Indian Journal of Science and Technology, Vol 9(46), DOI: 10.17485/ijst/2016/v9i46/107837, December 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Analyzing the Effects of Coding Education through Pair Programming for the Computational Thinking and Creativity of Elementary School Students

Young-Ho Seo* **and Jong-Hoon Kim**

Department of Computer Education, Jeju National University, 61, iljudongro, Jeju-si, Jeju, 63294, Korea; ho2832@gmail.com, jkim0858@jejunu.ac.kr

Abstract

Objectives: This study examines the effects of coding education applying pair programming to improve elementary school students' computational thinking and creativity. Coding education program focuses on geometry in a math curriculum. **Methods/Statistical Analysis:** In executing coding education program, the pair programming method was applied to the experimental group, and the general educational technique of learning via lectures/practice was applied to the comparison group. Computational cognition tests A and B, developed in Kim's study, were employed as the testing tools for computational thinking. Figure A, among Torrance's Torrance Tests of Creative Thinking (TTCT), was used to assess creativity. **Findings:** To verify the effectiveness of combining the coding education program and pair programming in improving computational thinking, A and B types of computational cognition tests were performed before and after the education program and the results were analyzed. The groups did not show significant differences when compared; however, when the results were compared within each group, the experimental group showed a significant increase in computational thinking, whereas the comparison group did not. Next, to verify the effectiveness of combining coding education program and pair programming in improving creativity, Figure A type of TTCT tests were performed before and after the education program and the results were analyzed. Again, the two groups did not show significant differences when compared; however, when the results were compared within each group, the experimental group showed significant increase in the areas of "Creativity Index," "creativity average," "fluency," and "originality," whereas the comparison group showed significant differences in the areas of "originality," and "resistance to premature closure". **Improvements/Applications:** This study is significant because pair programming, a cooperative learning approach, was applied to the coding education program for elementary school students. This facilitated in increasing their computational thinking and creativity.

Keywords: Coding Education, Computational Thinking, Creativity, Geometry in Elementary School Math, Pair Programming

1. Introduction

Many countries have already integrated coding education into their school syllabi. For instance, the governments of the U.S. and the U.K. create and manage those countries' coding education curriculum[1].

The Republic of Korea also plans to include software training as a mandatory part of the courses in elementary and secondary schools, based on coding education[2].

Schools, however, believe it is hard for students to learn software since it requires a complex cognitive capacity based on logical concepts. In addition, teachers believe it is not easy to elaborate proper individual activities given the differences of each student.

In order to resolve these problems, various teaching and learning strategies have been suggested to enhance academic performance; for example, project-based learning, peer instruction, and constructional approaches. Peer tutoring and cooperative learning always positively affect creative problem-solving abilities, academic achievement, and attitudes[3].

*Author for correspondence

In[4] is among those who have shown that pair programming, a cooperative learning approach, increases learners' self-esteem and has the effects of peer tutoring[4]. Via pair programming, academic performance and interest in learning can be increased[5].

Pair programming implies programming by two people using one computer through collaborative division of labor. The two programmers use a single computer and cooperate on design, algorithm, coding, and debugging while programming[4].

Pair programming has clearly demonstrated performance improvement through expert pair combination in the reduction of application-specific defects[6].

Pair programming was used for professional programmers; however, there are cases that have shown its educational effect.

In[4] is among those who have shown that pair programming, a cooperative learning approach, increases learners' self-esteem and has the effects of peer tutoring[4].

In[5] claimed that pair programming could positively affect academic achievement, motivation for studying, and related strategies. This implies that studies can show the effect of pair programming not only in academia but also many other additional areas[5].

However, most of the above studies are examples applied to more than middle school students. Therefore, this study focuses on the development of a coding education program for 44 students in the third, fourth, and fifth grades in elementary schools, who applied to an educational donation program during the winter vacation. The program was implemented via pair programming, a cooperative learning technique.

2. Materials and Methods

2.1 Survey to Develop Coding Education Contents

To operate a creative computer class at Jeju National University during vacation, a survey was conducted involving 34 current elementary school teachers and 162 elementary school students within the Jeju area to develop the relevant educational content.

Table 1 show that the teachers recommend Scratch and Entry for coding education.

Table 2 shows that the most recommended lecture styles for coding education were collaborative problem solving and project.

Table 3 shows the most recommended areas for math-related coding education are Rules and Problem solving, followed by Figure.

The survey on the effect of coding education revealed that it would help improve the capacity to think logically and problem solving skills, as shown in Table 4.

Table 5 shows that students selected Entry and Scratch as the programs they experienced when learning coding.

Table 6 shows that the most desired learning style for coding education is Collaborative problem solving type, followed by Project type.

Table 7 reveals that the recommended areas when practicing math-related coding education are "Rules and Problem Solving" and "Figure."

Table 8 shows the results of the survey, which demonstrates that the coding education can positively affect creativity and problem solving skills.

Entry was selected as the tool for educational programming language. Similar to Scratch, Entry is a block-based programming language and easy to be approached by

Table 1. Program we want to give recommends(teachers)

	Scratch	entry	App Inventor	Physical computing	Other
N	23	17	7	6	1

Table 2. Lecture-style educational program of the coding education (teachers)

	Collaborative problem solving type	Personal problem solving type	Project type	Lecture / training type
N	23	8	20	4

Table 3. Mathematics content recommendation for the coding education (teachers)

	The numbers and operations	Figure	Measure	Probability and Statistics	Rules and Problem solving
N	3	19	1	6	25

Table 4. The effect of the coding education(teachers)

	Problem solving skills	Creativity	Capacity to think logically	Computational Thinking	Information utilization capability
N	20	10	22	17	5

Table 5. Kind of education programs experience the coding education(students)

	Scratch	Entry	App Inventor	Physical computing	None
N	42	80	12	9	30

Table 6. Learning style of education programs of the desired Software Education

	Collaborative problem solving type	Personal problem solving type	Project type	Lecture / training type
N	76	38	58	44

Table 7. Mathematics content recommendation for The SW Education

	The numbers and operations	Figure	Measure	Probability and Statistics	Rules and Problem solving
N	44	45	19	60	32

Table 8. The effect of the SW Education

	Problem solving skills	Creativity	Capacity to think logically	Computational Thinking	Information utilization capability
N	54	57	27	50	34

elementary school students. In addition to being recommended by a number of students and teachers in the survey, the study by Kim and Lee (2016) has shown that elementary school students are not familiar with programming and have considerable difficulty in learning programming languages based on text; therefore, a block-based programming language was recommended to promote interest in programming[7].

The content for programming education linked to math curriculum was configured with the Figure part, the most recommended area by both teachers and students.

Lecture/training type would be used as the class format for programming education when learning basic grammar but proceeding classes will utilize pair programming, a collaborative problem solving technique.

2.2 The Content of Education Programming

To improve the computational thinking and creativity of elementary school students, the subjects of this study, 10 topics were selected from elementary school math textbooks as seen in Table 9.

Table 9. Education program

Hour	Step	Topic
1-2	Orientation	What is Entry? Make and deal with the object
3-4	Triangle	Drawing a triangle To use 'repeat blocks'
5-6	Various squares	Drawing various squares Making patterns using a square
7-8	Regular polygon	Drawing a regular polygon Drawing shapes utilizing a variety of figure
9-10	Circle	Drawing a circle with a variety of sizes and colors Making a Taegeuk pattern
11-12	Pushing Shape	Drawing several snowman Drawing the Olympic flag
13-14	Turning and flipping	Drawing a circle with various figures Creating flip program for lines and dots
15-16	Repeat figure with rules	Drawing repeatedly figures that create rules
17-18	Repetitive figures with the size varies	Making drawing program repetitive figures with the size varies
19-20	Using flower shape	Decorating with a flower shape.
21-24	Synthesis	Create a entry program on the basis of learned
25-28	Announce	Announced they have created to scratch Feedback

2.3 Testing Tools

Computational cognition tests A and B, developed in [8] were employed as the testing tools for computational thinking[8]. Figure A, among Torrance's Torrance Tests of Creative Thinking (TTCT), was used to assess creativity[9].

2.4 Experimental Design

Before the pair-programming education, preliminary tests of computational cognition and TTCT were performed on the experimental and comparison groups. The two groups are confirmed to be homogeneous, and the education program was performed for 7 days with 28 classes.

To verify the educational effect on both the groups, computational cognition tests and TTCT were conducted again after the program completion. Table 10 displays the study design.

3. Proposed Work

3.1 Comparing the Pre- and Post-tests within the Computational Thinking Group

The Wilcoxon signed-rank test, a nonparametric statistical method, was used since the test of normality for the experimental group showed that this group did not satisfy normality as seen in Table 11.

The experimental group showed a statistically significant increase in the computational cognition as seen in Table 12. The comparison group did not show a statistically significant difference in the computational cognition as seen in Table 13.

3.2 Comparing the Pre- and Post-tests within the Creativity Group

A parametric statistical method, the paired t-test, was applied to the creativity index, the creativity average, and fluency, whose normality was satisfied in both groups. The Wilcoxon signed-rank test, a nonparametric statistical method, was applied to originality, abstractness, elaboration, and resistance, which did not satisfy normality as seen in Tables 14 and 15.

Table 10. Experimental design

	Pre-test	Treatment	Post-test
G1	O1, O2	X1	O5, O6
G2	O3, O4	X2	O7, O8

G1: Experimental group
G2: Comparison group
O1, O3: Computational cognition test (A style)
O2, O4: Creativity test (Figure A style)
O5, O7: Computational cognition test (B style)
O6, O8: Creativity test (Figure A style)
X1: Pair programming
X2: Learning via lectures/practice

Table 11. Normality test of the experimental group computational cognition tests

Group	Descriptive Statistics(N=16)				stat	p
	M	SD	Max	Min		
Experimental (N=22)	8.1	5.1	16	1	.908	.043*
Comparison (N=22)	11.9	4.1	18	3	.961	.515

*p<.05

The experimental group showed a statistically significant increase in the creativity index, the creativity average, fluency, and originality as seen in Table 16. While the comparison group shows a significant increase in originality, resistance to premature is closure as seen in Table 17.

Table 12. Analysis of the experimental group

Period	N	M	SD	z	p
Pre	22	8.136	5.083	-2.309	.021*
Post	22	9.818	4.895		

*p<.05

Table 13. Analysis of the comparison group

Period	N	M	SD	z	p
Pre	22	11.864	4.063	-1.511	.131
Post	22	12.636	3.971		

*p<.05

Table 14. Normality test of the experimental group creativity tests

Subscales	Descriptive Statistics(N=22)				stat	p
	M	SD	Max	Min		
Creativity index	75.5	15.2	118	57	.915	.059
Creativity average	41.4	14.3	111	55	.941	.204
Fluency	98.4	19.9	131	57	.943	.230
Originality	89.3	13.5	116	54	.904	.035*
Abstractness	49.6	38.5	143	0	.905	.037*
Elaboration	126.2	19.3	150	79	.917	.065
Resistance	10.2	23.6	84	0	.504	.000*

*p<.05

Table 15. Normality test of the Comparison group creativity tests

Subscales	Descriptive Statistics(N=22)				stat	p
	M	SD	Max	Min		
Creativity index	82.9	12.6	104	60	.970	.719
Creativity average	82.4	12.2	104	60	.970	.707
Fluency	106.1	19.5	135	71	.953	.358
Originality	97.0	16.0	144	62	.927	.108
Abstractness	68.2	43.1	145	0	.930	.124
Elaboration	137.0	12.6	150	97	.838	.002*
Resistance	3.6	11.8	40	0	.332	.000*

*p<.05

Table 16. Analysis of the experimental group

Subscales	Period	M	SD	t	p
Creativity index	Pre	75.546	15.155	-3.838	**.001***
	Post	84.636	12.666		
Creativity average	Pre	74.727	14.307	-3.620	**.002***
	Post	83.182	11.713		
Fluency	Pre	98.364	19.944	-6.524	**.000***
	Post	120.364	16.828		
Subscales	**Period**	**M**	**SD**	**z**	**p**
Originality	Pre	89.273	13.460	-2.761	.006*
	Post	102.455	21.400		
Abstractness	Pre	49.591	38.514	-.877	.381
	Post	45.091	37.340		
Elaboration	Pre	126.227	19.282	-.605	.545
	Post	129.727	14.580		
Resistance	Pre	10.227	23.580	-1.472	.141
	Post	18.182	31.458		

*p<.05

Table 17. Analysis of the comparison group

Subscales	Period	M	SD	t	p
Creativity index	Pre	82.909	12.630	-.764	.453
	Post	85.727	15.147		
Creativity average	Pre	82.364	12.230	-.704	.489
	Post	84.909	14.858		
Fluency	Pre	106.136	19.463	-1.194	.246
	Post	112.682	20.101		
Subscales	**Period**	**M**	**SD**	**z**	**p**
Originality	Pre	96.955	16.004	-2.090	**.037***
	Post	104.773	16.405		
Abstractness	Pre	68.227	43.057	-1.754	.079
	Post	50.091	44.494		
Elaboration	Pre	137.000	12.627	-1.424	.154
	Post	129.727	21.092		
Resistance	Pre	3.636	11.770	-2.572	**.010***
	Post	27.227	35.951		

*p<.05

4. Conclusion

This study examines the effects of coding education applying pair programming to improve elementary school students' computational thinking and creativity. Coding education program focuses on geometry in a math curriculum.

To verify the effectiveness of combining the coding education program and pair programming in improving computational thinking, A and B types of computational cognition tests were performed before and after the education program and the results were analyzed. The groups did not show significant differences when compared; however, when the results were compared within each group, the experimental group showed a significant increase in computational thinking, whereas the comparison group did not. Next, to verify the effectiveness of combining coding education program and pair programming in improving creativity, Figure A type of TTCT tests were performed before and after the education program and the results were analyzed. Again, the two groups did not show significant differences when compared; however, when the results were compared within each group, the experimental group showed significant increase in the areas of "Creativity Index," "creativity average," "fluency," and "originality," whereas the comparison group showed significant differences in the areas of "originality," and "resistance to premature closure". This seems in accordance with the findings of [4,5].

6. References

1. Park H. Global software educational status and tools trends. Korea Internet and Security Agency Report focus 3; 2014.
2. Ministry of Education [Internet]. [cited 2015]. Available from:http://www.moe.gov.jm/.
3. Kim M. Alternative instructional methods and strategies for effective computer programming education. The Journal of Korean association of computer education. 2002; 5(3):1–8.
4. Williams L, Yang K, Wiebe E, Ferzli M, Miller C. Pair programming in an introductory computer science course: Initial results and recommendations. ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications; 2002. p. 20–7.
5. Han K, Lee E, Lee Y. The effects of pair programming on achievement and motivated strategies in programming course. The Journal of Korean Association of Computer Education. 2006; 9(6):11–28.
6. Sunitha K, Nirmala K. Correlation study on defect density with domain expert pair speed for effective pair programming. Indian Journal of Science and Technology. 2015 Dec; 8(34):1–7.
7. Kim S, Lee Y. The analysis on research trends in programming based STEAM education in Korea. Indian Journal of Science and Technology. 2016 Jun; 9(24):1–11.
8. Kim B. Programming education program based on PPS to improve computational thinking ability: Jeju National University of Education doctoral dissertation; 2014.
9. Kim K. Is creativity unidimensional or multidimensional? Analyses of the Torrance tests of creative thinking. Creativity Research Journal. 2006; 18(3):251–60.