I.J. Modern Education and Computer Science, 2016, 11, 1-7
Published Online November 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2016.11.01

Comparing Students' Scratch Skills with Their Computational Thinking Skills in Terms of Different Variables

Ali OLUK
Kastamonu University, Taşköprü Vocational School, Kastamonu, Turkey
Email: alioluk85@gmail.com

Özgen KORKMAZ
Amasya University, Faculty of Technology, Department of Computer Engineering, Amasya, Turkey
Email: ozgenkorkmaz@gmail.com

***Abstract*—**This study aimed to compare 5th graders' scores obtained from Scratch projects developed in the framework of Information Technologies and Software classes via Dr Scratch web tool with the scores obtained from Computational Thinking Levels Scale and to examine this comparison in terms of different variables. Correlational research model was utilized in the study that 31 students participated in. Students were taught basic programming by using Scratch during a 6-week period. At the end of training, students' programming skills were measured via Dr. Scratch web tool. Computational thinking skills were measured using Computational Thinking Levels Scale which includes 5 factors: creativity, problem solving, algorithmic thinking, collaboration and critical thinking. Data were analyzed for internal reliability to calculate scale reliability. Cronbach Alpha reliability coefficient was found to be 0.809. It was found that scores obtained by students by using any of the measurement tools did not differ according to gender or period of computer use, however, a high level significant relationship was observed between students' programming skills with Scratch and their computational thinking skills.

***Index Terms*—**Scratch, Thinking Skills, Computational Thinking, Dr. Scratch

I. Introduction

It can be argued that advanced technology has changed the skills that the individuals are expected to have such as computational thinking skills. It is believed that computational thinking skills will be necessary for everyone in mid-21st century [1]. It can be claimed computational thinking skills are basic skills that need to be acquired by all and not just by computer experts [2,3] Teaching these skills especially to children is a topic of discussion in educational systems [4]. It is known that many countries have recently attempted to include computational thinking skills in their curriculums [5]. Computational thinking seeks to solve problems, designs systems and comprehends human behaviors with the help of basic computer sciences [1]. Computational thinking is an important skill to educate individuals to identify problems, have a command of problem solving process and solve problems more productively [6]. Computational thinking aims to highlight skills such as creativity, logical thinking and critical thinking to develop problem solving skills with the help of computers [7]. International Society for Technology in Education (ISTE) [8] states that computational thinking cannot be fully defined without sub skills such as creative thinking, algorithmic thinking, critical thinking, cooperative learning and communication skills. Definitions of computational thinking skills point to the importance of training in programming to acquire sub skills [9-11]. Hence, it can be argued that training is significant to teach computational thinking skills [12].

For beginners, programming seems to be a hard task which can only be managed by individuals with high level training [13, 14]. This belief may be related to the fact that programming involves the use of many skills in combination such as generalization, abstraction and critical thinking [15]. In order to facilitate training, it is suggested to use programming tools and methods such as cooperative learning, game based learning, project based learning, simulation and drag and drop [14]. Instead of dealing with the complex structure of traditional programming languages, primary school students' computational thinking skills can be measured via programming tools such as Scratch [12]. There are programming tools like Scratch, Alice, Microsoft Small Basic and Toontalk that emphasize visual aspects such as drag and drop. It is suggested that beginners use these programming tools [12,16] . Since they are free and have easy-to-use interfaces, it is thought that these programming tools provide significant advantages for beginners and primary school students.

Among these vial programming tools, Scratch is distinguished due to its interface, language support and its feature of sharing projects [10]. It is known that Scratch is a program for beginners in programming, logic and algorithmic thinking skills [17 – 19].

One of the most important problems facing educators in using Scratch support in lesson planning is related to tools developed to assess projects prepared with Scratch [20]. Dr. Scratch is a web tool that is used to score projects by entering source files or URL addresses of Scratch projects. Dr Scratch has been developed to analyze projects generated by primary and secondary school students using Scratch programming tool and to provide feedback regarding the development of computational thinking skills [20]. Dr. Scratch offers web based analysis and supports teachers and students in calculating the computational thinking scores of Scratch projects in 7 headings: abstraction, synchronization, analogy/paralellism, information, user interactivity, digital logic and control flow [20]. Dr. Scratch assigns three points to each heading cited above and the highest score that can be obtained is 21. In addition to supporting teachers in the assessment of Scratch projects, Dr. Scratch also assists students who want to develop programming skills on their own [20].

Literature includes many studies focusing on the contribution of Scratch visual programming tool to programming education [19, 21-27]. There is also research that focus on the projects developed with the help of Scratch visual programming tool in terms of computational thinking skills [20,28,29]. The current study differs from other studies since it asesses students’ computational thinking skill scores with two different measurement tools and compares them. This study aimed to compare the scores obtained by 5th graders from Scratch projects developed in the framework of Information Technologies and Software classes via Dr Scratch web tool with the scores obtained from Computational Thinking Levels Scale and to examine this finding in terms of different variables. Research problems are provided below:

Research Questions

a) Do Dr. Scratch scores for student projects differ according to gender?

b) Do Dr. Scratch scores for student projects differ according to period of computer use?

c) Do students’ Computational Thinking Levels scores differ according to gender?

d) Do students’ Computational Thinking Levels scores differ according to period of computer use?

e) Are there significant relationships between Dr. Scratch scores for student projects and their Computational Thinking Levels scores?

II. Methods

A. Study Design

The study utilized correlational research model. Correlation studies examine the relationship between two or more variables without any intervention [30]. This study aimed to investigate the relationship between the achievement scores obtained by Dr. Scratch web tool and skill scores obtained from Computational Thinking Levels Scale.

B. Study Group

5th graders in Kastamonu province participated in the study. A total of 31 students took part in the research.

Table 1. Gender Distribution

	Gender				Total
	Female N	%	Male N	%	N
Total	16	51,6	15	48,4	31

16 of the participating students were females and 15 were males. Students were taught Information Technologies and Software classes with Scratch support for 6 weeks and the projects generated at the end of the class were assessed.

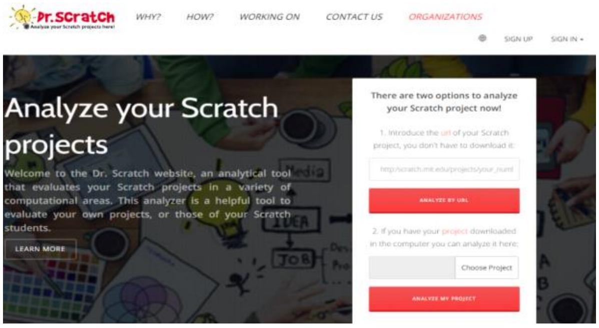
C. Teaching Implementation/Treatment

Study group students were provided with Scratch programming tool training for 6 weeks in the framework of 2-hours a week Information Technologies and Software classes. Scratch programming tool user interface was presented to students in the first week of the study and samples of projects generated with the help of Scratch were displayed. Students were provided with information in the upcoming weeks about the use of Scratch. Code blocks were used to show how drag and drop process worked. They were shown how to register projects and how to upload registered projects to internet. Scratch web site explains how Scratch projects are developed. The use of blocks used in coding in Scratch program and how they are used in projects were presented with examples. Students were provided with opportunities to practice at the end of presentation during each class hour. Students were asked to generate authentic projects during the last two weeks of the process. The teacher acted only as a guide during project development phase.

D. Data Collection Tools

Computational Thinking Levels Scale: Computational Thinking Levels Scale, adapted to secondary school students, was developed by Korkmaz, Çakır and Özden [2] to describe secondary school students’ Computational Thinking Levels. Factor, reliability and validity analyses were undertaken for the measurement tool. The scale is a 5-pont Likert type scale with 22 items and five factors: creativity, problem solving, algorithmic thinking, cooperation/collaboration and critical thinking. The result of confirmatory factor analysis which was undertaken by using maximum likelihood technique showed item regression values between 0.507 and 0.872. Item-test correlation coefficients were between 0.655 and 0.862. Internal reliability analysis of the data was conducted to calculate the reliability of computational thinking scale and Cronbach Alpha reliability coefficient was found to be 0.809.

Dr. Scratch Web Tool: Students' programming skills in Scratch were measured with the help of Dr. Scratch web tool. This tool was developed to calculate computational thinking scores by downloading the related Scratch project files from the computer or the Internet.



Analyze your Scratch projects

Welcome to the Dr. Scratch website, an analytical tool that evaluates your Scratch projects in a variety of computational areas. This analyzer is a helpful tool to evaluate your own projects, or those of your Scratch students.

There are two options to analyze your Scratch project now!

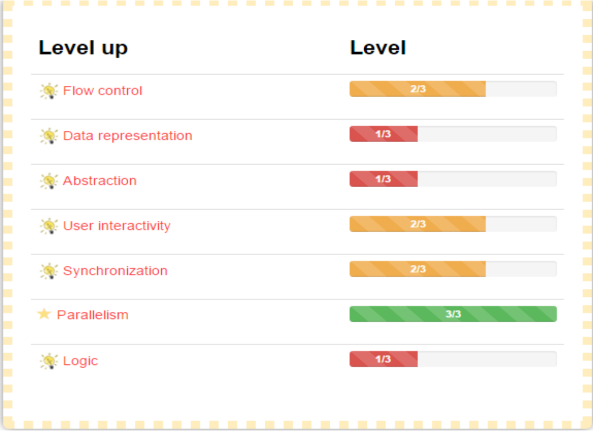
1. Introduce the url of your Scratch project, you don't have to download it:

http://scratch.mit.edu/projects/your_num

2. If you have your project downloaded in the computer you can analyze it here:

Fig.1. Dr. Scratch Web Tool Upload Screen

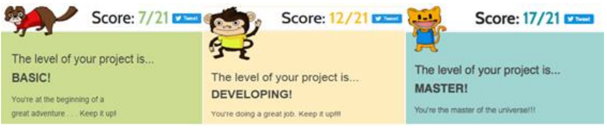
Assessment on the web tool provides 3 points for each level and the highest score is 21. The web tool provides separate cores for each level.



Level up	Level
Flow control	2/3
Data representation	1/3
Abstraction	1/3
User interactivity	2/3
Synchronization	2/3
Parallelism	3/3
Logic	1/3

Fig.2. Dr. Scratch Score Levels

The final score obtained from scoring identifies the projects to be in basic, developing or master levels.



Score: 7/21

The level of your project is...

BASIC!

You're at the beginning of a great adventure . . . Keep it up!

Score: 12/21

The level of your project is...

DEVELOPING!

You're doing a great job. Keep it up!!!

Score: 17/21

The level of your project is...

MASTER!

You're the master of the universe!!!

Fig.3. Dr. Scratch level screenshots

Dr. Scratch web tool assesses Scratch projects under 7 headings: flow control, data representation, abstraction, user interactivity, synchronization, analogy/parallelism and logic. Each heading provides 21 points by assigning 1 point to each of the three features included in this heading. While scoring flow control group; programming assigns one point when the feature is added to start the program, one point for the feature that allows the program to end instead of continuing forever and one point for programs that generates a cycle by eliminating code repetitions. While scoring data representation group, one point is assigned to identify the dimension and location of characters, one point for the program that involves new variables and one point for projects that creates and names lists. While scoring abstraction group, one point is assigned separately to group code blocks with different tasks, one point to design new blocks and one point to program and copy an object. While scoring user interactivity group, one point is assigned to starting the program with user interactivity instead of automatic start, one point is assigned to adding movement of characters in the program with user intervention and one point is assigned to ensuring sound and character interaction. While scoring synchronization group, one point is assigned to mutual interaction between two characters, one point is assigned to providing messages in character interaction and one point is assigned to uploading features such as changing stage. While scoring analogy/parallelism group, one point is assigned to work of more than one code group together, one point is assigned to assigning keys for parallel work and one point is assigned to forming conditions to generate events. While scoring logic group, one point is assigned to generate warning as a result of the created condition, one point is assigned to using more than one condition block and one point is assigned to using logical operation blocks. Addition of these points provides Dr. Scratch web tool score.

E. Data Analysis

SPSS 12.0 statistical program was used for data analysis. Spearman's rho correlation test was utilized to investigate the relationship between the analysis scores obtained by students from Dr. Scratch web tool and the scores obtained Computational Thinking Levels Scale. Mann Whitney U test was used to examine the differences in Dr. Scratch web tool score and Computational Thinking Levels Scale achievement scores.

III. Findings

A. Students' Dr. Scratch web tool scores based on gender

Mann Whitney U test was used to examine whether students' Dr. Scratch web tool scores differed according to gender. Table 2 presents the data obtained from the test.

Table 2. Investigation of Students’ Dr. Scratch Web Tool Score According to Gender

Gender	Number of Participants (N)	Mean Rank	Sum of Ranks	U	p
Female	16	15.69	251.00	115.000	,842
Male	15	16.33	245.00		

Table 2 shows no significant differences in students’ Dr. Scratch web tool scores according to gender, U=115,000, p>,05. Based on this finding, students’ Dr. Scratch web tool scores did not differentiate in terms of gender. Based on this result, it can be claimed that gender does not play a significant role on students’ Dr. Scratch web tool scores

B. Students’ Dr. Scratch web tool scores based on period of computer use

Mann Whitney U test was conducted to examine whether students’ Dr. Scratch web tool scores significantly differed according to period of computer use. Table 3 presents the data obtained from the test.

Table 3. Investigation of Students’ Dr. Scratch Web Tool Score According to Period of Computer Use

Period of computer use	Number of Participants (N)	Mean Rank	Sum of Ranks	U	p
Less than 1 hour	20	16.40	328.00	102.000	,739
1 – 3 hours	11	15.27	168.00		

Table 3 shows that students’ Dr. Scratch web tool scores were not statistically significant based on period of computer use, U=102,000, p>,05. Therefore, Dr. Stretch scores of students’ Scratch projects did not differ according to period of computer use. Hence it can be argued that period of computer use was not effective on students’ Dr. Scratch web tool scores.

C. Students’ Computational Thinking Levels Based on Gender

Mann Whitney U was conducted to examine whether the scores students obtained from the Computational Thinking Levels Scale differed according to gender. Table 4 presents test results.

Table 4. Investigation of Student Scores Obtained From Computational Thinking Levels Scale According To Gender

Gender	Number of Participants (N)	Mean Rank	Sum of Ranks	U	p
Female	16	15.31	245.00	109.000	,663
Male	15	16.73	251.00		

Table 4 shows no statistically significant differences in student scores in Computational Thinking Levels Scale based on gender, U=109,000, p>,05. Therefore, scores students obtained from Computational Thinking Levels Scale did not differentiate according to gender. It can be argued that gender did not play an effective role on student scores in Computational Thinking Levels Scale.

D. Students’ Computational Thinking Levels Based on Period of Computer Use

Mann Whitney U was conducted to examine whether the scores students obtained from the Computational Thinking Levels Scale differed according to period of computer use. Table 5 presents test results.

Table 5. Investigation Of The Scores Students Obtained From Computational Thinking Levels Scale According To Period Of Computer Use

Period of computer use	Number of Participants (N)	Mean Rank	Sum of Ranks	U	p
Less than 1 hour	20	16.27	325.50	104.500	,820
1-3 hours	11	15.50	170.50		

Table 5 presents no statistically significant differences in student scores in Computational Thinking Levels Scale based on period of computer use, U=149,500, p>,05. It can be argued that period of computer use did not play an effective role on student scores in Computational Thinking Levels Scale.

E. Relationship between Students’ Dr. Scratch Web Tool Scores and Their Computational Thinking Levels

Table 6 summarizes the findings regarding the relationship between students’ Dr. Scratch web tool scores and their computational thinking levels.

Table 6. Relationship Between Students’ Dr. Scratch Web Tool Scores And Their Computational Thinking Levels

			Computational Thinking Levels
Spearman's rho	Dr. Scratch Web Tool Scores	Correlation Coefficient	,934**
		Sig. (2-tailed)	,000
		N	31

Table 6 displays a positive relationship between student scores obtained from assessing students’ Scratch projects via Dr. Scratch Web tool and their Computational Thinking Levels, r=,934, p<,01. Correlation coefficients between “0.70 and 1” point to a high level relationship [30]. Findings suggest that increases in students’ programming skills in Scratch will cause increases in their computational thinking skills as well or increases in students’ Computational Thinking Levels will generate increases in their Scratch skills.

IV. Conclusions and Discussion

Especially in the last decade, studies have been conducted at schools about student acquisition in programming and computational thinking skills [1,12]. Scratch is regarded as a usable tool in teaching programming or ensuring that students acquire computational thinking skills [12,16]. This study compared the scores obtained by 5th graders from Scratch projects developed in the framework of Information Technologies and Software classes via Dr Scratch web tool with the scores obtained from Computational Thinking Levels Scale. Results obtained from the current study are presented below:

Scores obtained from assessing Scratch projects via Dr. Scratch web tool and students’ Computational Thinking Levels do not differ based on gender. In other words, gender is not influential on students’ project assessment scores. While literature presents extensive proof for the impact of gender on students characteristics related to computers or programming [31-34], it is also clear from the literature that there is a plethora of computing or programming features on which gender is not effective [35-37]. As a result, it is not possible to make a clear distinction in terms of computer use or programming related to gender

Scores obtained from assessing Scratch projects via Dr. Scratch web tool and students’ Computational Thinking Levels do not differ based on the period of daily computer use. It is possible to find evidence in literature that student skills regarding computer use or programming develop based on their computer or internet skills [34,38,39]. However, literature also presents that period of computer use does not affect programming skills [40]. It can be thought that casual interest on computers will naturally develops students’ game reflexes and increase their skills in using basic software and internet tools. However, training in programming is basically training in thinking. Therefore, it should not be expected that computer use will develop these skills randomly without a targeted and planned training on programming. Result of this study supports this view.

There is a significantly high relationship between students’ Scratch skills and their computational thinking skills. In other words, development in students’ programming skills in Scratch will cause similar increases in their computational thinking skills or improvements in their Computational Thinking Levels will generate increases in their Scratch skills. Literature provides extensive proof that the process of programming is not a mechanical process, but a thinking discipline [41]. As a matter of fact, ISTE’s [8] computational thinking concept is established in this ground. On the other hand, Dr. Scratch web tool aims to analyze Scratch projects based on sub headings such as logical thinking, problem solving, abstraction and analogy/parallelism [20]. ISTE’s [8] computational thinking definition provides all these sub headings and thinking skills are defined with creativity, problem solving, algorithmic thinking, cooperation, communication and critical thinking. In this context, the high level and significant relationship between these two skills can be regarded as a natural outcome.

V. Suggestions

Instead of letting students to spend casual time on computers in a random manner and providing them with basic programming education, students should be provided with planned training activities in terms of programming and thinking and in this context, thinking skills based programming training should be included in all phases of education from basic education to university grades.

While developing programs for lessons on codes, thinking exercises should be directly provided rather than teaching algorithms based on memorization and include the general structure of programming languages and history of programming

References

[1] J. Wing. Computational Thinking. Communication of ACM, 2006, 33-35.

[2] Ö. Korkmaz. R. Çakır. and M. Özden. Computational Thinking Levels Scale (CTLS) Adaptation For Secondary School Level. Gazi Journal of Education Sciences, 2015, 143-162.

[3] A.Yadav. C. Mayfield. N . Zhou. S . Hambrusch. and J. Korb. Computational Thinking in Elementary and Secondary Teacher Education. ACM Transactions on Computing Education, 2014, 14(1).

[4] J. Wing. Computational Thinking and Thinking About Computing. Phil. Trans.R. Soc. A, 2008, 366, 3717–3725 doi:10.1098/rsta.2008.0118.

[5] J. León. and G. Robles. Analyze your Scratch projects with Dr. Scratch and assess your Computational Thinking skills. Scratch2015AMS, 2015 Amsterdam, Netherlands.

[6] B. Czerkawski. And E. Lyman. Exploring Issues About Computational Thinking in Higher Education. TechTrends, 2015, 59(2), 57-65.

[7] Ö. Korkmaz. R. Çakır. M. Özden. A. Oluk. and S. Sarıoğlu. Investigation of Individuals' Computational Thinking Skills in terms of Different Variables. Ondokuz Mayis University Journal of Faculty ok Education, 2015, 68-87 doi: 10.7822/omuefd.34.2.5.

[8] ISTE. 2015. Computational Thinking Leadership Toolkit Firs Edition. Retrieved from http://www.iste.org/docs/ct-documents/ctleadershipt-toolkit.pdf?sfvrsn=4.

[9] Y. Akpınar. and A . Altun. Bilgi Toplumu Okullarında Programlama Eğitimi Gereksinimi[Programming Educational Needs in Information Society SchoolProgramming Educational Needs in Information Society School]. Elementary Education Online, 2014, 13(1), 1-4.

[10] D. Karabak. and A .Güneş. Curriculum Proposal For First Class Secondary School Students In The Field Of Software Development. Journal of Research in Education and Teaching, 2013, 163-169.

[11] S. Shin. P. Park. and Y. Bae. The Effects of an Information-Technology Gifted Program on Friendship Using Scratch Programming Language and Clutter. International Journal of Computer and Communication Engineering, 2013, 2(3), 246-249.

[12] S. Lye. and J. Koh. Review on teaching and learning of computational thinking through programming: What is next for K-12? Computers in Human Behavior, 2014, 41, 51-61.

[13] Z. Genç and G. Karakuş. Learning Through Design: Using Scratch In Instructional Computer Games Design. 5th International Computer & Instructional Technologies Symposium, 22-24 September 2011 (s. 981-987). Elazığ: Fırat University.

[14] [14]M. Başer. M. Bilgisayar Programlamaya Karşı Tutum Ölçeği Geliştirme Çalışması.[Attitude Scale Development Study Against Computer Programming]. The Journal Of Academic Social Science Studies, 2013, 6(6), 199-215.

[15] A. Gomes. and A. Mendes, A. Learning to Program - Difficulties and Solutions. International Conference on Engineering Education – ICEE 2007. Coimbra, Portugal.

[16] Ş. Çatlak, M. Tekdal. and F. Baz. The Status of Teaching Programming with Scratch: A Document Review Work. Journal of Instructional Technologies & Teacher Education, 2015, 4(3), 13-25.

[17] N. Çağıltay Ercil. and M. Fal. Scratch İle Programlamayı Öğreniyorum.[I learn programming with scratch.], 2013, Ankara: METU Pub.

[18] M. Armoni. O. Meerbaum – Salant. and M. Ben – Ari. From Scratch to " Real" Programing. ACM Transactions on Computing Education, 2015, 14(4), 10-25.

[19] J. Maloney. M. Resnick. N. Rusk. B. Silverman. and E . Eastmond. The Scratch Programming Language and Environment. ACM Transactions on Computing Education, 2010, 10(4), 1-16.

[20] J. León. G. Robles. and M. González. Dr. Scratch: Automatic Analysis of Scratch Projects to Assess and Foster Computational Thinking. RED-Revista de Educación a Distancia, 2015, 1-23.

[21] L. Begosso. and P. Silva. Teaching computer programming: a practical review. In 2013 IEEE Frontiers in Education Conference (FIE), 2013, (s. 508-510).

[22] Ö. Korkmaz. The Effects of Scratch - Based Game Activities on Students' Attitudes, Self - Efficacy and Academic Achievement. I.J. Modern Education and Computer Science, 2016, 16-23.

[23] B. Kaučič. and T. Asič. Improving Introductory Programming with Stratch? MIPRO 2011, 1095-1100.

[24] D. Ozoran. N. Çağıltay. and D. Topalli. Using Scratch In Introduction To Programing Course For Engineering Students. 2nd International Engineering Education Conference (IEEC2012), 125 - 132.

[25] A. OLUK. and F. SALTAN. Effects of Using the Scratch Program in 6th Grade Information Technologies Courses on Algorithm Development and Problem Solving Skills. Participatory Educational Research (PER), 2015, 10-20. http://dx.doi.org/10.17275/per.15.spi.2.2

[26] S. Nikou. and A. Economides. Transition in Student Motivation During a Scratch and an App Inventor Course. Proceedings of Global Engineering Education Conference (EDUCON). 2014, İstanbul Turkey.

[27] S. Federici. A Minimal, Extensible, Drag-and-drop Implementation of the C Programming Language. Proceedings of the 13th Annual Conference on Information Technology Education. 2011, New York.

[28] J. León. and G. Robles. The Europe Code Week (CodeEU) initiative. 2015 IEEE Global Engineering Education Conference (EDUCON). Tallinn, Estonia: Tallinn University of Technology.

[29] M. Resnick. J. Maloney. A. Hernandez. N. Rusk. E. Eastmond. K. Brennan. . . . Y. Kafai. Scratch: Programing for All. Communications Of The ACM, 2009, 52(11), 60-67.

[30] Ş. Büyüköztürk. E. Kılıç Çakmak. Ö. Akgün. Ş. Karadeniz. and F. Demirel. Bilimsel Araştırma Yöntemleri[Research Methods]. 2014, Ankara: Pegema Akademi.

[31] Ö. Özyurt. and H. Özyurt. A Study For Determining Computer Programming Students' Attitudes Towards Programming And Their Programming Self - Efficacy. Journal Of Theory And Practice In Education, 2015, 11(1), 51-67.

[32] W.W.F. Lau. and A.H.K. Yuen. Modeling programming performance: Beyond the influence of learner characteristics. Computers & Education, 2011, 571(1), 1202-1213

[33] S. Beyer. M. DeKeuster. K. Walter. M. Colar. and C. Holcomb. Changes in CS students' sttitudes towards CS over time: an examination of gender differences. SIGCSE Bull., 2005, 37(1), 392-396. doi: 10.1145/1047124.1047475

[34] S. Toker. An Assessment Of Pre-Service Teacher Education Program In Relation To Technology Training For Future Practice: A Case Of Primary School Teacher Education Program, Burdur.(Unpublished master's thesis), 2004, Gradute School of Education,METU

[35] A. Altun. S. Mazman. Programlamaya İlişkin Öz Yeterlilik Algısı Ölçeğinin Türkçe Formunun Geçerlilik ve Güvenirlilik Çalışması.[Validity and Reliability Study of Self-Efficacy Scale Turkish Form on Programming]. Journal of Measurement and Evaluation in Education and Psychology, 2012, 3(2), 297-308.

[36] P. Byrne. and G. Lyons. The Effect of Student Attributes on Success in Programming. Proceedings of ITICSE, 49-52.

[37] M. Yağcı. Effect of attitudes of information ,technologies (IT) preservice teachers and computer programming (CP) students toward programming on their perception regarding their self-sufficiency for programming. International Journal of Human Science, 2016, 13(1), 1418-1432, doi:10.14687/ijhs.v13i1.3502.

[38] S. Köse. A. Savran Gencer. and K. Gezer. Vocational High School Students’ Attitudes Toward Computer and Internet. Pamukkale University Journal of Education, 2007, 21(1), 44-54.

[39] S. Deniz. İ. Görgen. and H. Şeker. Tezsiz yüksek lisans öğretmen adaylarının teknolojiye yönelik tutumları. Eğitim Araştırmaları[Thesis of Master Teachers Attitudes Towards Technology.], Eurasian Journal of Educational Research, 2006, 6 (23), 62–71.

[40] B. Akkoyunlu. F. Orhan. Bilgisayar ve Ögretim Teknolojileri Egitimi (BÖTE) Bölümü Ögrencilerinin Bilgisayar Kullanma Öz Yeterlik İnancı ile Demografik Özellikleri Arasındaki İliski.[Computer Education and Instructional Technologies (CEIT) Department of Computer Use Self-Efficacy Beliefs of Students with long relationship between demographic characteristics] The Turkish Online Journal of Educational Technology, 2003, 2(3), 86-93.

[41] A.P. Ambrosio. F. M. Costa. L. Almeida. A. Franco. and J. Macedo. Identifying cognitive abilities to improve CS1 outcome. Paper presented at the Frontiers in Education Conference (FIE), 2011.

Authors’ Profiles

Ali OLUK was born in Kırıkkale in 1985. He graduated from Selçuk University Faculty of Education in 2008 and started teaching computer education and instructional technology the same year. He is still pursuing his MA degree at Science Institute, Department of Computer Education and Instructional Technology in Amasya University. He is currently working as a lecturer at Kastamonu University at Taşköprü Vocational School. His research interests include computer programming, computer based learning and augmented reality.

Ozgen KORKMAZ was born in Konya in 1972. He graduated from Faculty of Industrial Arts Education and started teaching at Gazi University Department of Computing in 1993. He completed his MA degree in Science Institute Department of Computer Education at Afyon Kocatepe University. In 2007, he completed his PhD at the Institute of Educational Sciences, Department of Educational Technology at Gazi University. He is currently working as an Associate Professor at Amasya University, Faculty of Technology, Department of Computer Engineering. His research interests include computer programming, web-based learning, blended learning, message design and web-based programming technologies.

How to cite this paper: Ali OLUK, Özgen KORKMAZ,"Comparing Students' Scratch Skills with Their Computational Thinking Skills in Terms of Different Variables", International Journal of Modern Education and Computer Science(IJMECS), Vol.8, No.11, pp.1-7, 2016.DOI: 10.5815/ijmecs.2016.11.01