Assessment of Eighth Grade Students' Domain-General Computational Thinking Skills

Halit Karalar[1]
Muhammet Mustafa Alpaslan[1]

[1]Muğla Sıtkı Koçman University

DOI: 10.21585/ijcses.v5i1.126

Abstract

The aim of this study was to assess the domain-general CT skills of 8th grade students in Turkey. In the study, first, the domain-general CT scale was adapted to Turkish, and then relations of the CT skills with gender, computer possession, internet access, and programming experience were examined. This survey research was conducted with the 284 eighth grade students. The data were analyzed through confirmatory factor analysis, independent sample t-test and Pearson correlation test. The results of the confirmatory factor analysis and Cronbach alpha showed that the Turkish version of the domain-general CT scale was valid and reliable. T-test results showed no significant difference in the CT skills of the students according to gender, computer possession and internet access. A statistically significant difference in algorithm, evaluation, generalization, and general CT skills was found between students who learned programming and those who did not. Correlational tests revealed that there was a positive and significant relationship between the programming experience of students who learn programming and their CT skills. As students' programming experience increased, their CT skills also increased. The results of the research were discussed, and recommendations for policy-makers and implementers were included.

Keywords: Computational thinking, domain-general computational thinking, scale adaptation, middle school students

1. Introduction

It is important for students to have Computational Thinking (CT) skills in order to be successful in professions in today's digital world (Vallance & Towndrow, 2016). CT plays the role of a bridge between the fields of science, technology, engineering, and mathematics (STEM), which are the main source of innovation of 21st century digital economies, and computer science, and is an important skill for the development of computer literacy and thinking skills including problem solving (Barr & Stephenson, 2011). Since CT requires deep learning compared to superficial or rote learning (Wing, 2006), it can be transferred to other areas and problem-solving contexts. In this respect, it is argued that CT is a necessary skill not only for STEM professions but also for all other professions (Denning, 2017). What is more, CT, which is one of the skills sets that all students should have such as reading, writing and mathematics in the current century, has become a necessary skill not only for computer science but also for all fields of learning (Wing, 2006).

It is necessary to integrate CT into education systems in order to provide students with the skills they will need in the future (Grover & Pea, 2013). For this reason, CT has been included in the Next Generation Science Standards (NGSS) in the USA and integrated into STEM curricula at K12 levels (Tang et al., 2020). In parallel with these developments, CT has been integrated into education programs in many countries including Finland, Norway, South Korea, Israel, Poland, New Zealand, Estonia (Tikva & Tambouris, 2021). Studies examining CT studies published between 2006 and 2018 reported that CT was integrated into primary school curricula in 52 countries (Tang et al., 2020).

Although there is no common ground on the definition of CT, which has attracted the attention of educators for the last 15 years, it is widely accepted that CT is a thinking skill that includes a cognitive problem-solving process that can be developed in many ways, not only through computer programming (Aho, 2012; Guzdial, 2008; Selby & Woollard, 2013; Shute et al., 2017; Tsai et al., 2021; Wing, 2011; Yadav et al., 2016). Similarly, Denning (2017) argues that CT is not only the way computer scientists think, but also a necessary thinking skill for other fields. Shute et al. (2017) argued that CT can be defined as "the conceptual foundation required to solve problems effectively and efficiently (i.e., algorithmically, with or without the assistance of computers) with solutions that were reusable in different contexts". Previous studies have examined students' domain-specific CT skills (computer science or programming) in different contexts. (Erümit et al., 2020; Günbatar, 2020; Korucu et al., 2017; Oluk et al., 2018; Oluk & Korkmaz, 2016; Wu & Su, 2021) However, little is known about how the programming skills acquired from different programming environments change their domain-general CT skills; whether there is a difference between the domain-general CT skills of students who gain programming skills and those who do not especially in middle school. For this reason, two purposes were adopted in the study. First, due to lack of domain-general CT scale in Turkish version, it was aimed to adapt the "Computational Thinking Scale for Computer Literacy Education" (CTSCLE) developed by Tsai et al. (2021) to Turkish. Secondly, it was aimed to assess students' domain-general CT skills. The following questions were sought in the study:

- What is the reliability and validity of the domain-general CT scale?
- Is there a significant difference between students' domain-general CT skills and gender?
- Is there a significant difference between students' domain-general CT skills and programming experiences?
- Is there a significant difference between students' domain-general CT skills and computer possession?
- Is there a significant difference between students' domain-general CT skills and internet access?
- Is there a significant relationship between students' domain-general CT skills and programming experiences?

1.1 CT Definition

There is no consensus definition on CT. Although the first emergence of CT was based on the book "Mindstorms: Children, computers, and powerful ideas" written by Papert in 1980 (Zhang & Nouri, 2019), it was Wing (2006) who was first to introduce the concept of "Computational Thinking". He stated that CT includes problem solving, system design and understanding human behavior by using concepts related to computer science. Wing later changed the definition to highlight that CT is a thinking process, and CT is defined as a process that involves presenting problems and their solutions in a form that can be effectively processed by an information-processing agent (Wing, 2011). Later, Aho (2012) stated that the concept of formulating the problems emphasized in this definition includes the algorithm. According to Aho's definition, CT is emphasized as a thinking process in which the solution of its problems can be presented through algorithms containing sequential steps. In the functional definition developed by the Computer Science Teachers Association (CSTA) and the International Society of Technology in Education (ISTE), it was emphasized that CT is a thinking skill consisting of but not limited to 6 skill sets (ISTE & CSTA, 2011). These skill sets: (1) Formulate problems so that they can be solved by information processing units. (2) Organize and analyze data rationally. (3) Presenting data through abstractions such as simulations and models. (4) Automating problem-related solutions using algorithmic thinking. (5) Identifying, organising and implementing appropriate solutions in the most efficient and effective way. (6) Transferring the solution of the problem to other problems, making generalisations. Although the definitions differ, it can be stated that CT is a thinking process that includes problem solving and consists of a set of skill sets obtained by modeling the mental processes that a computer scientist uses while solving the problem.

CT is an umbrella concept that includes many sub-skill sets. In the literature, different classifications are made regarding the sub-skill sets included in CT (Barr & Stephenson, 2011; Brennan & Resnick, 2012; Denner et al., 2012; Grover & Pea, 2013; ISTE & CSTA, 2011; Kalelioglu et al. al., 2016; Selby & Woollard, 2013). As stated by Tsai et al. (2021), approaches other than Selby and Woollard (2013) include sub-skills specific to computer science or programming, making it difficult to measure domain-specific CT skills. The Selby and Woollard (2013) approach, which includes domain-specific CT sub-skills, consists of five sub-skills: abstraction, decomposition, algorithmic thinking, evaluation, and generalization. Abstraction refers to a mental process that involves focusing on the main points rather than the details to solve a problem. Decomposition states a mental process that involves breaking down the problems into small and manageable parts in order to solve them. Algorithmic thinking refers to a mental process in which it is determined how to solve a problem step by step. Evaluation is a mental process

that involves determining the best solution from different solutions to a problem, taking into account the resources. Generalization states a mental process that involves determining how to solve certain problems and applying them in solving other similar problems.

1.2 CT Assessment Tools

Due to the lack of a commonly accepted definition of CT, different approaches have emerged regarding how to measure CT skills. Measurement tools developed for these approaches are based on Román-González et al. (2019) under seven titles. CT diagnostic tools are the tools aiming to measure the CT capability level. CT result evaluation tools are the tools that aim to measure whether the learners have gained sufficient content knowledge thanks to a training on CT skills. CT process assessment tools are intended to provide feedback to the student, which are usually provided by automated means to develop and increase learners' CT skills. CT data mining tools are to aim to provide feedback to students by applying data mining on students' interaction records in online environments. CT skill transfer tools are intended to measure whether students are able to transfer their CT skills to different types of problems, contexts, and situations. CT perception and attitude tools are tools that aim to measure learners' perceptions and attitudes towards CT and related subjects (computer, computer science, programming, digital literacy, etc.). CT word evaluation tools are amied to measure CT dimensions and components, which are emphasized as CT language and can be expressed verbally. Tsai et al. (2021) emphasized that all of the above evaluation approaches are specific to computer science or programming, and that there is a need for domain-general CT measurement tools. Similarly, Alsop (2019) suggested that children's learning of CT skills cannot be measured by focusing only on programming constructs (CT concepts), it should also include learning behavior and metacognitive practices.

In the literature, there are several Turkish instruments to assess students' CT skills. However, these scales are domain dependent scales that are specific to programming, or they include 21st century skills. For example, "Computational Thinking Levels Scale" developed by Korkmaz, Çakır, and Özden (2015, 2017) for university and middle school students includes five skills such as "algorithmic thinking", "problem solving", "critical thinking", "creative thinking" and "cooperativity". However, this scale does not include the core CT skills such as abstraction and decomposition stated by Selby and Woodland (2013). The other available scale for assessing CT skills of Turkish middle school students is "Self-Efficacy Perception Scale for Computing Thinking Skills" developed by Gülbahar, Kert, and Kalelioğlu (2019). This scale includes factors related to programming, which make it a domain-specific scale. Since CT skills are problem-solving-based thinking skills that are necessary to not only computer science but also other fields including science, mathematics etc., there is a need for domain-general (non-computer science or programming-specific) Turkish scales. In this study, it was aimed to adapt the CTSCLE scale developed by Tsai et al. (2021) into Turkish in order to provide a domain-general CT skill scale.

1.3 Factors Related to CT Skills

Studies addressing students' CT skills have associated the different factors on students' CT skills. Some studies focused on effect of gender on CT skills. The effects of gender on CT skills are controversial. Some studies reported that there was no difference between CT skills mean score of male and female students. For instance, Alsancak (2020) examines the gender difference on CT skills with 722 Turkish secondary school students. T-test results showed that there was no significant difference between female and male students' CT skills mean score, on the one hand, ($t(719) = -.98$, $p = .33$). On the other hand, Gulbahar et al. (2020) reported that there was significant difference between female and male CT skills in favor of female students ($t(43748) = 7.42$, $p < .01$). Therefore, there is a need to test the relations between gender and CT skills measured in the CTSCLE.

Other studies addressing CT skills mostly focused on effects of having a computer, Internet access, mobile device, or other technological tools, or programming experience on CT skills. For instance, Oluk and Korkmaz (2016) investigated that the relations of CT skills with Scratch programming skills and computer use. They found that Scratch programming skills were highly correlated with the CT skills (Spearman rho= .93, $p < .01$) whereas no relation between computer use and CT skills ($U = 102$, $p = .74$). In another study, Alsancak Sarıkaya (2019) reported that students did not differ based on their computer experience ($F(5,253) = 2.25$ $p = .05$) but they did in internet use ($F(5,253) = 2.27$ $p = .048$). In another study, İbili et al. (2020) reported that CT skills did not differ based on students' programming experience ($F(2, 588) = 1.423$, $p > .05$). These findings of aforementioned studies indicate that relations between programming experience and CT skills are controversial. Thus, it would be important to examine the links of these variables with the CT skills by utilizing a domain-general CT skills scale, the CTSCLE, to better understand the role of computer related experiences on CT skills.

2. Method

2.1 Research Method

In this study, one of the quantitative research models, the survey model was used. Survey model is based on collecting data from the sample of the population and interpreting these data in order to determine the characteristics of a population or to reveal the existing situation (Fraenkel et al., 2012). This method was chosen because the aim of this study is to examine the domain-general CT skills of middle school students. In addition, comparative analysis of CT skills according to various variables were made using the relational research method. Relational research model is a method used to examine the relationships between variables (Fraenkel et al., 2012).

2.2 Participants

The population of this study is 8th grade students in Turkey. Since collecting data from the entire population is not possible in terms of economy and time, the researcher can define an accessible population. Therefore, one of Turkey's western provinces where it could be easily accessed by researchers was determined to be the accessible population. Six public schools (3 rural and 3 urban areas) in this province were randomly determined. 284 middle school students studying in these schools voluntarily participated in the study. Characteristics of the participants were given in Table 1.

Table 1. Demographic characteristics of the participants

Demographic feature		Frequency	Percent
Gender			
	Boy	137	48.2
	Girl	147	51.8
Computer possession			
	No	103	36.3
	Yes	181	63.7
Internet access			
	No	23	8.1
	Yes	261	91.9
Programming environments experienced (multiple choice be selectable)			
	Scratch	152	0.54
	Code.org	103	0.36
	Arduino	22	0.08
	Other (Google CS First, Hour of Code, Blockly.games, CodeMonkey, Codecademy, Codesters etc.)	78	0.27
	No programming experience	60	0.21

2.3 Data Collection Tools

The data collection tool in this study consisted of two parts. In the first part, the personal information form aimed at collecting the students' demographic information (gender, etc.) was included and in the other part, the Turkish version of the CTSCLE was placed.

The CTSCLE was developed by Tsai et al. (2021) with 19 items in five factors (abstraction, decomposition, algorithmic thinking, evaluation, and generalization). The abstraction, algorithmic thinking, evaluation, and generalization sub-dimensions of the scale consist of four items and the decomposition sub-dimension consists of three items. The scale was originally administrated with 388 students (255 boys, 132 girls, mean age 14.58) studying at a secondary school in Taiwan. A 5-point Likert-type scale was used in the form of Strongly Disagree (1) and Strongly Agree (5). Tsai et al. (2021) found that 19 items were grouped under five factors and the total variance explained by 19 items was 64.03%. It was reported that the Alpha reliability coefficient value of the whole

scale was 0.91 and varied between 0.74 and 0.83 in the sub-dimensions (See Table 2). For the discrimination validity of the scale, the AVE (average variable extracted) values of each sub-dimension were examined and whether the square roots of the AVE values were greater than the correlation values between the sub-dimensions. Since the AVE values of the scale sub-dimensions were greater than 0.50 (0.74 - 0.83) and the square roots of the AVE values of each sub-dimension were larger than the correlation values in the related sub-dimensions, it was concluded that the scale provides the decomposition validity. The reliability and validity analysis showed that the scale was reliable and valid in Taiwanese sample.

Table 2. Sample example of sub-dimensions in the CTSCLE and their Cronbach values

Sub-dimension	Example item	Item number	Cronbach alpha*
Abstraction	I usually try to analyze the common patterns of different problems.	4	.81
Decomposition	I usually think about how to split a big problem into several small ones.	3	.74
Algorithmic thinking	I usually try to lay out the steps of a solution.	4	.77
Evaluation	I usually try to find the most effective solution for a problem.	4	.83
Generalization	I usually think about how to apply a solution to other problems.	4	.75

* Cronbach alpha values reported by Tsai et al. (2021).

2.4 Process and Data Analysis

Turkish version of the CTSCLE was prepared by following steps suggested by Çapık et al. (2018). First, the permission was granted from the authors who developed the scale. Later, two doctoral experts who graduated from the United States translated the scale into Turkish. In the translation form English to Turkish, semantic, and conceptual deduction were made. Next, the opinions of three researchers from the departments of Turkish Language Education, Computer Education and Instructional Technologies, and English Language Education were taken about the semantic translation. According to their opinions, revisions were made. Then, back-translation to English was made by two independent experts and compared with the original version of the scale to ensure that each version was semantically compatible with each other. Later, two information technologies in-service teachers checked the Turkish scale to determine its comprehensibility by the target audience. After taking the opinions of the information technologies teachers, the Turkish version of the scale was finalized.

More than one statistical technique was used in data analysis. First, outliers and missing data were checked. Then, the normality values of the data were examined. Since the skewness and kurtosis values were in the range of -1 and +1, the data were accepted as having a normal distribution (Hair et al., 2019). Next, Cronbach alpha coefficient was calculated to determine the level of reliability. In the literature, data with a Cronbach alpha value above .70 can be considered reliable (Hair et al., 2019). In order to examine the factor structure, measurement model or confirmatory factor analysis (also known as the measurement model or confirmatory factor analysis [CFA]), which is one of the structural equation model methods, was performed in the MPlus 7.0 program. CFA, a statistical technique that tests the construct validity of the scale, was used to test how fit the data obtained to the previously determined model. CFA is recommended to use when the factor structures are pre-determined. Because the factor names and structures had been already known in the study, CFA was used to confirm that. The maximum likelihood method was used as the estimation method in the CFA because the normal distribution assumptions were not severely violated. Parallel to the literature, good fit values were accepted as Chi-square / df ratio less than 3, RMSEA value lower than .06 and CFI value greater than .95 (Hu & Bentler, 1999). In addition, t-test and one-way ANOVA tests were used to test the difference according to demographic variable groups, since the data showed normal distribution. Lastly, Pearson correlation test was used to test the relationship between the programming environments experienced and their domain-general CT skills.

3. Results

3.1 Reliability and Validity of the Adapted CTSCLE

In this study, the validity and reliability analysis of the scale were done before interpreting the data. First, Cronbach's alpha value was calculated for each sub-dimension for reliability. In Table 3, Cronbach's alpha values for each sub-dimension were given along with descriptive statistics. Since the Cronbach alpha values were higher than the acceptable level of .70, the scale was at a reliable level.

Table 3. Descriptive statistics

	Mean	SD	Skewness	Kurtosis	Alpha
Abstraction	3.49	0.95	-0.35	-0.33	.86
Decomposition	3.37	1.01	-0.36	-0.28	.86
Algorithmic Thinking	3.75	0.99	-0.76	0.20	.90
Evaluation	3.83	0.99	-0.80	0.29	.89
Generalization	3.62	0.91	-0.51	0.25	.82
Total	3.61	0.85	-0.70	0.53	.96

Results of CFA showed that the chi-square value was found to be 325.25, $\chi 2(142) = 325.37$. The chi-square / df ratio was calculated as 2.29. Although this ratio is lower than the 3.00 value accepted in the literature, the chi-square value is a statistic that is affected by the number of people in the sample. For this reason, other fit values are recommended to be checked.

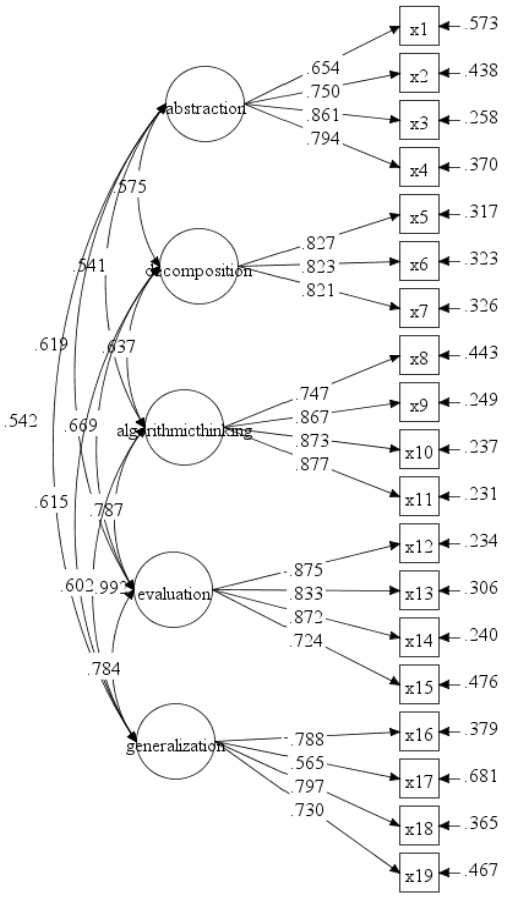

Figure 1. Factor loadings of the CTSCLE

The most accepted and more reliable (robust) fit indices in the literature are generally RMSEA, TLI and CFI values. The RMSEA value was found as .057 CI (.047, .067). This value is smaller than the limit value .06 and indicates a good fit. In addition, CFI and TLI values were calculated as .95 and the limit value was around .95. SRMR value was found as .035. These values showed that the factor structure of the scale was in good fit with the data and accepted as evidence for the construct validity of the scale. In addition, factor loadings higher than .30 indicated that each item was related to the factor (Figure 1). Finally, there was a high level of correlation between the factors of the scale. This shows that the structure is similar. After testing the validity and reliability of the scale, the average value for each factor was calculated and given in Table 3. According to these average values, it showed that students had intermediate level skills in abstraction, decomposition and generalization sub-dimensions, and high skill level in algorithm and evaluation sub-dimensions (low level: 1.00-2.33, medium level: 2.34-3.66 and high level: 3.67- 5.00). While the lowest mean score was in the decomposition dimension, the highest one was in the evaluation sub-dimension.

3.2 Students' CT Skills and Gender

It was found for females in the highest mean score in evaluation and the lowest one was in the decomposition sub-dimension. Similar findings were observed for male students. Whether the students' CT score changed according to their gender was tested with the independent sample t-test. Although female students had a higher average than boys in all sub-dimension and total, results of the t-test showed that these differences were not statistically significant (See Table 4).

Table 4. Gender comparison on the domain-general CT skills

	Boy (n = 137)		Girl (n = 147)			
	Mean	*SD*	Mean	*SD*	*t*	Cohen's *d*
Abstraction	3.45	0.98	3.53	0.92	0.74	0.08
Decomposition	3.31	1.05	3.42	0.96	0.90	0.11
Algorithmic Thinking	3.66	1.01	3.83	0.96	1.48	0.17
Evaluation	3.78	1.04	3.88	0.94	0.90	0.10
Generalization	3.57	1.00	3.66	0.81	0.84	0.10
Total	3.55	0.91	3.67	0.80	1.10	0.13

Note. df = 282.

3.3 Students' CT Skills and Programming Experiences

The CT skills of students who had never learned programming and those who have taken any programming course were compared in order to test the effect of students' programming experiences on their CT skills. According to the results of the independent sample t-test, there was a significant difference in some sub-dimensions while not in others (See Table 5). A statistically significant difference was found in favor of students taking programming language in algorithm, evaluation, generalisation, and total averages. The highest significant difference was in the evaluation sub-dimension (Cohen's d = .37), while the lowest significant difference was calculated in the total mean score (Cohen's d = .30). These findings showed that Algorithmic Thinking, Evaluation and Generalization skills can be gained with programming skills while Abstraction and Decomposition skills can be improved without programming experiences.

Table 5. Students' programming experiences comparison on the CT skills

	Yes (n = 224)		No (n = 60)			
	Mean	*SD*	Mean	*SD*	*t*	Cohen's *d*
Abstraction	3.53	0.92	3.32	1.04	1.54	.21
Decomposition	3.39	0.97	3.29	1.13	0.68	.09
Algorithmic Thinking	3.83	0.92	3.46	1.16	2.59 *	.35
Evaluation	3.91	0.94	3.53	1.10	2.68 *	.37
Generalization	3.68	0.87	3.38	1.01	2.32 *	.32
Total	3.67	0.81	3.40	1.00	2.21 *	.30

Note. * $p < .05$. df = 282.

3.4 Students' CT Skills and Computer Possession

CT skills of students who had a computer and those who did not was compared by utilizing independent sample t-test. Descriptive statistics and results of t-tests were given in Table 6. Students who had a personal computer rated the highest mean value on the evaluation sub-dimension whereas the lowest one was the decomposition. Similarly, students who did not have a personal computer did the highest mean value on the evaluation and the lowest was on the decomposition. Results of t-tests yielded no significant difference between students who had a personal computer or not on their CT skills.

Table 6. Students' computer possession comparison on the CT skills

	Yes (n = 181)		No (n = 103)			
	Mean	*SD*	Mean	*SD*	*t*	Cohen's *d*
Abstraction	3.46	0.97	3.54	0.92	.71	.08
Decomposition	3.33	1.05	3.43	0.92	.80	.09
Algorithmic Thinking	3.73	1.00	3.79	0.98	.52	.05
Evaluation	3.84	1.00	3.82	0.97	.14	.02
Generalization	3.63	0.94	3.60	0.85	.26	.03
Total	3.60	0.87	3.64	0.82	.38	.03

Note. df = 282.

3.5 Students' CT Skills and Internet Access

CT skills of students who had internet access and those did not was compared by utilizing Welch's t-tests because of extremely unequal sample size across the groups. Descriptive statistics and results of t-tests were given in Table 7. Both student groups rated the highest mean value on the evaluation sub-dimension whereas the lowest one was the decomposition. Results of Welch's t-tests yielded no significance difference between students who had internet access or not on their CTs.

Table 7. Students' internet access comparison on the CT skills

	Yes (n = 261)		No (n = 23)			
	Mean	*SD*	Mean	*SD*	*t*	Cohen's *d*
Abstraction	3.49	0.94	3.46	1.06	.17	.03
Decomposition	3.38	1.00	3.26	1.09	.53	.11
Algorithmic Thinking	3.78	0.97	3.40	1.09	1.76	.37
Evaluation	3.85	0.98	3.62	1.07	1.08	.22
Generalization	3.62	0.90	3.58	0.99	.23	.03
Total	3.62	0.84	3.46	0.98	.87	.18

Note. df = 282.

3.6 The Relationship between Students' CT Skills and Programming Experiences

Students can take computer and programming courses as elective or compulsory courses at different grade levels and generally learn programming in Code.org, Scratch, Ardunio and other programming environments. Scoring was made according to the order of importance of programming languages in order to determine the level of programming courses. Accordingly, Code.org and others (Google CS First, Hour of Code, Blockly.games, CodeMonkey, Codecademy, Codesters etc.) created a scale-type measurement tool with one score, Scracth two points, and Ardunio three points. In this scale, the scores were between the highest score "7 (experiencing all programming environments)" and the lowest "0 (no programming experiences)". Programming experiences average value was calculated as 1.94 (SD = 1.61) which showed that students had low-level programming experiences. The correlation coefficient between the level of programming environments experienced and CT skills were given in Table 8.

All correlation coefficients between CT sub-dimensions were statistically significant and the highest correlation coefficient was found between abstraction and decomposition (r = .78). On the other hand, a statistically significant relationship of the programming experiences was found to be the algorithm, evaluation, and the total score. Accordingly, this result showed that as the students' experience in different learning environments increases,

algorithmic thinking, evaluation and total computational thinking skills would increase.

Table 8. Relationship between students' programming experiences and CT skills

		1	2	3	4	5	6	7
1	Abstraction	1						
2	Decomposition	.78**	1					
3	Algorithmic Thinking	.75**	.74**	1				
4	Evaluation	.72**	.65**	.63**	1			
5	Generalization	.68**	.65**	.69**	.77**	1		
6	Total	.78**	.76**	.71**	.70**	.65**	1	
7	Programming experiences	.11	.06	.15*	.18**	.11	.14*	1

Note. * p <.05. ** p <. 01.

4. Discussion

Little has been known about how different programming environments would promote students' domain-general CT skills and whether there is a difference between the CT skills of the middle school students who have programming experience and those who do not. In this study, due to the lack of domain general CT skill scale in Turkey, first, it was aimed to adapt the CTSCLE developed by Tsai et al. (2021) to Turkish and then, to examine the domain-general CT skills of the students through this scale.

Results showed that there was no significant difference in CT skills of the students according to gender. This finding is parallel to many studies (Alsancak Sarıkaya, 2019; Atmatzidou & Demetriadis, 2016; İbili et al., 2020; Korucu et al., 2017; Mindetbay et al., 2019; Oluk & Korkmaz, 2016; Tsai et al., 2021; Wu & Su, 2021; Yağcı, 2018). In the light of these findings, it can be said that gender does not play an active role in CT. However, there are also studies that contrast with this finding. A study conducted with 5th and 6th grade level 2015-2016-2017-2018 year Bebras (International Challenge on Informatics and Computational Thinking) activity to participate in the 97.494 students in Turkey reported that girls were more successful than boys in CT skills (Gulbahar et al., 2020). The large sample size and the environments used in teaching programming may have caused this difference. There may also be differences in CT skills of students in different programming learning environments (Ardito et al., 2020; Pala & Mıhçı Türker, 2019; Wu & Su, 2021; Yildiz Durak, 2020).

One important finding of the study is that there was a significant difference in the CT skills of the students according to whether they have gain programming skills or not. In this study, it was also found that the algorithmic thinking, evaluation, and generalization skills of students who gained programming skills in different environments were statistically higher than students who did not learn programming skills whereas no significant difference in abstraction and decomposition skills was found. This finding showed that students could improve their abstraction and decomposition skills even without learning any programming. In the study conducted by Günbatar (2020), it was found that middle school students who took the Information Technologies & Software (IT&S) course and learned programming had significantly higher CT skills than those who did not take the course. The development of students' CT skills after they are exposed to programming courses is actually expected. Moreover, learning any programming language is seen as one of the most effective ways to improve students' CT skills (Lye & Koh, 2014). In the previous studies, it was found that there was a high-level significant relationship between Scratch programming skills of middle school students and their CT skills (Oluk & Korkmaz, 2016). Also, students 'algorithmic thinking skills were improved through learning any programming language (Grover et al., 2016). Furthermore, programming teaching with Scratch improved middle school students' CT and algorithmic thinking skills (Erümit et al., 2020; Oluk et al., 2018) and problem solving, algorithmic thinking and creative thinking skills (Alsancak Sarıkaya, 2019; Yünkül et al., 2017).

Another result of the research was that there was no significant difference in the CT skills of the students who had a computer or not and internet access or not. In previous studies, CT skills did not differ in terms of having a computer (İbili et al., 2020), duration of computer use (Oluk & Korkmaz, 2016), weekly internet usage duration, mobile technology usage qualification and mobile technology (Korucu et al., 2017). In a study, Alsancak Sirakaya (2020) found that students' CT skills differed according to their internet experience, mobile device experience, mobile internet experience and daily mobile internet usage time but did not differ according to their computer experience, the number of times they checked their mobile devices in a day and the purpose of using mobile

technology.

A significant positive relationship was found between the programming experience and CT skills. This finding implies that as students' programming experience in different environments such as Code.org, Scartch, Arduino increase, their CT skills also increase significantly. In the study conducted by Oluk and Korkmaz (2016), it was found that there is a highly significant relationship between the Scratch programming skills of middle school students and their CT skills. Tsai et al. (2021) found that students with 1 year or more programming experience had higher CT skills than those who did not. These findings show that students' programming experience is an important factor in terms of CT skills. However, there are studies that contradict these findings. In the study conducted by Alsancak Sarıkaya (2019), it was concluded that there was no significant difference in CT skills between students who saw themselves as novice and intermediate level programming skills. In another study, Ibili et al. (2020) reported that CT skills did not differ based on students' programming experience. That may be due to the difference in educational approaches, techniques, and activities in the teaching programming environment (Erümit et al., 2020).

As a result of the study, it was found that some students were about to graduate from middle schools without programming experiences; however, their abstraction and decomposition skills developed in some way, but algorithmic thinking, evaluation and generalization skills were not developed sufficiently. Results of this study suggest to complete the lack of computer laboratories in these schools and to employ qualified information technology teachers in order to provide CT skills, especially for students at schools located in rural areas where they suffer the lack of equipped computer laboratories. By doing so, students in rural areas will be provided with equal opportunities in order for them to be effectively prepared for 21st century professions by improving their CT skills.

This study has some limitations. The first limitation of the study is that the number of middle school students is small; therefore, the generalizability should be approached with caution. Nevertheless, the number of the sample in the study was above the recommended participants for statistical tests and research findings have potential to portray middle school students' CT skills in Turkey. The second limitation of the study is that different variables other than gender, programming experience, computer possession, and internet access could be used to explain the CT skills. Lastly, the future studies could examine students' CT skills in more depth with qualitative studies.

5. Conclusion

In this study, the Computational Thinking Scale for Computer Literacy Education, developed by Tsai et al. (2020), was adapted to Turkish, and its validity and reliability were examined. The CFA and Cronbach alpha results showed that the adapted scale was valid and reliable. Although the scale adapted in the study was a domain-general, as suggested by Tsai et al. (2021), the scale can be used for measuring students' domain-specific CT skills in other disciplines including physics and STEM by adding appropriate words in expression. Addition to this, it was emphasized that the scale can be used to measure CT skills of all participants at the secondary school level and above (Tsai et al., 2021). Moreover, the scale should be used by researchers working in different fields to measure CT skills, regardless of domain-general or domain-specific.

It was also concluded that the domain-general CT scale was suitable for Turkish culture, and there was no significant difference in students' CT skills according to gender, computer possession and Internet access. A statistically significant difference in algorithmic thinking, assessment, generalization, and general CT skills was found between students who learned programming and those who did not, in favor of students learning programming. In the study, it was concluded that there was a positive and significant relationship between the programming experience of students who learn programming and their CT skills. As students' programming experience increased, their CT skills also increased.

References

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 833–835. https://doi.org/10.1093/comjnl/bxs074

Alsop, Y. (2019). Assessing computational thinking process using a multiple evaluation approach. *International Journal of Child-Computer Interaction, 19*, 30–55. https://doi.org/10.1016/j.ijcci.2018.10.004

Alsancak Sarıkaya, D. (2019). Programlama öğretiminin bilgi işlemsel düşünme becerisine etkisi [The effect of programming teaching on computational thinking]. *The Journal of Turkish Social Research, 23*(2), 575–590.

Alsancak Sirakaya, D. (2020). Investigation of computational thinking in the context of ICT and mobile

technologies. *International Journal of Computer Science Education in Schools, 3*(4), 50–59. https://doi.org/10.21585/ijcses.v3i4.73

Ardito, G., Czerkawski, B., & Scollins, L. (2020). Learning computational thinking together: Effects of gender differences in collaborative middle school robotics program. *TechTrends, 64*(3), 373–387. https://doi.org/10.1007/s11528-019-00461-8

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. https://doi.org/10.1016/j.robot.2015.10.008

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. https://doi.org/10.1145/1929887.1929905

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Annual American Educational Research Association Meeting, Vancouver, BC, Canada*, 1–25. https://doi.org/10.1.1.296.6602

Çapık, C., Gözüm, S., & Aksayan, S. (2018). Kültürlerarası ölçek uyarlama aşamaları, dil ve kültür uyarlaması: Güncellenmiş rehber [Intercultural scale adaptation stages, language and culture adaptation: Updated guideline]. *Florence Nightingale Journal of Nursing 26*(3), 199–210. https://doi.org/10.26650/fnjn397481

Denner, J., Werner, L., & Ortiz, E. (2012). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts? *Computers & Education, 58*(1), 240–249. https://doi.org/10.1016/j.compedu.2011.08.006

Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM, 60*(6), 33–39. https://doi.org/10.1145/2998438

Erümit, A. K., Şahin, G., & Karal, H. (2020). YAP programlama öğretim modelinin öğrencilerin bilgi-işlemsel düşünme becerilerine etkisi [The effects of YAP programming teaching model on students' computational thinking skills]. *Kastamonu Education Journal, 28*(3), 1529–1540. https://doi.org/10.24106/kefdergi.3915

Fraenkel, J. R., Wallen, N. E., & Hyun, H. H. (2012). *How to design and evaluate research in education* (8th ed.). Boston, MA: McGraw Hill.

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

Grover, S., Pea, R., & Cooper, S. (2016). Factors influencing computer science learning in middle school. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16, March*, 552–557. https://doi.org/10.1145/2839509.2844564

Guzdial, M. (2008). Education: Paving the way for computational thinking. *Communications of ACM, 51*, 25–27. https://doi.org/10.1145/1378704.1378713

Gülbahar, Y., Kalelioğlu, F., Doğan, D., & Karataş, E. (2020). Bilge Kunduz: Enformatik ve bilgi-işlemsel düşünmeyi kavram temelli öğrenme için toplumsal bir yaklaşım [Bebras: An approach for concept based learning of informatics and computational thinking]. *Ankara University Journal of Faculty of Educational Sciences, 53*(1), 241–272. https://doi.org/10.30964/auebfd.560771

Gülbahar, Y., Kert, S. B., & Kalelioglu, F. (2019). Bilgi işlemsel düşünme becerisine yönelik öz yeterlik algısı ölçeği: Geçerlik ve güvenirlik çalışması [The self-efficacy perception scale for computational thinking skill: validity and reliability study]. *Turkish Journal of Computer and Mathematics Education, 10*(1), 1–29. https://doi.org/10.16949/turkbilmat.385097

Günbatar, M. S. (2020). Computational thinking skills, programming self-efficacies and programming attitudes of the students. *International Journal of Computer Science Education in Schools, 4*(2), 24–35. https://doi.org/10.21585/ijcses.v4i2.96

Hair, J. F., Blacks, W. C., Babin, B. J., & Anderson, R. E. (2019). *Multivariate data analysis* (8th ed.). Andover, Hampshire, UK: Cengage Learning.

Hu, L. T., & Bentler, P. M. (1999). Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. Structural Equation Modeling, 6(1), 1–55. https://doi.org/10.1080/10705519909540118

İbili, E., Günbatar, M. S., & Sarıkaya, M. (2020). Bilgi-işlemsel düşünme becerilerinin incelenmesi: Meslek liseleri örneklemi [An examination of the computational thinking skills: Sample of vocational high schools]. *Kastamonu Education Journal, 28*(2), 1067–1078. https://doi.org/10.24106/kefdergi.683577

ISTE, & CSTA. (2011). *Operational definition of computational thinking for K-12 education*. http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf

Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing, 4*(3), 583–596.

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior, 72*, 558–569. https://doi.org/10.1016/j.chb.2017.01.005

Korkmaz, Ö., Çakır, R., & Özden, M. Y. (2015). Bilgisayarca düşünme beceri düzeyleri ölçeğinin (BDBD) ortaokul düzeyine uyarlanması [Computational thinking levels scale (CTLS) adaptation for secondary school level]. *Gazi Journal of Educational Science, 1*(2), 143–162.

Korucu, A. T., Gencturk, A. T., & Gundogdu, M. M. (2017). Examination of the computational thinking skills of students. *Journal of Learning and Teaching in Digital Age, 2*(1), 11–19.

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. https://doi.org/10.1016/j.chb.2014.09.012

Mindetbay, Y., Bokhove, C., & Woollard, J. (2019). What is the relationship between students' computational thinking performance and school achievement? *International Journal of Computer Science Education in Schools*. https://doi.org/10.21585/ijcses.v0i0.45

Oluk, A., & Korkmaz, Ö. (2016). Comparing students' Scratch skills with their computational thinking skills in terms of different variables. *International Journal of Modern Education and Computer Science, 8*(11), 1–7. https://doi.org/10.5815/ijmecs.2016.11.01

Oluk, A., Korkmaz, Ö., & Oluk, H. A. (2018). Effect of Scratch on 5th graders' algorithm development and computational thinking skills. *Turkish Journal of Computer and Mathematics Education (TURCOMAT), 9*(1), 54–71. https://doi.org/10.16949/turkbilmat.399588

Pala, F. K., & Mıhçı Türker, P. (2019). The effects of different programming trainings on the computational thinking skills. *Interactive Learning Environments*, *0*(0), 1–11. https://doi.org/10.1080/10494820.2019.1635495

Román-González, M., Moreno-León, J., & Robles, G. (2019). Combining assessment tools for a comprehensive evaluation of computational thinking interventions. In S.-C. Kong & H. Abelson (Eds.), *Computational Thinking Education* (pp. 79–98). Singapore: Springer. https://doi.org/10.1007/978-981-13-6528-7_6

Selby, C., & Woollard, J. (2013). *Computational thinking: The developing definition*. https://eprints.soton.ac.uk/356481/

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*, 142–158. https://doi.org/10.1016/j.edurev.2017.09.003

Tang, K. Y., Chou, T. L., & Tsai, C. C. (2020). A Content analysis of computational thinking research: An international publication trends and research typology. *Asia-Pacific Education Researcher, 29*(1), 9–19. https://doi.org/10.1007/s40299-019-00442-8

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers and Education, 148*, 1-22. https://doi.org/10.1016/j.compedu.2019.103798

Tikva, C., & Tambouris, E. (2021). Mapping computational thinking through programming in K-12 education: A conceptual model based on a systematic literature Review. *Computers and Education, 162*, 1-23. https://doi.org/10.1016/j.compedu.2020.104083

Tsai, M.-J., Liang, J.-C., & Hsu, C.-Y. (2021). The computational thinking scale for computer literacy education. *Journal of Educational Computing Research, 59*(4), 579–602. https://doi.org/10.1177/0735633120972356

Vallance, M., & Towndrow, P. A. (2016). Pedagogic transformation, student-directed design and computational thinking. *Pedagogies: An International Journal, 11*(3), 218–234. https://doi.org/10.1080/1554480X.2016.1182437

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

https://doi.org/10.1145/1118178.1118215

Wing, J. M. (2011). *Research notebook: Computational thinking - what and why? The Link*. http://www.cs.cmu.edu/sites/default/files/11-399_The_Link_Newsletter-3.pdf

Wu, S. Y., & Su, Y. S. (2021). Visual programming environments and computational thinking performance of fifth- and sixth-grade students. *Journal of Educational Computing Research, 2*. https://doi.org/10.1177/0735633120988807

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in K-12 classrooms. *TechTrends, 60*(6), 565–568. https://doi.org/10.1007/s11528-016-0087-7

Yağcı, M. (2018). A study on computational thinking and high school students' computational thinking skill levels. *International Online Journal of Educational Sciences, 10*(2), 81–96. https://doi.org/10.15345/iojes.2018.02.006

Yildiz Durak, H. (2020). The effects of using different tools in programming teaching of secondary school students on engagement, computational thinking and reflective thinking skills for problem solving. *Technology, Knowledge and Learning, 25*(1), 179–195. https://doi.org/10.1007/s10758-018-9391-y

Yünkül, E., Durak, G., Çankaya, S., & Abidin, Z. (2017). Scratch yazılımının öğrencilerin bilgisayarca düşünme becerilerine etkisi [The effects of Scratch software on students' computational thinking skills]. *Necatibey Faculty of Education Electronic Journal of Science and Mathematics Education, 11*(2), 502–517.

Zhang, LC, & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers and Education, 141*. https://doi.org/10.1016/j.compedu.2019.103607