The Impact of Teaching Scratch on Growth of Computational Thinking of Seventh Grade Students

Abolfazl Rafiepour [1,2*], Mohammad Radmehr [2]

[1] *Faculty of Education and Arts (FLU), Nord University, Levanger, Trondelag, Norway*

[2] *Department of Mathematics Education, Faculty of Mathematics and Computer- Shahid Bahonar University of Kerman, Kerman, Iran*

*Corresponding author email *Abolfazl.rafiepour@nord.no* and *Rafiepour@uk.ac.ir*

Abstract

In recent years many educational systems around the world start to integrate some sort of computational thinking (e.g. programming and coding) in their school curriculum. In Iran educational system, it can be identify vein and root of computational thinking in both school, curriculum and after school program. In current study we investigate the potential benefits and possibility of using Scratch as blocky programming in enhancing middle school students capabilities in computational thinking. So, the main purpose of the study was to measure the effect of programming with Scratch on the level of computational thinking of 7th grade students. In this study 50 seventh grade students from two lower secondary school) were participated. Results of analysing students' scores upon Dr. Scratch show that the teaching blocky programming could improve seventh grade students capability in CT.

Keywords: Scratch, Blocky Programming, Computational Thinking, Middle School Students, School Curriculum,

INTRODUCTION

One of the latest educational perspectives is to introduce computer programming in classrooms from K-12 for developing students' computational thinking which defined by Wing (2006). So that today, governments in many countries around the world have made decision to put programming in their national curriculum. Indeed, human beings in society have become more and more technology-based in the 21st century and must have appropriated knowledge to perform their work in a more efficient manner, in which such a society is heavily integrated with technology (Bocconi et al., 2016). One of the important aspects that everyone has to know is Computational Thinking (CT) as Wing (2006) suggested, "to reading, writing, and arithmetic, we should add CT to every child's analytical ability" (p. 33). Shodiev (2014) defines CT as a way of thinking algorithmically using design trees from computer science as a guiding structural, and sometimes metaphorical, framework. Hoyles and Noss (2015) consider CT as abstraction, algorithmic thinking,

decomposition, and pattern recognition. CT involves concepts (e.g. loops, conditions) and practices (e.g., abstraction and debugging) (Lye & Koh, 2014; Kafai & Burke, 2013). However, CT is not simply programming and rote skill, but it is a conceptualizing and fundamental skill; a way that humans think (Wing, 2006).

CT changes the nature of some contemporary researches in the mathematics domain. For example, we can see the computer-based proof in mathematics (e.g. four-color theorems). For another example, a new domain of research related to mathematics and computation such as Bioinformatics emerged recently. In this regard, the European Mathematical Society (2011) recognized an emerging way of engaging in mathematical research: “Together with theory and experimentation, the third pillar of scientific inquiry of complex systems has emerged in the form of a combination of modelling, simulation, optimization, and visualization” (p. 2). Weintrop et al. (2016) try to address CT as a more sophisticated concept upon literature review and interview experts who use CT in their carrier. They develop a taxonomy of CT skills which has close relation with the third pillar of scientific inquiry. This taxonomy contains four main categories: data practices, modelling and simulation practices, computational problem-solving practices, and systems thinking practices.

Educational system in many countries start to introduce computational thinking in their students at K-12. For example coding and programming integrated into the school curriculum at Japan, Canada, Cyprus, Greece, Italy, Luxembourg, Hungary, UK(England), Slovakia, Portugal, Ireland, France (Balanskat and Engelhardt, 2015). Investigation of the Iranian school curriculum show that computational thinking gradually inter to the some grade textbooks. For example, there is a chapter about building a computer game using Scratch software in the grade nine “work and technology” textbook (Esmaili et al., 2020). Recently, from 2022-2023 school year, we can see some sort of programming and coding in the textbooks at several grade levels. However, there were several after school programs about computational thinking from several years ago, especially in math houses.

In the current study we focused on Scratch as blocky programming and use it at a lower secondary school to examine this research question. Does the working with scratch as a tools for blocky programming could affect the level of computational thinking of seventh grade students?

LITERATURE REVIEW

One of the pioneer in using programming in education was Symour Papert who created the famous and well-known programming language named Logo. In Papert's (1991) constructionism perspective, learning means building relationships between old and new knowledge in interactions with others while building collective artifacts. Papert mentions in his writings that one day he was passing through an art class at the university and he saw how the students were trying to make different shapes and forms by cutting of a bar of soap and from it. From that moment, Papert started to think about how we can use such experiments in math classroom. Indeed, from that time the idea of Logo software development began. Papert and Harrell (1991) have described the concept of constructionism, that is, learning by making. Papert (1980) briefly described the early evolution of the idea of laboratory microworld such as turtle geometry, which was a mathematical land where certain kinds of thinking occurred on that. Mathematics could develop with particular ease in these "small lab worlds," each with its own set of assumptions and constraints. In doing so, students learn to transfer habits of discovery from their personal lives to the formal domain of scientific theory building.

Recently the ideas of Papert to use programming as microworld alive again in a new soul which is blocky programming such as Scratch. In this type of programming, students in all age and grade level could engage in programming project and construct their own thinking and learn mathematics and other subjects which are important for 21st citizen. Martin's (2016) case study, specifically investigating the effectiveness of the Scratch programming language for exploring computational thinking concepts and methods in sixth grade students, provided promising evidence that a well-designed student task or assignment, It can provide all students with the opportunity to develop the ability to use computational thinking concepts, and students who use Scratch are easily able to quickly acquire computational thinking concepts and methods. In Martin (2016) study, many students who worked on a programming project with minimal help from their teacher, gained experiences in the field of computational thinking concepts and methods only by spending four hours training. In another study, Maloney et al. (2008) examined Scratch as a tool for students in the computer Clubhouse organization, by examining 536 Scratch projects and analyse student interviews. They concluded that students' coding abilities and understanding of computational thinking concepts were developed through the using of Scratch.

In current study we use Brennan & Resnick (2012) theoretical framework for studying and assessing computational thinking which used by several studies in literature (e.g. Xue and Liu, 2021). This framework summarized at table 1.

Table1. Framework for Assessing of CT (Brennan & Resnick, 2012)

Key components of CT	**Concept**	**Definition**
Computational thinking **Concepts** that were used in Scratch	Sequences	Sequences is a series of individual steps or instructions that can be executed by the computer.
	Loops	A sequence can be repeated in a loop for a certain number of times or an unlimited.
	Events	What happens on a computer can cause something else to happen. Events are an essential component of interactive media.
	Parallelism	sequences of instructions could happening at the same time.
	Conditionals	Conditionals – the *if* block – means the ability to make decisions based on certain conditions, which will be caused multiple outcomes.
	Operators	Operators enabling the programmer to perform numeric and string manipulations.
	Data	Data involves storing, retrieving, and updating values. Two types of data in Scratch are as below. • variables (which can maintain a single number or string) • lists (which can maintain a collection of numbers or strings)
Computational	Incremental	Coding is not a clean, look like final program that

thinking P**ractices** that were used in Scratch	and Iterative	work and well-design. Instead, it is an adaptive process, in which changed gradually through feedback and new ideas.
	Testing and Debugging	A developed program usually does not work immediately as written, but finding logic errors and debugging is an essential part of coding.
	Reusing and Remixing	Coding upon available programs and developing the work of others projects.
	Abstracting and Modularizing	Abstracting and modularizing means to build large program with using several small programs.

METHOD

The current study had adapted through a quasi-experimental quantitative method. In this regard, 50 seventh grade students from two lower secondary school in city of Rafsanjan in Kerman province (at south-east of Iran) was selected for participating at this study. Data collection occurred in 2020-2021 school year. In the time of current study the pandemic of covid-19 virus exist in all over the world and schools in many countries provide lessons in virtual mood for students. In Iran, also like other part of the world, students participated at virtual class. They use their special platform namely SHAD which designed at national level for supporting teaching and learning process of Iranian students.

All of students announced that they didn't have any sort of knowledge or experience in programming and coding. Almost all of them have the same level in their education progress (according to their scores in math and science). All fifty students participate in four 45-minute online sessions which became familiar with Scratch blocky programming and its capability for designing different program and animation. All of the class time video recorded and uploaded in apparat.com which is a platform for producing and sharing videos in Iran. So, students have access to the content of the class through apparat.com and student educational network (SHAD).

After each training session, the students programmed their creative ideas in their projects and sent them to the researchers. The researchers saved their projects and analysed them with the help of Dr. Scratch program, which examines the students' projects and assigns a computational thinking score. Authors saved and record the score of students in an excel file. In the next section, brief results of the students will be reported.

In literature, there are several tools for assessing computational thinking, for example in Román-González, Moreno-León and Robles (2019) writers try to review evaluation framework in domain of computational thinking and then introduce a comprehensive model intends to assess CT. However, in current study we use Dr. Scratch as a assessment tools of CT which designed upon component of Brennan & Resnick (2012) framework.

Dr. Scratch (Moreno-León et al., 2015) is a free and open-source web application that analyses, in an automated way, projects programmed with Scratch language. The score that Dr. Scratch assigns to a project is based on the degree of development of seven dimensions of CT competence which are abstraction and problem decomposition, logical thinking, synchronization, parallelism,

algorithmic notions of flow control, user interactivity, and data representation. These dimensions are evaluated through analysing project codes and given points from 0 to 3 and results will be reported in a total at ranges from 0 to 21. Upon final results, Dr. Scratch consider three level of progress which are basic, developing and master level. In the basic level, students get 0-7 points at total. In the developing level, students get 7-11 points at all seven dimensions of CT competence. In the master level, students get 14-21 points at total. Furthermore, Dr. Scratch also generates some feedback for students to improve their abilities in programming.

RESULTS

Descriptive analysis of student score upon Dr. Scratch tools show that most of the students capabilities in programming has grown well (see table 2). As shown in table 2. 86% of the students who participate at this study scored by Dr. Scratch at developing and master levels of computational thinking, which show that using blocky programming (in this case Scratch) has a positive effect on the growth of the level computational thinking of seventh grade students.

Table 2. Level of students CT capability

	Level of CT	Number of Students	Percentage of Students
	Basic	7	14 %
	Developing	24	48 %
	Master	19	38 %
Sum		50	100 %

In order to get a general summary of the amount of changes in the computational thinking capabilities of all students, the computational thinking score of all projects that students submitted at each stage was calculated with the help of the Dr. Scratch program. Then, the average computational thinking scores were calculated in each stage, and in total, to show the progress of the students, The broken line graph of the average computational thinking scores in each stage was drawn (See diagram 1).

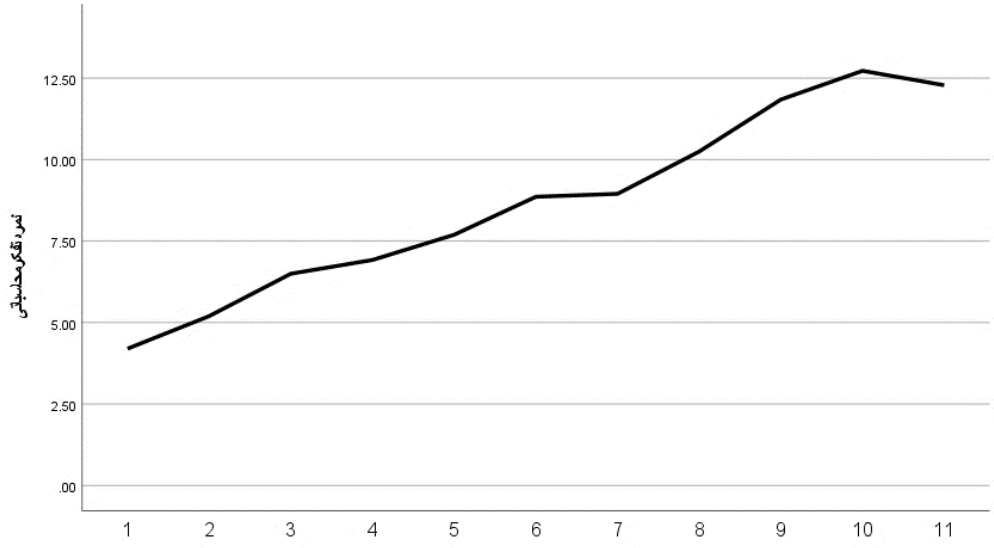

Diagram 1. The broken line diagram of the average changes in the computational thinking score of all students

The ascending graph in diagram 1 generally shows the progress of computational thinking of the participating students. Therefore, according to these evidence we can claim that the teaching blocky programming could improve seventh grade students capability in CT.

DISCUSSION AND CONCLUSION

One of the important things that everyone citizen in 21 century should be known is computational thinking. In this regard, after reading, writing and arithmetic, we need to add computational thinking to every child's analytical ability. CT is not only related to computer science, but it is also located in the heart of many other sciences such as mathematics, science, engineering, etc. A suitable platform for developing students' computational thinking is blocky programming (e.g. Scratch).

In the current research, with a constructionism point of view, we attempt to develop a course for teaching scratch programming and examine gradually develop of students' programming skills and computational thinking abilities. Upon table 2 and figure 1, we can see that the score of students in their CT related project which provide by Dr. Scratch dramatically improved during four 45-minute session training. The results of this research are similar to the other researches from literature review (e.g. Maloney and colleagues (2008), Burke (2012) and Martin (2016))

According to the research paths that was taken in Iran and other part of the world, it can be concluded that the research on Scratch programming education is generally growth more and more and provides many benefits such as improving programming learning, enhancing problem solving skills and create more positive motivation for learning between students.

REFERENCES

Anja Balanskat, Katja Engelhardt. (2015). Computing our future: computer programming and coding-priorities, school curricula and initiatives across europe. European Schoolnet. Retrevied from d3780a64-1081-4488-8549-6033200e3c03 (eun.org) by December 2022.

Brennan, K., & Resnick, M. (2012). New Frameworks for Studying and Assessing the Development of Computational Thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, Vol. 1, Vancouver, 13-17 April 2012, http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education: Implications for policy and practice*. European Union. Retrieved from: http://publications.jrc.ec.europa.eu/repository/bitstream/JRC104188/jrc104188_computhinkreport.pdf

Burke, Q. (2012). The markings of a new pencil: Introducing programming as writing in the middle school classroom. *The Journal of Media Literacy Education*, 4 (2), 121-135.

Esmaili, M., Bahrami, M., Khalilarjmand, M., Sosanabadi, F., Shojaieardakani, M., Sadegh, M., Zeighami, H., Tarzi, M., Farjad, S., Kiani, S., Hodarzi, R., Mostafavi, M., Makinayeri, A., Niazi, M., & Niknejad, B. (2020). *Work and Technology for Grade 9*. Tehran: Ministry of Education of Iran.

Hoyles, C., & Noss, R. (2015). *Revisiting programming to enhance mathematics learning*. Paper presented at the Math + Coding Symposium, Western University.

Kafai, Y. B., & Burke, Q. (2013). Computer programming goes back to school. *Phi Delta Kappan*, *95*(1), 61-65. https://doi.org/10.1177/003172171309500111

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation*. New York, NY: Cambridge University Press. https://doi.org/10.1017/CBO9780511815355

Martin, S. A. (2016). A descriptive and explorative case study of a scratch programming experience involving the creation of a lunar Simulation/Model with grade six learners (Doctoral dissertation, University of Calgary (Canada).

Maloney, J., Peppler, K., Kafai, Y. B., Resnick, M., & Rusk, N. (2008). *Programming by choice: urban youth learning programming with scratch*. ACM SIGCSE Bulletin, 40 (1), 367-371. Retrieved from http://dl.acm.org.ezproxy.lib.ucalgary.ca/citation.cfm?doid=1352322.1352260

Moreno-León, J., Robles, G., & Román-González, M. (2015). Dr. Scratch: Automatic analysis of scratch projects to assess and foster computational thinking. *RED. Revista de Educación a Distancia*, *15*(46), 1–23. Retrieved from http://www.um.es/ead/red/46/moreno_robles.pdf.

Papert, S., & Harel, I. (1991). Constructionism. New York: Ablex Publishing Corporation.

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

Román-González, M., Moreno-León J. and Robles, G. (2019). Combining Assessment Tools for a Comprehensive Evaluation of Computational Thinking Interventions. In Siu-Cheung Kong and Harold Abelson (Eds.) *Computational Thinking Education.* Springer. Pp. 79-98.

Shodiev, H. (2014). Computational thinking and simulations in teaching science and mathematics Paper presented at the *AMMCS-2013 Interdisciplinary Conference Series: Applied Mathematics, Modelling, and Computational Science*.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal for Science Education and Technology*, *25*, 127-147. https://doi.org/10.1007/s10956-015-9581-5

Wing, J. M. (2006). Computational thinking and thinking about computing. *Communications of the ACM*, *49*(3), 33-35. https://doi.org/10.114/1118178.1118215

Xue, C. and Liu, Y. (2021). The Analysis of Research Hotspots and Frontiers of Computational Thinking Based on CiteSpace. *Open Journal of Social Sciences*, **9**, 1-16. doi: 10.4236/jss.2021.99001.