International Journal of Computer Science Education in Schools, December 2021, Vol. 5, No. 2
ISSN 2513-8359

A Computer Science Unplugged Activity: *CityMap*

Merve Yıldız[1]
Hasan Karal[2]

[1]Karadeniz Technical University
[2]Trabzon University

DOI: 10.21585/ijcses.v5i2.110

Abstract

The purpose of this study is to introduce a developed activity which is named *CityMap* and to present an example of the implementation and evaluation process of a computer science unplugged activity through this activity. The aim of this activity is to write algorithms of going from one place to another using step by step instructions. The rules are also to reach the destination with the shortest way and the least number of steps using correct instructions. Accordingly, a map and a worksheet was designed to implement the *CityMap* which is related to daily life and scenario based. For the evaluation, an answer key was prepared and scoring criteria were determined. Then, the case study was used as a research method and the process of writing algorithms using step by step instructions of students was examined by implementing the the activity with 15 sixth grade students. Both individual and group evaluation were made and for this process game components were used. The findings revealed that, in general, students could wrote algorithms step by step instructions for the tasks determined in the activity. In addition, during the implementation, it was observed that using of the gamification made the activity more enjoyable.

Keywords: computer science unplugged, writing algorithms using step by step instructions, computational thinking, gamification

1. Introduction

In recent years, the concept of computational thinking has come to the fore as computer science has become a multidisciplinary field and the teaching of programming has spread rapidly at the K-12 education. Computational thinking, which is described as one of the 21st century skills (Grover 2018; Tabesh 2017), is seen as a basic literacy skill for digital age learners (Wing 2006). When the literature is examined, the concept of computational thinking was first put forward by Papert (1980) as *computational ideas*. According to this, in the most general sense, Wing (2006) defined computational thinking as *problem solving, system design and understanding of human behavior by making use of the concepts of computer science*. In another study, Aho (2012) described computational thinking as a thinking process that involves in formulating problems so their solutions can be represented as computational steps and algorithms. On the other hand, the International Society for Technology in Education (ISTE) and Computer Science Teacher Association (CSTA) published an operational definition about computational thinking. According to this definition, computational thinking is a problem solving process that includes (but is not limited to) the following characteristics: -Formulating problems in a way that enables us to use a computer and other tools to help solve them, -Logically organising and analysing data, -Representing data through abstractions such as models and simulations, -Automating solutions through algorithmic thinking, -Identifying, analysing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources, -Generalising and transferring this problem solving process to a wide variety of problems (CSTA and ISTE 2011). Although there are different definitions of the concept of computational thinking, it is emphasized in the literature that computational thinking is mostly related to problem solving and algorithmic thinking (Şahiner 2017; Kalelioğlu et al. 2016; ISTE 2015; Bundy 2007). In this context, for this study, computational thinking is considered as writing algorithms using step by step instructions towards the solution of a problem.

In the globalizing world, it is crucial to gain computational thinking skills (Barr and Stephenson 2011). In the Horizon Report (2017), it is stated that computational thinking skill is one of the six subjects that should be integrated into education at K-12 level. Similarly, in the report published by the European Commission, justifications for integrating computational thinking into educational programs are presented, such as improving 21st century skills and strengthening employment opportunities (Bocconi et al. 2016). For this reason, computational thinking has become the focus of studies in the field of education. When the literature is examined, it is seen that there are different teaching methods for the improvement of computational thinking skill. One of the popular tools that play a role in the improvement of computational thinking skill is computer science (Grover 2018). It can be said that there are two different approaches in the teaching of computer science: plugged and unplugged (Brackmann et al. 2017). For plugged; coding and robotics workshops, programming course, various digital tools and games, and for unplugged; paper-pencil activities, coloring, puzzles and card games can be given as examples.

Unplugged activities are one of the approaches that provide the teaching of computer science-related knowledge and skills that have become popular over the past few years. This approach can be adapted to almost any environment, as it enables educators to carry out their educational activities without the need for electricity, computers, internet or similar technological tools (Şendurur 2019; Nishida et al. 2009). Researches have shown that computer science unplugged activities contribute to the acquisition of basic concepts related to computer science (Hermans and Aivaloglou 2017; Wohl et al. 2015; Taub et al. 2009), support improvement of computational thinking (Leifheit et al. 2018; Jagušt et al. 2018; Rodriguez 2015), provide an entertainment element (like a magic show) for the lesson (Bell and Vahrenhold 2018; Curzon 2014) and help to overcome obstacles such as misconception or negative attitude towards programming (Bell and Vahrenhold 2018; Şendurur 2018). Additionally, it is possible to come across studies emphasizing that the use of game components for teaching computer science is an effective source of motivation (Tsarava et al. 2017; Kotini and Tzelepi 2015; Nishida et al. 2009), increases class participation performance (Ibanez et al. 2014; Swacha and Baszuro 2013) and strengthens collaboration (Li et al. 2013). Also Voigt et al. (2010) state that creating a competition learning environment with computer science unplugged activities provides a more enjoyable learning process. From this point of view, *CityMap*, which is related to daily life, scenario based and includes gamification, was developed. Thus, it is aimed to present an example of the implementation and evaluation process of a computer science unplugged activity through this activity.

2. Literature Review

The CS Unplugged (2019) organization, founded by Tim Bell, Ian Witten and Michael Fellows, was the first to define unplugged activities. They state that the purpose of these activities, which they call computer science unplugged, is to provide an understanding of the philosophy behind computer science, considering the difficulties experienced in teaching programming. Although programming is perceived as a difficult and complex process, the underlying algorithmic structure is actually a phenomenon that exists in daily life. Each of the daily routine tasks is an algorithm and these algorithms (tasks) are performed after they are modeled in mental processes. At this point, it is stated that computer science unplugged activities act as a bridge that facilitates this mental process (Şendurur 2018). Hence, Bell and Vahrenhold (2018) argue that computer science unplugged activities are a fun and flexible pedagogical approach that does not require any technological tools and are based on learning by doing and experiencing.

In a computer science uplugged activity conducted with 160 middle school students without programming experience, a significant increase was determined between the pre-test and post-test scores of the students in terms of programming and computational thinking (Threekunprapa and Yasri 2020). Leifheit et al. (2018) discussed computational thinking as algorithmic concepts in their study with 3rd and 4th grade students. As a result of the short test they applied at the end of each computer science unplugged activity, they found that students' interest and motivation towards programming were high and they concluded that the game-based computer science unplugged activity approach was useful in teaching algorithmic computational thinking. Besides that Brackman et al. (2017), in their quasi-experimental study with 5th and 6th grade students, found that there was a significant difference between the experimental and control groups in the development of students' computational thinking skills and reached the conclusion that this difference provided an evidence about the effectiveness of computer science unplugged activities.

In another experimental study, computer science unplugged activities were applied to students before teaching block-based programming (Hermans and Aivaloglou 2017). It was found that students used the concepts they learned about programming more and their self-efficacy was higher. Similarly, Wohl et al. (2015) compared Scratch, Cubelets, and unplugged activities in teaching computer science to children aged 5-7, and stated that computer science unplugged activities made more contribution to the understanding of the concept of algorithm than others. Taub et al. (2009), in their study with middle school students, affirmed that unplugged activities are more effective in understanding computer science. Besides these, Nishida et al. (2009), in their study with 10 university students in the programming course, concluded that computer science unplugged activities are a fun way to understand and learn concepts and provide high motivation. In a three year longitudinal study conducted with grades 4-6 students, Jiang and Wong (2018) aimed to examine the effects of plugged and unplugged activities on the development of problem solving skills associated with students' computational thinking skill and their motivation for coding. The researchers suggested that students learned programming concepts well, gained problem solving skills, and plugged and unplugged activities both provided intrinsic motivation.

In summary, the findings of the studies indicate that computer science unplugged activities are effective method in teaching computer science and support the improvement of computational thinking skill such as problem solving and algorithmic thinking, which are sought in individuals in the 21st century. At this point, it is expected that this study, which sets an example for the design and implementation of computer science unplugged activities, will help instructors who are the practitioners, in planning lessons.

3. Method

Case study is one of the qualitative research approaches used to examine one or more situations in depth (Creswell 2009). The focus of the case study is to define an event as it exists, to examine the situation in its real environment and to describe it in detail (Leymun et al. 2017). Although there are different classifications regarding the case study in the literature, Yin (2014) divided the single and multiple case studies into holistic and embedded designs in line with the analysis units. According to this classification, holistic single case study design is used where a single unit of analysis is involved and a single case is analyzed. For this reason, since the purpose of the activity is to examine the process of writing algorithms using step by step instructions of the students in the study group via the *CityMap*, the study was designed with a holistic single case study.

3.1. Participants

In line with the purpose of the study, convenient sampling method, which is one of the sampling methods for qualitative research, was used because of easily accessible and applicable (Patton 2014). The study group consists of 15 (9 girls, 6 boys) sixth grade students who attended the "Coding and Robotics Education" course.

3.2. The Role of The Researcher

One of the researchers, the first author, is also the instructor of the course. In the first two weeks of the course, different activities were carried out for problem solving and writing algorithms step by step. And then in the third week, the developed *CityMap* activity was applied. During the implementation, the researcher only provided guidance on how to do the activity and and did not make any intervention. In addition, the researcher made an observation about the students' behavior.

3.3. Development of The CityMap Activity

CityMap was developed as a result of reviewing various projects and literature on computer science unplugged activities and taking as basis. Especially, the activities were written by Bell, Witten and Fellows (1998), on the csunplugged.org, in the Information Technologies and Software Curriculum (MEB 2018) and the model was developed by Nishida et al. (2009) were examined. Based on these examinations, an activity table for *CityMap* was created (Table 1).

Table 1. The Computer Science Unplugged Activity *CityMap*

Activity name:	CityMap
Aim of activity:	Writing algorithms of going from one place to another using step by step instructions (go x square, turn right, turn left).
Rules of activity:	The rule is, using the correct instructions, to reach the destination with the shortest way (minimum number of squares) and the least number of steps (minimum number of code lines).
Learning outcomes:	• Solves a problem using the given instructions. • Writes algorithms using step by step instructions.
Age/Grade:	5th and 6th grade (can be adapted for all age groups according to the complexity of the given map and tasks)
Individual / Group Work:	Both individual and group work
Required preliminary skills:	Basic literacy skill Knowing the basic directions

Required materials:	A map and a worksheet
Evaluation tools:	Preparation of answer key Determination of scoring criteria Designing badges* (*Optional: can be used as a performance indicator at the end of the evaluation)

The aim of this activity is to write algorithms of going from one place to another using step by step instructions (go x square, turn right, turn left). The rule is, by using the correct instructions, to reach the destination with *the shortest way (minimum number of squares)* and *the least number of steps (minimum number of code lines).* The activity was constructed based on a scenario in order to create sense of story (Bell and Vahrenhold 2018; Kelleher et al. 2007). In order to establish the relationship between the activity and daily life, daily routines are taken as a basis according to the scenario. Five different jobs character, namely *teacher, police, doctor, taxi driver and fireman*, were determined in the scenario. Later, three different tasks to be done in one day routine were created for the jobs characters. First task: to go from home to workplace, Second task: to go from workplace to the task place, Third task: to return home from the task place.

For example; as shown in Table 2, according to the scenario, the *teacher* lives in no 17, and his/her workplace is the school. The *teacher*'s task is to go to the museum with the students. He/she will return home after the museum. In this case, the students who chose the *teacher* character were requested to write algorithms using step by step instructions, the way from home to school for the first task, from school to museum for the second task, and from the museum to home for the third task.

Table 2. The Scenario of *CityMap*

Jobs Character	Home	Workplace	Task place	Task (in the scenario)
Teacher (T)	No 17	School	Museum	*"Today is the day of sightseeing .. Let's go to the museum with the students"*
Doctor (D)	No 13	Hospital	Beach	*"There is a case of injury at the beach .. Emergency!"*
Police (P)	No 23	Polis Station	Market-1	*"Oops!! There is a thief in the market.. "*
Taxi Driver (TD)	No 22	Taxi Station	No 11	*"You are expected from no 11."*
Fireman (F)	No 28	Fire Department	No 16	*"There was a fire at no16 .. Run and run .."*

3.4. Implementation Process of The CityMap Activity

Before the implementation of the *CityMap* activity, the researcher wrote the houses and the tasks determined for five jobs character and pasted them on the map. During the implementation process, the students were asked to form groups of five and the students in each group to choose one of the five jobs characters. Since the study group consists of 15 students, three groups was formed. Later, materials were distributed as a map to each group and a worksheet to each student, the rules and what to do in the activity was explained. Everyone wrote algorithms using step by step instructions for three tasks on the worksheet individually in line with the jobs characters they chose (Figure 1). At the end of the lesson, the worksheets were collected to be checked.

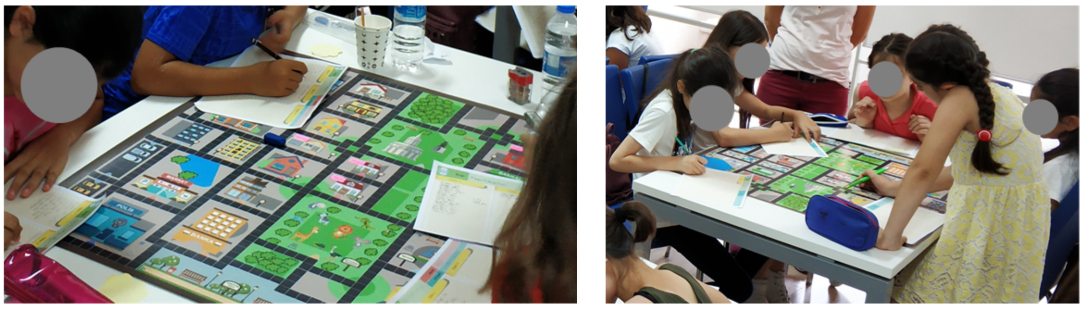
Figure 1. Implementation process of *CityMap*

3.5. Data Collection and Data Analysis

In order to collect data, a map and a worksheet designed by the researchers were used for the *CityMap* activity in the study (Figure 2). The ways on the map were designed square by square, like a grid. In accordance with the scenario, buildings and houses such as hospitals, schools, parks, markets, which may be in a city, were placed around the ways on the map. In order to avoid confusion, door numbers were determined for the houses. On the other hand, the worksheet is divided into three separate sections for writing the determined tasks and each section is divided into lines. Small square spaces were added to the edge of each line where the steps are written. Thus, it was easier for the students to see how many steps (code lines) they did in the task. In order to determine how many squares the task was completed, it was asked to write the total number of squares at the end of the worksheet.

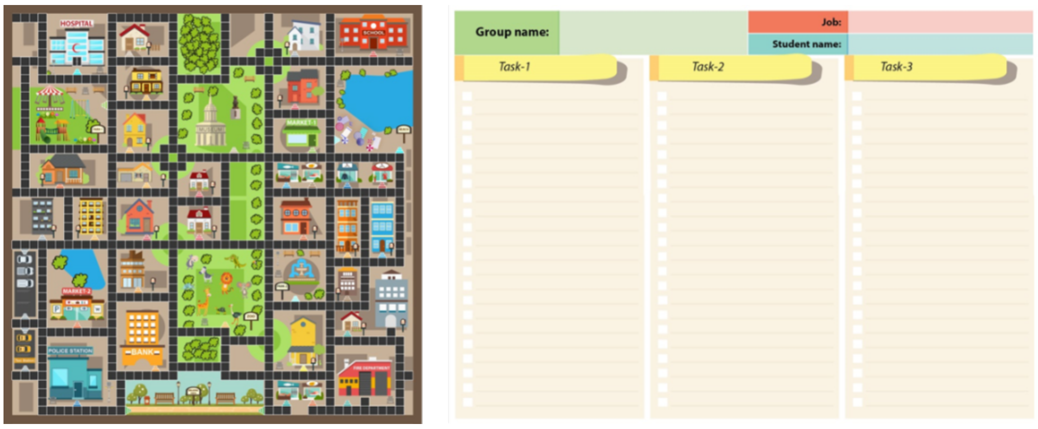

Figure 2. Map and worksheet designed for *CityMap*

The data collected via the worksheet was checked with the answer key prepared by the researchers. The answer key was created separately for the three tasks determined for each jobs character. According to the map, there are many alternative ways to go from one place to another in order to fulfill the determined tasks. But there is only one way that satisfies the activity rules. As an example, solutions are given in Table 3 regarding the three tasks of a *teacher*.

Table 3. An Example of Answer Key

Task1	Task2	Task3
Home → School	School → Museum	Museum → Home
1. Get out of the home (start)	1. Get out of the school (start)	1. Get out of the museum (start)
2. Move forward 1 square	2. Move forward 1 square	2. Move forward 1 square
3. Turn left	3. Turn right	3. Turn right
4. Move forward 3 square	4. Move forward 5 square	4. Move forward 6 square
5. Turn left	5. Turn left	5. Turn left
6. Move forward 3 square	6. Move forward 11 square	6. Move forward 1 square
7. Turn right	7. Turn right	7. Turn left
8. Move forward 1 square	8. Move forward 11 square	8. Move forward 1 square
9. Turn left	9. Turn right	9. Turn left
10. Move forward 1 square	10. Get in the museum (finish)	10. Move forward 3 square
11. Turn right		11. Turn rigt
12. Move forward 17 square		12. Move forward 3 square
13. Turn left		13. Turn right
14. Move forward 11 square		14. Get in the home (finish)
15. Turn right		
16. Move forward 5 square		
17. Turn left		
18. Get in the school (finish)		
Number of squares: 42	Number of squares: 28	Number of squares: 15
Number of code lines: 18	Number of code lines: 10	Number of code lines: 14

While evaluating, it was first checked that the students wrote algorithms using step by step instructions correctly. If the steps for each task were written correctly and completely, 10 points were given. If it is empty or mostly incomplete, no points were given. The errors were grouped under 5 categories: *1- no writing start/finish step, 2- mixing direction (turn right/left step), 3- lacking of one or a few steps, 4- counting number of squares incorrectly, 5- using incorrect instructions.* If there are errors in the wiriting algorithms using step by step instructions, 2 points were deducted from 10 points for each category according to these five error categories. Then, *the shortest way* (minumum number of squares) and *the least number of steps* (minimum number of code lines), which is the rule of the actitivty, was controlled. In this case, students who write algorithms using step by step instructions in accordance with both rules were given 4 points, 2 points for each rule. If it was written in accordance with only one rule, 2 points were given.

In case studies, it is recommended to use more than one data collection tool in order to obtain rich data about the situation under study (Patton 2014; Yıldırım and Şimşek 2013). In this study, the observation technique was used in addition to the worksheet. The researcher, who was the course instructor, made observations in the study environment as a *participant observer* and kept unstructured field notes about students' behavior. The data obtained from the field notes were used to support the findings.

3.6. Gamification and Assessment Framework

Gamification can be defined as the use of game philosophy, game components and game design techniques out of the context of game theory to increase motivation and encourage problem solving (Werbach and Hunter 2012; Deterding et al. 2011; Zichermann and Cunningham 2011). Although teaching via games or gamification basically has the same structural factors, while teaching factors are integrated into games in teaching via games, game components are integrated into existing teaching factors in gamification (Çağlar and Kocadere 2015). In other words, there is no game in the gamification of a teaching environment. Wherein the subject to integrate game components such as star, badge, level, leaderboard with the teaching environment. Thus, gamification, which is based on games, yields similar to the effects of games (Huotari and Hamari 2012).

In the researches were emphasized that the game components, which are the source of external motivation, have positive effects such as providing fun in the learning environment (De-Marcos et al. 2014; Kocadere and Çağlar 2015), increasing motivation (Sillaots 2014; Kocadere and Çağlar 2015, Su and Cahang 2015), ensuring engagement with the environment and increasing academic success (Ibanez et al. 2014; Su and Cahang 2015; Hanus and Fox, 2015). Therefore, points, badges and leaderboard were used in the evaluation of the *CityMap* activity. For individual evaluation; the answers of all students were checked and after the scoring was completed, the students who chose each jobs character were compared among themselves. For example; the scores of the students coded as T-1, T-2, T-3 for *teacher* were listed and the student with the highest score was given the *teacher badge* (Figure 3). If the scores were equal, the total number of errors was checked. For group evaluation; the teams that got the most badges to their group were determined and written on the board as the leaders of the week.

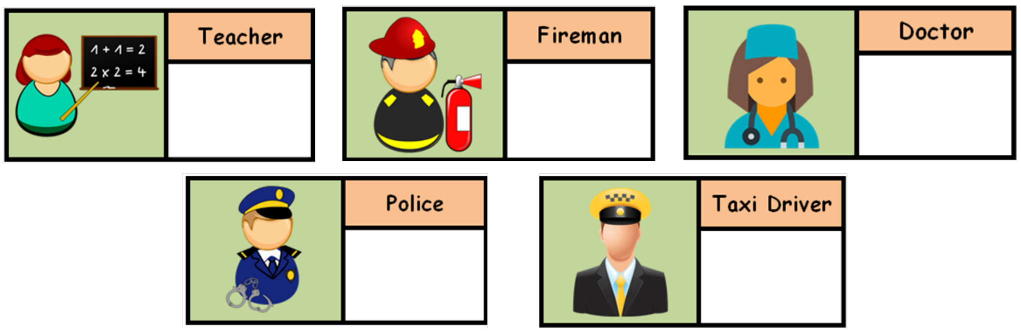

Figure 3. Badges designed for the evaluation of the *CityMap*

4. Results

While giving information about the implementation process of the activity at the beginning of the lesson, the researcher stated that this activity would be like a competition and that they could win badges individually and be enrolled on the leaderboard as a group at the end of the evaluation. The researcher observed that almost all students were excited and motivated to do the activity. The worksheets filled in by the students were checked through the prepared answer key. When Table 4 is examined, it is seen that 6 students in the first task, 4 students in the second task, and 6 students in the third task wrote algorithms using step by step instructions correctly and completely. According to this finding, although the number of students who wrote algorithms using step by step instructions without any error seemed low, when the number of errors was examined, it was found that some students made almost no errors (f = 1) in some tasks. For instance; while the student with code D-1 did not make any error in the first and second tasks, in the third task, he made only one error in the category of *counting number of squares incorrectly*. As a result of the observations, it is thought that this situation is caused by the excitement of the students about the activity.

Written algorithms using step by step instructions by one student in the first task, three students in the second task, and two students in the third task were determined that most of them were incomplete or the activity was empty. According to Table 4, the student with the code P-3 completed the first and third tasks without error. When his/her worksheet was examined in detail, it was seen that he/she wrote only a few lines of the second task but did not complete it. On the other side, the student with the code D-2 did not complete the second and third tasks, while the student with the code F-3 student did not complete all three tasks. It was observed that these students were uninterested towards the lesson.

When the errors made regarding the writing algorithms using step by step instructions were examined, it was found that the most errors were made in the categories of *lacking of one or a few steps* (f = 31) and *counting the number of squares incorrectly* (f = 22). It is thought that this situation may be due to the long distance between the two places given in the relevant task or many alternative routes. Because it was observed that the students try too many times on the map to find the best solution. This situation may have led to confusion when writing algorithms using step by step instructions. Another situation is that one map was given for a group. Although the map was large, it was observed that sometimes, it was difficult for students to work on a map with five students.

Other errors related to writing algorithms using step by step instructions are *using incorrect instructions* (f = 17), *mixing direction* (f = 15) and *no writing start/finish step* (f = 2). However, when the number of errors is examined, it is seen that almost all of the errors in *using incorrect instructions* belong to the student with the code of T-3 (Table 4). When the worksheet of this student was examined in detail, it was determined that the student used incorrect instructions such as turn 5 square top, go up, not the given instructions. It can be said that the student made these errors because of misunderstanding about the use of the instructions. Similarly, it is seen that the majority of the number of errors in the *mixing direction* category belongs to the student with the code D-3. Here, it is the situation that the student confuses his/her own direction by the direction of relative to the map.

When the completion of activity according to the rule of *the shortest way* is examined, the number of students who finds correct solution is 10 in the first task, 8 in the second task, and 12 in the third task. The number of students who completes their tasks with *the least number of steps* is 9 for the first task, 6 for the second task, and 11 for the third task. These findings can be interpreted as students took into account the rules and focused on the activity to get points.

In addition to this, both individual and group evaluations were made by using the gamification. For individual evaluation; each jobs character group was evaluated within itself. The students who wrote the algorithms using step by step instructions correctly by following the rule of the shortest way and the least number of steps were ranked according to their total scores. In references to Table 4, the *teacher badge* is T-1, the *doctor badge* is D-1, the *police badge* is P-2, the *taxi driver badge* is TD-3 and the *fireman badge* was earned by F-2. During the announcement of the students who received the badges, it was observed that the students were curious and excited. For group evaluation; the group / groups that earned the most badges were enrolled on the leaderboard. In this case, the first and second group earned two badges each, and the third group just one. The two groups, who were the leaders of the week by getting equal badges, were observed that have shared their pleasure by hugging each other.

Table 4. Scoring of Writing Algorithms Using Step By Step Instructions for The *CityMap*

	Task1									Task2									Task3										
		Errors					Rules				Errors					Rules				Errors					Rules				
	writing algorithms using step by step instructions	no writing start/finish step	mixing direction (turn right/left step)	lacking of one or a few steps	counting number of squares incorrectly	using incorrect instructions	the shortest way	the least number of steps	Score	writing algorithms using step by step instructions	no writing start/finish step	mixing direction (turn right/left step)	lacking of one or a few steps	counting number of squares incorrectly	using incorrect instructions	the shortest way	the least number of steps	Score	writing algorithms using step by step instructions	no writing start/finish step	mixing direction (turn right/left step)	lacking of one or a few steps	counting number of squares incorrectly	using incorrect instructions	the shortest way	the least number of steps	Score	Total number of errors	Total score
	+10p	-2p	-2p	-2p	-2p	-2p	+2p	+2p		+10p	-2p	-2p	-2p	-2p	-2p	+2p	+2p		+10p	-2p	-2p	-2p	-2p	-2p	+2p	+2p			
T-1	√						-	-	10	√						+	+	14	√						+	+	14	0	**38**
T-2	/				1		+	-	10	/			2	3		-	-	6	/		1				+	+	12	7	28
T-3	/					7	-	-	8	/					5	-	-	8	/					4	-	-	8	16	24
D-1	√						+	+	14	√						+	+	14	/				1		+	+	12	1	**40**
D-2	/			5	3		-	-	6	X						-	-	0	X						-	-	0	8	6
D-3	/		5	3	1		+	+	8	/		3	3			+	+	10	/		3	1			+	+	10	19	28
P-1	/			1			+	+	12	/		1	3	1		+	-	6	√						+	+	14	6	32
P-2	√						-	-	10	√						+	-	12	√						+	+	14	0	**36**
P-3	√						+	+	14	X						-	-	0	√						+	+	14	0	28
TD-1	√						+	+	14	/				1		-	-	8	/	1		1	2		+	-	6	5	28
TD-2	√						+	+	14	/			3	1		+	+	10	/			3			+	+	12	7	36
TD-3	/			1			+	+	12	/			1	2		+	+	10	√						+	+	14	4	**36**
F-1	/		1		2		+	+	10	/		1	3	2		-	-	4	√						+	+	14	9	28
F-2	/	1		1	1	1	+	+	6	√						+	+	14	/				1		+	+	12	5	**32**
F-3	X						-	-	0	X						-	-	0	X						-	-	0	X	0
According to categories total number of errors		1	6	11	8	8					0	5	15	10	5					1	4	5	4	4					

T: Teacher, **D:** Doctor, **P:** Police, **TD:** Taxi Driver, **F:** Fireman

-1: Group1, **-2:** Group2, **-3:** Group3

√: Correct and complete, **/:** Some little errors, **X:** Empty or mostly incomplete

5. Conclusion and Discussion

The rapid advancement of science and technology, changing needs and expectations have started to differentiate the characteristics and competencies sought in individuals, and recently some skills such as problem solving, critical thinking, and creativity have come to the fore. These characteristics, called 21st century skills, are considered important for social innovation and economic growth, as well as they reveal individual differences. In particular, many smart systems called industry 4.0 enable data exchanges and production technologies to be automated and produce higher quality, cheaper and faster. The creation of these automatic systems is possible with artificial intelligence, robotic systems and various software. Thus, programming has become the new communication language of the digital world. So, unlike other skills, this has brought to the agenda the computational thinking skill, the most important feature of which is the opportunity for people to cooperate with computers (Demir and Seferoğlu 2017). In this context, computational thinking has taken its place in the development plans of many countries in order that learners can have skills that can keep up with the times (Bocconi et al. 2016) and started to become the focus of the studies in the field of education.

Computational thinking is a basic literacy skill for everyone, not only for computer scientists (Wing 2006). The most effective way to improve this skill is seen as programming, that is, computer science (Lye and Koh 2014; Wing 2011). Because programming is the process of developing an algorithm that will serve to solve a problem and its implementation (Akçay and Çoklar 2016), and many findings in the literature supported that programming improves students' cognitive skills (Chao 2016; Gülbahar and Kalelioğlu 2014; Fessakis et al. 2013). However, when it comes to programming teaching, which is perceived as a complex and difficult process, it is indicated that unplugged activities are an alternative teaching method for computer science (Thies and Vahrenhold 2016). From this point of view, CityMap was developed and thus, it is aimed to present an example of the implementation and evaluation process of a computer science unplugged activity through this activity. In line with this purpose, *CityMap* activity applied to 15 sixth grade students and the process of writing algorithms using step by step instructions of students, which is considered as computational thinking skill, was examined.

When the process of writing algorithms using step by step of students was examined; it was determined that most errors were made in the categories of *lacking of one or a few steps* and *counting the number of squares incorrectly*. It is thought that this situation may be due to the long distance between the two places given in the relevant task or many alternative routes or the use of a map by five students. Other common errors are *using incorrect instructions* and *mixing direction*. Almost all of the errors in *using incorrect instructions* belonged to one student. This situation indicates that the student does not understand the instructions. It can be said that in future studies, it may be beneficial to present the instructions to the students in written form, such as supported by visual examples, as well as verbally expressing them.

Similarly, it was determined that the majority of the error of *mixing direction* belonged to another student. When the student's answers were examined in detail, it was understood that the student confused his/her own direction by the direction of relative to the map. In a similar implementation, it is thought that giving a map to each student will be a solution to this problem. Another result is that most of the students have reached the solution with the shortest way and the least number of steps, as required by the rules of the activity. This indicates that the students can understand the rules and use the instructions correctly.

In addition to these, while giving information about the implementation process of the activity at the beginning of the lesson, it was observed that almost all students were excited when it was stated that they could win badges individually and be enrolled on the leaderboard as a group at the end of the evaluation. This finding indicates that the game components help to make the lesson more fun. Hence in the literature, it is asserted that the inclusion of gamification and entertainment elements in unplugged activities is an effective source of motivation (Bell and Vahrenhold 2018; Tsarava et al. 2017; Voigt et al. 2010). As a conclusion, in this study, it was presented an example of the implementation and evaluation of a computer science unplugged activity and some suggestions were made for the implementation of this activity. In future studies, the effects of these and similar activities on students' computer science achievements or computational thinking skill can be examined.

Acknowledgements

A part of this study was presented at 7th International Instructional Technologies and Teacher Education Symposium (ITTES), Antalya, Turkey, 30 October-1 December 2019.

Funding information

This study was supported by The Scientific and Technological Research Council of Turkey (Project No: TÜBİTAK - 118R034).

References

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

Akçay, A., & Çoklar, A. N. (2016). Bilişsel becerilerin gelişimine yönelik bir öneri: Programlama eğitimi. In A. İşman, H. F. Odabaşı and B. Akkoyunlu (Eds.), *Eğitim Teknolojieri Okumaları 2016* (pp. 121-139). Ankara, TOJET.

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

Bell, T., & Vahrenhold, J. (2018). CS unplugged - How is it used, and does it work?. In H. J. Böckenhauer, D. Komm, and U. W. (Eds.) *Adventures Between Lower Bounds and Higher Altitudes. Lecture Notes in Computer Science* (pp. 497-521). Springer, Cham. https://doi.org/10.1007/978-3-319-98355-4_29

Bell, T. C., Witten, I. H., & Fellows, M. (1998). Computer Science Unplugged: Off-line activities and games for all ages. Retrieved April 25, 2020 from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.114.7908&rep=rep1&type=pdf

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education - Implications for policy and practice*. Seville: Join Research Center (European Commission). https://doi.org/10.2791/792158. Retrieved May 15, 2020 from: https://publications.jrc.ec.europa.eu/repository/bitstream/JRC104188/jrc104188_computhinkreport.pdf

Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., & Barone, D. (2017). Development of computational thinking skills through unplugged activities in primary school. In Proceedings of the 12th Workshop on Primary and Secondary Computing Education (pp. 65-72).

Bundy, A. (2007). *Computational thinking is pervasive.* Retrieved May 23, 2020 from: https://www.inf.ed.ac.uk/publications/online/1245.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. https://doi.org/10.1016/j.compedu.2016.01.010

Creswel, J. W. (2009). *Research design: Qualitative, quantitative, and mixed methods approaches.* Los Angeles, University of Nebraska-Lincoln.

CS Unplugged (2019). *About.* Retrieved October 25, 2019 from: https://csunplugged.org/en/about/

CSTA & ISTE (2011). *Operational definition of computational thinking for K-12 education.* Retrieved November 30, 2017 from: http://www.iste.org/docs/ct-documents/computational-thinking-operational-definitionflyer.pdf

Curzon, P. (2014). *Unplugged computational thinking for fun.* Retrieved April 25, 2020 from: https://publishup.uni-potsdam.de/opus4-ubp/frontdoor/deliver/index/docId/8257/file/cid07_S15-27.pdf

Çağlar, Ş., & Kocadere, S. A. (2015). Çevrimiçi öğrenme ortamlarında oyunlaştırma [Gamification in online learning environments]. *Journal of Educational Sciences & Practices, 14*(27), 83-102.

De-Marcos, L., Domínguez, A., Saenz-de-Navarrete, J., & Pagés, C. (2014). An empirical study comparing gamification and social networking on e-learning. *Computers & Education*, 75, 82-91. https://doi.org/10.1016/j.compedu.2014.01.012

Demir, Ö. & Seferoğlu, S. S. (2017). Yeni kavramlar, farklı kullanımlar: Bilgi-işlemsel düşünmeyle ilgili bir değerlendirme. In H. F. Odabaşı, B. Akkoyunlu, and A. İşman (Eds.). *Eğitim Teknolojileri Okumaları 2017* (pp. 468-483). Ankara, TOJET.

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: "Defining gamification". In Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments (pp. 9-15).

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5-6 years old kindergarten children in a computer programming evironment: A case study. *Computers & Education, 63*, 87-97. https://doi.org/10.1016/j.compedu.2012.11.016

Grover, S. (2018). *The 5th 'C' of 21st century skills? Try computational thinking (not coding).* Retrieved October 25, 2019 from: https://www.edsurge.com/news/2018-02-25-the-5th-c-of-21st-century-skills-try-computational-thinking-not-coding

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

Hanus, M. D., & Fox, J. (2015). Assessing the effects of gamification in the classroom: A longitudinal study on intrinsic motivation, social comparison, satisfaction, effort, and academic performance. *Computers & Education, 80,* 152-161. https://doi.org/10.1016/j.compedu.2014.08.019

Hermans, F., & Aivaloglou, E. (2017). To Scratch or not to Scratch? A controlled experiment comparing plugged first and unplugged first programming lessons. In Proceedings of the 12th workshop in primary and secondary computing education (pp. 49–56).

Horizon Report (2017). *The NMC/CoSN Horizon Report: 2017 K–12 Edition.* Retrieved November 13, 2019 from: https://library.educause.edu/~/media/files/library/2017/11/2017hrk12EN.pdf

Huotari, K., & Hamari, J. (2012). Defining gamification: a service marketing perspective. In Proceeding of the 16th International Academic MindTrek Conference (pp. 17-22).

Ibanez, M. B., Di-Serio, A., & Delgado-Kloos, C. (2014). Gamification for engaging computer science students in learning activities: A case study. *IEEE Transactions on Learning Technologies, 7*(3), 291-301.

ISTE (2015). *CT leadership toolkit.* Retrieved October 30, 2017 from: http://www.iste.org/docs/ct-documents/ct-leadershipt-toolkit.pdf?sfvrsn=4

Jagušt, T., Krzic, A. S., Gledec, G., Grgić, M., & Bojic, I. (2018). Exploring different unplugged game-like activities for teaching computational thinking. In 2018 IEEE Frontiers in Education Conference (FIE) (pp. 1-5). IEEE.

Jiang, S., & Wong, G. K. (2018). Are children more motivated with plugged or unplugged approach to computational thinking?. In Proceedings of the 49th ACM technical symposium on computer science education (pp. 1094-1094).

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

Kelleher, C., Pausch, R., Pausch, R., & Kiesler, S. (2007). Storytelling alice motivates middle school girls to learn computer programming. In Proceedings of the SIGCHI Conference On Human Factors in Computing Systems (pp. 1455–1464). ACM.

Kocadere, S. A., & Çağlar, Ş. (2015). The design and implementation of a gamified assessment. *Journal of e-Learning and Knowledge Society, 11*(3), 85-99.

Kotini, I., & Tzelepi, S. (2015). A gamification-based framework for developing learning activities of computational thinking. In *Gamification in Education and Business* (pp. 219-252). Springer, Cham.

Leifheit, L., Jabs, J., Ninaus, M., Moeller, K., & Ostermann, K. (2018). Programming unplugged: An evaluation of game-based methods for teaching computational thinking in primary school. In 12th European Conference on Game-Based Learning (pp. 344)

Leymun, Ş. O., Odabaşı, F., & Yurdakul, I. K. (2017). Eğitim ortamlarında durum çalışmasının önemi. *Eğitimde Nitel Araştırmalar Dergisi, 5*(3), 367-385.

Li, C., Dong, Z., Untch, R. H., & Chasteen, M. (2013). Engaging computer science students through gamification in an online social network based collaborative learning environment. *International Journal of Information and Education Technology, 3*(1), 72-77.

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

MEB (2018). *Bilişim Teknolojileri ve Yazılım Dersi Öğretim Programı.* Retrieved October 25, 2019 from: http://mufredat.meb.gov.tr/Dosyalar/2018124103559587-Bili%C5%9Fim%20Teknolojileri%20ve%20Yaz%C4%B1l%C4%B1m%205-6.%20S%C4%B1n%C4%B1flar.pdf

Nishida, T., Kanemune, S., Idosaka, Y., Namiki, M., Bell, T., & Kuno, Y. (2009). A CS unplugged design pattern. *ACM SIGCSE Bulletin, 41*(1), 231-235. https://doi.org/10.1145/1508865.1508951

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas.* Basic Books, New York.

Patton, M. Q. (2014). *Nitel araştırma ve değerlendirme yöntemleri.* [Qualitative research and evaluation methods].(Trans. Eds. M. Bütün and S. B. Demir). Pegem Akademi, Ankara.

Rodriguez, B. R. (2015). Assessing computational thinking in computer science unplugged activities. *Doctoral dissertation,* Colorado School of Mines, Arthur Lakes Library.

Sillaots, M. (2014). Achieving flow through gamification: a study on re-designing research methods courses. In European Conference on Games Based Learning (vol. 2, pp. 538-545).

Su, C. H., & Cheng, C. H. (2015). A mobile gamification learning system for improving the learning motivation and achievements. *Journal of Computer Assisted Learning, 31*(3), 268-286.

Swacha, J., & Baszuro, P. (2013). Gamification-based e-learning platform for computer programming education. In X world conference on computers in education (pp. 122-130).

Şahiner, (2017). Komputasyonel düşünme kavramı ile ilgili 2006-2016 yılları arasındaki bilimsel yayınların incelenmesi: Doküman analizi çalışması. *Yayınlanmamış Yüksek Lisans Tezi,* Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü.

Şendurur, P. (2019). Investigation of pre-service computer science Teachers' CS-unplugged design practices. *Education and Information Technologies, 24*(6), 3823-3840. https://doi.org/10.1007/s10639-019-09964-6

Şendurur, P. (2018). Programlama öğretiminde bilgisayarsız etkinlikler. In Y. Gülbahar and H. Karal (Eds.), *Kuramdan Uygulamaya Programlama Öğretimi* (pp. 189-235). Pegem Akademi, Ankara.

Tabesh, Y. (2017). Computational thinking: A 21st century skill. *Olympiads in Informatics, 11*, 65-70. https://doi.org/10.15388/ioi.2017.special.10

Taub, R., Ben-Ari, M., & Armoni, M. (2009). The effect of CS unplugged on middle-school students' views of CS. *ACM SIGCSE Bulletin*, *41*(3), 99-103.

Thies, R., & Vahrenhold, J. (2016). Back to school: Computer science unplugged in the wild. In Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education (pp. 118-123).

Threekunprapa, A., & Yasri, P. (2020). Unplugged coding using flowblocks for promoting computational thinking and programming among secondary school students. *International Journal of Instruction, 13*(3), 207-222.

Tsarava, K., Moeller, K., Pinkwart, N., Butz, M., Trautwein, U., & Ninaus, M. (2017). Training computational thinking: Game-based unplugged and plugged-in activities in primary school. In European Conference on Games Based Learning (pp. 687-695).

Voigt, J., Bell, T., & Aspvall, B. (2010). Competition-style programming problems for computer science unplugged activities. In: Verdu, E., Lorenzo, R., Revilla, M., Regueras, L. (Eds.), *A New Learning Paradigm: Competition Supported by Technology* (pp. 207–234). CEDETEL, Boecillo.

Yin, R.K. (2014). *Case study methods: design and methods* (5th ed.). Thousand Oaks: Sage Pbc.

Werbach, K., & Hunter, D. (2012). *For the win: How game thinking can revolutionize your business*. Wharton Digital Press.

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

Wing, J. M. (2011). Research notebook: Computational thinking - What and why. *The link magazine,* 6. Retrieved January 18, 2020 from: https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

Wohl, B., Porter, B., & Clinch, S. (2015). Teaching computer science to 5-7 year-olds: An initial study with Scratch, Cubelets and unplugged computing. In Proceedings of the Workshop in Primary and Secondary Computing Education (pp. 55-60).

Yıldırım, A. & Şimşek, H. (2013). *Sosyal Bilimlerde Nitel Araştırma Yöntemleri* (9.baskı). Ankara, Seçkin Yayıncılık.

Zichermann, G., & Cunningham, C. (2011). *Gamification by design: Implementing game mechanics in web and mobile apps.* O'Reilly Media, Inc..