coding training is carried out under the supervision of the IT teachers and in their use with a single interactive board. It is considered that it is important to provide coding laboratory facilities for student-centered practice to realize effective learning. The number of coding classes that have the infrastructure installed is quite limited. So, it would be wrong to expect to achieve success in coding with just the program change. Besides, secondary schools can be considered a bit too late for coding training.

Teaching coding to children in early childhood before elementary school education may enable long-term economic payoffs. Investments in early childhood interventions are associated with lower costs and longer-term impacts than later interventions in childhood. (Heckman, 2006; Heckman & Masterov, 2007; Reynolds, et al., 2011).

5. Conclusion

Besides, special education students are the most disadvantaged students in this education process. Measures can be taken to ensure that all disadvantaged students receive coding training. For example, Al-Khalifa & AlSaeed (2020) found out that tactile teaching strategies were effective in the programming education of students with vision impairment. In this research, it was also found out that students' problem-solving skills are insufficient. But problem-solving skill is an important skill for the development of life skills (Prajapati, Sharma & Sharma, 2017; Wurdinger & Qureshi, 2015). Similarly, special education students need to develop life skills to maintain their daily lives (Smith, Cihak, Kim, McMahon & Wright, 2017). For this reason, it is thought that education and practices that support problem-solving should be given importance. Based on the research results; it can be indicated that coding education can provide many educational opportunities to support the problem-solving skills of special education students. Some of these opportunities are the ability to break down problems into smaller parts and to draw on both logic and creativity to figure out the best ways to solve them (Lechelt et al., 2018). To inform and educate the future generation, companies, universities should make investments and serious ventures for coding and robotic applications education. Soon, new professions will emerge, and many occupations will not be needed. Therefore, it is very important to teach 21st-century skills to all children including students who need special education. It is thought that it is of great importance to increase the competencies and expertise of teachers in this field to develop students' problem-solving skills. In this context, it is thought that it would be beneficial to provide teachers with training or in-service training on how to use the applications for the development of problem-solving skills and how to evaluate the dimensions of problem-solving skills of the students. According to the findings of the study, the following suggestions were made.

• Coding education applications and web-based platforms like scratch, code.org should be integrated into the K12 curriculum.

• In-service or training courses should be given to teachers and teacher candidates for problem-solving steps, development, and evaluation of problem-solving skills.

• Coding laboratories including robotic and unplugged application materials should be set.

• Experimental studies that investigate the traditional programming course and unplugged programming/robotic

ISSN 2513-835

programming course can be made.

Limitations

The research was carried out in a vocational high school where mild intelligent students have been given special education. Studies conducted with different samples of intellectual disability can be made. Also, in this research, unplugged coding education materials were used. Comparative studies can be conducted on the effectiveness of computerized and unplugged coding education.

References

Adams, K. D., & Cook, A. M. (2013). Programming and controlling robots using scanning on a speech generating communication device: A case study. *Technology and Disability, 25*(4), 275-286. doi:10.3233/TAD-140397

Ahmad, M., Karim, A. A., Din, R., & Albakri, I. S. M. A. (2013). Assessing ICT competencies among postgraduate students based on the 21st century ICT competency model. *Asian Social Science*, *9*(16), 32. doi:10.5539/ass.v9n16p32

Alkan, A. (2019). The effect of code game lab software on the level of problem-solving skills in programming language teaching. M*ehmet Akif Ersoy University Faculty of Education Journal,* (50), 480-493.doi:10.21764/maeuefd.486061

Allan, V. H., & Kolesar, M. V. (1996). Teaching Computer Science: A Problem-Solving Approach that Works. National Educational Computing Conference 1996, Minneapolis, MN. Retrieved from https://files.eric.ed.gov/fulltext/ED398878.pdf

Allsop, Y. (2019). Assessing computational thinking process using a multiple evaluation approach. *International journal of child-computer interaction*, *19*, 30-55. doi: 10.1016/j.ijcci.2018.10.004

Alotaibi, H., S Al-Khalifa, H., & AlSaeed, D. (2020). Teaching Programming to Students with Vision Impairment: Impact of Tactile Teaching Strategies on Student's Achievements and Perceptions. *Sustainability*, *12*(13), 5320. doi: 10.3390/su12135320

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. Journal of *Educational Technology & Society, 19*(3), 47-57. Retrieved from https://www.jstor.org/stable/pdf/jeductechsoci.19.3.47.pdf

Antonia, P., Panagiotis, V., & Panagiotis, K. (2014). Screening Dyscalculia and Algorithmic Thinking Difficulties "1st International Conference on New Developments in Science and Technology Education" Proceedings Manuscripts.

Aranda, G., & Ferguson, J. P. (2018). Unplugged Programming: The future of teaching computational thinking? *Pedagogika, 68*(3), 279-292. Retrieved from https://pages.pedf.cuni.cz/pedagogika/?attachment_id=11945&edmc=11945

Arora, A., Arunachalam, V. S., Asundi, J., & Fernandes, R. (2001). The Indian software services industry. *Research policy*, *30*(8), 1267-1287. Retrieved from http://www1.ximb.ac.in/users/fac/dpdash/dpdash.nsf/23e5e39594c064ee852564ae004fa010/fc16012dc5a4d1cae52568b200183115/%24FILE/Soft_industry1.pdf

Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, *75*, 661-670.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *Acm Inroads*, *2*(1), 48-54.doi: 10.1145/1929887.1929905

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology*, *38*(6), 20-23. Retrieved from https://files.eric.ed.gov/fulltext/EJ918910.pdf

Barut, E., Tugtekin, U., & Kuzu, A. (2016). An Overview of Computational Thinking Skills with Robotic Applications. *The 3rd International Conference on New Trends in Education (ICNTE)*. Retrieved from http://journals.sfu.ca/onlinejour/index.php/i-jet/article/download/6097/4264

Bransford, J. D., & Stein, B. S. (1993). *The ideal problem solver: A guide for improving thinking, learning and creativity (2nd ed.)*. NY: W.H. Freeman.

Brennan, K., & Resnick, M. (2012). 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. 1-25). Retrieved from http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf

Carnevale, A. P., & Smith, N. (2013). Workplace basics: The skills employees need and employers want. *Human Resource Development International, 16* (5), 491-501. Retrieved from https://cew.georgetown.edu/wp-content/uploads/2014/11/HRDI.Editorial.pdf

Chao, P. Y. (2016). Exploring students' computational practice, design and performance of problem-solving through a visual programming environment. *Computers & Education*, *95*, 202-215. doi: 10.1016/j.compedu.2016.01.010

Curzon, P., Dorling, M., Ng, T., Selby, C., & Woollard, J. (2014). *Developing computational thinking in the classroom: a framework* Swindon, GB. Retrieved from https://eprints.soton.ac.uk/369594/1/DevelopingComputationalThinkingInTheClassroomaFramework.pdf

Demirer, V., & Sak, N. (2016). Programming education and new approaches around the world and in Turkey. *Journal of Theory and Practice in Education*, *12*(3), 521-546. Retrieved from http://acikerisim.lib.comu.edu.tr:8080/xmlui/bitstream/handle/COMU/1443/Veysel_Demirer_Makale.pdf?sequence=1&isAllowed=y

Doleck, T., Bazelais, P., Lemay, D. J., Saxena, A., & Basnet, R. B. (2017). Algorithmic thinking, cooperativity, creativity, critical thinking, and problem solving exploring the relationship between computational thinking skills and academic performance. *Journal of Computers in Education*, *4*(4), 355-369. doi: 10.1007/s40692-017-0090-9

Duff, C., McPherson, A. C., King, G., & Kingsnorth, S. (2019). Deconstructing residential immersive life skills programming through a pedagogical lens: mechanisms that can facilitate learning for youth with disabilities. *Journal of Research in Special Educational Needs*. doi: 10.1111/1471-3802.12470

Durak, H. Y., & Şahin, Z. (2018). Investigation of the contribution of coding training in teaching candidates to the development of lifelong learning competencies. *Journal of Ege Education Technologies*, *2*(2), 55-67. Retrieved from http://dergipark.gov.tr/download/article-file/618107

Erdem, E. (2018). *The investigation of different teaching strategies during teaching programming process in block based environment in terms of different factors.* (Master's thesis, Başkent University Institute of Educational Sciences).

Erickson, A. S. G., Noonan, P. M., Zheng, C., & Brussow, J. A. (2015). The relationship between self-determination and academic achievement for adolescents with intellectual disabilities. *Research in Developmental Disabilities*, *36*, 45-54. doi: 10.1016/j.ridd.2014.09.008

European Commission (2018). Coding - the 21st-century skill. Retrieved from https://ec.europa.eu/digital-single-market/coding-21st-century-skill

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. doi:10.1016/j.compedu.2012.11.016

Good, T. L., & Brophy, J. (1995). *Contemporary educational psychology (5th ed.)*. NY: Longman Publishers.

Goffin, S., & Tull, C. (1985). Problem solving: Encouraging active learning. *Young Children, 40*(1), 28–32. Retrieved from https://psycnet.apa.org/record/1985-27772-001

Hagge, J. (2017). Scratching beyond the surface of literacy: Programming for early adolescent gifted students. *Gifted Child Today*, *40*(3), 154-162. doi: 10.1177/1076217517707233

Heckman, J. J. (2006). Skill formation and the economics of investing in disadvantaged children. *Science*, *312*(5782), 1900-1902. doi:10.1126/science.1130121

Heckman, J. J., & Masterov, D. V. (2007). The productivity argument for investing in young children. *Applied Economic Perspectives and Policy*, *29*(3), 446-493. Retrieved from https://www.nber.org/papers/w13016.pdf

Hohn, R., & Frey, B. (2002). Heuristic training and performance in elementary mathematical problem solving. *The Journal of Educational Research, 95*(6), 374–380. doi:10.1080/00220670209596612

Holt, D., Segrave, S., & Cybulski, J. L. (2013). E-Simulations for educating the professions in blended learning environments. In *IT Policy and Ethics: Concepts, Methodologies, Tools, and Applications* (pp. 1102-1123). IGI Global. doi: 10.3968/j.sll.1923156320110303.1200

Hooshyar, D., Ahmad, R. B., Yousefi, M., Fathi, M., Horng, S. J., & Lim, H. (2016). Applying an online game-based formative assessment in a flowchart-based intelligent tutoring system for improving problem-solving skills. *Computers & Education*, *94*, 18-36. doi: 10.1016/j.compedu.2015.10.013

Howland, K., & Good, J. (2015). Learning to communicate computationally with Flip: A bi-modal programming language for game creation. *Computers & Education*, *80*, 224-240. doi: 10.1016/j.compedu.2014.08.014

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. doi: 10.1016/j.compedu.2018.07.004

Israel, M., Wherfel, Q. M., Pearson, J., Shehab, S., & Tapia, T. (2015). Empowering K–12 students with disabilities to learn computational thinking and computer programming. *TEACHING Exceptional Children*, *48*(1), 45-53. doi: 10.1177/0040059915594790

Jameson, J. M., Riesen, T., Polychronis, S., Trader, B., Mizner, S., Martinis, J., & Hoyle, D. (2015). Guardianship and the potential of supported decision making with individuals with disabilities. *Research and Practice for Persons with Severe Disabilities*, *40*(1), 36-51. doi: 10.1177/1540796915586189

Jitendra, A. K., Petersen-Brown, S., Lein, A. E., Zaslofsky, A. F., Kunkel, A. K., Jung, P. G., & Egan, A. M. (2015). Teaching mathematical word problem solving: The quality of evidence for strategy instruction priming the problem structure. *Journal of Learning Disabilities, 48*(1), 51-72. doi: 10.1177/0022219413487408

Jonassen, D. H. (2000). Toward a design theory of problem solving. *Educational technology research and development*, *48*(4), 63-85. Retrieved from https://link.springer.com/article/10.1007/BF02300500

Jonassen, D. H., Howland, J., Moore, J., & Marra, R. M. (2003). *Learning to solve problems with technology*. Pearson Education.

Kafai, Y. B., & Burke, Q. (2014). *Connected code: Why children need to learn programming.* Cambridge, MA: MIT Press.

Kalelioglu, F., & Gulbahar, Y. (2014). The effects of teaching programming via scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education*, *13*(1), 33-50. Retrieved from https://www.mii.lt/informatics_in_education/pdf/infe232.pdf

Kanbul, S., & Uzunboylu, H. (2017). Importance of coding education and robotic applications for achieving 21st-century skills in North Cyprus. *International Journal of Emerging Technologies in Learning (iJET)*, *12*(01), 130-140. doi: 10.3991/ijet.v12i01.6097

Karaman, G., & Karaman, U. (2019). Comparison of Informatics Technologies and Software Development Course Curricula in 2012 and 2017. *Kastamonu Education Journal*, *27*(1), 309-318. doi: 10.24106/kefdergi.2543

Karna-Lin, E., Pihlainen-Bednarik, K., Sutinen, E., & Virnes, M. (2006). Can robots teach? Preliminary results on educational robotics in special education. In *Sixth IEEE International Conference on Advanced Learning Technologies (ICALT'06)* (pp. 319-321). IEEE. doi: 10.1109/ICALT.2006.1652433

Kátai, Z. (2015). The challenge of promoting algorithmic thinking of both sciences-and humanities-oriented learners. *Journal of Computer Assisted Learning*, *31*(4), 287-299. doi: 10.1111/jcal.12070

Kazakoff, E. R. (2014). *Cats in Space, Pigs that Race: Does self-regulation play a role when kindergartners learn to code?* (Doctoral dissertation, Tufts University). Retrieved from http://ase.tufts.edu/devtech/Theses/EKazakoff_2014.pdf

Kereluik, K., Mishra, P., Fahnoe, C., & Terry, L. (2013). What knowledge is of most worth: Teacher knowledge for 21st century learning. *Journal of Digital Learning in Teacher Education*, *29*(4), 127-140. Retrieved from https://files.eric.ed.gov/fulltext/EJ1010753.pdf

Kotthoff, L. (2016). Algorithm selection for combinatorial search problems: A survey. In *Data Mining and*

Constraint Programming (pp. 149-190). Springer, Cham. doi: 10.1007/978-3-319-50137-6_7

Lazakidou, G., & Retalis, S. (2010). Using computer supported collaborative learning strategies for helping students acquire self-regulated problem-solving skills in mathematics. *Computers & Education*, *54*(1), 3-13. doi:10.1016/j.compedu.2009.02.020

Lechelt, Z., Rogers, Y., Yuill, N., Nagl, L., Ragone, G., & Marquardt, N. (2018, April). Inclusive computing in special needs classrooms: designing for all. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (p. 517). ACM. doi: 10.1145/3173574.3174091

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

Retrieved from https://journals.sagepub.com/doi/pdf/10.1177/107621751103400208

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J. ,& Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads*, *2*(1), 32-37. Retrieved from https://users.soe.ucsc.edu/~linda/pubs/ACMInroads.pdf

Malik, S. I., Mathew, R., & Hammood, M. M. (2019). PROBSOL: A web-based application to develop problem-solving skills in introductory programming. In *Smart Technologies and Innovation for a Sustainable Future*, 295-302. doi: 10.1007/978-3-030-01659-3_34

Mechling, L. C., & Ortega-Hurndon, F. (2007). Computer-based video instruction to teach young adults with moderate intellectual disabilities to perform multiple step, job tasks in a generalized setting. *Education and Training in Mental Retardation and Developmental Disabilities*, *42*(1), 24. Retrieved from http://daddcec.org/portals/0/cec/autism_disabilities/research/publications/education_training_development_disabilities/2007v42_journals/etdd_200703v42n1p024-037_computer-based_video_instruction_teach_young_adults.pdf

Miller, P. (2009). Learning with a missing sense: What can we learn from the interaction of a deaf child with a turtle?. *American Annals of the Deaf*, *154*(1), 71-82. Retrieved from https://www.jstor.org/stable/26234580

Mohaghegh, D. M., & McCauley, M. (2016). Computational thinking: The skill set of the 21st century. *International Journal of Computer Science and Information Technologies, 7*(3), 1524-1530. Retrieved from https://unitec.researchbank.ac.nz/bitstream/handle/10652/3422/ijcsit20160703104.pdf?sequence=1&isAllowed=y

Ormrod, J. E. (2000). *Educational psychology: Developing learners. 3rd ed*. Merrill Prentice Hall.

Özsoy, G., & Ataman, A. (2017). The effect of metacognitive strategy training on mathematical problem solving achievement. *International Electronic Journal of Elementary Education*, *1*(2), 67-82.

Polya, G. (1973). *How to solve it.* Princeton NJ: Princeton University Press.

Plerou, A., & Vlamos, P. (2016). Algorithmic thinking and mathematical learning difficulties classification. *Am. J. Appl. Psychol*, *5*(5), 22. doi: 10.11648/j.ajap.20160505.11

Prajapati, R. K., Sharma, B., & Sharma, D. (2017). Significance of Life Skills Education. *Contemporary Issues in Education Research*, *10*(1), 1-6. Retrieved from https://files.eric.ed.gov/fulltext/EJ1126842.pdf

Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science*, *45*(5), 583-602. doi: 10.1007/s11251-017-9421-5

Ratcliff, C. C., & Anderson, S. E. (2011). Reviving the turtle: Exploring the use of logo with students with mild disabilities. *Computers in the Schools, 28*, 241–255. doi:10.1080/073805 69.2011.594987

Reynolds, A. J., Temple, J. A., Ou, S. R., Arteaga, I. A., & White, B. A. (2011). School-based early childhood education and age-28 well-being: Effects by timing, dosage, and subgroups. *Science*, 1203618, 360-365. doi:10.1126/science.1203618

Root, J., Saunders, A., Spooner, F., & Brosh, C. (2017). Teaching personal finance mathematical problem solving to individuals with moderate intellectual disability. *Career Development and Transition for Exceptional Individuals, 40*(1), 5-14. doi: 10.1177/2165143416681288

Salter, J. (2013). Coding for kids: schoolchildren learn computer programming. *The Telegraphy*, *27*, 2014.

Schalock, R. L., Borthwick-Duffy, S. A., Bradley, V. J., Buntinx, W. H. E., Coulter, D. L., Braig, E. M., . . . Yeager, M. H. (2010). *Intellectual disability: Definition, classification, and systems of support* (11th ed.). Washington, DC: American Association on Intellectual and Developmental Disabilities.

Smith, C. C., Cihak, D. F., Kim, B., McMahon, D. D., & Wright, R. (2017). Examining augmented reality to improve navigation skills in postsecondary students with intellectual disability. *Journal of Special Education Technology*, *32*(1), 3-11. doi: 10.1177/0162643416681159

Sternberg, R. (2003). *Cognitive psychology.* Thomson, Wadsworth.

Sternberg, R. J., & Sternberg, K. (2012). *Cognitive psychology*. Wadsworth.

Taylor, M. S. (2018). Computer programming with Pre-K through first-grade students with intellectual disabilities. *The journal of special education*, *52*(2), 78-88. doi: 10.1177/0022466918761120

Taylor, M. S., Vasquez, E., & Donehower, C. (2017). Computer programming with early elementary students with Down syndrome. *Journal of Special Education Technology, 32*, 149– 159. doi:10.1177/0162643417704439

Taylor, M., Harlow, A., & Forret, M. (2010). Using a computer programming environment and an interactive whiteboard to investigate some mathematical thinking. *Procedia-Social and Behavioral Sciences*, *8*, 561-570. doi:10.1016/j.sbspro.2010.12.078

Van Laar, E., van Deursen, A. J., van Dijk, J. A., & de Haan, J. (2017). The relation between 21st-century skills and digital skills: A systematic literature review. *Computers in human behavior*, *72*, 577-588. doi:10.1016/j.chb.2017.03.010

Virnes, M., Sutinen, E., & Kärnä-Lin, E. (2008, June). How children's individual needs challenge the design of educational robotics. In *Proceedings of the 7th international conference on Interaction design and children* (pp. 274-281). ACM.

Voogt, J., Erstad, O., Dede, C., & Mishra, P. (2013). Challenges to learning and schooling in the digital networked world of the 21st century. *Journal of computer assisted learning*, *29*(5), 403-413. doi: 10.1111/jcal.12029

Wainer, J., Ferrari, E., Dautenhahn, K., & Robins, B. (2010). The effectiveness of using a robotics class to foster collaboration among groups of children with autism in an exploratory study. *Personal and Ubiquitous Computing*, *14*(5), 445-455. doi: 10.1007/s00779-009-0266-z

Wang, D., Han, H., Zhan, Z., Xu, J., Liu, Q., & Ren, G. (2015). A problem solving oriented intelligent tutoring system to improve students' acquisition of basic computer skills. *Computers & Education*, *81*, 102-112. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360131514002231

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). doi: 10.1145/2157136.2157200

Werner, L., Denner, J., & Campe, S. (2014). Using computer game programming to teach computational thinking skills. In *Learning, education and games*, 37-53. Retrieved from https://dl.acm.org/doi/pdf/10.5555/2811147.2811150

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. doi:10.1098/rsta.2008.0118

Woods, D. R., Hrymak, A. N., Marshall, R. R., Wood, P. E., Crowe, C. M., Hoffman, T. W., Wright, J. D., Taylor, P. A., Woodhouse, K. A., & Bouchard, C. K. (1997). Developing problem solving skills: The McMaster problem solving program. *Journal of Engineering Education*, *86*(2), 75-91. Retrieved from https://onlinelibrary.wiley.com/doi/epdf/10.1002/j.2168-9830.1997.tb00270.x

Wurdinger, S., & Qureshi, M. (2015). Enhancing college students' life skills through project based learning. *Innovative Higher Education*, *40*(3), 279-286. doi: 10.1007/s10755-014-9314-3

Yu, K. C., Fan, S. C., & Lin, K. Y. (2015). Enhancing students'problem-solving skills through context-based learning. *International Journal of Science and Mathematics Education*, *13*(6), 1377-1401. Retrieved from https://link.springer.com/content/pdf/10.1007%2Fs10763-014-9567-4.pdf

ISSN 2513-835

Appendix A

Dear Students,

Please write your answers to the asked questions into the empty cell in the same row in the table below.

What exactly is our problem?	
How can we solve the problem? Why do we choose this strategy?	
How do the various pieces of information in the problem fit together? How can we use them?	

ISSN 2513-835

How much time, effort, etc. should I put into this problem? At which point should I intensify my concentration and my efforts?	
Am I on true track as I proceed to solve the problem? what extent does the initial process of my plan differ from the way I continue?	
Did I solve the problem correctly? If not Why?	