THE INFLUENCE OF PAIR-PROGRAMMING TECHNIQUE ON SECONDARY SCHOOL STUDENTS' CONFIDENCE AND ACHIEVEMENT IN COMPUTER PROGRAMMING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

HABİBE KARAOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER EDUCATION AND INSTRUCTIONAL TECHNOLOGY

MARCH 2018

Approval of the thesis:

THE INFLUENCE OF PAIR-PROGRAMMING TECHNIQUE ON SECONDARY SCHOOL STUDENTS' CONFIDENCE AND ACHIEVEMENT IN COMPUTER PROGRAMMING

Submitted by **HABİBE KARAOĞLU** in partial fulfillment of the requirements for the degree **Master of Science in Computer Education and Instructional Technology Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Soner Yıldırım
Head of Department, **Computer Education and Inst. Tech.** _____

Assist. Prof. Dr. Gülfidan Can
Supervisor, **Computer Edu. and Inst. Tech Dept., METU** _____

Examining Committee Members:

Assist. Prof. Dr. Halil Ersoy
Computer Education and Instructional Tech. Dept., Başkent University _____

Assist. Prof. Dr. Gülfidan Can
Computer Education and Instructional Tech. Dept., METU _____

Prof. Dr. Zahide Yıldırım
Computer Education and Instructional Tech. Dept., METU _____

Date: 28.03.2018

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Habibe KARAOĞLU

Signature :

ABSTRACT

THE INFLUENCE OF PAIR-PROGRAMMING TECHNIQUE ON SECONDARY SCHOOL STUDENTS' CONFIDENCE AND ACHIEVEMENT IN COMPUTER PROGRAMMING

Karaoğlu, Habibe
M.S., Department of Computer Education and Instructional Technology
Supervisor: Assist. Prof. Dr. Gülfidan Can

March 2018, 167 pages

The aim of this case study is to explore the possible influences of Pair-Programming Technique on Secondary School students' confidence and achievement in computer programming. Within an 8-week implementation with 35 students in 5th grade, the students were divided into Individual and Pair-Programmers. The number of pair students was 22 and there were 13 individual students in the study. Both qualitative and quantitative data were collected in the study. The main data sources were interviews, students' ratings on a confidence scale, and their achievement scores based on rubrics. The qualitative data were collected through interviews with 20 students who were either in pair and individual programming groups. The quantitative data were collected through confidence scale and achievement rubrics from the 35 students. Content analysis, descriptive statistics, and Independent Samples t-Test were conducted to analyze the data.

The result of the study revealed that Pair-Programming technique was found to be effective for improving students' confidence and achievement in computer programming. Pair-programming Technique was useful for students in terms of solving complex programming problems by helping each other, sharing knowledge

and correcting mistakes. Moreover, the motivation, task completion time, quality of products and learning was revealed as positive factors of Pair-Programming Technique. Although negative issues also emerged during the implementation such as disagreements among pairs, the positive effects overweighed. As programming is one of the key skills necessary for the future of students with the changing needs of information society, this study contributes to the literature and practice by suggesting Pair-Programming Technique as an effective method.

Keywords: Pair Programming, Computer Programming, Coding, Confidence, Achievement, Secondary School Students, Computer Education

ÖZ

BİLGİSAYAR PROGRAMLAMADA EŞLİ PROGRAMLAMA TEKNİĞİNİN ORTAOKUL ÖĞRENCİLERİNİN ÖZGÜVEN VE BAŞARISINA ETKİSİ

Karaoğlu, Habibe
Yüksek Lisans, Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü
Tez Yöneticisi: Dr. Öğr. Üyesi Gülfidan Can

Mart 2018, 167 sayfa

Bu durum çalışmasının amacı eşli programlama tekniğinin ortaokul öğrencilerinin başarı ve özgüvenine olan olası etkilerini keşfetmekti. 35 ortaokul 5. sınıf öğrencisiyle 8 hafta yapılan bu uygulamada, öğrenciler bireysel ve eşli olmak üzere ayrıldı. Bu çalışmada hem nitel hem de nicel veri toplandı. Röportajlar, öğrencilerin özgüven ölçeği puanlamaları ve rubrikler ile puanlanan başarı puanları çalışmanın temel veri kaynaklarıydı. Nitel veri eşli ve bireysel olan 20 öğrenci ile yapılan röportajlar ile toplandı. Nicel veri ise 35 öğrenciden özgüven ölçeği ve başarı rubrikleri ile toplandı. Bu veri kaynaklarının analizinde alan içerik analizi, tanımlayıcı istatistik ve bağımsız örneklemler t-testi kullanıldı.

Bu çalışmanın sonucu, bilgisayar programlamada eşli programlama tekniğinin öğrencilerin başarı ve özgüveninin gelişmesinde etkili olduğunu ortaya çıkardı. Eşli programlama tekniğinin kullanımının karmaşık programlama problemlerini yardımlaşarak, bilgileri paylaşarak ve hataları düzelterek çözmede faydalı olduğu ortaya çıktı. Dahası, motivasyon, etkinliği tamamlama süresi, ürün kalitesi ve öğrenme eşli programlama tekniğinin olumlu faktörleri olarak ortaya çıktı. Eşler arasındaki anlaşmazlık gibi olumsuz faktörler olarak ortaya çıkmasına rağmen, olumlu faktörler daha ağır bastı. Bilgi toplumunun değişen ihtiyaçları doğrultusunda programlama eğitimi öğrencilerimizin geleceği için gerekli olan önemli

yeteneklerden biri haline geldi ve bu çalışma eşli programlama tekniğini etkili bir metot olarak tavsiye ederek pratiğe katkı sağlamaya çalıştı.

Anahtar Kelimeler: Eşli Programlama, Bilgisayar Programlama, Kodlama, Özgüven, Başarı

To my lovely son

ACKNOWLEDGMENTS

I want to express my deepest appreciation to my supervisor Assist. Prof. Dr. Gülfidan Can for her guidance, suggestions and criticism for the study. Without her encouragement and friendly advices, I could not complete to this study.

I would like to thank to thesis committee, Prof. Dr. Zahide Yıldırım, Assist. Prof. Dr. Gülfidan Can and Assist. Prof. Dr. Halil Ersoy for their advices, support and criticism.

I would like to show my gratitude to my friends, Aynur Yılmaz, Serdar Toprak, Satı Burhanlı, Cezmi Çal, Fatih Çal, Burçin Nazlı, Sevde Nur Gümüş, Zafer Kadirhan, Ecenaz Alemdağ and Fatih İlhan for their help and motivation. They always supported me. I sincerely, thank them for their support.

I would also like to thank to my family for their pure love and encouragement. They always motivated me during the study.

Finally, I sincerely thank to my love for his support, patience and love. The deepest thanks and appreciation is for my 8-month son Yusuf Kerem to choose me as a mother and become the source of inspiration in my life.

TABLE OF CONTENTS

ABSTRACT v
ÖZ vii
ACKNOWLEDGMENTS x
TABLE OF CONTENTS xi
LIST OF TABLES xv
LIST OF FIGURES xvi

CHAPTERS

1. INTRODUCTION 1
    1.1 Background of the Study 1
    1.2 Problem statement 5
    1.3 Purpose of the Study 5
    1.4 Research Questions 6
    1.5 Significance of the Study 6
    1.6 Definitions of Terms 6

2. LITERATURE REVIEW 11
    2.1 Programming and Programming for Kids 11
    2.2 Programming Environments for Kids 12
        2.2.1 Scratch 13
        2.2.2 Code.org 16
        2.2.3 Alice 17
    2.3 Information Technologies and Software Lessons and Programming Education in Turkey 18
    2.4 Extreme Programming Approach and Pair-Programming Technique 19
        2.4.1 Extreme Programming Approach 19
        2.4.2 Pair-Programming 19

2.5 Cooperative Learning Instructional Method 21
2.6 Review of Related Studies 24
2.6.1 Research Studies about Pair-Programming for Adults and Kids 24
2.6.2 Programming for Kids on Scratch 26
2.6.3 Programming Education in Turkey 28
2.7 Summary of the Review of Literature 29

3. METHODOLOGY 31
3.1 Research Questions 31
3.2 Research Design 31
3.3 Context of the Study 34
3.3.1 Information Technologies and Software Course 34
3.3.2 Learning Environment 35
3.4 Participants 37
3.4.1 Participants of Confidence Scale 38
3.4.2 Participants of Interview 39
3.5 Implementation 39
3.5.1 Implementation of the Course 39
3.5.2 Implemetation of Pair Programming Technique 43
3.6 Data Collection Instruments 44
3.6.1 Confidence Scale 44
3.6.2 Interview Protocol 47
3.6.3 Lesson Plans and Rubric 48
3.7 Data Collection Procedure 49
3.7.1 Quantitative Data Collection Procedures 50
3.7.2 Qualitative Data Collection Procedures 52
3.8 Data Analysis 54
3.8.1 Qualitative Data Analysis 55
3.8.2 Quantitative Data Analysis 56
3.9 Quality of Research 56
3.9.1 Credibility 57

3.9.2 Transferability 58
3.9.3 Dependability 59
3.9.4 Confirmability 59
3.10 Researcher Role and Possible Biases 60

4. RESULTS 63
4.1 Preparation of Data and Assumptions 63
4.2 Programming Confidence of the Students 65
4.2.1 Quantitative Results for Programming Confidence of the Students 65
4.2.2 Qualitative Results for Programming Confidence of the Programmers 71
4.3 Programming Achievement of the Students 84
4.3.1 Quantitative Results for Programming Achievement of the Students 84
4.3.2 Qualitative Results for Programming Achievement of the Programmers 87
4.4 Emotions of StudentsDuring the Programming Process 95
4.5 Comparing, Combining, and Summarizing the Results 102

5. DISCUSSION 109
5.1 Programming Confidence and Programming Achievement of Pair and Individual Programming Students 109
5.2 Conclusion 124
5.3 Implications of the Study 125
5.3.1 Implications for Practice 126
5.3.2 Implication for Theory and Research 127
5.4 Limitations of the Study 128
5.5 Recommendation for Further Research 129

REFERENCES 131

APPENDICIES

A. Q-Q PLOTS AND HISTOGRAMS FOR NORMALITY 143

B. INTERVIEW PROTOCOL FOR STUDENTS (TURKISH) 147

C. CONFIDENCE SCALE (TURKISH) 149

D. CONFIDENCE SCALE (ENGLISH) 151

E. SCRATCH LESSON PLAN 153

F. ACHIEVEMENT RUBRIC (TURKISH) 157

G. ACHIEVEMENT RUBRIC (ENGLISH) 159

H. PARENT'S CONSENT FORM (TURKISH) 161

I. VOLUNTARY PARTICIPATION APPROVAL FORM (TURKISH) 163

J. ETHICS COMMITTEE APPLIED ETHICS APPROVAL FORM OF MIDDLE EAST TECHNICAL UNIVERSITY RESEARCH CENTER (TURKISH) 165

K. APPROVAL FORM OF MINISTERY OF EDUCATION (MEB) 167

LIST OF TABLES

TABLES

Table 3.1	Data Collection Process.....	51
Table 3.2	Summary of the Interviews.....	56
Table 4.1	Tests of Normality.....	66
Table 4.2	Descriptive Statistics for Confidence.....	69
Table 4.3	Group Statistics for Confidence Scale.....	71
Table 4.4	Independent Samples Tests for Confidence Scale.....	72
Table 4.5	Possible Factors that Influence Programming Confidence (NP: Pair, NS: Individual, NT: Total).....	74
Table 4.6	Descriptive Statistics of Achievement.....	87
Table 4.7	Group Statistics for Achievement Scores based on Rubric Evaluation.....	87
Table 4.8	Independent Samples Test for Achievement Scores based on Rubric Evaluation.....	88
Table 4.9	Group Statistics Mann-Whitney U for Achievement Scores based on Rubric Evaluation…….....	89
Table 4.10	Mann-Whitney U Test for Achievement Scores based on Rubric Evaluation…….....	89
Table 4.11	Factors Influence Achievement.....	90
Table 4.12	Emotions Influences Confidence and Achievement……	99

LIST OF FIGURES

FIGURES

Figure 1.1	The Scratch learning environment’s main parts.....	8
Figure 2.1	The Scratch learning environment.....	14
Figure 2.2	The Code.org learning environment.....	17
Figure 2.3	The Alice learning environment.....	18
Figure 3.1	The Scratch projects of students.....	38
Figure 3.2	The Scratch projects of students.....	39

CHAPTER 1

INTRODUCTION

The current chapter presents the background of the study, problem statement, research questions, purpose of the study, and significance of the study and definitions of terms.

1.1 Background of the Study

Developing programming skills is not only necessary for the students who are in the area of computer sciences but also for the K-12 students who want to meet the needs the of their life problems (Yoon, Kim, & Lee, 2016). Especially with the technological developments of 21th century, gaining computer programming skills has become popular for people in the areas of Mathematics, Engineering and Designing (Grover & Pea, 2013; Kinnunen, P, & Malmi, 2006). The new developments in the technology necessitated them to get profound knowledge about a computer programming language, because people realized that knowing a programming language is one of the valuable skills for their future (Hwang, Shadiev, Wang, & Huang, 2012). Since the computer programming is one of the core topics for this century and since it provides lots of opportunities for people, learning a programming language is demanded by the people who want to get a job for the future and gaining computational skills for their life problems (Sáez-López, Román-González, & Vázquez-Cano, 2016; Wilson & Moffat, 2010). The development of the technology also challenged to the students while gaining ability of computer programming skills and solving the problems of Mathematics, Science, and Engineering (Chang, 2014).

Moreover, being a creator of software is more popular than using or consuming technological products (Clark, Rogers, Spradling, & Pais, 2013). According to Kalelioğlu (2015), the 21st century people need to have the ability to be productive rather than being a consumer. With new development of technology and computer

systems, people strived to learn the ways to create the software for new technological devices. Innovating new technology and using programming languages also provide people to gain a variety of skills such as computational thinking, problem solving, designing systems, and problem representations (Kafai & Burke, 2013). Computer programming also develops the algorithmic thinking skills and problem solving skills of the programmers (Fessakis, Gouli, & Mavroudi, 2013).

Since the computer programming is a vital for the people in the century, the education of both adults and children were gained prominence internationally. The universities and schools mentioned the importance of the computer programming and so new curricula was prepared for the core topic of programming for both novice and expert education (Yoon et al., 2016). Although the computer programming is considered as a topic for adult education or software developers, Kalelioğlu (2015) stated that the new generation can adapt easily for the new changes in the technology and they can also adapt to computer programming and computer sciences. For the importance of programming education of children, there has been sudden impetus to develop materials, curriculums and methods for programming lessons in most of the developed and developing countries including Turkey. For instance, United Kingdom applied a new course of computer programming for the primary and secondary school students as a must course (Esteban, 2016; Grover & Pea, 2013). Estonia also implemented programming education in their education system. Moreover, Finnish education system included the programming courses into the curriculum to teach the students computational thinking and logical thinking skills (Sáez-López, 2016). In addition to these countries Israel, Russia, New Zealand, Australia and South Africa created computer science classroom and curriculum for K12 students (Grover & Pea, 2013).

Although learning computer programming is stated as a beneficial skill for both adults and children , the computer programming is a difficult process because it needs to have several cognitive skills and practical skills (Ambrósio, Costa, Almeida, Franco, & Macedo, 2011). Moreover, computer programming was stated as a complex subject for the students who firstly involved in computer programming in all age groups (Kelleher & Pausch, 2005). According to Wilson and Moffat (2010) most of the computer engineering students thought that the computer programming was hard to learn, therefore some of them gave up the computer programming

lessons or they got lower grades. The reasons behind the difficulties of learning programming by mostly adult learners are complex language of commands, confusing logic of syntax, learning new codes and lack of knowledge about programming (Kelleher & Pausch, 2005).

Similarly, K12 students also consider the computer programming as confusing, boring, too hard to excel so that most of them give up learning the programming (Wilson & Moffat, 2010). Wilson and Moffat (2010) also stated that correcting the mistakes in the textual programming environment is difficult for students in this age group.

Since the computer programming is a popular topic and it is thought as a difficult topic to learn by both adults and children, some improvements were applied in the programming education such as visual programming environments, robot programming, and online programming environments. There are many programming environments which were designed for children to make programming and algorithmic thinking accessible for them (Denner, Werner, & Ortiz, 2012). Moreover, to make the computer programming education easy for the university students, some learning strategies like pair-programming and extreme programming were designed according to the needs of computer programming students.

Pair-programming is one of the commonly used techniques that were adopted from the extreme programming and agile process method. The agile method is developed by Agile Kent Beck and his colleagues to make the programming process quick and productive (Beck, 1999). In pair–programming, programmers work in pairs and sit together to develop code (Berenson, Slaten, Williams, & Ho, 2004). The collaborative programming environment is created for the programmers. Thus, the programmers integrate their knowledge of programming on the same programming problem.

In the literature, the programming confidence and programming achievement of the pair-programming students were explored. Working with a partner provided more enjoyable learning environment with social interaction (Kelleher & Pausch, 2005). The result of the research showed that the pair programming was effective technique that improved the programming confidence of the students, problem solving skills and the technical knowledge about programming (Dongo, Reed, & Hara, 2016).

According to Hanks, McDowell, Draper, and Krnjajic (2004) pair-programmers were also more confident than the solo programmers. Programming confidence of programmers increased with the pair programming technique and pair –programmers wrote more functional codes than solo programmers (Arisholm, Gallis, Dybå, & Sjøberg, 2007). Moreover, the pair-programmers achieved more than solo programmers, they also learned more codes and completed the assignments more successfully (Hanks, McDowell, Draper, & Krnjajic, 2004). Hannay, Dybå, Arisholm, and Sjøberg (2009) also stated that pair programmers wrote more correct codes than solo programmers in the same time period. The benefits of the technique on programmers stated as more enjoyable programming environment, more confident learners, and improved cognitive development of learners compared to solo programmers.

Furthermore, a study by Cliburn (2003) on the experiences with pair programming indicated that working in pair on programming activities provided many advantages like enjoyment, high achievement, and less workload on the introductory programming class of the students. Nagappan et al. (2003) also stated that working with a partner on the programming task provided to have positive attitude towards learning computer programming. Moreover, Hwang et al. (2012) found that cooperative programming increased the learning motivation of the students. The number of students who submitted their homework correctly was increased among the pair-programmers (Hanks, McDowell, et al., 2004). According to a research conducted by Hannay et al. (2009) pair-programming was beneficial technique for the achievement of complex programming problems.

The pair-programming technique is commonly used and beneficial especially for novice learners and several studies have conducted with adults. However, there is little information about experiences of children, especially the secondary school students on the usage of pair programming technique. Its effects on secondary students on their programming confidence and programming achievement are still largely unexplored.

1.2 Problem Statement

Pair-programming is a commonly used technique for the adults' computer programming education. On the other hand, the pair-programming technique is rarely used for children programming education. The literature showed that the usage of pair-programming technique researched by several researchers and the outcomes, advantages and disadvantages of implementation of the technique for adults who were generally computer sciences department of universities and software engineers were reported. The conducted research studies showed that the pair-programming technique provides several advantages like fun, motivation, and confidence in computer programming and increased achievement in programming in the adult education. With the help of the results of these research studies, educators had a chance to analyze whether the pair programming technique was beneficial for the programmers or not. The results of studies in the literature show direction to educators while implementing the technique in the computer programming lessons. However, according to literature review of the computer programming is a new topic for the children education and pair-programming is rarely used in the computer programming education of children. For this reason, there is limited information about the usage of pair programming on the programming education of children. Because of this limitedness, it is difficult to implement the pair programming effectively and decide the technique is beneficial for the children.

Since the computer programming is a popular topic for students' education in most of the countries including Turkey and the pair programming technique is commonly used and beneficial for programming education of adults, there is a need to research the influences of pair programming technique for students' learning programming. Moreover, for the complexity of the programming and its popularity among children, it is clear that gaining better understanding about the usage of pair-programming and the results of the implementation of pair programming technique on the children' programming achievement and programming confidence is necessary.

1.3 Purpose of the Study

This study explores the possible influences of the implementation of pair programming technique on programming confidence and programming achievement of secondary school students.

1.4 Research Questions

The study explores the following research questions:

1. How does the application of pair-programming technique influence the confidence level of secondary school students during programming process?

2. How does the application of pair programming technique influence the achievement level of secondary school students during programming process?

1.5 Significance of the Study

Considering the gap in the literature, extensive exploration is needed to understand how or if the pair programming technique influences the secondary school students programming achievement and programming confidence. First, the results of the study may contribute to the literature by filling the gap in the literature in terms of pair-programming practice in programming education of secondary school students, especially visual programming and online programming environments. Second, the findings of the study can provide useful instructional strategies and techniques for computer science teachers, curriculum developers, and other educators to facilitate students' learning of programming using pair-programming method. The teachers in other contexts may decide whether to use this technique or not, may use appropriate suggestions of this study in their own classes and contexts to observe the differences and similarities while utilizing pair-programming technique. Finally, the reports of this study about the students' experiences, opinions, preferences, and perceived benefits of pair-programming technique and the problems experienced during the process can help the practice and guide the similar future studies.

1.6 Definitions of Terms

Extreme Programming Method:

Muller and Tichy (2001) defined the extreme programming method as "Extreme programming" is a software development method for a small group of people who dealing with rapidly changing needs of programming. Pair programming is one of the sub-methods of the extreme programming method.

Pair Programming:

"Pair programming" is defined as programming technique in which programmers work together at the same computer for completing the task on common programming environment and code collaboratively (Williams & Kessler, 1999). One of the programmers is called the driver and the other programmer is the navigator. The primary job of the driver is to type the code. The navigator looks for errors in the code as the driver types and suggests strategies for attacking various problems as the pair works. In some ways, the role of the navigator is more important than the driver as he or she has a more objective viewpoint and can think about the direction a particular line of thought may proceed as they program.

Pair programmers: In the context of this study, pair programmers are the programmers work together to complete the programming activity in the class. Two programmers sit shoulder to shoulder and work collaboratively during the programming process. One programmer chooses and combines the codes, the other one helps him or her in the logical sequence of the codes and finds the errors of the activities.

Non-pair programmer (individual programmer): In the context of this study, Non-pair or individual programmer is a programmer who works alone during the programming process in the class, completes all programming activities in the programming lesson all by him/herself.

Scratch*:*

Scratch is a visual programming environment which was developed by MIT Media Lab projects. It helps young learner to make programming easily and gains 21. Century skills such as thinking creatively and reasoning systematically (Resnick, Kafai, & Maeda, 2005). Scratch programming environment is free toolkit that contains drag and drop code blocks and enables to create games, animations and interactive art. The program includes four main parts which are block palette, stage, script area and sprite palette (Resnick, Kafai, & Maeda, 2005).

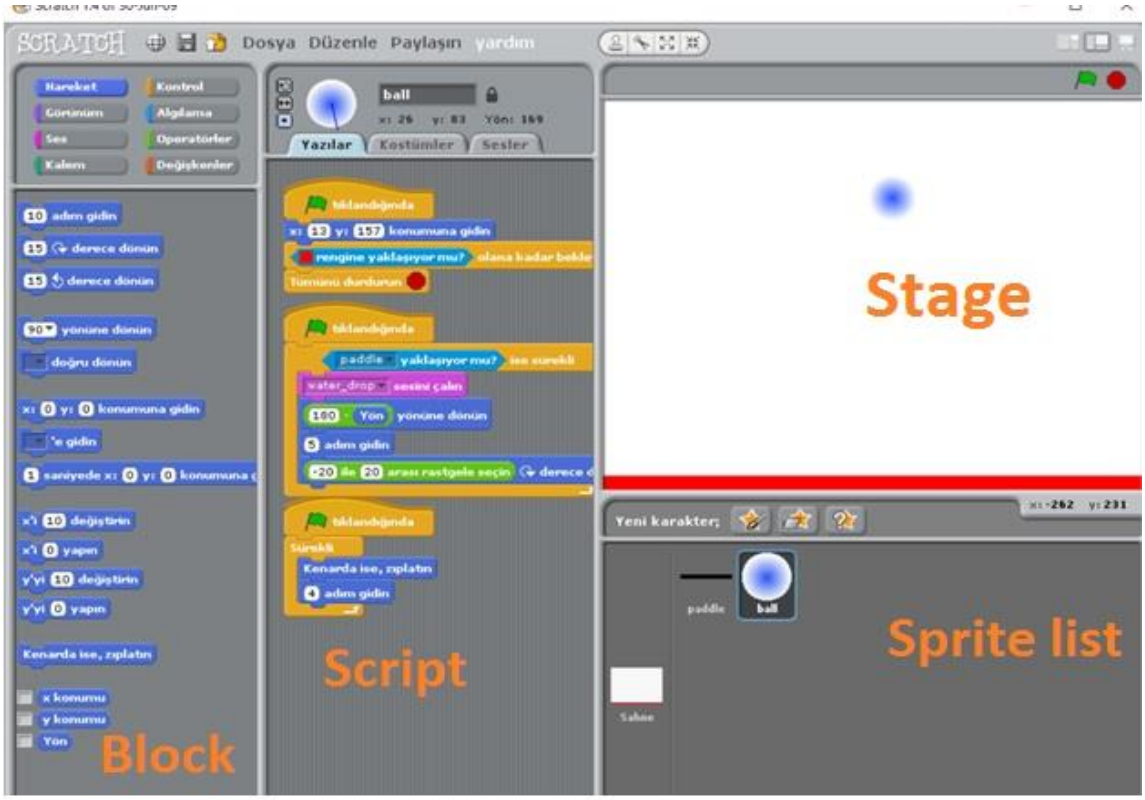


Figure 1.1 The Scratch learning environment's main parts.

Block Palette: The block palette is visual programming interface which includes colorful code blocks in different code categories (Resnick, Kafai, & Maeda, 2005). With the help of the code blocks, programmers can bring different blocks together to create small programs. Code blocks in the block palette looks like a piece of a puzzle so programmers create their programs just like doing a puzzle. Block palette contains several types of code blocks which are motion, looks, sound, pen, events, data, sensing, operators and control categorized according to functions in the programming environment. The categorization in the Scratch programming environment aims to facilitate the programming to the novice learners.

Script Area: Script area is a code screen of the Scratch programming environment. Programmers in the Scratch drag the codes block palettes and drop it the codes on script area. The programmers combine the codes according to aim of the programs in this area.

Stage: Stage is a part of Scratch environment that includes visual representation of characters and animation of these characters.

Novice Programmer: In the context of this study, novice programmers are defined as those who do not have enough experiences in programming environment. In the study novice programmers are 5^{th} grade secondary school students who have not taken any programming courses before. According to curriculum of Ministry of National Education in Turkey, the Information Technologies and Software course are stated to be given in the 5^{th} grade. Thus, 5^{th} grade students are novice programmers for the study.

Information Technologies and Software: Information Technologies and Software is a course that includes several abilities using of technology. According to Information Technologies and Software lesson curriculum , the course aims to teach the students both cognitive skills and technical skills (MEB, 2012). The course is compulsory in the 5^{th} and 6^{th} grade; in the 7^{th} and 8^{th} grade levels it is elective. In the study, the main topic is also teaching programming to the novice programmers.

Secondary school students*:* In Turkey, Secondary school students are students who have completed Elementary school and are typically in the age between 9 to 15.

Programming Confidence*:* In the study, programming confidence indicates that the trust and beliefs of the Secondary School programming class students in their own ability for programming activities.

Programming Achievement: In the context of this study, programming achievement is the success of Secondary School programming class students during the programming activities based on an internationally used rubric.

CHAPTER 2

LITERATURE REVIEW

This chapter aims to present the review of related research studies based on the pair programming method and programming skills of students. The review includes the pair–programming technique, programming environments for kids and the studies on programming for children. Firstly, the programming (coding) concepts for the children, the history, and the importance of programming are presented. Secondly, the Information Technologies and Software lesson in Turkey and other countries and the core programming topics are presented. Then, the programming environments that were created for the students are presented followed by pair programming method and the implementation and definitions. Lastly, a review of studies related with the pair-programming method, programming confidence and programming achievement of the secondary school students are presented.

2.1 Programming and Programming for Kids

Although most countries including Turkey have revised their education program according to relatively new phenomenon of programming for kids in the last few years, the programming education for children has started to develop in 1960s with the Logo programming environment for teaching mathematics (Feurzeig, Papert & Lawler, 2011). The Logo programming environment was developed for teaching mathematics in a logical way that supports programming thinking skills and provides activities for children (Feurzeig & Papert, 2011). The environment also aimed to change the teaching strategy of mathematics education by adding constructivist activities for classical learning topic of mathematics (Papert, 1991).

Since the programming education was considered useful to improve mathematical skills, problem solving skills, higher order thinking skills of the children (Papert, 1991), the TORTIS programming environment was also developed by the Perlman

for the use of programming to robotic devices with the Logo turtle (Perlman, 1976). The goal of designing the programming environment was to facilitate the programming language to the 3-4 years old kids and make the programming environment more accessible for them and one of the other major aims of the designing the programming environment was teaching kids to teach programming a robotic system with the Logo turtle (Solomon & Papert, 1975).

The first programming environments Logo was developed for both teaching programming to young children and kindergarteners and helping them gain some skills during the programming process. Many researchers and educators analyzed the importance of the environment on different perspectives. Logo improved the language and social skills of the kids and young children by providing the collaborative working programming environment (Strand, 1986). Moreover, Clements (2002) reported not only the pedagogical side of the Logo environment but also the development of thinking skills and problem solving skills of children. Lye and Koh (2014) stated that scaffolding, constructivist activities and the problem based learning were useful skills of the programming education of the K12 students.

The programming has been a core topic from past to present. In the last few years the popularity and importance of learning a programming language has increased in many countries of the world. For this reason, most of the software companies developed visual programming environments to facilitate the programming education of the children.

2.2 Programming Environments for Kids

There were several programming environments which were designed for teaching programming to the children such as Scratch, ToonTalk, Code.org, Alice, and Stagecast creator. These environments included features for young programmers, including simple programming syntax, simple commands and visual code blocks and drag and drop properties on the code blocks of instead of typing the code syntax one by one (Fessakis et al., 2013). Young programmers have some difficulties on the complex programming environment and the advanced level of programming environments in terms of writing syntax and correcting mistakes (Wislow, 1996). Because of this problem, the motivation, enjoyment, confidence, and the achievement of the children for programming could decrease and the fear and

anxiety could increase (Felleisen et al., 2004). For these reasons, according to Gross and Powers (2005) different programming environments were designed such as Logo, Alice, Scratch, Karel and StarLogo for children to make the programming concepts and syntax easy in the education of beginner programmers.

The well-designed programming environment could help young learners' programming development (Haugland, 1992). Programming environments provide learners to participate actively to the programming activity and support the development of the programming skill of the children by controlling the programming environment (Fessakis et al., 2013).

2.2.1 Scratch

Scratch is a visual programming environment designed by MIT Media Lab to facilitate teaching programming languages to kids (Resnick, Kafai, & Maeda, 2005).Scratch was designed according to constructivist learning approach and the Papert's Logo project (Papert, 1980). It is also a graphical programming environment that provides opportunities for learners to create games, interactive stories, animations, art and different multimedia (Brennan & Resnick, 2013). The Scratch programming environment provided enjoy visual learning platform for the children (Ota, Morimoto, & Kato, 2016). To develop the interactive project with Scratch, the environment presents approximately a hundred coding blocks into the eight categories that are motion, sound, look, pen sensing, control, operators and variables (Resnick, Kafai, & Maeda, 2005).With the increasing demand on the programming subject, Scratch becomes one of the preferred language environments for the kids (Chang, 2014). Scratch is one of the commonly used programming environments because programming with the Scratch is easier than other programming environments for the young programmers by presenting a lot of visual code blocks that look like a LEGO puzzle and scripts (Resnick, Kafai, & Maeda, 2005). Programming with the Scratch environment is easy for the students because the environment allows the students coding without syntax errors by using the colorful code blocks (Maloney et al., 2010). The other major reason for becoming the commonly used programming environment is that Scratch is a free and it is a media rich platform for the young programmers, teacher and parents (Resnick, Kafai, & Maeda, 2005)

The programming environment improves some skills of the programmers during the programming process. He, Chang, and Liu (2010) stated that learners gain various abilities in mathematics, arithmetic, creative thinking, problem solving, and it leads to joyful learning**.** The creator of Scratch programming environment stated that the environment develops some skills of the children like collaborative working, creative thinking, problem solving, systematic analysis and communication (Resnick, Kafai, & Maeda, 2005). During the programming process, some features of the environment like mathematic, arithmetic and statistic can add the programming activity to enrich the problem solving, creative thinking and collaborative thinking skills (He, Chang, & Liu, 2010).

The visual interface of the Scratch consists of four main parts with different features that are stage, sprite list, scripts area, and blocks palette (Chang, 2014). The figure (Figure 2.1) shows the parts of the Scratch.

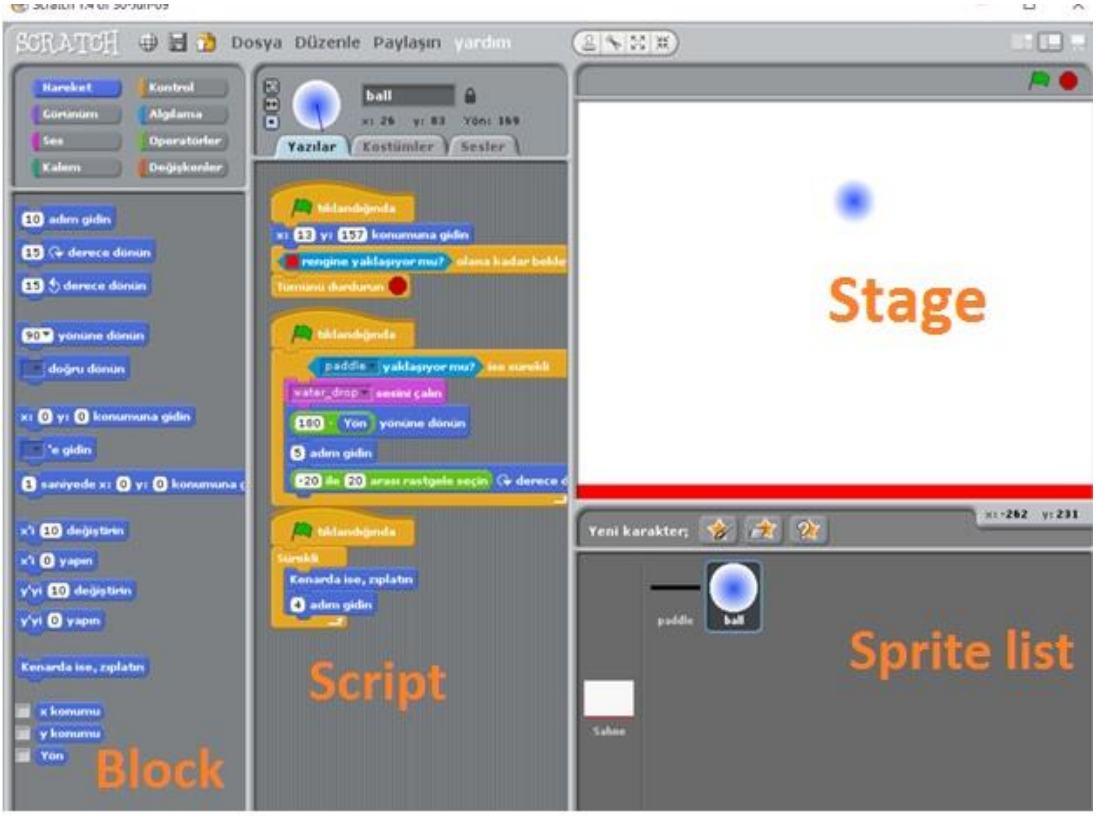


Figure 2.1 The Scratch learning environment.

Scratch programming environment was developed according to some core key features such as building block programming, rich media manipulation, and

collaborative working environment and support multiple language (Resnick, Kafai, & Maeda, 2005). The key features represent the following properties.

1. Building-block Programming: Programming in the coding environment depends on selecting the accurate code blocks and fitting them into accurate sequence. The code blocks look like puzzle pieces, so the young programmer does not need to write any code during the programming process. The code blocks have colors according to their functions, so the programmer can distinguish the type of code blocks easily (Resnick, Kafai, & Maeda, 2005).

2. Rich Media Manipulation: Scratch programming environment includes different type of media resources, so the user can develop a program by adding sound, animation, emotion, and graphics (Resnick, Kafai, & Maeda, 2005). By mixing different media types in the programming activity, the user can work in rich software and improve the programming skill (Resnick, Kafai, & Maeda, 2005).

3. Collaborative Working Environment: Scratch working environment provides the programmer ability to work with partners and share their projects with the other programmers in the web site of the Scratch. The programming environment help the user share their products and observe the products of other people in the world. With the web site of the Scratch, programmers can work with the partners and observe the different type of Scratch projects (Resnick, Kafai, & Maeda, 2005).

4. Support Multiple Languages: Scratch programming environment supports many languages so that the children can join the programming environment in their own languages. Scratch is developed approximately in 40 languages and allows the user creating programs with their own language. Supporting multiple language can provide to more achieved young children and share products of them in their own language (Resnick, Kafai, & Maeda, 2005).

2.2.2 Code.org

Code.org is an online visual programming environment designed in 2013 for the students who can create codes by dragging and dropping the code blocks for their coding activity (Code.org, 2014). The aim of designing the environment is explained as making the programming education to the students easy and fun. The launcher of the Code.org (2014) stated that the students can create codes in a collaborative and a creative environment. The online environment aims to teach the K-8 students computer programming with the skills of computer thinking, logical thinking and algorithmic thinking. The lessons in the environment include 20 steps from easy to difficult level. Students lead to the challenges in the lesson and work hard to pass each step in logical way (Code.org, 2014).

The environment teaches some key concepts to the novice programmers in a well-designed context (Figure 2.2). The online visual programming environment teaches their learners information about computer literacy, usage of applications, basic programming concepts including functions, loops and conditions and thinking critically by using algorithms (Code.org, 2014).

According to (Kalelioğlu, 2015) students can also observe the learning process of themselves during the programming process, analyze the outcomes of the activities in terms of which step is completed.

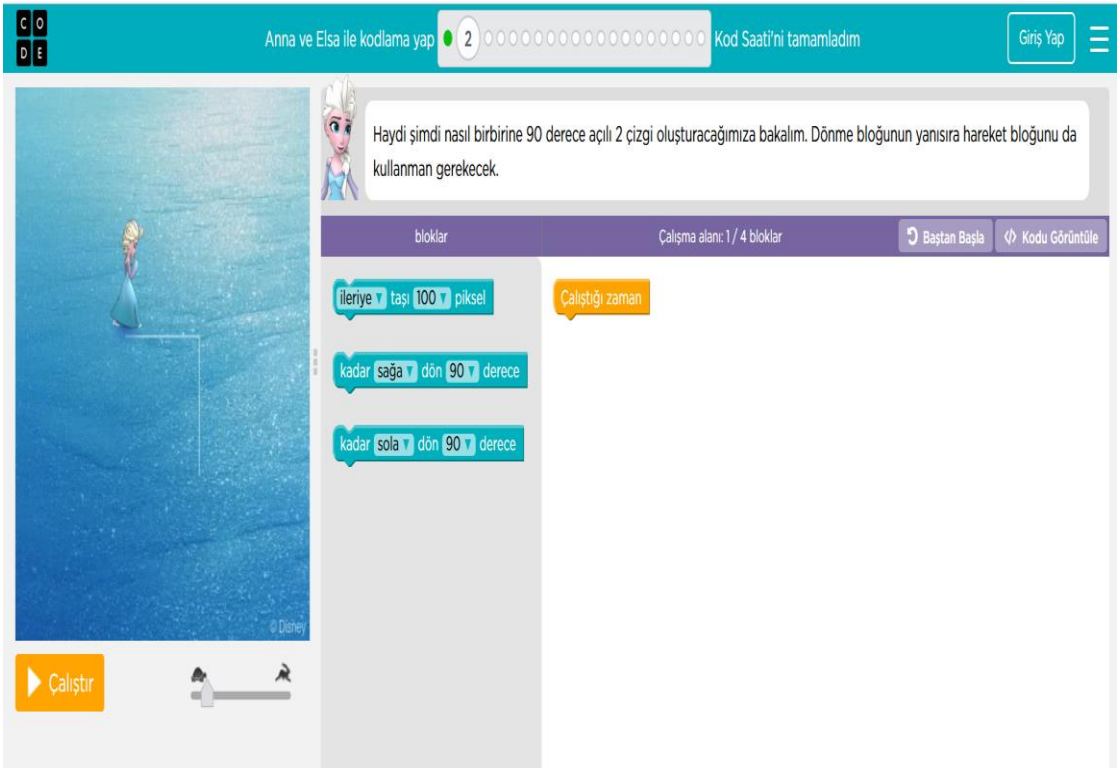


Figure 2.2 The Code.org learning environment.

2.2.3 Alice

Alice is an open source programming environment which presents to educators drag and drop interface to create 3D characters with adding the motion to the characters (Alice.org, 2017). The environment designed by the Carnegie Mellon University for the novice programmers to create the animations easily (Alice.org, 2017). The environment provides 3D motion environment with the code blocks that makes easy to programming to the characters (Figure 2.3).

Alice allows the novice programmers learn basic programming concepts in a 3D objects like vehicles, animals, people (Alice.org, 2017). The programming environment aims to teach the learners Alice and Java programming languages to make the programming easy and fun while creating video games, and animations (Moskal et al., 2004). Alice programming environment prepare the novice learners to the complex programming courses by facilitating the programming concepts and 3D object design (Cooper et al., 2003).

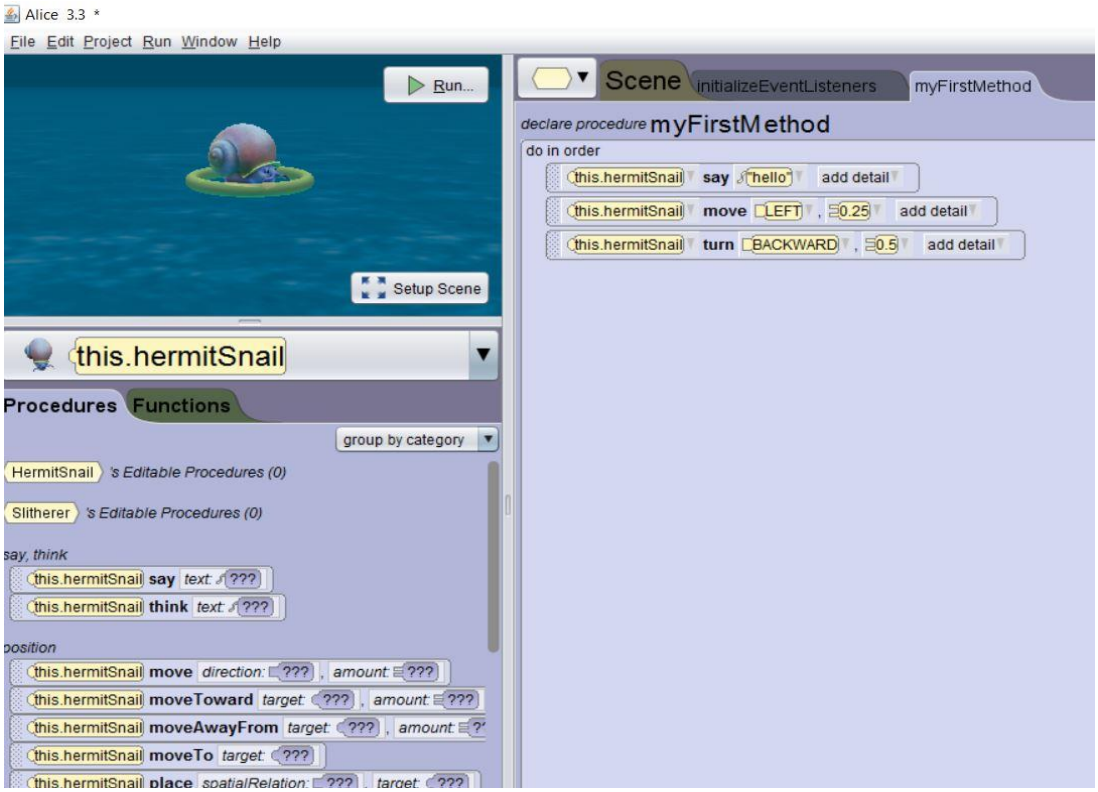


Figure 2.3 The Alice learning environment.

2.3 Information Technologies and Software Lessons and Programming Education in Turkey

Information Technologies and Software lesson is course that aims to teach the students technological innovations and the usage of computer applications for everyday use. The lesson was designed in 2005 as an elective course for the students 1st to 8th grade level by the Ministry of Education in Turkey (BTE, 2013).

In 2013, the name and the level of the lesson were changed such that the name of the lesson changed as "Information Technologies and Software" and the course offered 5th and 6th grade as 2-hours must course, 7th and 8th grade as 2-hours elective course (BTE, 2013). The curriculum of the new lesson was designed to make the students producers rather than consumers of technology, so the course includes some changes including new technological trends and innovations like programming, 3D environment design (MEB, 2018).The curriculum of the Information Technologies and Software course also includes many several programming subjects like creating

small software using visual programming environments, problem solving with the programming, creating 3D objects for improving the creativity of the students (MEB, 2018). Programming, mobile learning, 3D environments started to be used after the curriculum was changed in 2013.

2.4 Extreme Programming Approach and Pair-Programming Technique

2.4.1 Extreme Programming Approach

Extreme programming (XP) is a software development approach designed by the Kent Beck and his colleagues which aims to achieve productivity with the excellent software products by collaborating the team members clearly (Beck, 1999). With the design of the XP approach, the usage of traditional software engineering methods was decreased (Muller & Tichy, 2001).

According to Beck (1999) with the extreme programming approach, the development of the software depends on many values such as communication, feedback, simplicity, courage, and respect. The values implemented on the production of the software and a team searches the solutions of the problems together. Extreme programming approach is developed for the software engineers' team to increase the productivity by working the team and provide quick solution to the problem of the software (Muller & Tichy, 2001).

2.4.2 Pair-Programming

Pair-programming is defined as a collaborative programming method in which all the programming steps are completed by the two programmers who works on the same computer (Berenson et al., 2004). Although the pair programming method is an old concept that is one of the key elements of 12 elements of Extreme programming approach, pair programming become a popular technique for the software engineers and programmers recently (Beck, 1999).

To create the computer software with the pair programming method, two programmers use one computer, mouse and keyboard (Hanks et al., 2004). According to Williams and Kessler (2002) the two programmers called as driver and navigator in which drivers create codes, design the environment, test the programs and controls the programming environment; the navigators observe the driver's codes and give some suggestions about the work and correct the mistakes of the

driver during the programming process.

The driver is always active in typing the codes and design of the code environment. On the other hand, the navigator detects the problems in the coding environment, asks questions about the process of driver and gives suggestions to their partner to develop the program (Nicolescu & Plummer, 2003). The Cliburn (2003) stated that the role of the navigator is more important in the programming process of two programmers because the navigator should be active, creative and objective in the programming tasks. In addition to this, the two programmers (navigator &driver) should communicate to create new solutions to their problem. The navigator and the driver should communicate with each other 45 to 60 seconds periodically (Williams et al., 2001).

Working "shoulder to shoulder at one computer" aims to produce more qualified programs and decrease the some tactical deficiencies and misspelling codes in the software (Williams & Kessler, 1999). To create the quality products with the pair-programming method and give the equal role to the programmers, the role of the two programmers should be change every one hour (Bevan, Werner &McDowell, 2002).

Several methods can be implemented while composing the pair groups for the pair-programming method. The researchers who used pair programming method used different strategies to create pair groups. Williams and Kessler (1999) used the pair programming method in their study and formed the group according to characteristic of the students like active working students or passive worker students. McDowell et al. (2004) did not apply any rule and students chose their partners freely. On the other hand, Napaggan et al. (2003) applied different strategy that the software created the groups randomly. In the Napaggan's research, the partner changing was done during the study, but some of the groups worked with the same partner throughout the semester.

Nicolescu and Plummer (2003) stated that pair programming method should be applied according to some rules as follows;

1. The pair group member works together on the same computer, keyboard and the mouse to complete the task successfully. They complete the entire task together.

2. Each member has a role in the pair programming method such that one member is navigator who observes the driver, check the written codes and correct the mistakes make brainstorming and give suggestions, the other member is driver who types codes and make all of the work on the computer.

3. The navigator and the driver decide the design of the software and the type of the codes together.

4. The driver and the navigator always communicate with each other and ask questions and discuss to solve the problem of the task.

5. The role of the driver and navigator are changed during the software development, thus the group members gain experience on both roles.

2.5 Cooperative Learning Instructional Method

Pair programming technique is not an instructional method; however it has many similarities with cooperative learning. Several research studies in the literature referred to the relationship between the instructional method of cooperative learning and the software development technique of pair programming. The study Preston (2006) explored the usage of cooperative learning in order to improve pair-programming technique. Preston (2006) resulted that the cooperative learning with pair programming technique, the students became more active during the learning process and the weaker students were involved more into the projects.

In this section, brief information is given about cooperative learning, its difference with collaborative learning, and why pair-programming technique was associated with cooperative learning in this study.

Cooperative learning is an instructional method in which students work together to achieve a goal, solve problem, or complete the classroom assignments. The cooperative learning provides helping each other, preparing the study by searching together and sharing knowledge to solve the problems (Sharan & Sharan, 1987). Cooperative learning is also defined as a student-centered instructional method in which students work together by interacting with each other in the same group (Lam et al., 2013). In cooperative learning the students works in a small group and learn with the same learning materials, the teacher gives the necessity information about the subject at the beginning (Slavin, 1987). The role of the teachers' in cooperative

learning was designing the learning environment, supporting the students in learning (Panitz, 1999; Cooper & Mueck, 1990).

Cooperative learning is explained as an effective and important instructional method for the students because the method enhances learning of students by providing interaction between the students in working groups and sharing the learning tool for common goals (Doymuş et al., 2004). According to Slavin (1987) the cooperative learning improves the interaction between the pairs, motivation of the students, higher order thinking skills of the students. The cooperative learning develops social interaction while learning the academic subjects (Davidson & Major, 2014). The cooperative learning includes five basic elements which are positive interdependence, face-to-face promotive interaction, individual and group accountability, development of team-work skills, and group processing (Johnson, Johnson, & Smith, 1998).

The usage of cooperative learning method with the key elements of method supports the learning in the working groups. The study about Johnson and Johnson (1989) resulted that the cooperative learning provided more successful learners, productivity, good interactions and socialization, and self-confidence. The integration of cooperative learning to the education was designed by Slavin (2011) indicated that each group member's work is based on the group goals and the motivation is a factor for learning and enthusiasm of group mates. The cooperative group members socially interact and their motivation helps students' learning. In the cooperative learning pair tutoring, pair modeling, pair practice and pair assessment, and correction improves students' learning (Slavin, 2011). The pair programming technique supports pair assessment and correction, social interaction, encourage group mates, and helping group mates during learning.

Collaborative learning, on the other hand, is an instructional method in which students work together to achieve a common goal. In collaborative learning, each group member helps each other and responsible for each other's learning (Gokhale, 1995). Collaborative learning is also defined as social learning process for the students in which the students worked together to solve the problems, help each other, and increase the motivation of group members (Teague & Roe, 2008). The

teacher has the role of a facilitator in collaborative learning (Gokhale, 1995). Yerion and Rinehart (1995) described the advantages of collaborative learning including deeper learning, higher motivation, improved cognitive skills, and enhanced social interaction. The achievement, confidence and enjoyment were improved with the collaborative learning method (McKinney & Denton, 2006). The satisfaction of the students and their enthusiasm were also enhanced with the collaborative learning method (Yerion & Rinehart, 2002).

The collaborative and cooperative learning methods have lots of similarities as mentioned in this part of the study. Although, both of the methods have several similarities, there are also differences among these methods. The differences provide to explain the reason of choosing the cooperative learning method in the study. The role of the teacher is one of the important reasons of using cooperative learning instead of collaborative method for this study. The role of the teacher is defined as the facilitator in both of the methods. While teachers do not particularly observe the learning process or provide the information about topic in collaborative learning, the learning process is specifically monitored by the teachers and teacher provides information to the students in the cooperative learning method (ResourcEd, 2018). In the present study, similar with the cooperative learning, the teacher also observed the learning process especially the students' roles as drivers and navigators in the pair programming technique. The teacher also provided necessary information about programming in the first lesson and the students used the information in the programming activities in second lesson.

The organization and structure of the working groups is another factor of choosing cooperative learning method for the present study. According to Panitz (1999) collaborative learning needed to have more detailed and complex preparation process. The organization of the working groups is structured and the role of the students are assigned by the teachers in cooperative learning method (ResourcEd, 2018). In the present study, similar to the cooperative learning, the teacher also organized the pair programming groups. The teacher assigned the pair groups and role of the students (driver or navigator). The learning environment and the rules of the pair programming technique were also organized by the teacher. Teacher also checked the students' role changing process in every 2 weeks during the semester.

As a result, the cooperative learning instructional method was preferred to support the pair programming software development technique in the study.

2.6 Review of Related Studies

The pair programming method was started to be used by software industry, software development process of computer science program students, and teaching strategy of programming to the adult education. The method was not used as the teaching method for children. The study aims to utilize the popular method of pair programming used by adults with helping kids learn programming. The review of the study was divided into three categories as 1) pair programming for adults and kids, 2) Programming for kids with Scratch, and 3)Programming Education in Turkey.

2.6.1 Research Studies about Pair-Programming for Adults and Kids

In pair programming technique, programmers collaborate on the same task and one of the students develops codes and the other student observes and helps the partner while dealing with complex problems and mistakes (Gómez et al., 2017). This technique was commonly used in the computer programming and software development area. The studies show that the pair programming is an efficient technique for programmers who need to support for the improvement of their programming. On the other hand, some of the results showed that the method is not effective for the software development of programmers as discussed in the following.

Performance of pair programming groups who were adult learners was better than the solo- programmers in learning, quality of software (Alves, Salge, & Berente, 2016; Lye & Koh, 2014). The pair programming technique was useful because it developed critical thinking skills, programming skills, and provided interaction between programmers who were high school students (Bailey & Africa, 2017). The significant results were resulted that the ratio of programming confidence and fun increased with the pair programming technique by changing the partner's role of 6^{th} grade elementary school students in 5 minute periods (Zhong et al., 2018). Similar to the literature that reported the effects of pair programming technique on programmers as being more productive, confident and having fun. Pair groups were more confident and enjoyed more than individual programming groups and the achievement levels of all children were better (Hanks et al., 2004; Salleh, Mendes, & Grundy, 2011). The study conducted by Werner (2009) resulted that the pair

programming was an effective technique for the middle school students programming education in term of gaining ability of problem solving.

Pair groups adult programmers can answer the questions of the task and able to manage the problem of the software by working as team (Williams & Kessler, 2002). Williams and Kessler (2002) reported that the pair groups never gave up the work during the laboratory session but some of the pair groups gave up the task because they could not find the answer of the questions in the programming environment. Waite, Jackson, and Leonardi, (2004) resulted that the group work in computer sciences provided improved performance and high satisfaction for the programmers who were Computer Engineering university students. Pair programming method also helped adult learners to find solution to their problems by working on the same computer and on the same programming task (Cliburn, 2003; Williams & Kessler, 2000).

Pairing students produced better quality software than individual students and the students also achieved more on tests, and get higher scores on the programming course which were for the adult learners (Williams & Kessler, 2000; Faculty, 2016; Nagappan et al., 2003). In the study of Hannay, Dybå, Arisholm, and Sjøberg (2009), the pair programming reported as a method which not only provide quality product production but also develop the social interaction and collaboration of the group members that support the confidence and the motivation of the professional adult programmers in 3 different countries. Li, Plaue, and Kraemer (2013) resulted that the pair programming technique increased the productivity of the university students in programming. Alves et al. (2016) also resulted that pair programming students developed codes quicker than individual programmers in computer programming course in university. Moreover, the pair programmers' programs were differentiated from the individuals' in terms of being shorter and easier to manage in the future (Cockburn & Williams, 2001). The adult programming student divided into groups and pair programmers used the time effectively and complete the programming activity quicker than solo programmers (Nagappan et al., 2003).

The research results about adult learners showed that pair programming provided interaction between students, higher grades, confidence and motivation for learning of the computer programming (Salleh et al., 2011). Williams and Kessler (2002) explained the benefits of pair programming method as quicker software production

(approximately half time) than solo programmers, knowledge transfer between the adult programmers, improved learning environment, and collaborative learning environment.

Zacharis (2001) compared the effectiveness of pair programming with solo programming among the adult learners and the results showed that pair programmers created less error (50% fewer) in their software than solo programmers. McDowell and Werner, 2006) studied the improvement of confidence and programming quality of programming class students in elementary school by using pair programming and the results showed that the enjoyment and the confidence level of the pair programmers were higher than solo programmers. The achievement of the adult students was also better in the pair programming groups when the pair programming technique used accurately (Nagappan et al., 2003; Umapathy & Ritzhaupt, 2017).

Although, most of the relevant studies showed that the pair programming was an effective method for producing software and learning computer programming, some of the studies also showed the negative outcomes of the pair programming method. In the study of the usage of pair programming in adult education, arranging time and finding well-organized partner for working together for the project was difficult for most of the pair programming groups (Bevan et al., 2002). The completion time of programming activities by the 6^{th} grade pair programming student was found slower than individual students because of the time lost in communication between the pairs (Lewis, 2011).The poor partner matching caused the unexpected outcomes for the process of project and pair programming method did not work in this situation (Cliburn, 2003). The study of Arisholm (2007) resulted that the pair programming adult groups did not create expected software product in terms of correctness of codes.

2.6.2 Programming for Kids on Scratch

The Scratch visual learning environment provided to help kids learn computer programming by creating projects, stories and games (Maloney, Resnick, Rusk, Silverman, & Eastmond, 2010). Perlman (1974) showed that the visual environment provides solutions of the syntax problems, typing command problems with the visual code blocks.

The result of the study of Sáez-López (2016) showed that the visual programming environment (Scratch) is an effective educational tool for both education of computer programming to the kids and gaining some abilities like mathematics, art, history and music. Scratch programming environment was differentiated from the other programming environments in terms of touching different type of interests that the students engage the programming to show their feelings and talents, and present rich media sources for designing the programming environments (Maloney et al., 2010).

Scratch programming environment facilitated the development of computer programs and increased the motivation and engagement of the students with the visual code blocks and visual programming environment(Maloney et al., 2010). The visual programming environment including Scratch made the advance programming concepts simple for the novice students to engage them into the programming environment (Kelleher & Pausch, 2005). Kelleher and Pausch (2005) stated that the environment was well designed for the beginners of programming so the motivation and enjoyment of the students were increased in the programming lessons. Students improved the ability of programming by creating advanced programs and designing the environment with the friendly interface environment of Scratch (Idlbi, 2009). According to Idlbi (2009) with different type of sources of Scratch, all of the students enjoyed at the end of the programming sessions.

The result of the case study about the Scratch usage in the elementary school resulted that the confidence, motivation, fun, engagement and attitudes were benefits of creating code blocks in Scratch environment (Sáez-López, 2016). Moreover, the Scratch environment facilitated the programming learning of students quicker than text based programming environments and keeps the enthusiasm of the little programmers for a long time period (Maloney et al., 2010). In the study of usage of Scratch programming environment to produce a game, the research resulted that although the time was limited to learn programming and producing a game, the students could easily learn the environment and programming concepts in 6 weeks and created programs for the game production effectively (Yoshihara & Watanabe, 2016). Moreover, the result of the study of Urban Youth Learning Programming with Scratch resulted that the environment of Scratch increased the popularity of programming between the youth urban programmers' and the number of programmer was increased in the programming club (Maloney, Peppler, Kafai, Resnick, & Rusk,

2008). The study of Maloney et al. (2008) also stated that Scratch increased the motivation of the students by providing support and having user friendly interface for the novice programmers.

Scratch programming environment includes both the functions of programming and also conditional statements, loops, variables which provide easy assessment for the students, the result of the study indicated that the students were motivated for the Computer programming subject and teachers could able to assess their students easily (Ota et al., 2016).

2.6.3 Programming Education in Turkey

Similar to other countries, programming education is a popular subject in Turkey. The curriculum was updated with the needs of programming education. Moreover, special courses were introduced in order to teach programming and robot coding to the secondary school students. Sayginer and Tüzün (2015) stated that since the programming education of children is important in Turkey, some developments were made in order to make computer programming prevalent for kids. According to Sayginer and Tüzün (2015), Ministry of Education in Turkey developed a portal (EBA) for the programming education and with the help of the portal, teachers and students could share their projects and examine the other projects.

Since the programming education is an important subject in Turkey, several studies were also conducted to understand the effects and importance of the programming education of children. According to Demirer and Sak (2016) the programming education improves the problem solving skill and analytical thinking skills of students. The study of Genç and Karakuş (2011) reported that Scratch programming environment provides students to gain some skills including creative thinking skills, algorithmic thinking skills, mathematical and computational thinking skills. Çetin (2012) conducted a study that analyzed the effects of programming on problem solving skills of children. The study reported that the programming education of children contributed to problem solving skills of children positively (Çetin, 2012). The Scratch programming environment provided significant results on the problem solving skills, algorithmic thinking skills and the creative thinking skills of children (Yünkül, Durak, Çankaya, & Abidin, 2017).

The computer programming developed the positive attitudes for the school environment and for the lessons (Demirer & Sak, 2016; Akpınar & Altun, 2014). Genç and Karakuş (2011) stated that the Scratch programming environment provided the collaborative working environment for the children. Çatlak et al. (2015) also resulted that Scratch programming environment increased the motivation of students and the interests of the students towards lessons with its friendly interface.

However, the study by Kalelioğlu and Gülbahar (2014) explored the effects of Scratch programming environment on problem solving skill of primary school students. It resulted that the Scratch programming environment did not show any significant effects on problem solving skills of primary school students.

Only few studies explored pair-programming technique with secondary school students. Demir and Seferoğlu (2017) conducted a research to compare the pair programming and individual programming process of secondary school students by using Scratch learning environment. It resulted that the pair programmers completed the programming quickly and with less mistakes in the codes. The study Demir and Seferoğlu (2017) also reported that pair programming provided more enjoyable working environment for the students while creating code blocks in Scratch.

2.7 Summary of the Review of Literature

Although several studies present the advantages using pair-programming technique in computer programming process, there are limited number of studies researching the influence of pair-programming on programming confidence and programming achievement of secondary school students.

As mentioned in this chapter, computer programming is a valuable skill but learning computer programming is difficult process for both adults and children. To facilitate the programming, several improvements were implemented such as visual programming environments and online programming environments. A programming environment with easy to use interface provides various benefits for the programmers while learning and implementing code blocks. Several studies about visual programming environments showed that the environments facilitate the computer programming process with the easy to use interface options. According to studies in the literature about computer programming education, the researchers mainly focused on the visual programming environments and the benefits and limitation of

these environments. However, computer programming requires cognitive skills, higher order thinking skills, and problem solving skill (Papert, 1991). To achieve the cognitive skills of programming, some methods and techniques were implemented on the education of computer engineers, software developers such as agile method, extreme method, and pair-programming technique. The literature showed that pair-programming technique is commonly used in the adult education. Related studies about pair-programming reported that the technique is beneficial for programmers in terms of motivation, fun, productivity, confidence and achievement. Although the pair programming technique is commonly used in adult education, there are limited studies for children's programming education.

Since the technique is beneficial for the programmers, the current study aimed to combine the visual learning environment of Scratch and pair programming technique to get in-depth information about secondary school students' experiences. The results of the literature search failed to result in a study about the usage of pair programming on the secondary school students' computer programming confidence and achievement. The literature mostly searched the influences and factors of pair programming technique on adult education.

Therefore, this study aims to deeply analyze the influence of pair programming technique with Scratch on secondary school students' confidence and achievement. To implement the technique on the secondary school students' computer programming lessons, the outcomes of the techniques should be analyzed deeply. Thus, the study explored how pair programming technique influences the programming achievement and programming confidence of secondary school students in the programming process. Getting in-depth knowledge about this problem is necessary because of limited research in the literature, therefore the case study design of the research used to help to analyze the experiences of secondary school students on pair programming technique with Scratch.

CHAPTER 3

METHODOLOGY

The current chapter presents the research questions and research problems of the study, the design of the study, sampling strategy and participants, data collection instruments, data collection procedures, data analysis, quality of the study, researcher role, and limitations of the study.

3.1 Research Questions

The study explored how the experience of pair programming influences programming confidence and programming achievement of secondary school students.

The study investigates the following research questions:

1. How does the application of pair-programming technique influence the confidence level of secondary school students during programming process?

2. How does the application of pair programming technique influence the achievement level of secondary school students during programming process?

3.2 Research Design

It should be noted that this study is qualitative in nature. However, both qualitative and quantitative data were collected to triangulate data and understand students' experiences from various perspectives. An overview of the research design is provided here in this part to help readers easily understand the rest of the chapter.

In a class with 35 students and only 27 computers, she arranged 11 pair-programmers and 13 individual programmers. This situation is not unique to this research, as many computer classrooms in Public schools in Turkey do not have adequate number of working computers for each student. Therefore, in one classroom both pair and

individual students are naturally formed. The difference of this study is that, the initial arrangement of pairs and individuals was systematic and did not change throughout the study to make sure students' experiences are formed in a long-period of time that the results represent their unique experiences being in either situation (pair vs. individual). Also, the pair groups were informed about their roles as being navigators and drivers. Therefore, pair groups utilized pair-programming technique in this study. In the first week of the implementation, the teacher also provided information regarding the rubrics and provided examples.

For 8 weeks, the teacher utilized Scratch lesson plans and rubrics which were provided in Scratch website for teachers. Each activity was different. In 2-hour (40+40 minutes) class duration, the teacher gave directions of the activity in the first hour and then she gave the activity to the students in the second hour. The students were asked to do the activities by themselves and were not allowed to ask any questions to the teacher. At the end of the lesson, the teacher answered their questions, gave feedback to the students, and evaluated the activities by using the rubrics. She completed all the evaluations at the end of the lesson and re-checked them within a week to ensure accuracy. The rubric scores were used for part of their course grade.

After the 3th lesson, the teacher implemented the confidence scale. At the end of the 8-week period, confidence questionnaire was applied again and interviews were conducted with students to understand their own opinions of their achievement and confidence in programming and their experiences in these 8 weeks.

Case Study

Research method of this study is case study. According to Creswell (2007), "Case studies explore an issue through one or more cases within a bounded system" (p.73). Case study method is commonly used for gaining deep understanding of a phenomenon in the natural life situations in the fields such as education, psychology, and health (Merriam, 2009). Main conditions of using case study design are answering how and why questions, lack of the manipulation of participants' behaviors, discovering conditions of environment and uncertainty of the boundaries between the phenomenon and context (Yin, 2009).

Gaining an in-depth understanding a phenomenon needs some requirements such as collecting data from different sources, being active participant in the phenomenon as a researcher and the in-depth description of the result of the phenomenon (Creswell, 2007). Yıldırım and Şimşek (2013) stated that the results of each phenomenon are different from each other, so the similar cases are not compared and the researcher cannot make any generalization about the result of the phenomenon.

In this study, researcher explored the experiences of students during programming in class where there are two different forms of programming (individual and pair). The study investigated the phenomenon without any manipulation of students for the responses they give, and it gave detailed information about participants and their conditions. The researcher was actively involved in the context and deeply analyzed students' experiences by collecting various types of data.

For the study, the qualitative data were collected and supported by the quantitative data which is defined as embedded case research design. Embedded case study is a type of case study that is structured by several sub-units of analysis (Yin, 2003). Moreover, embedded case study eases the combination of quantitative and qualitative methods in one research study (Scholz & Tietje, 2002; Yin, 2003).

In this research, secondary school students worked on programming activities in 8 weeks either individually or as a pair. At the beginning of the intervention, a programming confidence scale was given to the students. Each week, their achievements on activities were assessed by using rubrics. At the end of the intervention, programming confidence scale was given again and interviews were conducted with the students. In qualitative part of the study the main purpose was gaining an in-depth understanding of novice child programmers', specifically secondary school students' programming confidence and programming achievement with the usage of pair-programming technique. Qualitative data were collected by conducting interviews with the students. Therefore, with the collection of these data, rich-information was obtained about the issues of programming confidence and programming achievement of students with pair programming technique.

In quantitative part of the study the main purpose was supporting the qualitative results of the study by using the programming confidence scale and programming achievement rubric results. In the quantitative data collection, programming

achievement rubric and programming confidence scale were administered on the process of programming activity. The pair groups performed a programming activity by collaborating with their partners. On the other hand, the individual participants worked alone and performed the activity by themselves.

3.3 Context of the Study

3.3.1 Information Technologies and Software Course

"Information Technologies and Software" is a course offered in 5th and 6th grade as must course and 7th and 8th grade as an elective course (Talim Terbiye Kurulu, TTK). Before the changes of Board of Education and Discipline in 2013, the course was given as an elective course from 1st grade to 8th grade as one hour. The name of the course was also changed from "Computer Course" to "Information Technologies and Software course" in 2013 and the course was given in 2 hours as a must course for the 5th and 6th graders.

In the 5th grade Information Technologies and Software curriculum, programming is one of the core topics of the course. In addition to the programming topic, information about several technologies, hardware and software parts of computers, Internet security and ethical issues of Internet usage, beneficial applications and software usage, Windows Office applications are the other topics of the curriculum of Information Technologies and Software course of 5th grade.

In previous years in the school where the data were collected in this study, this course was given as a course which included several applications and technologies according to curriculum of Talim Terbiye Kurulu (TTK). The grading policy of the Information Technologies and Software lesson had included written exam, project works and activity rubric results of each lesson. In the written exam, the questions were about general programming concepts, software and hardware of the computer, algorithm of programming and so forth. Multiple choices, true and false and open-ended questions were asked in the exam. Moreover, students completed several programming activities in the lessons, but they were mostly completed individually if there were adequate numbers of computers. The activities were measured with the rubrics that are presented in the Scratch official web site. The project works were also one of the assessment methods for the course.

In the year when the research was conducted, Information Technologies and Software course was given for two semesters. In the first semester curriculum did not include programming. The second semester, students started to learn programming. However, in the first semester, students got general information about what programming is. The research was conducted in the second semester and pair-programming technique was used. Instead of paper-pencil tests, performance-based assessment was used dominantly. The course grade consisted of 50 % of activities that are part of this research, 20% of exam result, and 30% of project. Not only their achievement but also their programming confidence was evaluated. Similar to the previous semesters, the students used exercises in Scratch website and assessed using the rubrics.

3.3.2 Learning Environment

For the usage of pair programming technique on programming activities of the children, a programming environment, named "Scratch" was used. The environment was developed by MIT Media Lab projects. Scratch is free programming environment that launch several projects on its web site from the users of different countries. The Scratch programming environment helps young learners to learn programming easily. It also helps learners to gain 21. Century skills like thinking creatively and reasoning systematically. According to Maloney et al. (2010) the environment was defined as; "Scratch is a visual programming environment that lets users create interactive, media-rich projects. People have created a wide range of projects with Scratch, including animated stories, games, online news shows, book reports, greeting cards, music videos, science projects, tutorials, simulations, and sensor-driven art and music projects" (p.16).

In this environment, students encounter four main parts that are block palette, script area, stage, and sprite palette. The environment presents drag and drop blocks for users to make programming easily. This environment is commonly used because it makes it easier to learn programming for kids. Scratch programming environment developed the motivation and needs of the young programmers (8 to 16 years old) who join the Intel Computer Clubhouses after school time (Resnick et al., 2005).

One of the important properties of this environment is having a web site for discussion and sharing the projects of the users. This helps the user examine the code blocks of different projects. The web site is also supported by the documents and resources part for the parents, teachers and the students. The lesson plans were implemented in the programming lessons of the secondary school students' lessons.

Some screenshots of projects of the students who were the participants of the study are provided below (Figure 3.1 and Figure 3.2).

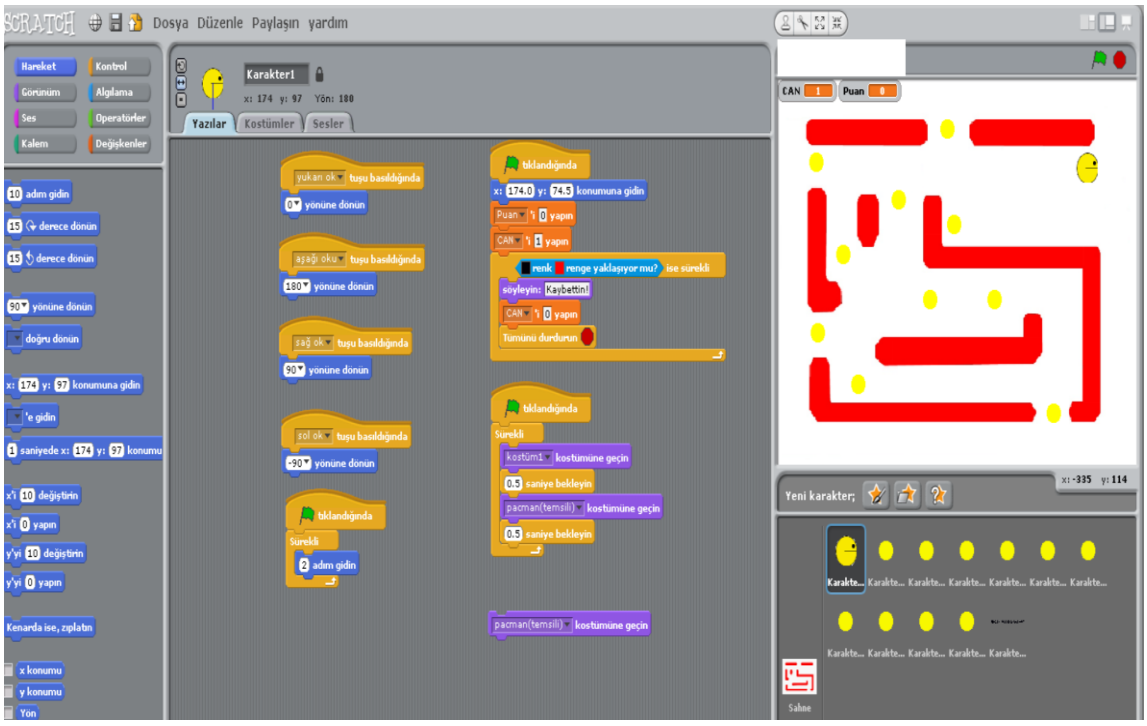


Figure 3.1 The Scratch projects of students.

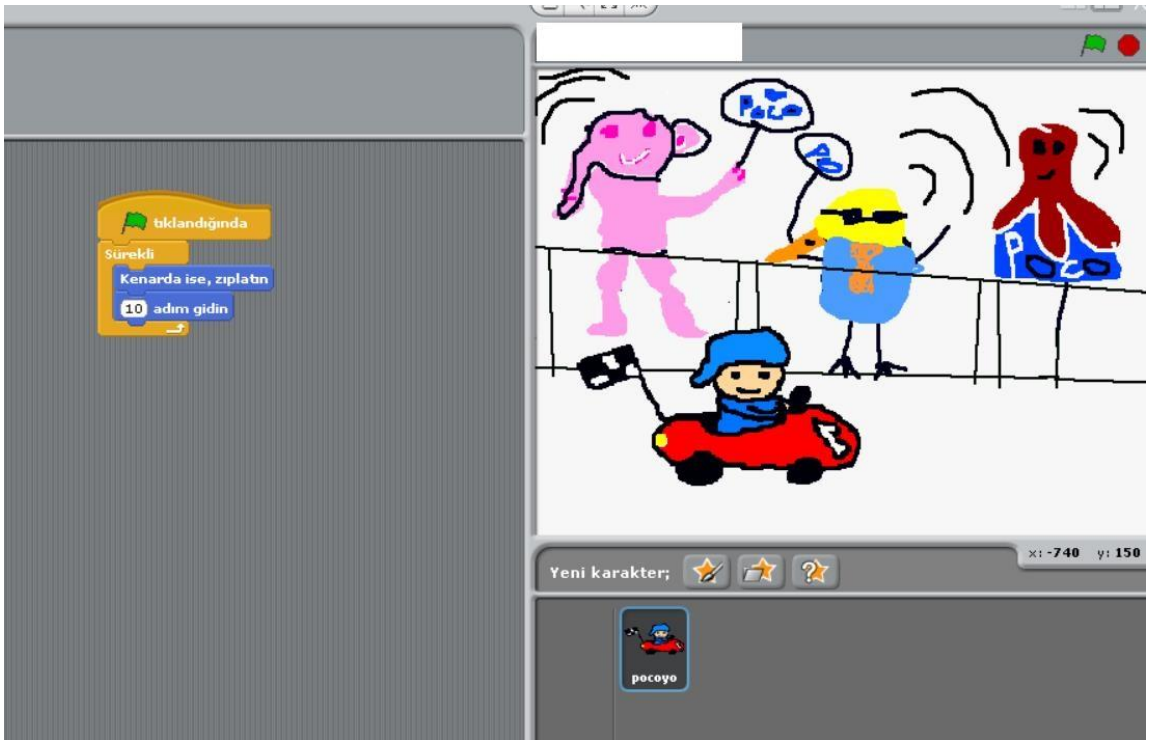


Figure 3.2 The Scratch projects of students.

3.4 Participants

The participants of the study were 5th grade students from the Yenikent İlksan secondary school in Sincan in Ankara. According to the records of the counseling service in Yenikent İlksan public school, the economic status of the students in the school was low. For the 5th grade students in the study, most of the parents had low income and the salaries were under the poverty rate according to the poverty rates in Turkey reported by TUIK (2017).

The reason behind choosing this school was the researcher is a computer literacy teacher of this school. The Yenikent İlksan secondary school is a school with bilateral training applied for the students of primary and secondary schools. The school is a public school and has 1207 secondary school students and 1570 primary school students in 2017. The average number of students in classrooms is 40 students. The school has one computer laboratory. The interactive white boards and Internet connection are available in each classroom. The Computer laboratory was

opened in 2008 by the MEB. Most of the computers have Windows XP and a few of them have Windows 7 operating system with 2GB memory. The number of computers in the laboratory 27 and each computer has Internet connection.

The participants were selected from 5th grade programming class students in the Yenikent İlksan secondary school in Ankara. The purposeful sampling strategy was used for selection of the participants. With the purposeful selection information, rich cases are supplied to gain in depth information about the phenomenon (Patton, 1990; Creswell, 2007). As a teacher, the researcher had twelve 5th grade level classes in the semester, so one class was chosen in order to get in-depth information about the pair programming experiences of the students. The first reason behind choosing the particular class from the 12 classes was that there were fewer students than other classes, which enables more effective observation and care for individual students. The second reason why 5th grade students were chosen as participants was that the 5th grade students have not attended to the programming course before. Since they did not have any programming course before, the relationship between the pair programming technique and their achievement and confidence on programming could be observed effectively. According to the result, 32 (91%) students did not have any programming experience before and only 3 (8.6%) students had experience before the programming lectures according to the school records and students' self-reported information.

3.4.1 Participants of Confidence Scale

The confidence scale was conducted to the same 35 5th grade students in the Yenikent İlksan public school. Two more students joined to the class in the middle of the semester. Since the intervention had already started, their data were not included in the results. Among 35 students, there were 11 groups which consist of 2 students in each group and 13 individual students. The scale was implemented twice and in the second implementation there were 35 students who filled the scale.

Demographic information of the participants was achieved with the result of the confidence scale form. The number of female students was 19 (54.3%) and the number of the male students was 16 (45.7%). The age of the participants was between 10 and 11. The 14 students were 10 years old (40%) and 21 students were 11 years old (60%).

3.4.2 Participants of Interview

The interviews were conducted to 20 students out of 35 5th grade students. The students are the same with the confidence scale participants but only 20 of them volunteered to join the interview. The numbers of individual interviewer participants were 7 students and their genders were both male and female. Moreover, the numbers of pair-group interviewers were 13 and they were also both male and female. The age of the students was between 10 and 11. These 20 interview participants' and not-interviewed students' rubric scores were similar, and they were similar in terms of the composition of gender and age (35 students' average rubric score M = 72.02, 20 students' average rubric score M = 71.85).

3.5 Implementation

3.5.1 Implementation of the Course

Implementation process of this study consists of five main steps. Each step is explained below.

Step 1: (Week1)

As stated above, there were two participant groups (pair or individual) in this research. Before starting to programming activities, the teacher informed the students about the curriculum of the course for the semester. Teacher also informed the students about the rules of the computer laboratory and pair programming technique. After the necessary information about the process of course was given, the students were randomly assigned to pair groups or individual. Moreover, the computers that they study throughout the semester were assigned to them and not to be changed.

After the design of the classroom determined teacher informed the students about how they implement the programming activities with their partners or individually. According to this rule, students were applying the given activity in two ways. For the individual programmers, they completed the activity by themselves without receiving any help of their teacher and classmates. The individual novice programmers worked alone. They only used the knowledge and example activities that gained in the lessons. On the other hand, the pair-programming groups worked as a group. They shared their experience in the light of the information that gained in the programming lessons. Similarly, they also received no help from the teacher throughout the

activity. The teacher answered students' questions about their activities after the lesson, started evaluating the activities using the rubric, and gave feedback for their activities at most within one week duration after they complete their activity. The rules and structure of the pair programming technique as also mentioned to the students that were;

1. The individual student worked alone but the pair programming students worked together on the same computer and they complete the given activities together.
2. The roles of the students in pair programming were assigned as driver or navigator in the first week but the role of the students changed every two weeks during the implementation. The students in the role of "driver" created code blocks and used computer to complete the activity. The driver also talked with the partner for the suggestions and the mistakes of the code blocks. On the other hand, the "navigator" observed the code screen and give suggestions about code blocks. Both driver and navigator always communicated each other and worked together during the programming process.
3. The navigator and driver decided the codes and design elements together. If one of the students did not obey the rules, the group member could inform the teacher immediately.
4. The pair programming student got point from the programming activities that they completed together so they needed to work together and create better codes.

Since their completion time of the activities was different from each other in the class, students were allowed to create different projects or examine the Scratch projects in the software. The computer game or surfing in the Internet was forbidden during this time. Teacher also shared the measurement tools for grading of the students. The structure of achievement rubrics was described to the students and the grading policies of the activities were presented.

For record keeping and tracking students' performances in each activity, the teacher prepared sheets that include the list of the students in terms of pair programming students and individual programming students so she added the grades of the students easily.

Teacher evaluated their works with the rubric in each activity throughout the semester. The rubrics were used in the study for all of the weekly activities of the lesson. Each activity has a rubric and teacher of the class evaluated the programming activities by using the items on the rubric. Since the rubric criteria were presented on the board and the teacher explained to the students, they knew how they get points or lose points.

Step 2 (Week2-Week3-Week4 of the Course)

The lessons plans 1 and 2 and 3 were implemented with the students in week2, week3 and week4. The programming activities included basic code blocks about the Scratch programming environment. The lesson plans for the activity, topics, and rubrics were taken from the official web site of the Scratch programming environment which is prepared open source platform for the programming lessons of the students. Before starting the activity part of each lesson, the teacher implemented some example projects with their students. The teacher assessed the performance with the rubric which was also taken from open source platform of Scratch. The rubrics were presented on the smart board. Thus, students could see the evaluation criteria in this way.

The first three lessons plan focused on the code blocks applications of programming. The design of the Scratch programming environment was not important for the three programming activities. The students got point with the code blocks usage. The students created similar code platforms in the second and third programming activities.

In the 3rd programming activity, the teacher firstly evaluated the students' works according to rubrics. Moreover, another teacher came to the classroom and she also evaluated the students' activities according to rubrics. Two instructors were evaluated the programming activities to provide reliability of the study. The teacher accustomed to the Scratch projects because she also taught computer programming to her students.

After the third activity completed the first implementation of confidence scale was applied to the students. The reason why the scale was implemented in this week was that the 4th programming activity included the topic of designing the programming environments so the researcher prevented the possible confusion of the students

about programming. The researcher thought that starting the 4th activity might affect the students' responses on the confidence scale because the 4th activity did not include any code blocks. Since the students were 5th grader, they did not have any experiences about filling 5 point Likert Scale. Thus, the teacher explained them how to fill the scale.

Step 3 (Week5-Week6-Week7-Week8-Week9-Week10 of the Course)

In week 5 to 9 the lesson plans of Scratch were implemented to the students weekly. The lesson plans included programming objectives for each lesson. Teacher used the first exercise in the lesson plan to teach the programming concepts. After that students applied the activity which is showed on the screen of smartboard. They worked as the rule mentioned in step 1. The weekly lesson plans were prepared on the basis of basic programming concepts to complicated ones so students completed the activities different time period and quality. In the first lesson of each week, students learned different code blocks. They also gained some design skills and programming skills in each week.

In the 7th week of the course, the activity was evaluated using the same rubric by two teachers again. Firstly, students completed the activity of 7th week and teacher1 measured the activity by using the rubric. After that tracher2 came to the class and evaluated the students' works one by one.

In the 10th week of the course, the last activity was completed by the students and they had knowledge about several programming concepts and codes. After the 8th activity was completed, the scale was administered by the teacher at the end of the lesson. They filled the confidence scale easier than first implementation because in the first implementation, they did not know how they to fill the 5 point Likert Scale. At most, the students completed the scale within 30 minutes.

Step 4: (Week11, Week12 of the Course)

In the 11th and 12th week of the semester, the teacher continued Scratch activities with the students to reinforce their knowledge and skills in programming and provide feedback for any difficulties the students were having. In these weeks, the teacher conducted the interview protocol of the study. The interview time was determined according to schedule of the classroom. The time of the computer literacy course was the last two hour for the class so the teacher interviewed the students after the course

ended. The computer laboratory and libraries of the school were used for the interview. The interviews were conducted with individual students.

Step 5: (Week13 of the Course)

In the last week, teacher summarized the programming activities weekly by creating a pong game in the Scratch programming environment and the students played the game that they created in Scratch.

3.5.2 Implementation of Pair Programming Technique

The pair programming technique required students to work together on the same computer, keyboard and mouse during the programming process. Each pair had roles "driver" or "navigator" and the roles of the students could be changed different time periods while creating codes. Changing roles in the pair programming technique provided to gain different experiences in computer programming. According to Rostaher and Hericko (2002) the frequency of changing the roles in pair programming depended on the programming experience of the students. In the present study, the time of changing roles was determined as 2 weeks for pair programmers because the 5^{th} grade students had no experience about programming and pair programming technique before. Working the same role in two weeks facilitated to implement pair programming technique in this age group. In the study the pair programming technique was implemented in the course as follows;

1. The role of the driver and navigator was assigned to the pair programming groups. The teacher reminded the students that they had to switch their roles every two weeks. They worked together by communicating each other and they had to be active in programming process.
2. The teacher observed the adaptation of the pair students in the first week. Since the pair groups were assigned randomly, the observation was necessary for seeing that each pair worked accordantly.
3. The teacher observed all of the students in the class whether they worked for programming activities or interesting in doing other things in the lesson. The teacher also checked the role of the students by using a student list weekly.
4. To prevent the attendance problem of students in pair programming, teacher reminded the students that they will lose points when they do not attend to the class.

5. The students did not get any help from their teachers during the activities so they had to work individually or with their partners. Teacher gave feedback to their students after the programming activities completed.
6. The teacher reminded the students about importance of learning the usage of code blocks in activities because the usage of code blocks helped them in their project work and their exam. The passive students could be reported by their partners immediately. This was also beneficial to prevent the unbalanced workload among the pairs in pair programming technique.
7. The pair programming students were informed that they needed to work with their partner to achieve a given goal. If one of the pairs did not care about the activity, the teacher reminded them that both of the students had to take the same responsibility.
8. Teacher also stated that she will not answer the questions of the student in programming process; the pair programming students had to talk each other to achieve the solution to their problems. The role of the teacher in pair programming technique implementation was encouraging the students and observing them during the programming process.

3.6 Data Collection Instruments

In the study, there were different types of instruments to collect data for case study research. Yin (2003) suggests that several types of data collection sources for the case study research such that documents, archival records, interviews and observation and physical artifacts. Moreover, Yin (2003) adopts the both qualitative and quantitative approaches for the case study. In the embedded case study design both qualitative and quantitative knowledge was integrated and analyzed in the research. The design also allows using both qualitative data sources like interview and quantitative data sources such as surveys and questionnaires (Scholz & Tietje, 2002).In the study, 5 point Likert confidence scale, achievement rubrics, and interview protocol were used for data collection instruments.

3.6.1 Confidence Scale

The main purpose of the scale was gaining quantitative results for the programming confidence of the students with the usage of the pair programming technique.

Wiebe, Williams, Yang, and Miller et al. (2003) developed a scale that measures attitudes towards computer science and programming. The origin of the scale is a mathematics attitude scale developed by Fennema in 1976. The scale was modified to reflect the pair-programming attitude of the university students. It had several sub-scales and the confidence was one of them. The 5 point Likert type scale includes 11 questions for the confidence variables of the study with strongly agree to strongly disagree. The confidence scale consists of both negative and positive statements. The Cronbach's alpha level of the confidence part of the scale reported as 0.91 (Williams et al., 2003). In this present research, these 11 items in the confidence scale was used.

The numbers of the items were increased by adding another confidence scale at the end of the first scale, in order to increase reliability. Moreover, since the ages of the students were small in this study, increasing the number of questions provided to get more consistent answers from the 5^{th} grade secondary school students in the study. For this purpose, the second scale was combined to the first one, given one after another. The second scale was developed by TIMMS and PIRLS (2011) international study center in Boston College for mathematic education. The items in the scale translated into Turkish by Minister of Education (MEB, 2011) and the confidence scale was implemented to 8^{th} grade secondary school students in Turkey. This scale had been implemented in several countries including Turkey and the analysis and validation made by the TIMMS and PIRLS. The results of validation of confidence scale were published in the web site of the International Study Center. The Cronbach's alpha level of the scale was found 0.87 for Turkey (TIMMS & PIRLS, 2011). The scale includes 9 questions for the confidence level of mathematics lessons of 8^{th} grade students for several countries. These 9 questions were added to the previous scale of 11 items to collect data in the present research study.

The reason of choosing the mathematic confidence scale was that the items in the scale were similar with the Williams' programming confidence scale. Both of the confidence scale included some negative items. Moreover, the origin of William's programming confidence scale was also came from mathematic attitude scale, so combining these two scales was appropriate in terms of their origins. The first improvements in programming for the children started by creating Logo programming environment for teaching mathematic (Feurzeig, Papert & Lawler,

2011). Moreover, Byrne and Lyons (2001) resulted that the strong relationship was occurred between the programming and mathematic attitudes of students that the mathematic was important priority for learning computer programming. Gaining mathematical knowledge also provided to gain stronger programming ability for secondary school girls (Wiest, 2004). Mathematic was stated as prerequisite for gaining programming skills. The studies in the literature also resulted that computer programming enhanced the mathematical thinking skills (Denner, Werner, & Ortiz, 2012; Fessakis, Gouli, & Mavroudi, 2013; S He, Chang, & Liu, 2010; Sáez-López, 2016; Kalelioğlu, 2015). These studies showed that there was strong relationship between the computer programming and mathematic for the education of children. Since mathematic and computer programming had strong relationship, the mathematic confidence scale was used for the study.

In combining the two scales, a few changes were made for the new confidence scale. First of all 11 items programming confidence scale used as in original format. On the other hand, in the 9 items scale, the only changing was done in the word of "Mathematic" because the scale was measured mathematic confidence. The "Mathematic" word was changed by "Programming" for this study. Since the language of two confidence scale was in English, the items of the instruments were translated into Turkish. In order to provide consistency on the translation of the items, the translation was checked by two English teachers who worked in the Yenikent İlksan public school and who graduated from the department of Foreign Language in METU. Back-to-back translations were checked by the content experts.

The quality of the 20 items in the questionnaire was also tested with four students with similar backgrounds and educational levels as the original participants. Using Think-aloud procedure, students were asked to read each item and tell the researcher what they understood from the item and gave feedback about whether the item was clear. The content validity of the instrument was further checked from two instructors who have expertise in the field of Computer Education and Instructional Technology department in METU. For the scale, the items were examined by the faculty and they give their opinion about whether the items in the scale are representative or not. The items in the scale did not changed and the experts stated that the scale was representative for measuring the programming confidence. Content validity of the scale was approved in this way. Cronbach alpha of the final 20 items

of 5 point Likert confidence scale was also calculated. The Cronbach alpha was ranged between 0.81 and 0.88 for two implementations.

3.6.2 Interview Protocol

The semi-structured interview protocol was designed according to the needs of the research. The aim of the semi-structured interview of the study was getting in-depth information about programming confidence and programming achievement level of students on the experiences of the pair-programming technique in Scratch programming environment.

The questions of the interview were prepared for two different sections as 1) individual programming participants' interview and 2) pair-programming participants' interview. The interview questions were asked to the participants to gain in-depth information about the various experiences of individual and pair programmers in 8-week implementation regarding their confidence and achievement level in programming. The Think-aloud procedure was used for testing the prepared questions. The questions were asked to the 4 students from 6^{th} grade programming class students in order to check the understandability of the questions. The students reported what they understood from the questions. Then, the interview questions of interview were revised according to students' feedback and responses.

There were 5 main questions for pair and individual groups' interview protocol. The questions included both achievement and confidence. The content of the questions in two different interview sections (pair and individual) were matching with each other. The only difference was the issue of working in pairs or individual while doing programming. The results of the interviews were integrated and compared with other data about students' confidence (collected by the questionnaire) and achievement (evaluated by rubrics). In the interview, the students were asked their own opinions about their confidence and achievement.

The questions were checked by one computer science teacher who is working in the same school of Yenikent İlksan public school and a faculty in the university in order to provide content validity of the interview. Provided feedback was about revision of some sentences for easier understanding according to students' age and level, making some questions more open-ended, and replacing the possibly biased sentences. The final version of interview questions was analyzed for the content validity by two

content experts again in METU. The Human Subjects Ethics committee of the METU was also approved the interview protocol. Since the ages of the participants were small, the permission for conducting the interview was taken from the parents of the participants. The verbal permission was also taken from the participants before starting to interview.

3.6.3 Lesson Plans and Rubric

In the study, the programming lessons were taught according to lesson plans which were developed by Irish Software Engineering Research Centre. The Centre has been working for the students to encourage students to develop and discover computer programming and software development in Scratch learning environment since 2007.

The lesson plans were prepared from simple programming concepts to difficult ones. Each lesson plan included topic of the lesson, learning objectives, teacher tips, introduction part, programming concepts for the lesson (description of the code blocks and usage of them), challenge time1 and 2 and ultimate challenge time. The lessons plan provided making several exercises on the programming software. Thus, the students could gain a lot experience in their programming lessons.

To make a valid measurement for the performance of students in programming lessons, the rubrics were used as a measurement tool. The items in the rubric were also designed by Irish Software Engineering Research Centre. The measurement items were constructed according to sequence on the lesson plan. Thus, the achievement of the students on the activities could be measured with this tool accurately. The rubrics were translated into Turkish because to conduct inter-observer reliability, another teacher was asked to use the rubrics and the teacher did not know English. The items of the rubric were translated and checked by two experts in the field of Foreign Language Education-English. The items were also checked together with the other computer teacher to make sure the items were understood by the teachers as the same way. Finally, a content expert in the field of Computer Education and Instructional technology reviewed the rubrics and improvements were made.

3.7 Data Collection Procedure

Table 3.1 *Data Collection Process*

Course Week	Implementation	Rubrics	Other Data Collection
Week 1			
Week 2	Implementation Scratch Lesson 1	Rubric 1	
Week 3	Implementation Scratch Lesson 2	Rubric 2	Second Teacher Evaluation
Week 4	Implementation Scratch Lesson 3	Rubric 3	Confidence Scale
Week 5	Implementation Scratch Lesson 4	Rubric 4	
Week 6	Implementation Scratch Lesson 5	Rubric 5	
Week 7	Implementation Scratch Lesson 6	Rubric 6	
Week 9	Implementation Scratch Lesson 7	Rubric 7	Second Teacher Evaluation
Week 10	Implementation Scratch Lesson 8	Rubric 8	Confidence Scale
Week 11	Scratch Activity		Interviews
Week 12	Scratch Activity		Interviews
Week 13	Summary and Game		

Data collection of the study includes two types of the data collection procedures. First procedure is quantitative data collection which uses the instruments of confidence scale and achievement rubrics. The programming confidence and achievement of the students were collected via these instruments. Second procedure is qualitative data collection procedure which collects the data via interview protocol. The semi-structured interview protocol was used since the age of the students are under eighteen. Table 3.1 summarized the data collection process.

Before collecting both qualitative and the quantitative data, the researcher consulted the Ethics Committee of the METU and got permission for applying these instruments to the students. The necessary forms for the study were prepared including parents' approval form and voluntary participation form. After the Ethics Committee gave the permission for the research, the other permission was also received from the Ministry of National Education. The reason of why the permission was taken from the Ministry of National Education is that the participants of the

study are the secondary school students in a public school. The parents of the participants also signed the parents' approval form in order to collect data from their children. All of the parents were informed about the procedures and the topics of the study in the parent teacher meeting. Thus, all of the permissions were taken for conducting the study.

3.7.1 Quantitative Data Collection Procedures

The confidence scale and the achievement rubrics were used for the collection of the quantitative data of the study.

The confidence scale: The programming confidence of the participants was measured with the instrument of the 5-point confidence Likert scale. The scale was planned to be applied two times in the semester (table 3.1). The semester included 13 weeks for the programming activities. First implementation of the confidence scale was applied at the end of the third programming activities. The reason why the confidence scale was not applied before starting the activities was that, the students were not familiar with programming and the Scratch environment. Since the students did not have any information about the programming and programming environments, the teacher applied after the third lesson of implementation, before starting the 4^{th} lesson.

In the third week, 30 minutes was used for the implementation of the instrument. The principal of the school approved the schedule change in the class schedule. The students were informed about how they fill the scale because the type of the scale was different for the students. After all of the explanation was made clear, the students filled the scale and the researcher collected the confidence forms by checking whether there were empty questions or not. Since the ages of the students were small they were inclined to give empty answers. They might lose their focus while filling the confidence form.

The second implementation of confidence scale was made in 10^{th} week of the semester after the end of the completion of the activity. The reason why the time was chosen for implementation was that all of the programming activities were completed and students had in-depth information about computer programming. The other reason behind the implementation scale was that the pair programming technique was used throughout the semester and the students experienced more programming

activities with their partners or individually. Getting more experience in the programming lessons with the pair programming technique was needed to achieve in-depth information to the students so the scale was conducted in 10^{th} week of the semester. The instrument was not applied at the end of the semester because toward the end of the semester the students tend to have fewer attendances to the school.

The time also planned by the researcher according to the curriculum of the classroom. The instruments were also implemented in different lesson by getting permission of the teacher of the lesson and school manager. The students did not need information about how they fill the scale. The scale was implemented approximately 30 minutes. For both implementations, all 35 students participated.

Achievement rubric: The programming achievement of the participants was evaluated with the instrument of the achievement rubrics which were prepared by Irish Computer Center for the Scratch learning environment. The lesson plans and rubrics were used according to structure of this Center. The achievement rubrics were implemented 8 week of the semester (from week 2 to 10). The teacher taught the programming lessons according to weekly lesson plans of the Irish Computer Center. Teacher gave some information by following the steps in the lesson plan. The lesson plans included two challenge activities so the teacher made the first challenge activity of each week by informing the students about the usage of the codes. The second challenge activity was completed by the students who worked as pairs or individuals in each week.

The pair groups and individual working students were assigned at the beginning of the semester randomly. The teacher as a researcher organized the classroom environment according to needs of the pair programming technique. To get accurate and deep information about the usage of the pair programming technique, the teacher informed to their students that each student worked according to their group type and pair groups could discuss and work with each other but the individual programmers could not get any help from their classmates. The teacher did not answer the question of the students in both individual and pair students until the activity ended. After the activity of the lesson completed, the teacher answered their questions and gave feedback to the students.

At the beginning of lesson, teacher presented the activity to the students by using the smart board in the computer laboratory. The rubric of the activity presented on the board and the teacher informed the students 10 minutes about the items of rubric and the programming activity. After all the information was given to the students, teacher gave 30 minutes for students to complete the activity. During the time teacher observed their students. Since the time of the lesson was 40 minutes, the teacher used their 15-minutes break time for evaluating the students' performances. In order to make accurate evaluation teacher saved the activity files of the students and she re-examined them later in detail. Their files were collected as a name and week number. For example, pair groups save their activity as membername1_membername2_week1. This type of saving of the project facilitated the evaluation of the teacher. The attendance problem occurred while collecting data in some weeks of the implementation. However, the numbers of un-attending students were only 4 students (2 individual students and 2 pair students) throughout the semester and they did not attend the programming activities a few weeks during the semester.

3.7.2 Qualitative Data Collection Procedures

The interview protocol was applied at the end of the implementation procedures to explore students' experiences regarding the research questions of this study. To make the interview more comfortably and to give the sincere answers from the participants, small-talk and warm-up questions were used and the teacher made sure that the students are relaxed and comfortable answering the questions. Although the participants were accustomed to their teacher (or researcher), the interview started with the some talks to create comfortable interview environment. After this point, the aim the interview and the general information about the interview process were explained to the participants. The interviews were recorded with the digital voice recorder on the condition that the permission of the participants. During the interview, it was observed that the students were not bothered about the voice recorder.

The interview questions of the study were separated into two groups which were individual interview questions and pair-group questions. The questions were asked according to group type of the students. The questions for both group of interview protocol aims to get in-depth information about the confidence and achievement of

the programming skills of the secondary school students. The interview was implemented in an empty and silence rooms in the school (Computer laboratory and library) to record the quality voice and achieve the deep information about the study.

Since the interviews were conducted in the last two weeks of the semester, the teacher planned the interview hours according to the volunteer students' position in the attendance sheet. Although the number of the students in the classroom was 35, in week 11 and week 12, only 20 of these students were accessible and volunteer for the interview of the study. Each of the interview session took approximately 8 minutes. Summary of the interviews demonstrated in Table 3.2. The absent students' and the interviewed participants' rubric scores were compared and the results showed that these students are not different. The average score of 35 students on rubrics was M=72.02 and 20 interview participants' average rubric scores was M= 71.85.

Table 3.2 *Summary of the Interviews*

Student	Group (P) or individual (I)	Gender	Age	Duration	Place
St1	I	F	10	09.19	Computer laboratory
St2	I	M	11	06.45	Computer laboratory
St3	I	F	11	07.11	Computer laboratory
St4	I	M	11	06.22	Library
St5	I	M	11	06.43	Library
St6	I	F	10	08.08	Computer laboratory
St7	I	F	11	08.03	Library
St8	P	F	11	09.58	Library
St9	P	F	11	07.04	Computer laboratory
St10	P	F	11	06.37	Computer laboratory
St11	P	F	10	05.30	Library
St12	P	F	11	06.09	Library
St13	P	F	11	06.16	Computer laboratory
St14	P	F	11	04.10	Computer laboratory
St15	P	M	11	06.15	Computer laboratory
St16	P	M	10	06.11	Computer laboratory
St17	P	M	10	05.02	Computer laboratory
St18	P	F	11	04.42	Computer laboratory
St19	P	M	11	05.37	Computer laboratory
St20	P	F	10	04.18	Library

3.8 Data Analysis

In this part of the study the researcher analyzed the qualitative and quantitative data which were collected from different type of the instruments. To analyze the qualitative data, content analysis was used as the data analysis method. On the other hand, independent sample t-test, Mann Wilkson-U test, and descriptive statistics were used as the data analysis method for quantitative data.

3.8.1 Qualitative Data Analysis

Content analysis was utilized for analyzing qualitative data obtained from interview of pair programmers and individual programmers. In content analysis, the steps proposed by Yıldırım and Şimşek (2008) were followed. They are coding the data, searching for themes, organizing the data according to codes and themes and defining, interpreting the findings and producing the report (Yıldırım & Şimşek, 2008).

Before started to analyze the data coming from the interview of 5th grades programmers, the voice records of the interviews were transferred to the computer and the researcher checked the quality of records one by one. The recordings were listened several times to transcribe the data accurately. The sentences of the students were transcribed and read by the researcher to understand the participants' answers clearly. According to Yıldırım and Şimşek (2008) the content analysis requires the achieve codes and themes to understand the meaning of the data. The researcher listened the records of the students and read transcriptions to create the codes for the data. After that the codes were classified according to their relationships in the data set so the themes were created by the researcher by checking whether the meaningful code groups were under the appropriate themes or not. Then the themes and codes were organized and defined by the researcher in terms of programming achievement of pair and individual groups and programming confidence of pair and individual groups. The results of the content analysis were reported by the researcher in the result part by giving the comments of participants in the form of Turkish and English.

As presented in the "Quality of Research" section below, part of the qualitative data were analyzed by two researchers separately, themes and categories were developed, and they were compared and combined to increase the reliability of the qualitative data analysis results. Among 20 interviews, 2 interviews were inter-coded (10%). The researcher, then, continued analyzing the rest of the data and made revisions with the introduction of each new student's interview data. The saturation has been achieved for the pair programming students' (N=14) interview while examining 8th student's data. The saturation has been achieved for the individual programmers' (N=7) interview while examining 4th student's data that the categories and themes

were defined and clear, and there were no new data coming from the rest of the interviews.

3.8.2 Quantitative Data Analysis

Quantitative data obtained from confidence scale and achievement rubrics were analyzed according to properties of data type.

The items in confidence scale included both negative and positive items. The negative items in the confidence scale coded by reverse coding to validate the items in confidence scale. In confidence scale, Q6, Q7, Q8, Q9, Q10, Q13, Q14, Q16, and Q20 were negative items. While implementing reverse coding for these items, the points of the items were changed with the points of (1-5), (2-4), (3-3), (4-2) and (5-1) for the 5 point Likert type confidence scale. After this, Independent sample t-test was used in order to see the significance difference of programming confidence of pair programming students and individual programming students for two application of pre-test and post-test.

The data collected from the achievement rubrics in 8 weeks programming activities were analyzed with the independent sample t-test and Mann Whitney U-test statistical method to understand whether there was a meaningful difference between the groups. Both of parametric and non-parametric tests were used for programming achievement variable because of violation of the normality assumptions to confirm if they provide the same results.

This study is not experimental and the quantitative data were used only to supplement and triangulate the results of the interviews. The intention of this research is not to strictly differentiate two groups, but rather to understand their experiences of being in a classroom with both individual and pair students working on the same activities under the same conditions. Most importantly, the research intended to understand the advantages and disadvantages of using pair-programming technique on pair students. Therefore, the analyses of quantitative data were only interpreted in the light with interview data.

3.9 Quality of Research

The validity described as “meaningfulness, appropriateness and usefulness of the inferences researchers make based specifically on the data they collect, while

reliability refers to the consistency of these inferences over time, location, and circumstances" (Fraenkel, 2012, p.458). Moreover, the reliability is defined as "repeatability and consistency of the findings over time" (Twycross &Shields, 2004, p.36).

The term of validity and reliability are used in the quantitative research (Creswell, 2007). In the qualitative studies the term "Trustworthiness" is used as alternative terms of validity and reliability (Lincoln & Guba, 1985). According to Lincoln & Guba (1985) the terms of Trustworthiness includes credibility, transferability, dependability and confirmability. The terms of credibility, transferability, dependability and confirmability named in the quantitative study as internal validity, external validity, reliability, and objectivity (Merriam, 2009; Marshall & Rossman, 1999).

Since the method of the study was case design, the terms trustworthiness was used for checking the quality of the study. Each of the terms under the trustworthiness includes criteria for providing the quality of the study.

3.9.1 Credibility

The credibility refers to internal validity in the quantitative study. Miles and Huberman (1994, p.5746) defined the credibility issue as "confidence in the 'truth' of the findings". To provide the credibility issue, Creswell (2003); Lincoln and Guba (1985) suggested some procedures including that prolonged engagement, pair debriefing, and triangulation.

Prolonged engagement is a procedure that researcher spends adequate time with the participants in the social and cultural context of the phenomenon to facilitate constructing relationship between the researcher and participants (Lincoln & Guba, 1985). In the study, the researcher was also the computer literacy teacher of the students and she spent the two hour of each week throughout the two semesters. At the beginning, the students and teacher did not know each other closely but after a 4 week the students and teacher accustomed to each other. The researcher (as teacher) actively participated every session of the lesson with the participant and the data were collected in the second semester to gain the trust of students and to collect more information about the properties of participants and the environment during the first

semester. Researcher was also active in the development and design of the lesson during the two semesters.

Triangulation is a procedure that combines different data sources to increase the deep understanding of phenomenon (Miles & Huberman, 1994). In the study, the confidence scale, interview, and achievement rubrics as various data sources were combined to get deep understanding of the case.

Pair debriefing is a procedure that a pair or pairs who knows the subject and the methods of the study and ask question about the meaning of the study, method, and data analysis process to make the researcher interpretation honest and expand the researcher horizons (Lincoln & Guba, 1985; Merriam, 2009; Glesne & Peshkin, 1992). In the study three pairs contributed to the study. One of the pairs is MS students in the Computer Education and Instructional Technology department in METU, the other one is Ph.D. student in the same department. Third pair is computer literacy teacher in the Yenikent İlksan secondary school. All of the pairs knew the method and the topic of the study and periodically they asked the questions about the study.

3.9.2 Transferability

Transferability refers to applicability or transferability of the result of the study in other settings. The term transferability refers to external validity in quantitative study (Merriam, 2009). Since the case study explores a phenomenon with bounded system, the result is not appropriate for the generalization. Although the transfer or generalization of the case study findings is limited, Marshall and Rossman (1999) stated that the results of the case study can be generalized to different research studies. Transferring or generalization of the qualitative result of study in other context is possible with the technique of thick description. Lincoln and Guba (1985) presented a thick description technique for providing transferability of the qualitative study. Thick description is defined as giving detail information about the phenomenon, context of the study to facilitate the transferability of the results to the other settings of the study (Holloway, 1997).

In this qualitative research the context, assumptions, limitations, participants, data collection and analysis procedure were described in a detailed way to provide the transferability of the study to the different contexts. The thick quotations were also

provided in the results of the study. With the detailed description of the study, other researchers can decide whether the transferability of the research is appropriate for their research settings or not.

3.9.3 Dependability

The dependability refers to reliability in the quantitative study (Lincoln & Guba, 1985). The dependability was checked with inter-rater reliability, and pair examination in this study.

For the quantitate part of the study, the reliability of quantitative data was provided by checking the inter-rater's measurements. The two instructors measured the programming achievement of the pair and individual students by using achievement rubrics. The two of eight-week implementations were measured by two teachers. The similarity of first rating was found 82% and the second rating was also found 87%. According to Cohen's Kappa (1960), 0.81-1.00 is perfect agreement between inter-raters. Thus, the agreement of the inter-raters was found perfect for the study.

For analyzing the interview data, another M.S. student volunteered to conduct content analysis of the interviews. The researcher informed her partner about the aim and process of the present research. Since both the researcher and the partner were experienced in content analysis, they coded the interviews separately and developed categories and themes. As a sample, the partner analyzed 2 interviews (individual and pair student's interviews) and developed themes and categories. Then they came together to compare and combine the categories and themes. The codes were mostly similar for both coders but there were also a few differences. The similarities and differences were examined for both of the interviews and the two coders discussed on the different codes and revisions were made. Therefore, for 20 interviews, initially 2 interviews were conducted with a pair to develop the main categories and themes (10%), and the rest of the data were then analyzed by the researcher herself. If needed additional themes, categories, and sub-categories were added with each new students' data.

3.9.4 Confirmability

The confirmability refers to objectivity in the quantitative study. The confirmability issue of the qualitative study is defined as neutrality of the result of the study did not

include the biases, interests of the researchers (Lincoln & Guba, 1985). Marshall and Rossman (1999) stated that the findings should reflect the phenomenon or participants not be “fabrication” form the “bias and prejudice” of the researchers. To provide the confirmability of the case study Lincoln and Guba (1985) suggested that the reflexivity and triangulation issues.

Reflexivity issue is checked for the process of research deeply because of the bias, interests, past habits and role of the researcher may influence the findings of the results (Malterud, 2001). In this study the possible biases and researcher role are explained in the section of researcher role in a detailed way to provide confirmability of the study. Moreover, the researcher kept a diary throughout the research to record all of the steps such as participants’ selection, data collection, comments of the participants in the lectures and the comments of the researcher in the study to make sure the research has been reported accurately.

Triangulation is also a procedure for checking the confirmability of the qualitative study. Triangulation is explained as using different data sources to increase the confirmability of the study (Merriam, 2009; Miles & Huberman, 1994). Combining different data sources facilitate not only the deep-understanding of the phenomenon but also to compare the findings for more accurate representation of the results. In the study, multiple data sources such that interview, confidence scale, and achievement rubrics were used. The data obtained from several resources are compared and combined for triangulation.

3.10 Researcher Role and Possible Biases

The researcher has conducted the whole research process including implementation, data collection, data analysis, and reporting. The researcher was the computer science teacher of the class. The students were accustomed to the researcher which increased the rapport we have with each other and increased the chance of receiving honest responses from the students.

The motivation of the teacher about programming may affect the result of the measurement of the students. Since the researcher is a teacher of computer literacy, the researcher provided that every child is able to learn programming. This may cause a possible bias because every student has to learn programming and their achievement, motivation and confidence should be high. To prevent this personal

bias, instead of using the teachers' own teaching materials and lessons, the researcher used lesson materials that were designed by Irish Computer Center for Scratch learning environment. Using prepared lessons and materials by an Irish Computer Center provided more structured lessons for the research and reduced the possible biases.

The application of the instruments and the measurement process were important for the accuracy of the study. Therefore, to prevent the bias of the researcher, the teacher wrote every step in the process in a diary of preparing materials that she changed or translate in the lesson plans, rubrics and the confidence scale. Moreover, the researcher as a teacher implemented all of the plans and activities as reported in the data collection procedure section to provide accurate data collection.

As a researcher, the teacher also consulted several experts on the computer literacy lessons in the school. One of the teachers helped the researcher during the study to evaluate student's activities using a rubric for 2 weeks. The teacher's other role in the study was examining the data collection instruments to give some advice. The teacher especially gave her opinions on interview protocol and the programming lesson activity rubrics. Another M.S student helped the researcher with analyzing the interview data for inter-coder reliability.

CHAPTER 4

RESULTS

In this chapter, the results of the study are presented based on the research questions of the study. The preparation of data and assumptions, the findings of data which are 1) Programming confidence of students 2) Programming achievement of students 3) Emotions of students during programming process, comparing, combining and summarizing the result, and reflections of the researcher as the teacher during implementation are presented in this chapter.

4.1 Preparation of Data and Assumptions

The qualitative data were gathered from the interviews of pair and individual programmers. After data collection procedures ended, the researcher transcribed the interviews of both individual and pair students' records. All of the voice records listened and written textual format to facilitate content analysis process. The name of the students and their demographic information were omitted from the textual format because of the ethical consideration. The name of the individual students deleted and the students were symbolized by "S" with following id number (S1, S2, etc.). The name of the pair students were also symbolized by "P" values with following id numbers (P1, P2, etc.). The data were shared by the inter-coder by omitting the personal information. The researcher listened to the records several times to check the data transcriptions of each student. Then, the researcher prepared the Microsoft Excel sheets for the themes and categories of the students.

After the data were collected from confidence scale the 5 point Likert scale items were transferred into SPSS 22.0 file in numerical format. The achievement rubrics were also transferred into SPSS file for 8-week implementation. The data of both achievement rubric and confidence scale were checked by the researcher before analyzing the data. The missing values of the confidence scale and achievement

rubrics were also checked before the analysis. The researcher checked the missing values by using descriptive statistics. There was no missing value in the confidence scale data set. However, the achievement rubric data set included some missing data. There were 4 participants who did not attend to the class in one of the weeks out of 8 weeks. The researcher used single imputation method that was replacing the missing value with sample mean for these 4 participants.

The Independent sample t-test was used for the analysis of the quantitative data. According to Green and Salkind (2004) the statistical test of Independent Sample T-test required some assumptions including normality distribution of test variables, homogeneity of variances. The homogeneity of variances for confidence independent variable tested by the Levene's Test (See Table 4.4). The result showed that the homogeneity of variances was provided with the values of F= .10, and p= .74. The result showed that the variances were equal for confidence variable. However, the homogeneity of variances for achievement independent variable was violated with the values of F= 6.72, and p= .01.

The normality assumption was tested by Shaphiro- Wilk test. The S-W test suggested that the dependent variables (confidence, achievement) were approximately normally distributed for two dependent variables. The Table 4.1 showed the result of Shaphiro – Wilk test. Moreover, the Q-Q plots and histograms were also checked for the normality assumptions. The normality was provided by the histograms and Q-Q plots (See Appendix A).

Table 4.1 *Tests of Normality*

		Shapiro-Wilk		
		Statistic	df	Sig.
achievement				
	individual	0.87	13	0.06
	pair	0.78	22	0.00
confidence2				
	individual	0.95	13	0.74
	pair	0.92	22	0.08
confidence1				
	individual	0.89	13	0.12
	pair	0.98	22	0.99

Note. Confidence1= first implementation of confidence scale; Confidence2= second implementation of confidence scale)

According to Shapiro Wilk test results in Table 4.1 the pair independent variable had the value of p= .00, therefore the normality assumption was violated for achievement dependent variable. However, the normality assumption was provided for the individual programmers in the same achievement variable. According to Green and Salkind (2004) the "data for an independent-samples t test can also be analyzed by using nonparametric procedures" (p.173). Moreover, when the normality assumptions were not met, the non-parametric tests might be more powerful (Green & Salkind, 2004). Since one of the independent variable was normally distributed but other one was not, the non-parametric Mann Whitney U- Test was also conducted to support the result of Independent t test for the achievement dependent variable.

4.2 Programming Confidence of the Students

4.2.1 Quantitative Results for Programming Confidence of the Students

The result of the programming confidence was presented in this part of the study. The descriptive statistics of each item was showed in Table 4.2. According to result of descriptive statistics for programming confidence variable, the mean score of first implementation confidence scale ranged from 1.77 to 4.54 for pair students and 1.53 to 4.61 for individual programmers. Item 2 (see Table 4.2) was the highest mean (M= 4.54) that was about "I am sure that I can learn programming". The pair and individual students' means were differentiated in item2. While the pair students believed more that they can learn programming, the mean of the individual students was lower than pair programming students for this item. Moreover, in the descriptive results of the programming confidence variable the confidence of solving complex programming problems evaluated in items (item3 and item 17). The results suggested that individual students' mean scores were closer to pair students' in the first implementation of confidence scale. However, in the second implementation individual students' mean scores decreased for the item3 and item17. The individual students' confidence decreased in the solution of complex programming problems throughout the 8 week implementation. In the first implementation of confidence scale the mean of positive items mostly high for the pair and individual students but the individual students' mean scores decreased with the second implementation. In general, pair programming students' means were higher than individual students' in both implementations. At the first implementation, pair and individual students felt confident about taking good grades from programming but the means of individual

programmers decreased at the second implementation (item4). The confidence of the pair and individual students were higher at the beginning, in the item 5 the means of the both pair and individual students were high but in second implementation the confidence of both students decreased. The decrease of individual students' mean was higher than pair programming students for the item 5. According to descriptive statistics in Table 4.2, the total means of both pair and individual students were closer to each other in the first implementation. However, the means were differentiated in second implementation. The confidence scale total mean for pair students increased but the individual students' total mean score decreased at the second implementation.

Table 4.2 *Descriptive Statistics for Programming Confidence*

Items	Pair-Programmers		Individual Programmers	
	Confidence 1 *M (SD)*	Confidence 2 *M (SD)*	Confidence 1 *M (SD)*	Confidence 2 *M (SD)*
1. Bilgisayar alanında üst düzey çalışmalar yapabileceğime eminim. (I am sure that I could do advanced work in computer science.)	3.86 (0.71)	4.40 (0.66)	4.15 (0.89)	3.30 (0.85)
2. Programlamayı öğrenebileceğime kesinlikle eminim. (I am sure that I can learn programming.)	4.54 (0.85)	4.63 (0.58)	4.61 (0.76)	3.76 (0.92)
3. Daha zor programlama problemlerinin üstesinden gelebileceğimi düşünüyorum. (I think I could handle more difficult programming problems.)	3.68 (0.94)	3.90 (0.75)	3.84 (1.14)	3.07 (1.18)
4. Programlama dersinden yüksek not alabilirim. (I can good grades in programming course.)	4.04 (0.84)	4.54 (0.50)	4.15 (1.06)	3.61 (0.65)
5. Programlama konusunda kendime güvenim oldukça yüksektir. (I have a lot of self-confidence when it comes to programming.)	4.40 (0.73)	4.27 (0.93)	4.23 (1.01)	3.76 (0.92)
6. Programlamada başarılı değilim. **(*)** (I am no good at programming.)	1.95 (1.04)	2.31 (1.46)	1.53 (0.66)	2.30 (1.18)
7. Üst düzey programlama yapabileceğimi düşünmüyorum. **(*)** (I do not think I could do advanced in programming.)	2.63 (1.17)	2.36 (1.09)	3.15 (1.28)	2.84 (1.21)
8. Programlamayı iyi yapabilecek bir değilim. **(*)** (I am not the type to do well in computer programming.)	1.90 (1.01)	2.22 (1.23)	2.69 (1.37)	2.23 (1.36)
9. Nedense, çok çalışmama rağmen programlama bana çok zor geliyor. **(*)** (For some reason even though I work hard at it, programming seems unusually hard for me.)	2.36 (1.49)	2.13 (1.32)	2.00 (1.35)	2.30 (1.10)
10. Birçok dersin üstesinden gelebilsem de programlama problemlerinde mutlaka hata yapıyorum. **(*)** (Most subjects I can handle, but I have a knack flubbing up programming problems.)	2.50 (1.30)	2.36 (1.36)	3.07 (1.65)	3.38 (1.19)

Table 4.2 (cont'd)

11. Programlama dersi en kötü olduğum derstir. (Programming has been my worst subject.)	1.77 (1.19)	2.22 (1.30)	1.84 (1.28)	2.15 (1.06)
12. Programlamada genellikle iyiyimdir. (I usually do well in computer programming.)	4.36 (0.90)	4.22 (1.19)	3.69 (1.25)	3.38 (1.32)
13. Benim için programlama çoğu sınıf arkadaşıma göre daha zor. **(*)** (Programming is more difficult for me than for many of my classmates.)	2.27 (1.12)	2.50 (1.14)	2.46 (1.45)	2.30 (1.03)
14. Programlama güçlü yanlarımdan biri değil. **(*)** (Programming is not one of my strengths.)	2.13 (0.94)	2.50 (1.62)	2.30 (1.31)	2.38 (0.96)
15. Programlamadaki konuları çabukça öğrenirim. (I learn things quickly in programming.)	4.09 (1.01)	4.22 (0.75)	3.69 (1.54)	3.46 (1.19)
16. Programlama kafamı karıştırır ve beni gerer. **(*)** (Programming makes me confused and nerveous.)	1.81 (0.95)	2.36 (1.25)	2.15 (1.06)	2.23 (0.83)
17. Zor olan programlama problemlerini çözmekte başarılıyım. (I am good at working out difficult programming problems.)	3.63 (1.00)	4.09 (1.06)	3.53 (1.12)	3.00 (1.15)
18. Öğretmenim zor da olsa programlamayı iyi bir şekilde yapabileceğimi düşünür. (My teacher thinks I can do well in programming with difficulties.)	4.27 (0.82)	4.31 (0.94)	4.15 (0.89)	3.38 (0.86)
19. Öğretmenim bana programlamada iyi olduğumu söyler. (My teacher tells me I am good at programming.)	3.95 (0.95)	4.18 (1.05)	3.61 (1.12)	3.30 (0.63)
20. Programlama dersi benim için diğer derslerden daha zordur. **(*)** (Programming is harder for me than any other subjects.)	2.04 (1.43)	2.00 (1.19)	1.69 (0.94)	2.38 (0.86)
Total	62.27 (7.38)	65.82 (8.01)	62.61 (7.25)	59.54 (5.72)

Note. (*) = Reverse code

Table 4.3 *Group Statistics for Confidence Scale*

		N	Mean	Std. Deviation	Std. Error Mean
Confidence					
	individual	13	-3.07	7.12	1.97
	pair	22	3.54	6.55	1.39

Table 4.4 *Independent Samples Tests for Confidence Scale*

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
	Equal variances assumed	0.10	0.74	-2.79	33	0.00	-6.62	2.36	-11.43	-1.80
	Equal variances not assumed			-2.73	23.62	0.01	-6.62	2.41	-11.62	-1.62

Independent Sample T-test was conducted to compare the means of pair programming students' and individual programming students' confidence. The mean differences of confidence scale from first implementation to last implementation were also compared with Independent Sample T-test. The independent variables include two levels: pair programming students (n=22) and individual programming students (n=13). Table 4.3 included group statistics information of the programming confidence dependent variable. There was a significance difference in the scores of pair programmers' confidence M = 3.54 (SD = 6.55) and individual programmers' confidence M = -3.07 (SD = 7.12) with t (33) = -2.79, p = .00. The 95% confidence interval for the difference in means was ranging from -11.43 to -1.80. The Cohen's d effect size calculated was 0.96. This was large effect size of the result because the effect sizes are interpreted as small (0.2), medium (0.5) and large (0.8) (Green & Salkind, 2004).

4.2.2 Qualitative Results for Programming Confidence of the Programmers

According to content analysis result of the interviews with 5th grade pair-programmers, most of the students felt more confident with their partner during the programming activities. The programming confidence of the students was high when they worked with their partners during the programming lessons. Three themes were constructed according to the responses of the students about the programming confidence of pair programmers and individual programmers such that a) problem solving process, b) programming process c) Being in a pair or individual during the process. The Table 4.5 showed the themes and the categories of programming confidence interview result of programmers.

It should be noted that this research gives more priority to understanding the unique experiences of pair-programmers. Individual programmers' experiences were also important to understand having the shared environment with pair-programmers while engaging in the same tasks as baseline information and to compare with the pair-programmers' experiences. Moreover, this research intends to portray the experiences of both groups with the compositions of both pair and individual students engaging in the same tasks in the same classroom and to understand whether the implementation of 8-weeks influenced their opinions of their programming confidence and achievement.

Table 4.5 *Possible Factors that Influence Programming Confidence (NP: Pair, NS: Individual, NT: Total)*

PEER			INDIVIDUAL			
Categories	Participants Mentioned	NP	Categories	Participants Mentioned	NS	NT
Problem Solving Process			**Problem Solving Process**			
Sharing knowledge	P1,P2,P4,P5,P6,P8,P10,P11,P12,P13	10	Sharing knowledge	0	0	10
Helping each other	P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13	13	Helping each other	0	0	13
Finding solutions (easy)	P1,P2,P4,P5,P6,P10,P11,P13	8	Finding solutions (hard)	S1,S2,S3,S4,S5,S6,S7	7	15
Correcting mistakes	P1,P2,P3,P4,P6,P7,P8	7	Correcting mistakes	0	0	7
Source of knowledge for solving problem	0	0	Source of knowledge for solving problem	S1,S4,S5,S6,S7	5	5
Programming Process			**Programming Process**			
Task completion time (fast)	P3,P4,P5,P8,P11,P12,P13	7	Task completion time (slow)	S1,S2,S3,S5,S6,S7,P2	7	14

Table 4.5 (cont'd)

Quality of product (high)	P4,P6,P8,P11,P12	4	Quality of product (low)	S1,S2,S5,S6,S7	5	9
Learning of programming (high)	S5,P1,P4,P3,P6,P7,P9,P10,P12	9	Learning of programming (low)	S1,S4,S6,S7	4	13
Motivation (high)	P4,P5,P6,P8,P12	5	Motivation (low)	S5,S6,S7	3	8
Being in a pair or individual during the process			**Being in a pair or individual during the process**			
Disagreements	P1,P2,P3,P5,P6,P9	6	Disagreements	0	0	6
Programming ability differences	P1,P2,P3,P5,P6,P9,P10	7	Programming ability differences	0	0	7
Workload	0	0	Workload	S1,S2,S3,S4,S5,S6,S7	7	7

Note. NS= Number of individual programmers, NP=Number of pair programmers, NT= Total number of pair and individual programmers.

According to the content analysis results of programming confidence, problem solving process is one of the themes that include 5 categories. The pair programming participants reported that they shared their experience with their partner during the programming process. One of the core topics in their responses was the programming problems and how they solve their problems with their partners. The students mentioned that they encountered several problems while completing the programming activities. Moreover, individual programmers had also several problems during the programming process. Therefore, Problem solving in individual programming is one of the themes of the content analysis. The theme included 5 categories including that sharing knowledge, helping each other, finding solutions, correction of mistakes and source of knowledge for problems.

Most of the pair students (N=10) responded that they share their knowledge with their partners in order to solve problems of the activities. Sharing knowledge with the partner provided more confidence to the programmer according to responses of them. Sharing knowledge provided to help them get new strategies for the problem solving of computer programming activities. On the other hand, individual programmers said that they encountered some problems in programming but they did not have any chance to sharing their knowledge with some friends. Some individual programmers stated that they worked alone and they could not share their knowledge, so most of time they could not manage the activity accurately. Not finding the solution of the problems made them unconfident during the programming activities. The participants stated that;

> "My friend contributed me a lot. My friend showed me the codes that I could not find and lacking information about the codes. We are sharing information therefore we are successful. We shared our knowledge and we could solve the problems easily." (P4)

> *"Arkadaşımın katkısı çok oldu. Bulamadığım kodları (ıuıı) bilmediğim bilgileri (ıuıı) bana o gösterdi. Bildiğimiz bilgileri paylaşıyoruz ve böylece başarılı oluyoruz. Bilgilerimizi paylaştık ve sorunları kolayca çözebildik." (P4)*

> "Scratch programming environment was a little bit hard for me because it contains mostly codes. If I had had a friend working with together, we would work by speaking, thinking etc., this would be easier for me. …The following

programming activities were difficult for me so my confidence decreased. … Now, I wish I had a friend beside me; I would finish the activity quickly and produce better." (S2)

"Scratch programı benim için biraz zordu. Çünkü hep programlama var. Yanımda arkadaşım olsa biraz daha şey olurdu ikimiz konuşarak sonra düşünerek falan yapardık, benim için daha kolay olurdu. ... İleriki haftalara doğru çalışmalar zordu ve özgüvenim azaldı. ... Şimdi yanımda birisi olsa daha çabucak bitirir, iyi yapardım." (S2)

Some of the pair programming students (N=13) responded that they helped each other during the programming activities. Helping each other facilitated to complete the programming activities of students faster and easier. They said that getting help of a friend influenced their confidence positively. Individual programming students mentioned that they could not get any help because they had to work individually. They needed help while creating code blocks. In some activities, the sequences of code blocks were complicated for the student so they claimed that they needed help of a friend or a teacher in that time. The participants said that;

"We were helping each other in coding. For example, we could not know the sequence of the cod blocks. When I stepped forward 10 steps my friend stopped me when she came to near me. My group friend helped me. She assisted me to place the codes into correct places … She listened to you carefully in the lesson so he helped me. The help of my group friend increased my confidence. …I prefer working with a group member because she supported me during programming process." (P6)

"Kodlamalarda arkadaşımla yardımlaşıyorduk. Kodlamalarda mesela (ıuıı) yerlerini [sırasını] pek bilemiyorduk. Ben 10 adım ileri derken o kenara geldiğinde dur diyordu. Grup arkadaşım bana yardımcı oluyordu. Kodların yerini yapmamda yardımcı oldu. ... Derste sizi iyi dinliyor o yüzden bana yardımcı oldu. ...Grup arkadaşımın yardımı özgüvenimi arttırdı. ... Grup arkadaşımla çalışmayı tercih ediyorum çünkü o bana destek oluyor." (P6)

"When I encountered problems in programming, I had difficulties in solving the problems in code blocks. …I deduced from the samples which we had

learnt during our lessons. In some studies I needed more resources. Sometimes, I really had difficulty. In fact, if I had had a group friend, I would have done better. I think it would be better. When I could not get help, my self-confidence decreased." (S7)

"Problemle karşılaşınca çözmek için kodlarda çok zorlandım. ... Kendime örnekler çıkardım derste öğrendiklerimizden. Bazı çalışmalarda daha fazla kaynağa ihtiyaç duydum. ... Bazen gerçekten çok zorlandım. Aslında grup arkadaşım olsa daha çabuk yapardık, daha iyi olurdu bence. Yardım alamayınca özgüvenim düştü." (S7)

The pair programming students (N=8) stated that finding solution of the problems of programming activities is one of the important factors that influence their confidence. Solving the problems with their partner included several factors; trial error and brainstorming with a partner. According to their responses, when they worked with their partner they managed the programming activities easily. They could solve the problems by trial and error or making brainstorming with their partner to find the solution to programming problems. On the side of individual programmers, some students (N=7) stated that they had several difficulties on programming activities. One of the difficulties of them was hard to find solution to the problems. They said that problem solving was one of the biggest problems for them. The students said that;

"*I and my friend sometimes had difficulty, we were confused, we fought, we said let's not do it. We did not quit when we had trouble. While solving problems* we talked to each other how the code blocks were added and I said "OK Let's do it". My friend supported me whether it was good or not. My confidence increased with the help of the friend." (P5)

"Arkadaşımla bazen zorlandık (ıuuu) bazen karıştırdık, kavga ettik, olmaz yapmayalım dedik. Sorun yaşadığımızda pes etmedik. Sorunları çözerken… Arkadaşımla şöyle yapalım böyle yapalım diye konuştuk, bende tamam yapalım diyordum. Grup arkadaşım güzel olsa da kötü olsa da hep yanımda oldu. Özgüvenim arkadaşımla bu sayede arttı." (P5)

"I had difficulties during programming, simple things happened as well. Group programmers could collaborate in programming activities. The programmers who understood the programming could solve the problems, activities easily. When I encountered problems, I did not quit and I listened to lessons more attentively. I tried to complete it even if I had difficulty. I worked harder. I had uncompleted programming activities because I could not solve the problems. … My self-confidence decreased towards the last weeks." (S2)

"Programlama yaparken zorlandım, kolay şeyler de oldu. Grup olarak çalışanlar işbirliği yapabildi. Programlamayı anlayanlar sorunları kolayca çözebiliyor. ... Sorun yaşadığımda pes etmedim daha çok dinledim dersi. Zorlansam da yapmaya çalıştım. Daha çok uğraştım. ... Eksik şeylerim[etkinliklerim] oldu bazı sorunları çözemedim. ...kendime güvenim son haftalara doğru azaldı." (S2)

The pair programming students (N=7) also responded that they solved the programming problems by correcting mistakes of the partner during the programming process. Individual programming students mentioned that they could not find the mistake on the code blocks easily. In some activities, the sequence of code blocks was complicated and they could not see the mistake or they could not correct the mistake by themselves. This caused the decrease of their motivation. The students stated that;

"Working with my friend was beneficial for me. For instance, while you were teaching the subject I misunderstood some parts but my friend listened to the teacher correctly. … I made mistakes and she corrected my mistakes. In codes (ıııı), in effects of codes, when I chose the wrong codes, she corrected my fault. After that, my friend corrected the number of the variables that you presented to us. … For example, we did the activity of fish effect and I added the wrong code and my friend corrected." (P3)

"Arkadaşımla çalışmamın bana katkısı oldu. Mesela siz ders anlatırken benim yanlış dinlediğim bir şeyi arkadaşım doğru dinliyordu. ... Ben yanlış yapıyordum o düzeltiyordu. Kodlarda (ıuuuı) efektlerde falan ben yanlış şey [kod] seçince o düzeltiyordu. Ondan sonra verdiğiniz sayılarda [değişken] o

düzeltme yapıyordu. ... Mesela balık şişirme yapmıştık [balık efekti] ben değerleri [kodları] yanlış yaptım arkadaşım düzeltti." (P3)

"I had difficulty in coding, when I had difficulty I was not self-confident, I was panicky but I still relied on you (laughing). I started to struggle with coding, all the things I did came out wrong. I could not give attention and my mind had gone. Since I worked individually, I could not solve the problems. …My self-esteem decreased and I had difficulty as I made mistakes. I wished I had found my own mistakes by talking with my friends about wrong ones. I would do that because my friend told me to do that and I would trust myself." (S4)

"Kodlamada zorlandım, zorluk yaşadığımda kendime güvenim olmuyordu panik oluyordum yine de size güveniyordum (gülme). Kodlamalarda zorlanmalar başladı yaptığım her şey yanlış çıktı. ... Dikkatimi veremedim aklım gitti. Tek olunca problemleri çözemedim. ... Özgüvenim azaldı hata yaptıkça zorlandım. Arkadaşlarımla çalışıp o yanlış bu yanlış diye konuşarak kendi yanlışlarımı bulmak isterdim. Arkadaşım böyle yapmamı söyledi deyip kendime güvenir ve yapardım." (S4)

Some of the pair programming students responded that they did not need any extra source of knowledge for problem solving. On the other hand, the individual programming students (N=5) mentioned that they had limited knowledge about programming and they needed more information about code blocks while completing the activities. They said that if they had a friend or source for the solution of problems, they would solve the problems easily. Since they had to complete the activities individually, they needed more information and more sources than pair programmers. The participants said that;

"When we encountered the problems in coding we learned by trial and error we continued to work. We supported each other with my group friend. We reached solution of the problems by listening to you carefully and by trial." (P12)

"Problemlerle karşılaştığımızda deneyerek öğrendik, çalışmayı sürdürdük. (uuuu) grup arkadaşımla birbirimize destek çıktık. Öğretmenim sizi dinleyerek ve deneyerek çözüme ulaştık." (P12)

"I said programming was simple in the first week. As the activities became more difficult, I thought myself; if only there had been more resources. Maybe, I would create better if I had learned more. My self-confidence decreased" (S4)

"İlk hafta programlama kolaymış dedim. Etkinlikler zorlaştıkça daha fazla bilgi ve şey [kaynak] olsa diye kendi kendime düşündüm. Belki daha fazla öğrensem yapardım dedim. ... Özgüvenim azaldı." (S4)

Programming process is another theme of the content analysis with 4 categories including task completion time, quality products, learning of programming, and motivation. In the interview result of the pair programmers showed that pair programming technique influenced programming confidence of the pair programmers positively. The students stated that the effects of pair programming technique increased their confidence. The individual programmers also stated that the effects of the individual programming influenced the programming confidence negatively. They mostly claimed that working individually caused several problems and they had several negative effects of working individually.

Some pair programming students (N=7) claimed that they finished the activities quickly. They collaborated during the programming activities and solved problems as quick as they could. However, some of the individual programming students (N=7) stated that they needed extra time to complete activity. The given time was not enough for individual programming. Students mentioned that limited time for the completion of activities decrease their motivation and confidence.

"My friend contributed to me in programming .We finished coding faster with my friend and we completed it faster. Even if with hard activities; we believed that we would finish it quickly, we trusted ourselves." (P8)

"Arkadaşımın programlamada bana katkısı oldu. Arkadaşımla daha çabuk bitirdik daha hızlı kodlama yaptık. ... Hata yaptığımda arkadaşıma soruyordum. Zor çalışmalarda bile arkadaşımla çabucak bitireceğimize inandık, kendimize güvendik." (P8)

"Since I worked alone, I had to make more try to be successful in the programming activities. I had to try a lot of code blocks. The allocated time for

the activities was not enough for me. In some activities, I spent most of my time with design of program." (S7)

"Yalnız çalıştığım için, programlama etkinliklerini başarıyla bitirmek için daha çok denemek zorundaydım. Birçok şeyi [kod bloğunu] denemek zorundaydım. Etkinlikler için verilen (ıuı) süre azdı, zamanım yetmedi. ... Bazı aktivitelerde şeyimin [zamanımın] çoğunu program tasarımı için harcadım." (S7)

Some pair programming students (N=4) also responded that they produced better quality products with the help of their partners. Besides, the individual programming students (N=5) mentioned that they created some programs but the quality of the products were low. According to answers of the students some of their products were uncompleted or in lower quality than they wanted. They claimed that producing low quality product influenced their confidence negatively. The students stated that;

"We experienced more and we produced more quality products as a group. We confused the sequence of the code blocks and we completed by asking each other. We completed by helping each other." (P12)

"Grup olarak daha çok deneyim edindik ve iyi çalışmalar ürettik. Kodların sıralamasını karıştırıyorduk birbirimize sorarak daha iyi yaptık. Yardımlaşarak yaptık." (P12)

"Pairs were collaborating and producing better products. I had uncompleted activities because Scratch programming environment was difficult for me. …I cannot do it completely myself" (S2)

"Çift oturanlar iş birliği yapıyor ve daha iyi yapıyor ...Eksik yarım olan birkaç tane etkinliğim oldu. ...çünkü Scratch programı benim için biraz zordu. ... Tek olunca tam yapamam." (S2)

Moreover, pair programming students (N=9) responded that they learned better while working with a partner because the general opinion of the students was that they created more codes and they made more try-and-error with their partners and this provided better learning of programming. The individual programming students (N=4) stated that learning of programming was difficult for them because they had

several problems and they could not manage the programming activities accurately. The students said that;

> "We had some problems in the programming lesson, but we solved the problems by trying out the various codes that we had decided with my friend. Thus we learned more codes" (P1)

> *"Programlama dersinde bazı sorunlar yaşadık fakat arkadaşımla karar verdiğimiz değişik kodları deneyerek sorunları çözdük. Böylece daha çok kod öğrendik." (P1)*

> "I tried various things- code blocks to complete the programming activities. It was difficult for me to use some code blocks. Some of my friends worked together so I had to work hard and try a lot compared to many friends." (S5).

> *"Programlama etkinliklerini tamamlamak için çeşitli şeyleri [kod bloklarını] denedim. Bazı kod bloklarını kullanmak benim için zordu. Bazı arkadaşlarım beraber çalıştı bu yüzden birçok arkadaşıma göre çok çalışmak ve denemek zorundaydım." (S5)*

Finally, pair programming students (N=5) stated that working with a partner increased their motivation. With the support of the partner, students felt more motivated in the programming activities. On the other hand, the individual programmers (N=3) responded that working individually in programming was hard for them and felt un-motivated because of the complicated code blocks and programming problems. The students stated that;

> "My friend helped me during the programming process. My partner motivated me in the programming activities by giving advices and appreciating me." (P5)

> *"Arkadaşım programlamada bana yardımcı oldu. ... Şöyle yapalım, böyle yapalım, güzel yaptın diyordu, beni motive ediyordu" (P5)*

> "The activities included complicated codes and I had some difficulties during the lesson. I tried more to solve the problems and this confused my head. Thus, I needed more time for the activities. Some of my friends finished quickly. This situation decreased my motivation." (S6)

"Etkinlikler karmaşık şeyler [kodlar] vardı ve ders boyunca zorlandım. Problemleri çözmek için (uuuu) daha çok denedim bu benim kafamı karıştırdı, panik oldum. Bu yüzden etkinlikler için daha fazla zamana ihtiyaç duydum. ...Bazı arkadaşlarım hızlıca bitirdi bu benim motivasyonumu düşürdü." (S6)

Being in a pair or individual during process is the final theme of the programming confidences of students' content analysis result. The theme of being in a pair or individual during process was constructed with the categories of disagreement, programming ability differences, and workload. The students claimed that the differences among the partners caused some problems.

Firstly, some pair programming students (N=6) stated that they could not agree with the decision of each other during the programming activities. Every partner insisted on their decisions so they had problems with their partners. Besides the pair programmers, individual programmers responded that they needed alternative solution and ideas for the completion of the programming activity. They had to apply their own ideas and they could not find the accurate codes individually. The student said that;

"When creating code blocks, my friend helped me in the codes I forgot but in the design part, my friend did not choose the characters that I wanted to add and I did not like his / her preferences. We discussed for the reason of designing the environment. While working individually you had chance to try more and you can design what you want. However, my partner helped me in the code blocks; she/he told me the codes that I forgot. Thus, working with my friend increased self-confidence." (P1)

"Kodlama yaparken benim unuttuğum şeylerde [kodlarda] o yardımcı oluyordu ama balık seçerken [karakter] benim istediğimi o istemiyordu, onun istediği de benim hoşuma gitmiyordu. Aramızda kargaşa çıkıyordu tartışıyorduk. Tek olunca daha çok deneme şansın oluyor istediğin tasarımı yapabiliyorsun. ... Fakat kodlarda bana yardımcı oluyor benim unuttuğum kodları o bana söylüyor. Arkadaşımla çalışmak bu yüzden özgüvenimi arttırıyor." (P1)

Secondly, the individual programming students (N=7) said that workload in the programming activities is one of the disadvantages of programming process. Students claimed that the disadvantages caused to decrease confidence of them. However, pair programmers did not mention any workload in their programming process.

> "While designing the Scratch environment I could not draw some design elements. I mostly had difficulties in coding part. It was hard to me. I tried to solve problems by trial and error however; some of them finished some of them not. I did not give up; I tried not to be afraid. I made my brain work more. I tried to recall some codes. … My confidence declined when I could not get out of the programming problems." (S7)

> *"Tasarım yaparken bazı şekilleri çizemedim. Daha çok kodlama da sıkıntı yaşadım, bana zor geldi. Deneyerek çözmeye çalıştım ama bazıları oldu bazıları olmadı. Pes etmedim korkmamaya çalıştım beynimi daha çok yordum. Denedim hatırlamaya çalıştım. ... Sorunların içinden çıkamayınca seyimm (ıuı) özgüvenim azaldı" (S7)*

Finally, some pair programming students (N=7) complained about the programming ability differences among the partners in the programming. However, since the individual programming students worked individually they did not encounter this problem. The pair programming student stated that;

> "In coding, we were helping each other with my friend. For example, we could not know the place of the code blocks. We were adding different things and discussions occurred." (P6)

> *"Kodlamalarda arkadaşımla yardımlaşıyorduk. Kodlamalarda mesela (ıuuı) mesela yerlerini [sırasını] pek bilemiyorduk. ... İkimizde farklı şeyler ekliyorduk tartışma oluyordu." (P6)*

Both individual and pair-programming students related their confidence to different factors. Pair-programmers frequently stated that their confidence was increased with the advantage of having a pair while programming because pairs cooperated, supported, and motivated each other during programming process. Individual-programmers on the other hand, attributed their decrease of confidence in several

factors: not having a pair or information sources, increasing difficulty of activities, and unsatisfied acheivements in activities. While some individual students were motivated to work hard for solving the problems by themselves, they reported that increasing difficulty of the activities and decreasing the achievements in activities decreased their confidence in completing the programming activities. Individual students compared their performance with the pair-programmers in terms of duration, workload, and ability to complete the activities, and tend to attribute their difficulties to being an individual during programming.

4.3 Programming Achievement of the Students

4.3.1 Quantitative Results for Programming Achievement of the Students

The result of quantitative data for the programming achievement was represented in this part of the study.

Table 4.6 *Descriptive Statistics for Achievement*

	Pair-Programmers *M (SD)*	Individual-Programmers *M (SD)*
Activity 1	9.72 (0.45)	7.92 (1.38)
Activity 2	9.81 (0.39)	8.53 (1.61)
Activity 3	9.63 (0.65)	8.00 (1.52)
Activity 4	9.36 (1.25)	9.46 (0.77)
Activity 5	9.45 (1.01)	8.53 (0.96)
Activity 6	9.72 (0.45)	7.76 (1.53)
Activity 7	9.54 (0.80)	6.23 (1.87)
Activity 8	9.54 (0.50)	7.69 (1.31)
Total	76.27 (4.33)	63.38 (6.47)

The means and standard deviations of pair and individual students were given in Table 4.6. The means of the pair programming students ranged 9.36 to 9.81. The means of achievement of individual programmers also ranged 6.23 to 9.46. The means of the pair groups were closer to each other in 8 week implementations. On the other side, the means of individual programmers differentiated week by week. The highest mean was *M*= 9.46 in the activity 4. The lowest mean was in the activity 7 *M*= 6.23. The individual programming students had difficulties in activity 7 but the pair programming students' achievement was high in this week with the mean of *M*= 9.54. While the means of individual programmers' achievement were decreasing in some weeks, the pair students completed the activities with higher means.

Table 4.7 *Group Statistics for Achievement Scores based on Rubric Evaluation*

		N	Mean	Std. Deviation	Std. Error Mean
Achievement Scores					
	individual	13	63.38	6.47	1.79
	pair	22	76.27	4.33	0.92

Table 4.8 *Independent Samples Test for Achievement Scores based on Rubric Evaluation*

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Achievement Scores	Equal variances assumed	6.72	.014	-7.06	33	0.00	-12.88	1.82	-16.59	-9.17
	Equal variances not assumed			-6.38	18.45	0.00	-12.88	2.01	-17.12	-8.65

An independent-samples t-test was conducted to compare the programming achievement of pair programmers (n=22) and individual programmers (n=13) according to usage of pair programming method in programming course. Table 4.7 presented group statistics of the study. There was a significance difference in the scores of pair programmers' achievement *M*= 76.27 (*SD*=4.33) and individual programmers' achievement *M*= 63.38 (*SD*=6.47) with *t* (18.46) = 6.38, *p* = .00. The 95% confidence interval for the difference in means was ranging from -11.43 to -1.80. The Cohen's d effect size calculated as 4.87 for the programming achievement dependent variable and it was large effect since the effect sizes are interpreted as small (0.2), medium (0.5) and large (0.8) (Green & Salkind, 2004).

Table 4.9 *Group Statistics Mann-Whitney U for Achievement Scores based on Rubric Evaluation*

	peerindividual	N	Mean Rank	Sum of Ranks
meanach				
	individual	13	7.38	96.00
	peer	22	24.27	534.00
	Total	35		

Table 4.10 *Mann-Whitney U Test for Achievement Scores based on Rubric Evaluation*

	meanach
Mann-Whitney U	5.00
Wilcoxon W	96.00
Z	-4.75
Asymp. Sig. (2-tailed)	0.00
Exact Sig. [2*(1-tailed Sig.)]	0.00[b]

Mann-Whitney U test was conducted to evaluate to compare the means of pair and individual independent groups for the achievement dependent variable. The results of the test were significant *z* = -4.76, p = .00. The mean rank was *M* = 7.38 for individual programmers and the mean rank was *M* = 24.27 for pair programming groups in Table 4.9.

4.3.2 Qualitative Results for Programming Achievement of the Programmers

Table 4.11 *Factors Influence Achievement in Programming*

PAIR			INDIVIDUAL			
Categories	Participant Mentioned	NP	Categories	Participant Mentioned	NS	NT
Method of achieving programming			**Method of achieving programming**			
Knowledge sharing	P1,P2,P4,P5,P6,P8,P10,P11,P12,P13	13	Knowledge sharing	0	0	13
Being creative (creativity)	P4,P6,P8,P12	4	Being creative (creativity)	0	0	4
Testing codes	P1,P4,P5,P6,P8,P10,P11,P12	8	Testing codes	0	0	8
Access to the resources	0	0	Access to the resources	S1,S4,S5,S6,S7	5	5
Getting help from others	P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13	13	Getting help from others	0	0	13
Programming Process			**Programming Process**			
Grades of activities (high)	P3,P4,P6,P8,P9,P12,P13	7	Grades of activities (low)	S1,S4,S5,S6,S7	5	12
Amount of errors in codes (less)	P1,P2,P3,P4,P5,P6,P8,P10,P12,P13	10	Amount of errors in codes (more)	S1,S2,S3,S5,S6,S7	6	16

Table 4.11 (cont'd)

Effort to achieve (less)	0	0	Effort to achieve (more)	S1,S2,S3,S4,S5,S6,S7	6	6
Activity completion duration (quick)	P3,P4,P5,P8,P11,P12,P13	7	Activity completion duration (slow)	S1,S2,S3,S5,S6,S7	5	12

Note. NS= Number of individual programmers, NP= Number of pair programmers, NT= Total number of pair and individual programmers.

The results of the content analysis for the interview of programming achievement of 5th grade programmers revealed that some factors influenced the programming achievement of the students during the programming activities. According to content analysis result, two themes were constructed which were a) method of achieving programming and b) programming process. The pair groups believed that they achieved more with their partners. The Table 4.11 clearly presented the themes and the categories of the programming achievement content analysis results. According to content analysis results of the interviews, the individual programming students stated that they had several problems while programming individually. The students said that the programming activities were difficult for the individual programmers.

One of the themes of the content analysis is the method of achieving programming. The theme included five categories for both pair programming students' achievement and individual programming students' achievement.

The pair-programmer students (N=13) responded that, to achieve the programming activities, they shared their knowledge that they learned from their teacher. Transferring knowledge was a method of achieving programming skills with the partner according to their opinions. However, individual students did not state any comments about the knowledge sharing with a friend because they had to work individually so they did not share any knowledge in the programming process. The students said that;

> "… For example, she helped me where I could not. I sometimes misunderstood in the lessons and she shared with me the correct codes. My partner corrected my mistakes in codes. We came to upper levels as we learned the program." (P3)

> *"... Mesela yapamadığım yerlerde bana yardımcı oldu. Ben bazen yanlış dinliyordum o doğrusunu benimle paylaşıyordu. Yanlışlarımı düzeltiyordu sayılarda falan. ... programlamayı öğrendikçe üst seviyelere geldik." (P3)*

Moreover, the pair programming students (N=4) responded that thinking with a partner provided more creative results for the programming activities. According to students' responses, creativity was a way of achieving programming with a partner. The students stated that;

> "I and my friend thought different things (codes) and we tried alternative things (codes). We produced different code blocks to create different code programs." (P4)
>
> *"Arkadaşımla değişik şeyler düşündük ve değişik şeyler denedik. Farklı programlar oluşturmak için değişik kod blokları ürettik". (P4)*

The pair-programming students (N=8) also responded that working with a partner facilitated the programming activities by testing the codes regularly. The students claimed that checking the code blocks with a partner provided to produce more successful activities. The students said that;

> "While we were working with my friend we combined our information about coding and we worked better. He checked and corrected my mistakes. This made us more successful." (P8)
>
> *"Arkadaşımla çalışırken bilgilerimizi birleştirip daha iyi çalışmalar yaptık. Ben kodları ekledim o da kodların doğru olup olmadığını kontrol etti. ... Hatalarımı düzeltti ve kontrol etti. Bu daha başarılı olmamızı sağladı." (P8)*
>
> "At the beginning of the lesson I convinced myself that I would success. As the coding became more difficult my performance decreased. I added the wrong codes , I could not see my mistakes and could not correct them." (S7)
>
> *"İlk başlarda başarırım diye düşündüm kendimi inandırdım. Kodlamalar zorlaştıkça başarım azaldı. ...yanlış şeyler [kodlar] ekledim, yanlışlarımı göremedim ve düzeltemedim." (S7)*

Moreover, the individual students stated that they did not get any help from a friend or other resources. They claimed they needed help a friend during the programming activities. They stated that help of a friend while creating codes may increase their programming achievement. However, one student (N=1) stated that working individually was better for him/her because when he/she worked with a partner, they might disagree with each other while designing the coding environment. Moreover, the pair programming students (N=13) stated that they got help from their partners and working with a partner was beneficial for gaining better programming skills. The students said that;

"When I worked individually, there were parts that I could not complete. If I had had a partner, he/she would have completed my deficiencies about programming. I could also do the parts that my partner did not know." (S1)

"...ben bireysel olduğumda yapamadığım kısımlar oldu. Ama grup arkadaşım olsa benim bilmediklerimi, yapamadıklarımı böyle hiç bilmediklerimi (ıuıı) o bilir o yapardı. Onun bilmediklerini de ben yapardım." (S1)

"When I worked with my friend he helped me on the coding activities. My partner reminded me some codes that I forgot to add to the code blocks and we worked together. If I had worked individually, I would have not completed the programming activities easily, my friend helped me." (P10)

"Arkadaşımla oturduğumda kodlama da falan yardımcı oldu. Unuttuğum şeylerde [kodlarda] bana hatırlattı beraber yaptık. ...daha kolay yaptık, kendim otursaydım daha kolay yapamazdım, yardım etti." (P10)

"I would like to sit alone while coding because it is nice to sit alone. If we had worked as a group of two people we would want to add different characters in design of the coding environment coding. Coding would be easier however; two people might cause chaos. *." (S6)*

"Kodlama yaparken tek oturmak isterdim çünkü tek oturmak güzel. İki kişi otururşak o başka bir şey ister ben başka bir şey isterim tasarımda. Kodlama kolaylaşır fakat... iki kişi olunca kargaşa çıkabilir." (S6)

Furthermore, the individual programming students (N=5) said that they needed more resources than their friends who worked as a group because some activities require more knowledge of programming. The students stated that;

"It was necessary to revise what you said and I revised them at home. If I had known more and if I had had my notes with me, I would have done better." (S1)

"Sizin dediklerinizi tekrar etmek gerekiyordu, evde ben tekrar ettim. Daha fazla bilsem, yanımda notlarım olsa daha iyi yaparım dedim." (S1)

Programming process on achievement was one of the themes that influence the programming achievement of pair programming students' and individual programming students'. According to responses of the students, the theme was constructed with 4 categories. Both pair programming and individual programming students stated that there were some factors that influenced their achievement positively or negatively. The pair programmers generally stated that their achievement was high during the programming process but the individual programmers stated that their achievement was mostly influenced negatively in the programming activities.

The pair programming students (N=7) stated that they got high grades from the programming activities. They mostly claimed that working with their friends was beneficial for the students to get high grades. However, the individual programming students (N=5) responded that they got low grades from some activities because they had difficulties in the code blocks. The students stated that;

> "By working with my friend I became more successful than other friends. We received high grades because we completed the activity successfully." (P4)
>
> *"Arkadaşımla çalışarak diğer arkadaşlarımdan daha başarılı oldum. Etkinlikleri başarıyla tamamladığımız için yüksek notlar aldık." (P4)*
>
> "I could not complete some of the activities without any mistakes and I got low grades from some activities." (S5)
>
> *"Bazı etkinlikleri hatasız bir şekilde tamamlayamadım ve bazı etkinliklerden düşük notlar aldım." (S5)*

Moreover, the pair programming students (N=10) stated that they made less error in the programming activities. The pair programmers stated that their partner checked the code blocks to correct the mistakes. According to pair programmers, less error in the codes was one of the positive influences of programming process. On the other hand, the individual programmers (N=6) stated that they made more mistake while creating programs. Most of the students said that they did not realize their mistakes while working individually. They claimed that if they had worked with a partner they would make fewer mistake. The students said that;

"We believed our decisions about code blogs and we agreed on our decisions while creating codes. We made fewer mistake with the help of the group working." (P6)

"Kodlarla ilgili kararlarda arkadaşımla birbirimize güvendik ve şeyleri [kod bloklarını] oluştururken ortak düşündük. ...Grup çalışması sayesinde daha az hata yaptık." (P6)

"I made a lot of mistakes and I tried to correct the code blocks during the lesson. My friends worked together and they solved the problems together. I could not complete the programming activities individually without any mistakes." (S6)

"Birçok hata yaptım ve düzeltmek için ders boyunca uğraştım. Arkadaşlarım beraber çalıştılar ve problemleri beraber çözdüler. Ben programlama etkinliklerini hatasız tamamlayamadım." (S6)

Furthermore, some individual programming students (N=6) stated that they needed to make more effort to complete the programming activities. The students said that they did not complete the activities easily and they had to work more for the programming activities while pair-programming students did not report any serious problem about the completion of the activities. The students said that;

"I had difficulty with some activities. I was scared and panicked because of unsuccessful programming results and I could not manage complicated code blocks. ... Some of my friends worked together in the programming activities. For this reason, I had to work very hard and try more. (S5)

"Bazı çalışmalarda zorluk yaşadım. Korktum, panik oldum yapamadığım için (ıı) şeyleri karışık şeylerde baş edemedim. ...Bazı, bazı arkadaşlarımla beraber çalıştılar. Bu yüzden birçok arkadaşıma göre çok çok çalışmak ve çok denemek zorundaydım. (S5)

Finally, the individual programming students (N=5) mentioned that they finished the programming activities slowly. The time was one of the problems for the individual programmers. The students said that they needed more time because they worked

individually. The pair programming students (N=7) stated that they complete the programming activities quickly with the help of their partners. The students said that;

> "Since I worked alone in programming activities, I had to try more to complete the activities successfully. ... The given time for the programming activities was not enough for me so I needed extra time." (S7)

> *"Yalnız çalıştığım için programlama etkinliklerini başarıyla bitirmek daha çok denemek zorundaydım. ... Şey (ıuı) etkinlikler için verilen (ı) süre azdı. Zamanım yetmedi." (S7)*

> "My friend did not hinder me; instead he contributed to me. While working with my partner, we made coding quickly and we finished the programming activities faster. If I had worked individually, I would have been slower because I was asking the codes that I did not know to my friend." (P8)

> *"Arkadaşımın bana zararı olmadı katkısı oldu. Onunla çalışırken kodlamayı daha hızlı yaptık, çabuk bitirdik. ...Tek olsam daha yavaş olurdum çünkü bilemediklerimi ona soruyorum." (P8)*

Pair-programming students felt more achieved completing the activities on time and with success together. Individual students reported their extra hard-work trying to complete the activities by themselves without help and sometimes not being able to complete them successfully.

4.4 Emotions of Students During the Programming Process

While examining confidence and achievement of students, their emotions also emerged as another main issue to be examined during content analysis. The interview result of the 5th grade programming students showed that they felt some emotions during the programming process. The emotions occurred both negative and positive. The pair programming students mostly mentioned that they felt positive feelings about programming. On the other hand, the individual programmers mostly stated that they felt negative feelings while creating code blocks. According to response of the students, the emotions affected or being affected by their programming confidence and programming achievement. This part of the study showed that the emotions that the students felt and how these emotions were related to the

programming confidence and programming achievement of pair programmers and individual programmers.

Table 4.12 *Emotions Influence Confidence and Achievement*

PAIR			INDIVIDUAL			
Categories	Participant Mentioned	NP	Categories	Participant Mentioned	NS	NT
Emotions during programming			Emotions during programming			
confident	P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13	13	confident	0	0	13
non-confident	0	0	non-confident	S1,S3,S4,S5,S6,S7	6	6
relaxed	P3,P4,P6,P7,P12	5	relaxed	0	0	5
panicked	0	0	panicked	S1,S4,S5,S6,P6,P9	6	6
motivated	P4,P5,P6,P8,P12	5	motivated	0	0	5
non-motivated	0	0	non-motivated	S5,S6,S7	3	3
productive	P4,P6,P8,P12	4	productive	0	0	4
unproductive	0	0	unproductive	S1,S3,S4,S5,S6	5	5
had fun	P3,P6,P12	3	had fun	0	0	3
bored	0	0	bored	0	0	0
friendship	P1,P2,P3,P6, P12	5	friendship	0	0	5
isolation	0	0	isolation	S1,S6,S7	3	3
desperate	0	0	desperate	S1,S5,S7	3	3
afraid	0	0	afraid	S1,S4,S5,S6,S7	5	5

Note. NS= Number of individual programmers, NP= Number of pair programmers, NT= Total number of pair and individual programmers.

According to interview responses of students, the emotions during programming firstly organized as a theme of emotions of programmers both working with a partner and individually. In the Table 4.12, the theme included 14 categories which were both negative and positive emotions of the pair and individual programmers. The content analysis showed that the pair programmers mostly had positive emotions but the individual programmers mostly had negative emotions during the programming activities.

The responses of pair programming students stated that mostly they felt themselves more confident with their partners. All of the students (N=13) gave the same answer that they felt more confident while working with a partner during the programming process. However, most of the individual programmers (N=6) mentioned that they felt unconfident while working alone during the programming activities. The reason behind feeling unconfident in the programming lessons was mainly mentioned that they needed the help of a friend. The students stated that;

> "My group friends supported me with coding. In the first week, I gave 3 points to my self-confidence and 5 points for the last week. My self-esteem increased gradually" (P12)
>
> *"Grup arkadaşım bana kodlamalarda bana destek oldu. ... İlk hafta özgüvenime 3 puan verirdim son haftaya ise 5 puan veririm. Özgüvenim gitgide arttı çünkü." (P12)*
>
> "At the beginning of the programming lesson I felt a little bit confident about programming and I felt happy to sit a computer alone because I had chance to use computer more than my friends. However, I felt non-confident about programming as I experienced the code in the Scratch programming environments." (S7)
>
> *"Programlama dersinin başında biraz şeyim [özgüvenim] vardı ve (ıuı) bilgisayarda tek başıma oturmak daha çok bilgisayar kullanacağım için beni mutlu etti. Fakat şeydeee… [Scratch'te] kodları yaptıkça programlamayı karşı özgüvensiz hissettim." (S7)*

Some of the pair programming students (N=5) responded that they felt relaxed during the programming process since they worked with a group member and they

thought that they completed the activity easily by this way. On the other hand, the individual programming students (N=6) stated that they were panicked when they worked alone during the programming process. The students pointed out that;

> "I felt more relaxed and confident to work with a partner in programming activities; my self-confidence was high because my partner helped me to complete the programming activities which I was not able to complete." (P3)

> *"Daha çok güvende ve rahat hissettim, yani özgüvenim daha çok fazlaydı. Çünkü mesela benim yapamadığım konularda arkadaşım yardımcı oluyordu bana." (P3)*

> "I was panic in the difficult activities and I felt non-confident in these activities. I got more panic as the programs became harder." (S4)

> *"Zorlandığım etkinliklerde panik oluyordum, kendime özgüvenim olmuyordu. ...programlar zorlaştıkça daha çok panik oldum." (S4)*

The pair programming students (N=5) also responded that they felt motivated during the programming sessions. The pair programming students stated that working with a group member was the most powerful reason of feeling motivated in the programming activities. However, the individual programming students (N=3) stated that their motivation of programming decreased especially in last activities because they confused the code blocks and they did not get any help about the usage of code blocks. The students stated that;

> "I confused the place and usage of the codes but my friend helped me in the activities so this increased motivation of me." (P6)

> *"Kodların yerini ve nasıl kullanılacağını karıştırdım fakat arkadaşım bana yardımcı oldu. Bu benim motivasyonumu arttırdı." (P6)*

> "At the beginning of the programming activities, I thought that programming was easy for me but when I encountered the difficult programming activities I felt that I could not complete the activities successfully. My motivation disappeared like this." (S6)

"Programlamanın en başında programlama benim için kolay diye düşündüm fakat zorlandığım kısımlar olduğunda şeyleri [etkinlikleri] başarıyla tamamlayamayacağımı düşündüm. Motivasyonum böyle olunca kayboldu." (S6)

The pair programming students (N=4) stated that they felt productive while creating code blocks with a partner. On the other hand, the individual programming students (N=5) claimed that they felt unproductive while creating code blocks in programming activities. The students stated that;

"We shared our knowledge with my friend and we quickly completed the programming activities by correcting the mistake of the activity. As a group, we produced quality product and we got more experience about programming." (P12)

"Arkadaşımla bilgilerimizi birleştirerek ve hatalarımızı düzelterek programlama aktivitelerini hızlıca tamamladık. Grup olarak daha çok deneyim edindik ve iyi çalışmalar ürettik." (P12)

"I had some un-completed activities in the programming lesson. …I made a lot of mistakes while working individually and I produced incomplete encodes. I felt un-productive because I could not complete the activities as my teacher wanted." (S5)

"Programlama dersinde tamamlanmamış etkinliklerim oldu. ... Tek başıma çalışırken çok hata yaptım ve eksik kodlamalar yaptım. Öğretmenim sizin istediğiniz gibi kodlamaları tamamlayamadığım için iyi şeyler yapamadığımı [üretemediğimi] hissettim." (S5)

The pair programming students (N=3) stated that they felt fun while working with their partner in the programming activities. They stated that;

"Working with the partner was very enjoyable for me because my friend supported me and we believed each other that we would complete the activities." (P6)

"Grup arkadaşımla çalışmak çok eğlenceliydi çünkü arkadaşım beni destekledi ve biz etkinlikleri tamamlayacağımız konusunda birbirimize inandık." (P6)

While some of the pair programming students (N=5) responded that they felt friendship with their partner during the pair-programming process, the individual programming students (N=3) pointed out that they felt isolated from their friends in the programming activities. The individual students stated that they worked individually and some of their friends had chance to talk each other about programming. They stated that;

> "Creating code with the partner made me feel happy. The relationship between me and my friend improved so we shared our knowledge clearly." (P4)
>
> *"Arkadaşımla kod oluşturmak beni mutlu etti. Aramızdaki ilişki ilerledi, çok yakın hissettik birbirimize böylece bildiklerimizi açıkça paylaştık." (P4)*
>
> "Some of my friends worked together and completed the activities more quickly and successfully. Since I worked individually I felt bad myself. …I felt isolated from my friends." (S7)
>
> *"Bazı arkadaşlarım iki kişi olduğu için daha hızlı tamamlıyorlardı, daha çok başarıyorlardı. Ben tek çalıştığım için kendimi kötü hissettim. ... Arkadaşlarımdan uzaklaşmışım gibi oldum." (S7)*

The pair programming students (N=3) also pointed out that they felt desperate while working in the programming activities. They stated that creating code blogs was very complicated issue so when they had difficulties in coding they felt desperate about learning programming. The student said that;

> "I had some difficulties in the programming activities. Being the worst student among my friends disappointed me and made me feel sad" (S1)
>
> *"Zorluklar yaşadım, (ıuuıı) yaşadığım oldu. Arkadaşlarım mesela yaparken en sonuncu olmak beni biraz üzdü hayal kırıklığına uğrattı." (S1)*

The pair programming students (N=5) were afraid of programming because they said that the codes were complicated and they could not complete the activities of themselves accurately. They need to help of a friend in the programming activities. The students said that;

"While I was creating code blogs I confused the codes and I felt sad and I was afraid." (S6)

"Kodlama yaparken kodları karıştırdım, üzüldüm ve korktum." (S6)

The emotions of the students showed that the programming achievement and programming confidence were related to both positive and negative feelings. The pair programming students pointed out that they mostly felt positive feelings, so their programming confidence and programming achievement influenced positively. On the other side, the individual programming students mostly stated that they felt negative feelings because of working individually and needing help and source during the programming process. Several factors are reported to influence individual students' achievement and programming confidence negatively.

4.5 Comparing, Combining, and Summarizing the Results

The study aims to explore the influences of pair programming technique on programming confidence and programming achievement of secondary school students. The data were collected form 5^{th} grade students who were divided into pair programmers or individual programmers. The qualitative data collected form the interviews. Moreover, the quantitative data collected from confidence scale and achievement rubrics. Both types of data were analyzed to explore the possible influences of pair programming technique on secondary school students' programming achievement and programming confidence.

The independent sample t-test and Mann Whitney U test results showed a significance difference between the achievement scores of pair programmers and individual programmers in the activities implemented in 8-weeks. The result indicated that the pair programmers received significantly higher scores than individual programmers. Moreover, the programming confidence of 5^{th} grade students was analyzed with the independent sample t-test. The pre-test and post –test results of programming confidence of pair programming students and individual programming student showed that there was also a significance difference between these two groups of students. While the pair programming students' programming confidence increased during the 8 weeks programming activities, the individual programming students' confidence decreased.

The qualitative data findings of the study showed that some factors influenced the programming confidence of pair programmers and individual programmers differently. Firstly, the students stated that problem solving with a group member and individually lead different experiences for the students. The pair programming students indicated that when a problem occurred during the programming activities they mostly solved the problems by sharing their programming knowledge, supporting to each other, helping each other, correcting mistakes of the programs, finding solution of the problems of the activities and by using try and error method with their partner. The pair programming students stated that they felt more confident with a group member while solving the problems in the programming activities. On the other hand, individual programmers mostly reported that they had difficulty completing the activities without any resources or the help of the teacher, and therefore, they did not feel confident. They reported that they worked hard, but with the increasing difficulty of the activities, and decreasing scores, they started to lose their confidence. They frequently attributed the reason of not solving the problems on working individually. A few students showed that the reason behind this problem was having less information of programming. The students mentioned that if the students get needed information about coding, they could easily complete the activities. The individual programming students stated that they needed help of a classmate in the programming problems. They believed that if they had supported by a partner they could have been more confident.

Secondly, there were different results among pair and individual students in terms of learning the programming, task completion time, quality of product, and the motivation of the programmers. The pair programmers revealed that they learned by working with their partner more and their confidence increased in this way. Most of the pair programming students also shared their experiences about the time management by working with a partner as they finished the activities quickly and easily. All of the pair students indicated that they did not need any extra time for the activities with the help of their partner. Moreover, they yielded that they produced quality products because they weeded out the unnecessary codes and they mostly used practical code blocks for the activities. They mentioned that their confidence increased in this process. Furthermore, the pair programming students stated that they had enough motivation to complete the programming activities. They felt

motivated during the programming process with their partners. They indicated that the high motivation increased their confidence of programming.

On the other hand, the individual programmer students mostly mentioned that working individually during the programming process caused them to learn less information about coding, finishing the programming activities late, producing unsatisfactory programs, and demotivation during the programming process. The individual programming students mostly said that they could learn more with a group member because the group members provided more information to each other. They also indicated that they needed extra time for the completion of the programming activities. They mentioned that, especially the last activities required more time to complete for them. The individual programming students also thought that they could not produce the products that they wanted to. They also stated that their motivation was low because their pair programming classmates were completing their activities early while they could not complete their activities on time according to the criteria. They indicated that their confidence decreased for these reasons.

Although pair students reported positive experiences and feelings regarding their programming together, they also reported disadvantages of working as a pair. The programming abilities of the students and their partner might be different and they wanted to try their own programming style. They revealed that disagreements between the pair students influenced their works because the students disagreed especially in design part of the programming. On the other hand, individual programmers mostly indicated that their workload was more than pair-programming groups because they had to work individually. Moreover, the individual programming students revealed that time management was one of the disadvantages of working individually. The students stated the disadvantages of working individually decreased their confidence during the programming process.

The interview results of programming achievement of the students revealed that pair programming and individual programming students had different experiences while completing the programming activities according to the rubric. Pair programming students pointed out that, sharing knowledge with a partner, being creative, testing the code blocks, accessing the resources of Scratch programming environment and getting help from others were the factors of the way of achievement in the computer

programming. The pair programming students stated that they achieved the programming activities by these factors. On the other hand, the individual programming students stated that they had difficulties on the programming activities. They could not achieve some of the programming activities. They mentioned that they needed help of a friend during the programming activities. They also needed to access to the resources to complete the programming activities.

The interview results of programming achievement of the students in 8-week programming process also showed that the programming process was different for both groups. The pair programming students indicated that their programming ability was improved by working with their partners. They also said that their grades were high in the activities of 8 weeks. The pair programming students mentioned that their partners corrected their mistakes and they created quality code blocks. With the help of working with a partner, the students stated that they created code blocks easily and they did not need more effort for the activities. Finally, the pair programming students also pointed out that they completed their programming activities in the given time period. They stated that they did not need any extra time for the activities. However, the individual programmers stated that their programming ability was low because some of the code blocks were difficult for them. The individual programming students also said that the programming activities achievement scores were low since they could not complete some of the activities accurately. Moreover, the students had some mistake in the code blocks and they mentioned that they could not found the codes in the code blocks so they could not correct the code blocks. Furthermore, the individual programming students stated that they tried more to solve the problems in the programming activities by themselves and put more effort to be successful in completing the activities. Some of the students pointed out that they needed extra time for the activities because they could not complete some programming activities in given time. They said that working individually caused several problems and to solve these problems they needed more time.

Lastly, the interview result of programming experiences of 5^{th} grade students showed that some of the emotions were related to the programming achievement and programming confidence of the 5^{th} grade programmers. The pair programming students mostly mentioned that the positive feelings that they felt increased their programming achievement and programming confidence positively. The pair

programming students mostly mentioned that they felt more confident with their group member while creating code blocks. When the feelings of the students during the pair programming process were asked to the pair programming students and they responded that they felt positive feelings because of working with their partners. Being motivated, confident, productive, fun, were some of the positive feelings of the pair programmers. On the other hand, the students responded to the some interview questions with negative sentences. Most of the individual programmers mentioned that they felt negative feelings because they worked individually. They felt demotivated, unconfident, isolated, sad, and unproductive during the programming process. They said that the negative feelings caused the decreasing their programming confidence. As they receive lower grades and lower-quality programs with decreasing achievement, they felt less and less confident.

4.6 Reflections of the Researcher as the Teacher During Implementation

In the programming process, the qualitative and quantitate data were collected and analyzed to understand the influence of pair-programming technique on programming confidence and programming achievement of secondary school students. The findings of the study were presented in the result section. In addition to these findings of study, the teacher as a researcher observed some behaviors of students. These reflections were presented to reflect the 5^{th} grade students in the programming process.

In the first semester (previous semester before data collection), the interests of 5^{th} grade students were high at the beginning of the Information Technologies and Software lesson. I observed the reason behind this interest that the students expected to play games, surf on the Internet, and watch cartoons. In the first weeks, most of the students explained that their friends in the upper classes mentioned the lesson as a free time to play computer games and surf in social media. When I explained that there was not such kind of activities in the lesson, their interest and motivation decreased. Most of the 5^{th} grade classes gave the same reaction about the content of the lesson. Since their parents also limited their technological device usage at home, they expected to use them in the Information Technologies and Software class. These limitations on the Internet and social media caused same negative expectation about Information Technologies and Software lesson. As a teacher, I explained the

content of the lesson and tried to motivate them to the lesson. The expectation of the students was changing during the lesson. At the end of the first semester, the students adapted the rules and content of the lesson. Therefore, at the beginning of the second semester (the semester when the data were collected), the students' expectations and motivations were moderate.

Secondly, the students took the Information Technologies and Software lesson for the first time in 5th grade first semester. The general behavior that they demonstrated in the first lesson was quickly choosing the best computer and sitting alone, and they did not want to sit together with their classmates. Since the numbers of computers were inadequate and the technical properties were not enough for the students, some 5th grade class students had to work with their friends in class. In this condition, majority of group members in 5th grades behaved selfishly. While they were sitting together, they did not allow their pairs to use the computers, and their pairs were complaining about not being able to use the computer fairly. The teacher had to listen their complaints about their partners. When I talked the other teachers of 5th grade students, they also mentioned that the students tend not to share their materials, lesson notes with their friends, and do not want to cooperate. The teachers mentioned that 5th grade students usually in competition with each other. When a student made mistakes in the lesson, their classmates report that mistake to the teacher. Because these 5th grade students' selfish and competitive attitude with their classmates, most of the students try not to show their code blocks to their partners in the first semester while completing their activities. In all 5th grade classes, majority of students hid their code screens from their classmates. They turned the monitor to different sides or closed the screen with their notebooks or their hands. Although I said that they can work together with their classmates in free time programming activities, most of the 5th grade students hid their code blocks from their classmates.

Lastly, the 5th grade class students and their parents cared about the grades at the lesson very much. The average of the lesson was important for the students because they also got points in the exam for high school in Turkey. For this reason, most of the parents cared very much about the grade of the lesson. The 5th grade students compared their grades with their classmates. More than building knowledge, they were very focused on the grade. The 5th grade teachers also mentioned that the students memorized the knowledge to get high scores in exams, after that they forgot

most of the knowledge in the lesson. In the programming activities, I also observed that students worked hard. Individual programmers had difficulties more but they continued to work during the lesson. I felt that every lesson was in the mode of a competition. I observed that for students getting high scores in each activity was the major purpose. They did not care much about getting the information about programming as much as the scores they got. Some students came after the lesson and stated their feelings about getting low scores in the activities. Losing 1 or 2 points was a very big problem for them because the comparison between the classmates. Some parents also came to school who asked for higher grades for their students. Getting high scores from programming activities affected the motivation of the students in other activities. I observed that when the students got high grades in the programming activities their confidence on programming and their enthusiasm was increasing.

During the programming activities in the Information Technologies and Software lesson, students worked enthusiastically. The student liked the Scratch programming environment friendly interface. Although, the students complained about the difficulties of programming activities, they continued to work hard during the lesson. They wanted to achieve the activities so they did not give up the activities.

CHAPTER 5

DISCUSSION

This chapter discusses the findings of the study based on the research questions. The findings of programming confidence and programming achievement of pair and individual programmers and the relationship between programming confidence and programming achievement of the students are explained, interpreted and discussed in the light of the literature. The chapter also presents implications of the study including practical implications and theoretical implications, and recommendation for further research.

5.1 Programming Confidence and Programming Achievement of Pair and Individual Programming Students

The students reported that solving programming problems mostly influenced the programming confidence of pair-programming students. Both pair programmers and individual programmers thought that the reason of increasing or decreasing of the programming confidence was influenced by working with a pair. It can be said that the pair programming technique was effective for the solution of the programming problems with the most important factors of sharing knowledge, helping each other, correcting mistakes.

Sharing Knowledge

The findings of the study suggested that solving the programming problems by sharing knowledge stated as the factors of influencing programming achievement and the programming confidence of the students. The pair programming students mostly stated that they solved the programming problems by sharing the knowledge while the individual programming students wished to work with a partner in the solution of programming problems. The individual students also stated that the

partner may find solutions of the problems because the partner may know things that they did not. The achievement scores and the confidence scale scores results suggested significant difference between pair and individual programming students in favor of pair students. Although the present study suggested that pair students shared the knowledge during the programming process and the confidence and achievement of pair programmers were high, the study by Zieris (2015) resulted that pair students did not understand each other in programming process and they did not share their programming knowledge accurately. The difference might be occurred because of level of the students in pair groups. In the present study the level of students were the same in pair programming groups and the pair students did not report any communication and understanding problem during the semester.

In the literature, pair programming student shared the programming knowledge with each other which provided better learning (Dongo, Reed, & Hara, 2016). The present study also suggested the same conclusion with the literature that the pair programming students shared their knowledge with each other so they could achieve more in this way. Similar with the present study, the pair programming students had more fun while sharing knowledge with their partners and working collaboratively (Isong et al., 2016). The content analysis result about emotions during programming also suggested that the pair students felt fun while working with a classmate. The cooperative learning also was a way of working together to solve the problems by sharing knowledge (Sharan & Sharan, 1987). The cooperative learning enhanced the learning of the students by sharing their knowledge and experiences in a social platform (Bailey & Africa, 2017). It can be said that sharing knowledge during the programming process may enhance the learning of the programmers. Similar with the literature, the pair programming students' responses in the present study may suggest that pair programmers solved programming problems by sharing knowledge to achieve the goals of the activities. The pair programming students also suggested that their knowledge of programming increased by sharing their knowledge with their pairs. However, the study of pair programming technique revealed that although pair programming students may have larger knowledge about programming by sharing knowledge with a classmate, the knowledge that they got could not be deeply assimilated by both pairs (Ally, Darroch, & Toleman, 2005). Similarly, in the present study, the pair-programmers reported higher levels of learning because of sharing

with their pairs, and their rubric scores were significantly higher than individual students. However, their individual performances were not obtained in this research, and therefore, it is possible that some of the students in pair groups may not achieve as much as they achieved as a pair. On the other hand, pair-students' reports on increased learning may be due to the fact that they were novice programmers and they were learning the basic programming concepts and programming logic. As there are many topics to learn, the students may perceive they learned a lot.

Although most of the pair programming students responded that they shared the knowledge in programming problems, some students in both pair and individual programmers responded that the important thing was learning the content of the activities in the first lesson from the teacher and they mentioned that when someone could learn the necessary codes in the first hour, they could complete the activities. It can be said that some students may think the way of solving the programming problems was listening the content of programming in the first lesson of each week. The reason for this result may be due to lack of knowledge or experience on the programming subject. The students may complete the activities individually but being a novice in the programming context may cause them to prefer working with a partner and share the knowledge with a classmate.

Helping Each Other

One of the most important findings of the study suggested that getting help from a classmate or teacher eases the solution of problems in computer programming. The pair programmers responded that solving the complicated programming problems was easy with the help of the classmate. The students stated that the partners could find solutions that they did not solve in the programming activities. The confidence of pair programming groups was significantly different from the individual programmers that the items in the confidence scale evaluated high in the pair programmers. The responses of the students in the interview suggested that getting help of a classmate increased their confidence during the programming activities. The well-designed pair programming environment provided better learning by pairs helping each other while learning the programming context (Cliburn, 2003). The studies supported the present study with similar results that, the individual programming students also needed more help than pair programming student because

pair programming students could complete the programming activities by helping each other (Faculty, 2016). The socialization also occurred between the students since they communicate with each other to get help for the programming concepts (Zhong, Wang, & Chen, 2016). According to the responses of the students in the present study, partners highlighted that the success of implementing pair programming technique depended on the better communication of the pairs. Some students stated that they worked with their classmate by talking and sharing ideas. However, Williams et al. (2002) indicated that the miscommunication between the pairs sometimes caused the problems among the pairs and the students could not achieve their common goals. In contrast with the Williams' results in the literature, the presents study suggested that the communication between the pairs provided to complete programming activities successfully. The pair programming students also responded that the communication between the pair groups provided their friendship closer than before. Different from the literature, the pair students did not reported any miscommunication with their classmates in programming process. Bevan et al. (2002) also resulted that creating codes was difficult in pair programming because of the organization of pair groups and arranging the time for both of the pairs. The present study was differentiating from Bevan's research that the arrangement and organization of the pairs was done by the teacher and the pairs did not have any problems in organization of pair programming. The difference might be occurred because of the age and the grade level of pair programmers. In adult education, students may arrange and organize the pair programming sessions by themselves but in the present study the students were 5^{th} graders and the organization of the pair programming technique was made by the teacher.

The cooperative learning instructional method supported the pair programming technique in terms of the key elements of face to face promotive interaction, group processing and development of team work Johnson, Johnson, and Smith (1998) resulted that the key elements of cooperative learning enhanced the pair working to provide achievement of the students. The achievement scores suggested that the significance difference was occurred between the pair programming and individual programming students that pair programming students were achieved more in programming activities; the interview result also suggested that the students stated that they achieved more and solves programming problems with the help of their

partners. The study may suggest that the students mainly focused on the needs of a classmate in order to complete the programming activities with success.

Correcting Mistakes

According to the findings of the study, pair programming was effective technique that provides the opportunity of correction of mistakes in the codes easily. The programming students stated that they had some difficulties in programming activities especially in the last activities and several mistakes were done during the programming process. The individual programmers stated that they had several mistakes in the logic and sequence of the codes. Moreover, the pair programmers also indicated that complicated codes caused to make some mistakes in code blocks. Similar with the present study, the implementation of pair programming technique resulted that that the code correctness of pair programmers was higher than individual programmers since the mistakes were corrected by the partners' suggestions (Isong et al., 2016). The pair programming technique created fewer bugs in the codes (Begel, 2008). Declue (2003) also indicated similar result with the present study that pair programmers corrected the mistakes of each other and produced quality codes. Although the pair programming technique reported as a useful technique while correcting the mistakes of the programmers, some of the studies presented different result from the present study. One of the study resulted that the conflict between the pair programmers caused the ineffective working environment (Williams et al., 2002). The study Williams and Upchurch (2001) also indicated that pair –pressure between the pairs caused to work individual working in pair programming groups. Moreover, Arisholm (2007) resulted that pair programmers did not create codes without any mistakes. However, the present research suggested different result that the pair programming technique was useful for the completion of programming activities in terms of catching the mistakes of each other and correcting them quickly. The individual programming students also responded that they had difficulties on programming activities they could not find the mistakes of the code blocks by themselves. The result might be showed that there might be correlation between agreement of pairs and their confidence and achievement in computer programming. The pair pressure and conflict between the pairs in the present study might be prevented by teacher by observing the students'

agreement and checking the role of the students during the programming process. This may decrease the communication problems of the pair groups.

In the confidence scale, the related item which was “although I handle most of the courses I make mistakes absolutely on the problems of programming” with the correction of mistakes suggested that the individual students made more mistakes in programming problems than pair students. Pair programming students’ confidence was higher than individual programmers. It can be also said that some of the individual programming students may feel helpless in the complicated programming problems so the students may think that the pair programming technique was unique way of correction of mistakes in programming process. The programming confidence of the individual students may also decrease due to not being able to correct their mistakes. Although majority of the students believed that the pair programming technique helped them in the correction of mistakes in the code blocks, a few students indicated that if they had more information about programming or concentrate on the first lesson more, they would have completed the programming activities without mistakes either in pair groups or individually. Since all of the participants were novice, this response of the students might be due to their need for information on computer programming topic.

Although both of the groups made mistakes in the code blocks, the programming achievement of the pair programming students was significantly different from the individual programmers. The findings may suggest that computer programming was complicated for the novice programmers and the mistakes occurred because lack of knowledge about programming. The mistakes may also be caused by the limited resources that the students get during the programming activities. The students might be more successful in the activities but the searching Internet or a book was not allowed during the implementation.

Secondly, the programming process of pair students and individuals were influenced by the factors which were task competition time, quality of products and motivation. In the programming process, the students mentioned how the factors influenced their confidence.

Completion Time

According to responses of the students, completing the programming activities quickly improved the programming confidence of the students. The interview result of the present study suggested that the time was essential factor for completing the programming activities. The study suggested that pair programming students completed the activities quickly because the partners helped and solved the problems during the programming activities. A study in the literature showed that the pair programmers could complete two activities as spending 15% less time but the solo programmers used this time only for one activity (Williams & Kessler, 2000). The Williams's and Kessler's research supported the present study in terms of pair programmers finished the activities faster than individuals. However, the study by Plonka and Sharp (2012) resulted that the time pressure affected the motivation of the pair programmers negatively during the programming sessions because forming the groups as one novice and one expert programmers. In contrast with the study of Plonka and Sharp (2012), the present study suggested that the students finished the programming activities quickly with their partners and the motivation of the pair programmers were high. The reason might be the differences are capabilities between the expert and novice programmers but all of the students were novice programmers in the present study and they had equal knowledge about programming. Moreover, the study by Alves, Salge, and Berente (2016) resulted that although the observed result suggested that similar result with the present study that the pair programming students wrote the codes quickly, the statistical result indicated that there were no significant difference between the pair and individual programmers in terms of completion time. Furthermore, wasting the time on computer programming enhances the learning of programming concepts and this might be beneficial for the students in the process of creating quality products (Mcchesney, 2016). The suggestion of this study in the literature might be beneficial for the students but the present study was observed the pair and individual students in limited time period. The activity duration was 40 minutes in the present study and it cannot be extended due to being in the school schedule for students' other courses.

Many individual students reported in the interview that they could not finish their activity on time and got low scores, while pair-programmers finished their activities fast and with ease. Different from the individual students' responses in the present

study, the study conducted by Lewis (2011) 6th grade individual programmers completed the programming activities faster than pair programmers because pair programmers wasted their time for communication. Moreover, the study of Swamidurai and Umphress (2015) also presented different results that the completion time of the programming activities was not reduced with the pair programming technique. The difference between the present study and literature might be due to the characteristics of students and their experience on programming. The 5th grade students did not report any communication problems between the pairs and duration problem in programming activities. The importance of getting good grades among the 5th grade students might provide different results from the literature. Students may work hard to get good grades with their partners and this may provide them to finish the activities on time or before the given time. The individual students reported that the programming achievement was decreasing while not completing the programming activities on time. It was revealed that the students compared their pace and scores with the pair programmers. This may have caused to decrease programming confidence of the individual students.

Quality

Improving the quality of the product was one of the findings of the study which influenced the programming confidence of students. The responses of the students suggested that the pair-programming students produced quality products since they worked with their partner. The result highlighted that producing quality products depended on working with their partners. The studies of Nilsson (2003); Williams and Kessler (2002) stated that pair programmers created quality programs with the usage of pair programming technique. Moreover, the similar result with the present study showed that the pair programming students produce more quality products than individual programmers with less errors in codes and less codes in the software (Faculty, 2016). Furthermore, the study conducted by Salleh et al. (2011) correlated with the present study that benefits of pair programming technique indicated as productivity on technical issues, quality of design and programming and the achievement of programmers. The cooperative working provided to produce high quality codes with pair programming (Begel, 2008). While the present study were supported by several researches in the literature, Ally et al. (2005) resulted that the

quality of programs that pairs produced were lower quality products than individual programmers' because the management of the codes with a partner caused some problems. One possible reason for the conflicting results of this study with the present study might be due to the programming environments. The current study was implemented on the Scratch programming environment in which students create code blocks by drag and drop method. The other reason for this different result might be difference in the knowledge, experience, and age of students in pair groups.

The programming confidence scale suggested in the related items which was about creating complex programming activities with a success that pair programming students had more programming confidence in the last confidence scale results. The achievement score of the students suggested that the pair programming students achieved more and most of the pair programmers completed most of the programming activities without any mistakes. This result may suggest that the pair programming students produce quality products by working with a partner.

Apart from the suggestion of producing more quality products by pair programming technique, a few pairs in pair groups stated that they produce quality products because they knew more information than their partners. However, these students stated that if they had a chance to choose who their partners would be during the programming process, they would prefer to work with a partner although they knew more than their partners. They explained that even though they knew more than their partners; they felt more confident while someone sitting near to them. It can be revealed that some students may prefer working with a partner to feel more confident. This might be reason of being novice in computer programming. The programming confidence of pair programming students was high, so this may be an essential factor.

Learning

The findings of the study suggested that the implementation of the pair programming technique enhanced the learning of students in programming. The pair programming students stated that they learned more with the help of their partners. Students shared knowledge, questioned, guided each other while creating code blocks. The literature supported the present study that the students worked together and answered the questions of their projects, so the pair programming students understood and succeed

in most of the programming concepts (Cockburn & Williams, 2000). Moreover, the study by Hulks and Abrahamsson (2005) also presented similar result with the present study that pair programming technique was beneficial in a learning environment in terms of solving the programming problems, correction of mistakes, and creating complicated codes. Similar to the present study, the integration of pair programming technique into programming course provided to enhance learning of programming, and beneficial for the achievement of programmers in the programming courses (Isong et al., 2016). Moreover, the present study suggested that helping each other in programming process improved learning programming concepts, Dongo et al.(2016) also supported the present study that the pair programming students learned more by helping each other than individual students although the individual students was the hard worker students of the class. In the present study, the programming achievement of the pair programmers was found high and programming confidence scale also suggested that the confidence of pair programmers was high. Similar results revealed by Williams and Kessler (2000) that since the programming students was highly achieved with the pair programming technique, they had more confidence in the programming courses. With this literature, it can be said that there might be relationship between programming confidence and programming achievement of pair programmers'.

These findings may suggest that the novice programmers may needed to work with a partner in programming to be more confident and successful early in their programming experience. Since they may not be prepared for the programming problems yet, the pair programming technique may be a useful technique for better learning in computer programming. However, Braught, Eby, and Wahls (2008) resulted that there was no significance difference between the pair programmers and individual programmers' achievement score. Different from this literature, the present study suggested that pair programming students achieved more and they stated that they learned more codes with their pairs. The difference might be occurred because of the experience of students on computer programming. The pair programmers did not have any programming experience in present research so working with a classmate may provide better learning.

Motivation

The responses of the students showed that motivation was one of the essential factors for the improvement of the programming confidence and programming achievement of the students. The pair programming students responded that their motivation was high since they worked and completed the programming activities quickly. However, most of the individual programmers responded that the motivation of them was low because the pair groups completed the activities quickly and with fewer errors. The study may suggest the motivation was related to the programming achievement and programming confidence. Low motivation may be influenced by low achievement and in turn, it may result in low achievement and decreased confidence. The programming achievement scores of the pair and individual students may support the responses of the students that individual students achieved less than pair programmers in the activities. Since the motivation is an essential factor for learning, the getting low achievement may influence the motivation factor directly. The results of the study by Isong et al. (2016) supported the present study in which the implementation of pair programming provided significance difference on the motivation of the students in programming courses. Moreover, the study of Salleh et al. (2011) also stated the similar result with present study that pair programmers could create successful programs in terms of each pairs had same levels of programming skills and motivation. Different from the literature and present study, the result of an earlier study (Ally et al., 2005) showed that the method of forming of pair groups was vital for the motivation of the programmers since one of the students in pair programming groups may decrease the motivation of the pairs. The current study suggested different findings that the pair programming students worked harmoniously and there were not any reported problems of motivation differences among pairs for pair-programmers. The different result may occur due to the characteristics of the students in pair groups. The 5^{th} grade students in present study used to work each other so they worked harmoniously. In contrast with the present study, some students were unmotivated in pair groups because they could not adapted to each other and to the course (Hanks, 2004). In the present study, students could not report any problems about adaptation to their pairs or the course. This difference might be occurred because the 5^{th} grades students accustomed to each

other. Since the pair programming technique implemented in the second semester, the 5th grade participants were also accustomed to their teacher and the course.

According to responses of the individual programmers, it can be said that the individual programmers had high motivation at the beginning of the lesson but after the difficult activities their motivation decreased and they attributed this to their lack of help with a classmate. The students mainly claimed as a reason of low motivation was deficiency of a partner. The only difference between the two groups (pair programmers and individual programmers) was that pair programming students had partners during the programming process. As noted before the students completed the programming activities with the knowledge of their teacher gave them in the first lesson of the week and searching internet or asking to the teacher was forbidden while completing the activities. As individual students felt lack of resources during the difficulties of solving the problem, they wanted a pair. Although most individual students reported that they needed a partner, some other individuals reported the need for additional resources. The students stated that if they had more information and resources they would become more successful in the programming activities. It can be inferred that the students accustomed to research the unknown information in the portal of EBA which is prepared by Ministry of Education in Turkey for most of the lesson. The students may become more achieved and confident by searching the usage of code blocks in EBA and Scratch web platform. They also may not need help of a classmate by acquiring the resources about the computer programming.

Thirdly, being pair programmer or individual programmers influenced the programming confidence and programming achievement of the students. The interview result reported that disagreements, programming ability differences and workload were the factors that the programming confidence and programming achievement were influenced.

Disagreement

Agreement is important factor for the pair programming technique. The findings of the result suggested that the pair programming students mostly agreed with their partners during the programming process but disagreements occurred among the pairs mostly in the visual design of the Scratch environment. The individual programmer students did not mention any agreement or disagreement issue since

they had to study alone. However, the majority of studies in the literature were differentiating from the present study. According to Begel (2008) pair programmers had difficulties on agreement with the partner and the pair programming students wasted their time for deciding which decision was applied. The study of Begel (2008) also indicated that working with a partner who had difficult personality caused to occur discussions between the partners. The disagreement between the driver and navigator occurred while creating syntax and spelling error that the pair programming groups discussed to decide the accurate codes (Bryant, Romero, & Boulay, 1999). It can be inferred from the suggestion of the interview result of the study that since the students encountered with the computer programming first time in the 5th grade, they may fail to complete some parts of the code blocks. For the reason of this, they might allow to their partners to implement their ideas to complete the programming activities. As stated before, the participating students were concerned about the grading a lot during the implementation. Therefore, they might unwillingly compromise to successfully complete the activity on time. Even when they had disagreements about the design, for the sake of completing the activity on time, they may compromise.

Although the interview of the study suggested that the pair programming students agreed on the creating of code blocks, they had disagreements on the design or the programming environment in Scratch. It can be said that the students may care about the coding part of the study than design part because the students focused on getting higher grades from the activities. The reason behind this priority might be the distribution of points in the achievement rubrics in which the coding part provided more point than designing part. According to achievement rubrics, the programming achievement was higher in the students whom worked with their partners harmoniously and the programming confidence also high in these students. The programming confidence scale results suggested that the pair programmers were more confident than individual students. The results were also supported by the suggestions of interview of the students in this term. In contrast with the present study, Ally et al. (2005) revealed that individual programmers were more confident while making decision about the usage of coding and taking responsibilities about their decisions. This opposite result may be occurred in terms of the 5th grade students had disagreements on design part of the projects so the role of creating code

blocks was much more essential for the completion of programming activities successfully. Disagreement was not a big problem for the pair students in the present study because most of the disagreements occurred in design part but the programming confidence of the students may decrease due to disagreements between them regarding coding if one of the pair programming students or both of them were expert in computer programming.

Workload

The findings of the study suggested that the workload problem was a factor that influences the programmers negatively during the programming process. The interview result may suggest that pair programming students did not mention any workload problem during the programming process and they reflected that they worked with their partner mostly harmoniously. On the other hand, the workload problem was mentioned by the individual programmers. The individual programmers stated that they had to work more to solve their programming problems. As different form the interview result of the study, some of the research studies in the literature resulted that the pair programming students also coped with the workload problems. McDowell, Hanks and Werner (2003) stated that one of the pair students worked more and complete the most of the steps in the programming projects while other partners work less or any. Williams et al. (2002) also resulted that when the students in pair programming worked on the same role throughout the programming process, unbalanced work shared between the partners occurred. According to Williams et al. (2002) the students did not gain some programming skills because one of the partners worked more on the programming process. The interview results of the present study suggested that the pair-students did not report any unbalanced work for the completion of the activity while they sometimes reported their different levels of knowledge. The reason behind the differences between the interview results and some research studies in the literature may be explained due to differences of age, knowledge, and skills. While the 5^{th} grade students were all novice programmers and know little about programming other than listening to their teacher in the first lesson, their knowledge levels and abilities were not expected to be too diverse.

Goos, Galbraith, and Renshaw (2002) indicated that the pair programming develops the metacognition of the programmers when the programming students observe their

partners, analyze the process of programming and got information about the methods of correcting the mistakes in codes. Similar with literature, the present study suggested that the pair groups enhanced their learning by sharing knowledge, observing the partner and correcting the mistakes in code blocks. Moreover, the individual students also highlighted the importance of working with a partner during the programming process. The literature and the present study correlated in the improvement of students' metacognition in the novice pair programming groups. However, this result might be different in the expert pair programming groups. Since all of the students were novice in the present study, the students had to work harmoniously, observe each other or correct the mistake of each other to complete the programming activities successfully.

It can be inferred from the interview of the students that the 5^{th} grade students stated that the computer programming was difficult and they learned the programming subject first time, so working with a partner may provide more programming confidence and more programming achievement in their learning process. The study Bailey and Africa (2017) resulted that the IT teachers mentioned the significance of working pairs in programming by helping each other and sharing their knowledge but students wanted to work individually to experience the computer programming on their own. The present study suggested that although the students wanted to use the computer individually at the beginning of the year, they stated that they preferred to work with a partner during the programming activities because the pair groups coped with the problems of programming and they shared the works in the activities. However, it is possible that the students might prefer to work alone again after they have experiences with a pair. Therefore, instead of forming individual and pair programming in class for a long period of time, the formation can be changed regularly so that the individual students experience pair-work and pair-students challenge themselves individually.

The majority of the students in the current study mentioned that they needed to work with a partner because they had limited knowledge in computer programming. The cooperative learning instructional method also supported this claim in terms working in a group to enhance the programming learning of the students (Rogoff, 1998). The confidence scale in the present study also suggested that the programming confidence of the pair programmers were higher than individual programmers. The

confidence scale items of the scale suggested that the pair programming students had more programming confidence because they shared the works and they created the complicated codes easily. Moreover, the results of achievement rubrics indicated that the pair programming students achieved more. Sharing the works in the programming activities may provide more achievement since the students had chance to think and correct the mistakes of their partners.

5.2 Conclusion

This case study explored the influences of pair programming technique on secondary school students' programming confidence and programming achievement. According to results of the current study, pair students perceived that the pair programming technique enhanced their computer programming learning and improved their programming confidence. The pair programming students felt more confident than individual programmers. They commented that sharing knowledge, helping each other, correcting mistakes in code blocks were the most important factors for the improvement of their programming confidence and programming achievement.

Both of the groups (pair and individual) mostly preferred working with a classmate because the computer programming was a new subject for them and the students accustomed to work in groups in their courses. Most of the novice programmers commented that solving programming problems was a big problem while completing the programming activities and the getting help of a classmate was the common response of the students for the solution of programming problems. Alternative resources for solving programming problems were preferred less than working with a classmate from the students in the process of programming. The support of a classmate for the novice programmers of 5^{th} graders accepted as the most important factor that influenced their programming confidence and programming achievement.

There were also some positive and negative outcomes of working individual and pair in computer programming activities. The motivation, learning, quality of products and the completion time were the essential factors that influenced the programming confidence and programming achievement of the students. The result of the study indicated that the learning process of the students in computer programming was enhanced by the pair programming technique. The pair programming students commented that they learned more code blocks with the help of their partners. On the

other hand, the individual programming students reflected their needs of help and resources on the solution of programming problems. The needs of a partner occurred when they encountered difficult programming problems while they previously wanted to sit alone and use the computer for themselves. The difficulties in the code blocks compelled to students worked with a classmate. Most of the students commented that they preferred working in groups because their classmates could solve the problems that they could not solve.

The disagreement and workload of the pairs were the negative factors that influenced the pair and individual programming students during the programming process. The disagreement of pairs accepted as essential problems of the pair programming technique according to literature. However, the pair programming students did not mention that the disagreement problem as important as the literature. The current study resulted that the disagreement mostly occurred in pair programmers in the design of the Scratch working environment and not with the coding part which was essential for the success of the students in the course. The workload occurred some of the pair programming studies in the literature but in the present study the individual programmers had to work more than pair programming students. The reason explained in the result of the study as individual students had to try more codes to complete the programming activities.

5.3 Implications of the Study

The current study presents a number of implications for practice, theory and research. The practitioners of the study are students and teachers who deal with computer programming education. The influences of pair programming technique on programming achievement and programming confidence of secondary school students research subject also presents some implications for theory and research for the literature. The major implications of the current study may be providing deep understanding about the influence of pair-programming technique on programming education of secondary school students.

5.3.1 Implications for Practice

Since the method of the study is case study about the possible influences of pair programming on programming confidence and programming achievement of secondary school students, it may not be generalized to other settings, but practitioners can utilize the practices and research results in this study. Firstly, the findings of the study provided an understanding of the experiences of the pair programming students and individual programming students in several factors. The pair programming technique indicated in the study that the programming confidence and programming achievement of pair programming students influenced positively so the other computer programming teachers can benefit from the technique in the education of computer programming. The sitting scheme of the students could be designed according to pair programming technique implementation. The computer literacy teachers also can compare their instructional methods, technique or model with the pair programming technique and they can combine the pair programming technique in the education of computer programming.

Secondly, the factors of sharing knowledge, helping each other, correcting mistakes were the important factors for the solution of programming problems in the study. The teachers of computer literacy lesson, students who learn computer programming can benefit from the experiences of the pair programming students to determine beneficial way for the solutions of the problems that occur during the programming session. The comments of both pair and individual programming students can provide to create a framework for the education of computer programming for the teachers.

Finally, the usage of visual programming environment and pair programming technique can provide a new way for the computer programming education. The study indicated that the technique motivated to students in terms of getting support from a classmate. Moreover, learning of computer programming can facilitate with the usage of the technique. Thus, the teachers can evaluate the findings of the study from other studies and can have broad point of view about the different methods, techniques and strategies. This may be beneficial for the improvement of the teachers in the education of computer programming.

As stated in my personal reflections at the end of the “Results” section, before the start of the implementation, the students were very selfish in terms of their desire to sit on the computer alone, using it selfishly without allowing their pairs to use it fairly, and did not want to cooperate. However, the reports of the students showed that the students finally recognized the advantages of sitting together, working cooperatively on the same task, and most importantly sharing knowledge. The implementation had a positive influence on the students’ attitudes toward cooperation and sharing in my class. Therefore, the teachers who have similar student attitudes in class may utilize pair-programming technique not only to help them learn from each other, but also to appreciate the value of cooperation.

5.3.2 Implication for Theory and Research

The study was implemented to understand the influences of the pair programming technique on confidence and achievement of secondary school students on computer programming. Since the pair programming technique was commonly used in software industry and adult education, the limited research studies were available for practice in the secondary school students’ education. The study showed that the pair programming technique was coherent with the cooperative learning method in terms of the 5 key elements of cooperative learning. This contributed to analyze the pair programming method with an instructional method of cooperative learning. The study contributed to literature by providing deeper knowledge about the experiences of pair and individual programmers. Moreover, the study supported to the benefits of pair programming on the secondary school students learning, confidence, motivation and achievement.

The study may provide to close the gap in the literature about the secondary school students’ programming education with the pair programming technique. Since the limited studies were implemented for the secondary school students, the case study may provide to examine the experience of the pair and individual programmers deeply. Since the pair programming technique was mostly used in the software development process, the studies commonly focused on software quality, cost, and time factors. The study also may close the gap that the usage of pair programming in education by examining the different factors such that learning, sharing knowledge,

motivation. Finally, the study also may guide for the similar research studies on the understanding of influences of the pair programming technique for K12 students.

5.4 Limitations of the Study

The study had some limitations which are listed below;

- This study did not measure pair students' individual achievements after the implementation. Rather, the study aimed to compare the achievement of group and individual programmers with regard to the quality completion of the given tasks.

- The students' opinions might be influenced from each other because they were in the same classroom during the implementation. These opinions may also influence their achievement and confidence. Most individual students envied pair-students because of their success and completion time. However, this study which is mainly a qualitative research focused more on how these two groups experienced the implementation and whether their confidence and achievement changed after the implementation. This situation may be inevitable even without the special arrangement if the number of students is more than the available number of computers in the classroom. Therefore, exploring their experiences of being in the same environment of both individual and pair programmers were also valuable, because the students form their opinions by observing others, instead of experiencing only their own unique situation.

- The result of study is only limited for the one school and one class in Turkey. Design of the research was case study and the results might not to be generalized to in other contexts. However, it provides a description of the potential of pair-programming in novice programmers' confidence, possible influential factors, and achievement in one classroom to help especially practitioners and teachers to make informed decisions while designing their own programming courses.

- During the interviews, the students tended to give direct and short answers to the questions of interview due to their age level and therefore, the depth of their explanation was low.

5.5 Recommendation for Further Research

The study aimed to understand the influence of pair programming technique on programming confidence and programming achievement of secondary school students' in a detailed way. The process of the research showed that some recommendation can be done for the further research studies.

- The alternative research studies can be implemented for the pair programming students' individual achievement and individual confidence in order to understand the influences of the pair programming technique.

- In the study the pair programming and individual programming students worked in the same classroom so the further research studies can be implemented from two different classrooms to prevent the influences of two groups of students from each other.

- The design of the study was case study so the generalization could not be made in the finding of the study. The further research studies can use other research designs to make generalization and comparison from the findings.

- Scratch visual programming environment was used for the study; instead of Scratch programming environment, different programming environments can be used for the implementation of the pair programming technique.

- The participants of the study were secondary school students and the 5^{th} grade level was used for the study. Similar studies can be implemented with different levels of participants and in different contexts.

REFERENCES

Akpınar, Y., & Altun, A. (2014). Bilgi toplumu okullarında programlama eğitimi gereksinimi. *İlköğretim Online 1*(13), 1-4. Retrieved from http://ilkogretim-online.org.tr/index.php/io/article/view/2099

Alice.org. (2017). About Alice. Retrieved October 17, 2017 from http://www.alice.org/about/

Ally, M., Darroch, F., & Toleman, M. (2005). A framework for understanding the factors influencing pair programming success. *Lecture Notes in Computer Science*, 82–91. DOI:10.1007/11499053_10

Ambrósio, A. P., Costa, F. M., Almeida, L., Franco, A., & Macedo, J. (2011). Identifying cognitive abilities to improve CS1 outcome, 1–7. *Frontiers in Education Conference* (FIE). DOI:10.1109/fie.2011.6142824

Arisholm, E., Gallis, H., Dybå, T., & Sjøberg, D. I. K. (2007). Evaluating pair programming with respect to system complexity and programmer expertise. *IEEE Transactions on Software Engineering*, *33*(2), 65–86. DOI:10.1109/TSE.2007.17

Bailey, R., & Mentz, E. (2017). The value of pair programming in the IT classroom. *The Independent Journal of Teaching and Learning*, *12*(1), 90-103. Retrieved from http://journals.co.za/docserver/fulltext/jitl1_v12_a8.pdf?expires=1519340022&id=id&accname=guest&checksum=BA96B5C0A0D1E75141D8984D80DC0DB3

Bilişim Teknolojileri Eğitimcileri Derneği, BTE. (2013). Bilişim teknolojileri derslerinin tairihi Türkiye'de ilk okul ortaokullarda (ilköğretim) okutulan. Retrieved from https://bte.org.tr/bt-derslerinin-tarihcesi/

Beck, K. (2000). *Extreme programming explained: Embrace change.* Reading, Mass., Addison-Wesley.

Begel, A., & Nagappan, N. (2008). Pair programming : what's in it for me ? *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, 120–128. DOI:10.1145/1414004.1414026

Berenson, S. B., Slaten, K. M., Williams, L., & Ho, C. W. (2004). Voices of women in a software engineering course: reflections on collaboration. *Journal on Educational Resources in Computing (JERIC)*, *4*(1), 3. DOI:10.1145/1060071.1060074

Bevan, J., Werner, L., McDowell, C. (2002). Guidelines for the use of pair programming in a freshman programming class. *15th Conference on Software Engineering Education and Training*, 100-107. DOI:10.1109/csee.2002.995202

Braught, G., Eby, L. M., & Wahls, T. (2008). The effects of pair-programming on individual programming skill. In *ACM SIGCSE Bulletin,* 40(1), 200-204. DOI: 10.1145/1352322.1352207

Brennan, K., & Resnick, M. (2013). Imagining, creating, playing, sharing, and reflecting: How online community supports young people as designers of interactive media. *Emerging technologies for the classroom* 253-268. DOI:10.1007/978-1-4614-4696-5_17

Bryant, S., Romero, P., & Boulay, B. (2008). Pair programming and the mysterious role of the navigator. *International Journal of Human Computer Studies*, *66*(7), 519-529. DOI:10.1016/j.ijhcs.2007.03.005

Byrne, P., & Lyons, G. (2001). The effect of student attributes on success in programming. *In ACM SIGCSE Bulletin*, *33*(3), 49-52. DOI: 10.1145/507758.377467

Chang, C. K. (2014). Effects of using Alice and Scratch in an introductory programming course for corrective instruction. *Journal of Educational Computing Research*, *51*(2), 185-204. DOI:10.2190/EC.51.2.c

Clark, J., Rogers, M. P., Spradling, C., & Pais, J. (2013). What, no canoes? Lessons learned while hosting a scratch summer camp. *Journal of Computing Sciences in Colleges*, *28*(5), 204-210. DOI:10.2190/EC.51.2.c

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational psychology*, *76*(6), 1051. DOI:10.1037/0022-0663.76.6.1051

Cliburn, D. C. (2003). Experiences with pair programming at a small college. *Journal of Computing Sciences in Colleges*, *19*(1), 20-29. Retrieved from http://dl.acm.org/citation.cfm?id=948741&CFID=380881129&CFTOKEN=42051081

Cockburn, A., Williams, L. (2001). *The costs and benefits of pair programming. Extreme programming examined*. Addison-Wesley, Reading.

Code.org. (2014). About us. Retrieved october 25, 2017 from https://code.org/about

Cohen, J. A. (1960). *Coefficient of agreement for nominal scales*. Thousand Oaks, CA: SAGE Publications.

Creswell, J. W. (2007). *Qualitative inquiry and research design: Choosing among five approaches* (2nd ed.). Thousand Oaks, CA: SAGE Publications.

Çatlak, Ş., Tekdal, M., & Baz, F. Ç. (2015). Scratch yazılımı ile programlama öğretiminin durumu: Bir doküman inceleme çalışması. *Journal of Instructional Technologies & Teacher Education, 4*(3), 13-25. Retrieved from http://dergipark.ulakbim.gov.tr/nwsaedu/article/view/5000187408

Davidson, N., & Major, C. H. (2014). Boundary crossings: Cooperative learning, collaborative learning, and problem-based learning. *Journal on Excellence in College Teaching*, *25*. Retrieved from http://northweststate.edu/wp-content/uploads/files/BoundaryCrossings.pdf

DeClue, T. H. (2003). Pair programming and pair trading: effects on learning and motivation in a CS2 course. *Journal of Computing Sciences in colleges*, *18*(5), 49-56. Retrieved from https://dl.acm.org/citation.cfm?id=771843

Demir, Ö., & Seferoğlu, S. S. (2017). İşbirlikli problem çözmenin kodlama öğretimine yansıması olarak eşli kodlamanın incelenmesi. *5th International Instructional Technologies & Teacher Education Symposium* (ITTES 2017). Abstract retrieved from https://www.researchgate.net/publication/321824838_Isbirlikli_Problem_Cozmenin_Kodlama_Ogretimine_Yansimasi_Olarak_Esli_Kodlamanin_Incelenmesi

Demirer, V., & Sak, N., (2016). Programming education and new approaches around the world and in Turkey/Dünyada ve Türkiye'de programlama eğitimi ve yeni yaklaşımlar. *Eğitimde Kuram ve Uygulama*, *12*(3), 521-546. Retrieved from http://dergipark.gov.tr/download/article-file/262355

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

Dongo, T., Reed, A. H., & O'Hara, M. (2016). Exploring pair programming benefits for MIS majors. *Journal of Information Technology Education: Innovations in Practice*, *15*, 223–239. DOI:10.28945/3625

Doymuş, K., Şimşek, Ü. & Bayrakçeken, S. (2004). İşbirlikçi öğrenme yönteminin fen bilgisi dersinde akademik başarı ve tutuma etkisi. *Türk Fen Eğitimi Dergisi, 1*(2). Retrieved from http://efdergi.inonu.edu.tr/article/view/5000179999

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

Feurzeig, W., Papert, S. A., & Lawler, B. (2011). Programming languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments, 19*(5), 487-501. DOI: 10.1080/10494820903520040

Fraenkel, J. R., Wallen, N. E., & Hyun, H. H. (2000). *How to Design and Evaluate Research in Education*. (4th Ed.) Boston: McGraw-Hill.

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

Glesne, C., & Peshkin, A. (1992). *Becoming qualitative researchers*. White Plains, NY.

Gómez, O. S., Aguileta, A. A., Aguilar, R. A., Ucán, J. P., Rosero, R. H., & Cortes-Verdin, K. (2017). An empirical study on the impact of an IDE tool support in the pair and solo programming. *IEEE Access, 5*, 9175-9187. DOI:10.1109/access.2017.2701339

Gokhale, A. A. (1995). Collaborative learning enhances critical thinking. *Journal of Technology Education, 7*(1). DOI:10.21061/jte.v7i1.a.2

Goos, M., Galbraith, P., & Renshaw, P. (2002). Socially mediated metacognition: Creating collaborative zones of proximal development in small group problem solving. *Educational studies in Mathematics, 49*(2), 193-223. DOI: 10.1023/a: 1016209010120

Green, S. B., & Salkind, N. J. (2003). *SPSS for Windows and Macintosh: Analyzing and understanding data* (4rd ed.). Upper Saddle River, NJ: Prentice Hall.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38-43. DOI:10.3102/0013189x12463051

Hanks, B., McDowell, C., Draper, D., & Krnjajic, M. (2004). Program quality with pair programming in CS1. *ACM SIGCSE Bulletin, 36*(3), 176-180. DOI:10.1145/1007996.1008043

Hanks, B. (2006). Student attitudes toward pair programming. In *ACM SIGCSE Bulletin* 38(3), 113-117. DOI: 10.1145/1140123.1140156

Hannay, J. E., Dybå, T., Arisholm, E., & Sjøberg, D. I. (2009). The effectiveness of pair programming: A meta-analysis. *Information and Software Technology, 51*(7), 1110-1122. DOI:10.1016/j.infsof.2009.02.001

Holloway, I. (1997). *Basic concepts for qualitative research*. Wiley-Blackwell.

Hulkko, H., & Abrahamsson, P. (2005). A multiple case study on the impact of pair programming on product quality. *27th International Conference on Software Engineering,* 495–504. DOI:10.1109/ICSE.2005.1553595

Hwang, W. Y., Shadiev, R., Wang, C. Y., & Huang, Z. H. (2012). A pilot study of cooperative programming learning behavior and its relationship with students' learning performance. *Computers & Education*, *58*(4), 1267–1281. DOI:10.1016/j.compedu.2011.12.009

Idlbi, A. (2009). Taking kids into programming (contests) with scratch. *Olympiads in Informatics*, *3*, 17–25. Retrieved from http://educationdocbox.com/Homework_and_Study_Tips/66155166-Taking-kids-into-programming-contests-with-scratch.html

Isong, B., Moemi, T., Dladlu, N., Motlhabane, N., Ifeoma, O., & Gasela, N. (2016). Empirical confirmation of pair programming effectiveness in the teaching of computer programming. *2016 International Conference on Computational Science and Computational Intelligence* (CSCI). DOI:10.1109/csci.2016.0060

Johnson, D. W., & Johnson, R. T. (1989). *Cooperation and competition: Theory and research*. Edina, MN, US: Interaction Book Company.

Johnson, D. W., Johnson, R. T., & Smith, K. A. (1998). Cooperative learning returns to college what evidence is there that it works? *Change: the magazine of higher learning*, *30*(4), 26-35. DOI:10.1080/00091389809602629

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

Kalelioglu, F., & Gülbahar, Y. (2014). The effect of teaching programming via scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education*, *13* (1), 33–50. Retrieved from https://www.ceeol.com/search/article-detail?id=123695

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code.org. *Computers in Human Behavior*, *52*, 200–210. DOI:10.1016/j.chb.2015.05.047

Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming. *ACM Computing Surveys*, *37*(2), 83–137. DOI:10.1145/1089733.1089734

Lam, J., Li, K. C., Cheung, S. K. S., & Wang, F. L. (Eds.). (2013). Knowledge sharing through technology. *8th International Conference on Information and Communication Technology in Teaching and Learning*, *407,* 10-11. DOI: 10.1007/978-3-642-45272-7

Li, Z., Plaue, C., & Kraemer, E. (2013). A spirit of camaraderie : The impact of pair programming on retention. *26th International Conference on Software Engineering Education and Training* (CSEE&T), 209-218. DOI:10.1109/cseet.2013.6595252

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. Newbury Park, CA: SAGE Publications

Lewis, C. M. (2011). Is pair programming more effective than other forms of collaboration for young students? *Computer Science Education, 21*(2), 105-134.
DOI: 10.1080/08993408.2011.579805

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. DOI:10.1016/j.chb.2014.09.012

Maloney, J. H., & Smith, R. B. (1995). Directness and liveness in the morphic user interface construction environment. *Proceedings of the 8th Annual ACM Symposium on User Interface and Software Technology, 21-28.* DOI:10.1145/215585.215636

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., & Rusk, N. (2008). Programming by choice: Urban youth learning programming with Scratch. *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education, 40* (1), 367–371. DOI:10.1145/1352135.1352260

Maloney, J. H., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education (TOCE), 10*(4), 1–15. DOI:10.1145/1868358.1868363

Malterud, K. (2001). Qualitative research: standards, challenges, and guidelines. *The lancet, 358*(9280), 483-488. DOI:10.1016/s0140-6736(01)05627-6

Marshall, C., & Rossman, G. B. (1999). *Defending the value and logic of qualitative research. Designing qualitative research*, (3rd ed.). Thousand Oaks, CA : SAGE Publications

McChesney, I. (2016). Three years of student pair programming: Action research insights and outcomes. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, 84-89. DOI:10.1145/2839509.2844565

McDowell, C., Hanks, B., & Werner, L. (2003). Experimenting with pair programming in the classroom. *ACM SIGCSE Bulletin, 35*(3), 60. DOI:10.1145/961290.961531

McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM, 49*(8), 90–95. DOI:10.1145/1145287.1145293

McKinney, D., & Denton, L. F. (2006). Developing collaborative skills early in the CS curriculum in a laboratory environment. In ACM SIGCSE Bulletin, 38(1), 138-142. DOI:10.1145/1124706.1121387

MEB. (2012). Ortaokul ve imam hatip ortaokulu bilişim teknolojileri ve yazılım dersi (5, 6, 7 ve 8. sınıflar) öğretim programı. *Ankara: Milli Eğitim Bakanlığı Yayınları*.

MEB. (2018). Bilişim teknolojileri ve yazılım dersi öğretim programı (Ortaokul 5 ve 6. Sınıflar). Retrieved November 16, 2017 from http://mufredat.meb.gov.tr/ProgramDetay.aspx?PID=374

Mentz, E., Walt, J. L. Van Der, & Goosen, L. (2008). The effect of incorporating cooperative learning principles in pair programming for student teachers. *Computer Science Education 18(4),* 247–260. DOI:10.1080/08993400802461396

Merriam, S. B. (2009). *Qualitative research: A guide to design and implementation.* San Francisco, CA: Jossey-Bass.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook* (2nd ed.). Thousand Oaks, CA: Sage Publications.

Moskal, B., Lurie, D., & Cooper, S. (2004). Evaluating the effectiveness of a new instructional approach. *ACM SIGCSE Bulletin*, *36*(1), 75-79. DOI:10.1145/971300.971328

Muller, M. M., & Tichy, W. F. (2001). Case study: extreme programming in a university environment. *Proceedings of the 23rd International Conference on Software Engineering. ICSE 2001*, 537–544. DOI:10.1109/ICSE.2001.919128

Nagappan, N., Williams, L., Ferzli, M., Wiebe, E., Yang, K., Miller, C., & Balik, S. (2003). Improving the CS1 experience with pair programming. *ACM SIGCSE Bulletin*, *35*(1), 359-362. DOI:10.1145/611892.612006

Nawahdah, M., & Taji, D. (2016). Investigating students' behavior and code quality when applying pair-programming as a teaching technique in a Middle Eastern society. *In Global Engineering Education Conference (EDUCON),* 32-29. DOI:10.1109/educon.2016.7474527

Nicolescu, R., & Plummer, R. (2003). A pair programming experiment in a large computing course. *Romanian Journal of Information Science and Technology*, *6*(1-2), 199-216. Retrieved from https://www.researchgate.net/publication/228824452_A_Pair_Programming_Experiment_in_a_Large_Computing_Course

Nilsson, K. (2003). *Increasing Quality with Pair Programming* (Dissertation). Retrieved from http://urn.kb.se/resolve?urn=urn:nbn:se:bth-3964

Ota, G., Morimoto, Y., & Kato, H. (2016). Ninja code village for scratch: Function samples/function analyser and automatic assessment of computational thinking concepts. *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC*, 238–239. DOI:10.1109/VLHCC.2016.7739695

Özmen, B., & Altun, A. (2014). Undergraduate students' experiences in programming: difficulties and obstacles. *Turkish Online Journal of Qualitative Inquiry*, *5*(3), 1–27. DOI:10.17569/tojqi.20328

Panitz, T. (1999). Collaborative versus cooperative Learning: A comparison of the two concepts which will help us understand the underlying nature of interactive learning. Retrieved from https://eric.ed.gov/?id=ED448443

Patton, M. (1990). *Qualitative evaluation and research methods*. Beverly Hills, CA: Sage

Preston, D. (2006). Using collaborative learning research to enhance pair programming pedagogy. *ACM SIGITE Newsletter, 3*(1), 16–21. DOI:10.1145/1113378.1113381

Resnick, M., Kafai, Y., & Maeda, J. (2005). A networked, media-rich programming environment to enhance technological fluency at after-school centers in economically-disadvantaged communities. *Proposal to the National Science Foundation.* Retrieved from http://web.media.mit.edu/~mres/papers/scratch-proposal.pdf

ResourcEd. (2018). Collaborative learning vs. cooperative learning: what's the difference? Retrieved April 6, 2018 from https://resourced.prometheanworld.com/collaborative-cooperative-learning/

Rey, U., & Carlos, J. (2016). Code to learn with Scratch. *2016 IEEE Global Engineering Education Conference (EDUCON)*, 150–156. DOI:10.1109/EDUCON.2016.7474546

Rogoff, B. (1998). *Cognition as a collaborative process*. Hoboken, NJ, US: John Wiley & Sons Inc.

Rostaher, M., & Hericko, M. (2002). Tracking test first pair programming- an experiment. In *Conference on Extreme Programming and Agile Methods*, 174-184. DOI:10.1007/3-540-45672-4_17

Salge, C. A. de L., & Berente, N. (2016). Pair programming vs. solo programming: What do we know after 15 years of research? *49th Hawaii International Conference on System Sciences (HICSS), 5398-5406.* DOI:10.1109/hicss.2016.667

Salleh, N., Mendes, E., & Grundy, J. (2011). Empirical studies of pair programming for cs/se teaching in higher education: A systematic literature review. *IEEE Transactions on Software Engineering*, *37*(4), 509–525. DOI:10.1109/tse.2010.59

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using Scratch in five schools. *Computers & Education, 97*, 129-141. DOI:10.1016/j.compedu.2016.03.003

Sayginer, Ş., & Tüzün, H. (2015). İlköğretim düzeyinde programlama eğitimi : Yurt dışı ve yurt içi perspektifinden bir bakış. Retrieved from

http://ab.org.tr/ab17/bildiri/211.pdf

Scholz, R. W., & Tietje, O. (2002). *Embedded case study methods: Integrating quantitative and qualitative knowledge*. SAGE Publications.

Sharan, S., & Sharan, Y. (1987). Training teachers for cooperative learning. *Educational Leadership, 45*(3), 20-26. DOI:10.4324/9780203837290.ch20

Slavin, R. E. (1987). Developmental and motivational perspectives on cooperative learning: Reconciliation. *Child Development, 58*(5), 11-61. DOI:10.2307/1130612

Slavin, R. E. (2011). *Instruction based on cooperative learning*. *Handbook of research on learning and instruction* (4rd ed.). London: Taylor & Francis.

Swamidurai, R., & Umphress, D. (2015). Inverted pair programming. *SoutheastCon*, 1-5. DOI:10.1109/secon.2015.7133010

Tavakol, M. & Dennick, R. (2011). Making sense of cronbach's alpha. *International journal of Medical Education, 2*, 53-55. DOI:10.5116/ijme.4dfb.8dfd

Teague, D., & Roe, P. (2008, January). Collaborative learning: towards a solution for novice programmers. In Proceedings of the tenth conference on Australasian computing education, 78, 147-153.

Twycross, A., & Shields, L. (2004). Validity and reliability what's it all about? *Paediatric Nursing, 16*(10), 36–36. DOI:10.7748/paed.16.10.36.s22

Umapathy, K., & Ritzhaupt, A. D. (2017). A meta-analysis of pair-programming in computer programming courses: Implications for educational practice. *ACM Transactions on Computing Education, 17*(4), 1–13. DOI:10.1145/2996201

VanDeGrift, T. (2004). Coupling pair programming and writing: Learning about students' perceptions and processes. *In ACM SIGCSE Bulletin, 3*6(1), 2-6. DOI:10.1145/971300.971306

Waite, W. M., Jackson, M. H., & Leonardi, P. M. (2004). Student culture vs. group work in computer science. *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education, 3*6(1), 12-16. DOI:10.1145/971300.971308

Werner, L., & Denning, J. (2009). Pair programming in middle school: What does it look like? *Journal of Research on Technology in Education*, 42(1), 29-49. DOI:10.1080/15391523.2009.10782540

Wiest, L. R. (2004). Impact of a summer mathematics and technology program for middle school girls. *Journal of Women and Minorities in Science and Engineering, 10*(4), 317-339. DOI: 10.1615/JWomenMinorScienEng.v10.i4.20

Williams, L. A., & Kessler, R. R. (2000). The effects of pair-pressure and pair-

learning on software engineering education. *In Proceedings of the Conference of Software Engineering Education and Training*, 1-10. DOI:10.1109/csee.2000.827023

Williams, L. A., & Kessler, R. R. (2000). All I really need to know about pair programming I learned in kindergarten. *Communications of the ACM, 43*(5), 108–114. DOI:10.1145/332833.332848

Williams, L., Kessler, R., (2002). *Pair programming illuminated*. Addison-Wesley Longman Publishing, Boston, MA, USA.

Williams, L. A., & Kessler, R. R. (2003). *Pair programming illuminated*. Addison-Wesley Longman Publishing, Boston, MA, USA.

Williams, L. A., & Kessler, R. R. (2001). Experimenting with industry's pair-programming model in the computer science classroom. *Computer Science Education, 11*(1), 7-20. DOI:10.1076/csed.11.1.7.3846

Williams, L., Kessler, R. R., Cunningham, W., & Jeffries, R. (2000). Strengthening the case for pair programming. *IEEE Software, 17*(4), 19–25. DOI:10.1109/52.854064

Wilson, A., & Moffat, D. C. (2010). Evaluating Scratch to introduce younger schoolchildren to programming. *Proceedings of the 22nd Annual Psychology of Programming Interest Group (Universidad Carlos III de Madrid, Leganés, Spain*, 64-74. Retrieved from http://scratched.media.mit.edu/sites/default/files/wilson-moffat-ppig2010-final.pdf

Winslow, L. E. (1996). Programming pedagogy a psychological overview. *ACM SIGCSE Bulletin, 28*(3), 17–22. DOI:10.1145/234867.234872

Yerion, K. A., & Rinehart, J. A. (1995). Guidelines for collaborative learning in computer science. *ACM SIGCSE Bulletin*, *27*(4), 29-34. DOI: 10.1145/216511.216529

Yin, R. K. (2003). *Case study research: Design and methods* (3rd ed.). Thousand Oaks, CA: SAGE Publications.

Yin, R. K. (2009). *Case Study Research: Design and methods, essential guide to qualitative methods in organizational research* (4rd ed.). Thousand Oaks, CA: SAGE Publications.

Yıldırım, A. & Şimşek, H. (2008). *Sosyal bilimlerde nitel araştırma yöntemleri*, (6th ed.). Ankara: Seçkin Yayıncılık.

Yıldırım, A., & Şimşek, H. (2013). *Sosyal bilimlerde nitel araştırma yöntemleri* (9th ed.). Ankara: Seçkin Yayıncılık.

Yoshihara, K., & Watanabe, K. (2016). Practice of programming education using

Scratch and Nanoboardag for high school students. *10th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)*, 567–568. DOI:10.1109/CISIS.2016.118

Yoon, I., Kim, J., & Lee, W. (2016). The analysis and application of an educational programming language (RUR-PLE) for a pre-introductory computer science course. *Cluster Computing,* 19(1), 529-546. DOI:10.1007/s10586-016-0540-6

Yünkül, E., Durak, G., Çankaya, S., & Mısırlı, Z. A. (2017). Scratch yazılımının öğrencilerin bilgisayarca düşünme becerilerine etkisi. *Necatibey Eğitim Fakültesi Elektronik Fen ve Matematik Eğitimi Dergisi*, *11*(2), 502-517. DOI:10.17522/balikesirnef.373424

Zacharis, N. Z. (2011). Measuring the effects of virtual pair programming in an introductory programming java course. *IEEE Transactions on Education, 54*(1), 168-170. DOI:10.1109/te.2010.2048328

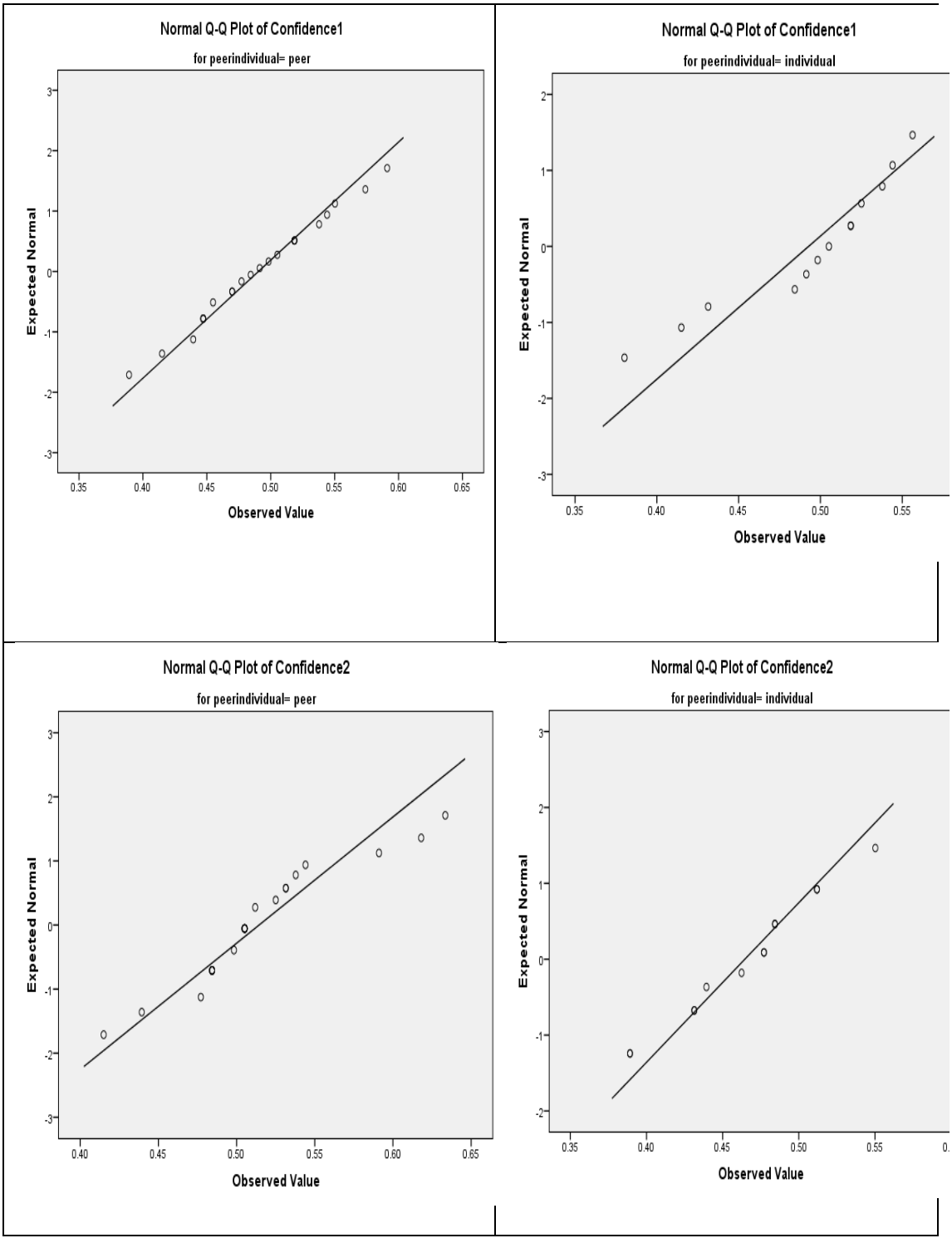
Zarb, M., Hughes, J., Richards, J., Wray, S., Williams, L. L. A., McCrickard, D. S., … Nagappan, N. (2010). How pair programming really works. *Communications of the ACM*, *9*(3), 1–11. DOI: 10.1109/Agile.2008.12

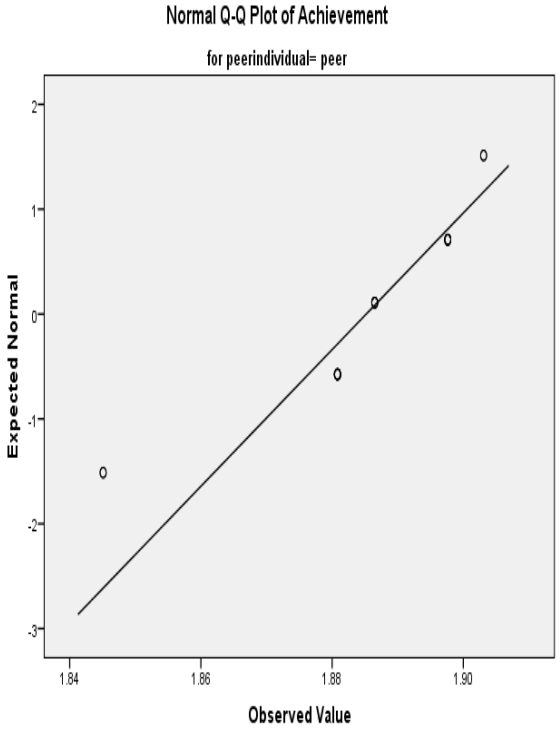
Zhong, B., Wang, Q., Chen, J., & Li, Y. (2017). Investigating the period of switching roles in pair programming in a primary school. *Journal of Educational Technology & Society, 2*0(3), 220-233. Retrieved from http://www.jstor.org/stable/26196132

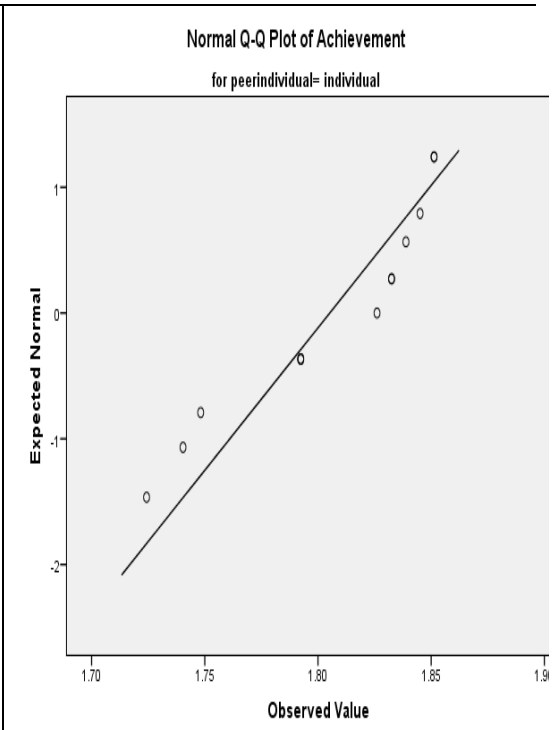
Zieris, F. (2015). Qualitative analysis of knowledge transfer in pair programming. *37th IEEE International Conference* on *Software Engineering*, *2*, 855-858. DOI: 10.1109/icse.2015.277

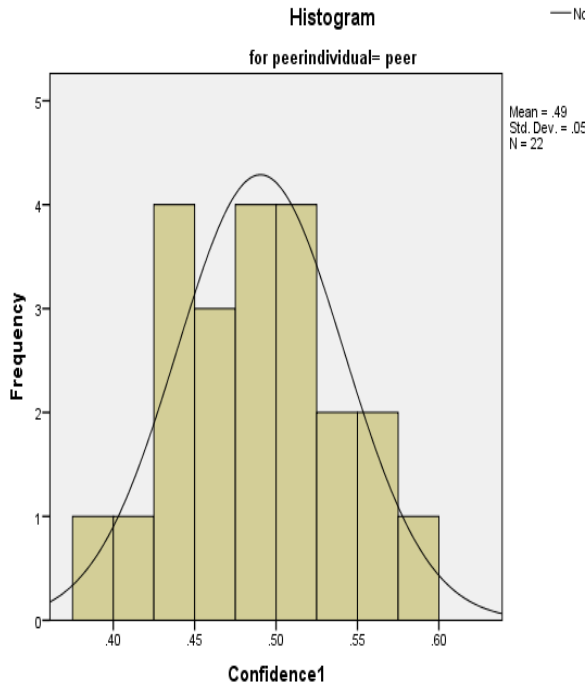
APPENDIX A

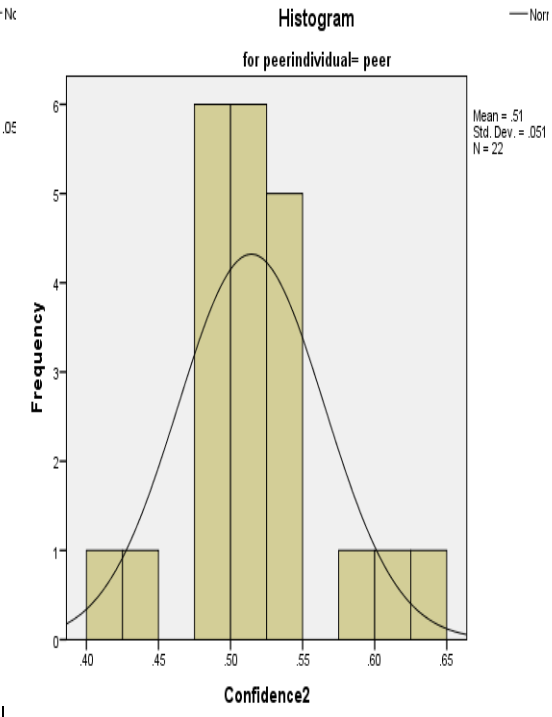
Q-Q PLOTS AND HISTOGRAMS FOR NORMALITY

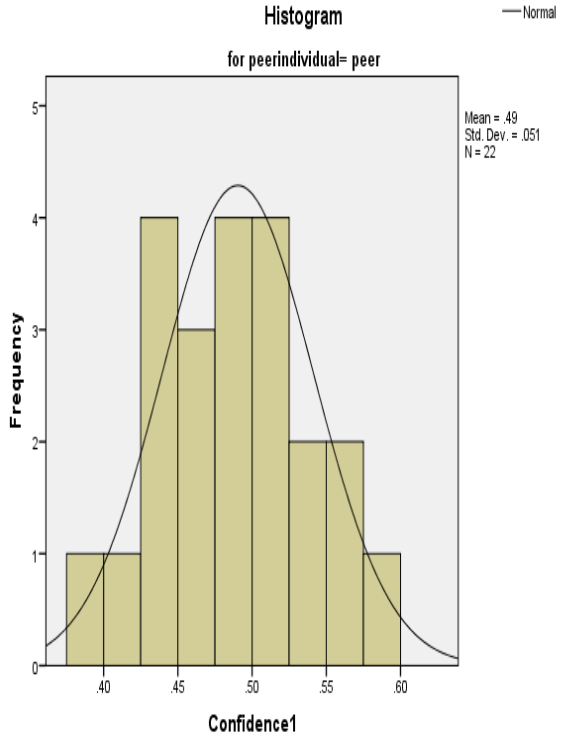

Normal Q-Q Plot of Confidence1
for peerindividual= peer
Expected Normal
Observed Value
Normal Q-Q Plot of Confidence1
for peerindividual= individual
Expected Normal
Observed Value
Normal Q-Q Plot of Confidence2
for peerindividual= peer
Expected Normal
Observed Value
Normal Q-Q Plot of Confidence2
for peerindividual= individual
Expected Normal
Observed Value

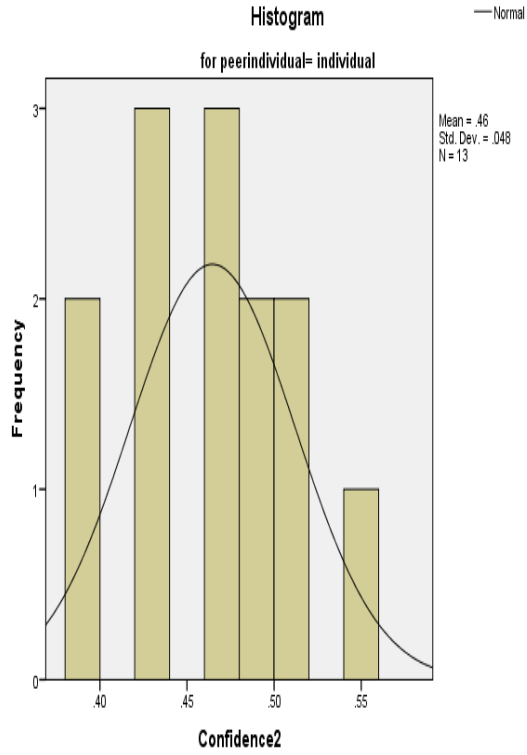

Normal Q-Q Plot of Achievement
for peerindividual= peer
Expected Normal
Observed Value
1.84
1.86
1.88
1.90

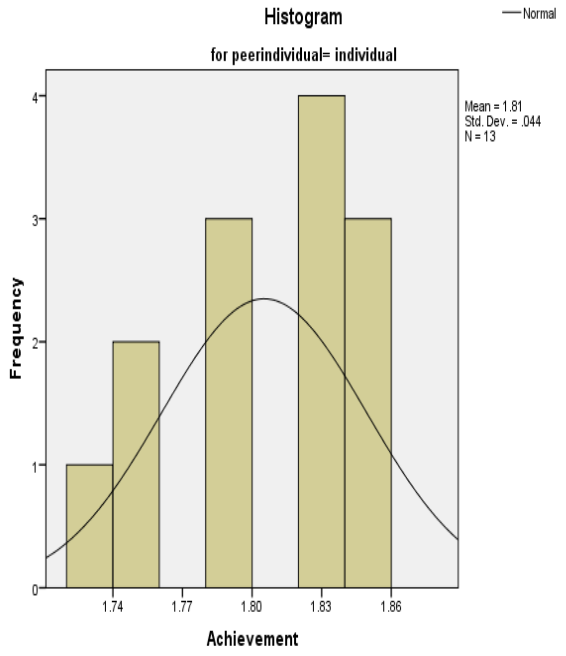

Normal Q-Q Plot of Achievement
for peerindividual= individual
Expected Normal
Observed Value
1.70
1.75
1.80
1.85

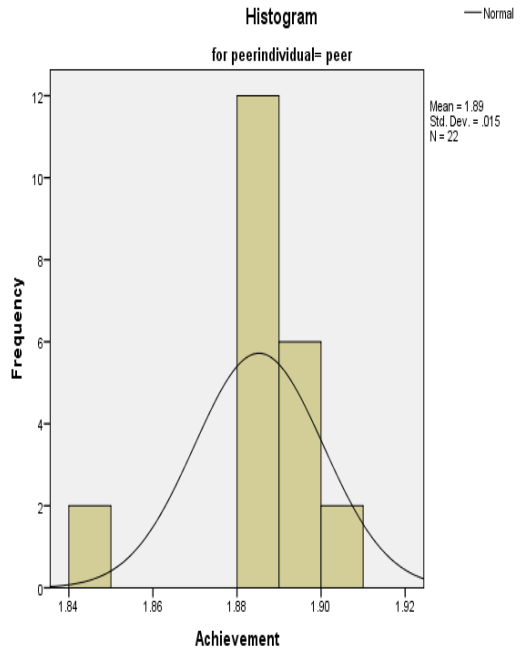

Histogram
for peerindividual= peer
Mean = .49
Std. Dev. = .05
N = 22
Frequency
Confidence1
.40
.45
.50
.55
.60


Histogram
for peerindividual= peer
Mean = .51
Std. Dev. = .051
N = 22
Frequency
Confidence2
.40
.45
.50
.55
.60
.65


Histogram
Normal
for peerindividual= peer
Mean = .49
Std. Dev. = .051
N = 22
Frequency
Confidence1


Histogram
Normal
for peerindividual= individual
Mean = .46
Std. Dev. = .048
N = 13
Frequency
Confidence2


Histogram
Normal
for peerindividual= individual
Mean = 1.81
Std. Dev. = .044
N = 13
Frequency
Achievement


Histogram
Normal
for peerindividual= peer
Mean = 1.89
Std. Dev. = .015
N = 22
Frequency
Achievement

APPENDIX B

INTERVIEW PROTOCOL FOR STUDENTS (TURKISH)

GÖRÜŞME FORMU

ARAŞTIRMA SORULARI:

1. How the application of pair-programming technique influences the confidence level of secondary school students during the programming process?

2. How the application of pair programming technique influences the achievement level of secondary school students during the programming process?

Giriş

Merhaba, adım Habibe KARAOĞLU. Orta Doğu Teknik Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı'nda hem yüksek lisans öğrencisi hem de Milli Eğitim Bakanlığında Bilişim Teknolojileri ve Yazılım öğretmeniyim. Ortaokul öğrencilerinin grup programlama tekniği ile kodlama öğrenmelerinin başarıya ve özgüvene etkisi üzerinde araştırma yapmaktayım. Görüşmemize geçmeden önce, görüşmemizin gizli olduğunu ve görüşmede konuşulanların yalnızca benim ve bazı araştırmacıların bileceğini belirtmek isterim. Uygulamadaki paylaşımlarınız başka kimseyle paylaşılmayacaktır. Bunun yanında araştırma raporunda isimleriniz kesinlikle yer almayacak, bunun yerine takma isimler kullanılacaktır.

Görüşmemize başlamadan önce sormak istediğin soru ya da belirtmek istediğin herhangi bir düşüncen var mı?

Görüşmelerin kaydedilmesi konusunda ne düşünüyorsun? Görüşme sonunda istemediğiniz bazı bilgileri silebilirsin.

Görüşmeye devam etmek istiyor musun?

Görüşmemizin yaklaşık yarım saat süreceğini tahmin ediyorum. İzin verirsen sorulara başlamak istiyorum.

GİRİŞ/ISINDIRMA SORULARI

1.1. Bilişim Teknolojileri dersine karşı düşüncelerin nasıl? Derste en çok ne yapmaktan zevk alıyorsun?

1.2. Bilişim dersinden önce animasyon ve kodlama ile ilgili herhangi bir deneyimin var mıydı? Varsa hangi kodlama programını kullandın tecrübelerinden bahseder misin?

1.3. Bilişim Teknolojileri dersi dışında kodlama bilgini geliştirmek için çalışıyor musun? (internetten araştırma yapmak, kursa gitmek gibi.)

İÇERİKLE İLGİLİ SORULAR

BİREYSEL PROGRAMLAMA SORULARI:

1. Programlama aktivitesini tamamlarken tek başına çalışmak sana ne hissettirdi? Zorluk yaşadın mı? Nasıl çözdün?
2. Ders içi programlama aktivitesinde karşılaştığın sorunlara karşı tavrın nasıl olur? Çalışmanı nasıl sürdürürsün? Pes eder misin?
3. Sence derste öğrendiğin bilgiler aktiviteyi tamamlamak içi yeterli miydi? Programlama aktivitesini tamamlarken daha fazla bilgi, kaynağa ihtiyaç duydun mu?
4. Programlama aktivitesinin her adımını başarıyla tamamlayacağına inandın mı? Zorlandığın kısımlarda neler düşündün?
5. Diğer derste yeni aktiviteyi tamamlarken grup halinde mi yoksa bireysel mi çalışmak istersin? Neden?

GRUP PROGRAMLAMA SORULARI:

1. Programlama yaparken arkadaşınla çalışmak nasıl bir durum? Sana neler hissettiriyor?
2. Ders içi programlama aktivitesinde karşılaştığın sorunlara karşı tavrın nasıl olur? Çalışmanı nasıl sürdürürsün? Pes eder misin?
3. Derste öğrendiğiniz bilgileri kullanırken grup arkadaşıyla çalışmak aktiviteyi tamamlamayı nasıl etkiledi.
   Katkısı var mı?
   Zararı var mı?
4. Programlama etkinliğini tamamlarken grup arkadaşınızla programlama yapabileceğinize dair özgüveniniz değişiyor mu? Ne düşünüyorsunuz?
5. Diğer derste yeni aktiviteyi tamamlarken grup halinde mi yoksa bireysel mi çalışmak istersin? Neden?

APPENDIX C

CONFIDENCE SCALE (TURKISH)

ORTAOKUL ÖĞRENCİLERİ İÇİN PROGRAMLAMA EĞİTİMİNDE ÖZGÜVEN

Bu anket programlama eğitiminde öğrencilerin özgüveninin değerlendirilmesi amacıyla hazırlanmıştır. Anketi doldurmak için:

- Her maddeyi dikkatlice oku ve uygun olan bölümü işaretle.
- Sorulara ait tek bir doğru ya da yanlış cevap yok bu nedenle gerçek düşüncelerini yansıt.
- Lütfen bütün soruları cevapla.

		Kesinlikle katılıyorum	Katılıyorum	Kararsızım	Katılmıyorum	Kesinlikle katılmıyorum
1	Bilgisayar alanında üst düzey çalışmalar yapabileceğime eminim.					
2	Programlamayı öğrenebileceğime kesinlikle eminim.					
3	Daha zor programlama problemlerinin üstesinden gelebileceğimi düşünüyorum.					
4	Programlama dersinden yüksek not alabilirim.					
5	Programlama konusunda kendime güvenim oldukça yüksektir.					
6	Programlamada başarılı değilim.					
7	Üst düzey programlama yapabileceğimi düşünmüyorum.					
8	Programlamayı iyi yapabilecek bir değilim.					
9	Nedense, çok çalışmama rağmen programlama bana çok zor geliyor.					
10	Birçok dersin üstesinden gelebilsem de programlama problemlerinde mutlaka hata yapıyorum.					
11	Programlama dersi en kötü olduğum derstir.					
12	Programlamada genellikle iyiyimdir.					
13	Benim için programlama çoğu sınıf arkadaşıma göre daha zor.					
14	Programlama güçlü yanlarımdan biri değil.					
15	Programlamadaki konuları çabukça öğrenirim.					
16	Programlama kafamı karıştırır ve beni gerer.					
17	Zor olan programlama problemlerini çözmekte başarılıyım.					
18	Öğretmenim zor da olsa programlamayı iyi bir şekilde yapabileceğimi düşünür.					
19	Öğretmenim bana programlamada iyi olduğumu söyler.					
20	Programlama dersi benim için diğer derslerden daha zordur.					

APPENDIX D

CONFIDENCE SCALE (ENGLISH)

PROGRAMMING CONFIDENCE SCALE FOR SECONDARY SCHOOL STUDENTS IN PROGRAMMING EDUCATION

This scale was prepared to evaluate the confidence of secondary school students in computer programming. To fill the survey:

- Read each statement carefully and mark your response.
- There are no rights or wrong answers. Do not be afraid to put down what you really think.
- Please complete all of the items.

		Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
1	I am sure that I could do advanced work in computer science.					
2	I am sure that I can learn programming.					
3	I think I could handle more difficult programming problems.					
4	I can good grades in programming course.					
5	I have a lot of self-confidence when it comes to programming.					
6	I am no good at programming.					
7	I do not think I could do advanced in programming.					
8	I am not the type to do well in computer programming.					
9	For some reason even though I work hard at it, programming seems unusually hard for me.					
10	Most subjects I can handle, but I have a knack flubbing up programming problems.					
11	Programming has been my worst subject.					
12	I usually do well in computer programming.					
13	Programming is more difficult for me than for many of my classmates.					
14	Programming is not one of my strengths.					
15	I learn things quickly in programming.					
16	Programming makes me confused and nervous.					
17	I am good at working out difficult programming problems.					
18	My teacher thinks I can do well in programming with difficulties.					
19	My teacher tells me I am good at programming.					
20	Programming is harder for me than any other subjects.					

APPENDIX E

EXAMPLE SCRATCH LESSON PLAN AS PROVIDED IN

HTTP://SCRATCH.IE/PRIMARY/LESSONPLANS/LESSON1

Lesson Plan 1
Under the Sea

Mathematical Skills / Concepts
Scratch Features
Curricular Links
Problem Solving, XY Coordinates
Moving Left and Right, Up and Down
Mathematics, Art, Science

MMXII:VI:VIII Version 1.01

Learning Objectives

The child will:

- Use problem solving and mathematical skills in an authentic setting.
- Create an animation incorporating movement and images.
- Create an animation of a natural habitat.

Teacher Tip

Encourage children to do as much independent experimentation as possible. They should be free to choose their own sprites, alter distances etc. This experimentation will increase children's understanding of Scratch.

Introduction

To support the children the teacher is encouraged to follow these steps using the interactive whiteboard.

- Ask the children to choose a sprite* from the file.

- Ask them to open the blue motion block and experiment for a few minutes.

Left and Right

- To move right add the block "move 10 steps" to the scripts area. Double click on this block to see it work.

- To move left use the same block. Highlight "10" steps by clicking on it. Change this to "-10". Double click on this block to see it work.

- Add a control to your script. Click on the control block and drag out "When space key pressed". Snap this onto your movement blocks.

- Add a repeat block to make a continuous movement. Insert the block "if on edge bounce" to stop your sprite disappearing from the screen.

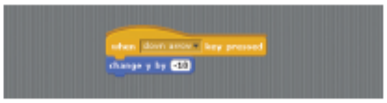
Challenge Time 1!

1. Select 2 fish / underwater sprites from file. Be as creative as you can.
2. Use the move blocks to make them move left and right across the screen.
3. Add the "wait 1 sec" block to one of your sprites. What happens?
4. Change the wait time and see what happens.

Up and Down

- To move up and down children will need to experiment with the y axis. Ask children to move the sprite on the stage. Observe the xy coordinates changing on the bottom righ of the stage. Try to move the sprite to (0,0). Show the children how the x coordinates change as the sprite moves left to right, and how the y coordinates change as the sprit moves up and down.
- Use the "when space key pressed" block. Using the drop down arrow alter it to "when up arrow key pressed". Add the movement block "change y by 10".

- Ask the children to demonstrate how to move their sprite downwards.



Challenge Time 2!

1. Open a new Scratch project.
2. Select the trampoline sprite from the sprite folder.
3. Select any other sprite.
4. Create a script which allows the sprite to bounce up and down on the trampoline.
5. Add another sprite which jumps more slowly.

*Sprites are the objects that perform actions in a Scratch project. While the Stage can also be programmed in a project, most projects have at least one sprite as well because only sprites can move.

APPENDIX F

EXAMPLE ACHIEVEMENT RUBRIC (TURKISH)

Ad-Soyad _____ Scratch Proje Adı _____

Deniz Altı
SCRATCH BAŞARI RUBRİK

Kategori	0	1	2
Proje Tasarımı **Yönlendirme**	Projeyi kullanacak diğer kişiler için yönlendirme veya açıklama yapılmamış.	Projeyi kullanacak diğer kişiler için yönlendirme yapılmış ancak net değil.	Projeyi kullanacak kişiler için gerekli yönlendirme ve açıklamalar yapılmış.
Proje Tasarımı Hareket	Herhangi bir hareket yok.	Karakterlere hareket kodu eklenmiş ancak karakterler beklenen şekilde hareket etmiyor veya sahneden kayboluyor.	Karakterlere gerekli kodlar yazılmış ve karakterler sahnede düzgün şekilde hareket ediyor. Aynı anda çalışıp aynı anda duruyor.
Proje Tasarımı **Arkaplan**	Arkaplan yok.	Arka plan eklenmiş fakat karakter bölümüne eklenmiş düzgün çalışmıyor. Ve uygun bir arka plan değil.	Uygun bir arkaplan eklenmiş.
Proje Tasarımı **Kod Blokları**	En az bir kod grubu kullanılmamış ve gerekli olan kod gruplarından görünüm, hareket ve kontrol gibi herhangi bir kod eklenmemiş.	Kod blokları kullanılmış ama görünüm, kontrol, hareket kod gruplarından kullanılması gereken bütün bloklar kullanılmamış.	Bütün kod blokları amacına uygun ve düzgün bir şekilde kullanılmış.
Proje Tasarımı **Kostümler/ Karakterler**	Eklenmesi gereken 2 karakter eklenmemiş ve bu karakterlere ait kostümler kullanılmamış.	2 karakter eklenmiş fakat bu karakterlere ait kostümler eklenmemiş.	Karakterler ve bu karakterlere ait kostümler eklenmiş.
TOPLAM			

APPENDIX G

EXAMPLE ACHIEVEMENT RUBRIC (ENGLISH) ADAPTED FROM WWW.HTTP://SCRATCED.GSE.HARVARD.EDU/RESOURCES/CREATING-SCRATCH-RUBRICS

Name _____Project name _____

Under The Sea
SCRATCH RUBRIC

Category	0	1	2
Project Design **Directions**	Does not provide explanation or a way for other people to interact with program.	Provide explanation or a way for other people to interact with program but not clear.	Provide explanation or a way for other people to interact with program.
Project Design **Music and Movement**	No movement	Appropriate motion blocks are included, and/or sprites are not moving and/or sprites get lost from the stage.	Appropriate motions are included, sprites are moving in stage and stop at the same time.
Project Design **Background**	No background.	Background or backgrounds are included but are not appropriate place in the program.	An appropriate background or backgrounds are included.
Project Design **Blocks**	Not used at least one code groups and and/or did not use at least 1 from each of the following: Control, looks, motion.	Used code groups but are not used at least 1 from each of the following: Control, Look, and motion.	All of the code groups and code blocks are used appropriately.
Project Design **Sprites**	Not created and use 2 sprites and/or did not use all costumes of the sprites.	Created and used 2 sprites but are not used the costumes for each sprite.	Created and used 2 sprites each with costumes.
TOTAL			

APPENDIX H

PARENT'S CONSENT FORM (TURKISH)

Veli Onay Mektubu

Sayın Veli,

Orta Doğu Teknik Üniversitesi, Bilgisayar ve Öğretim Teknolojileri eğitimi bölümünde yüksek lisans öğrencisiyim. Aynı zamanda çocuğunuzun bilişim teknolojileri ve yazılım dersi öğretmeniyim. Çocukların programlama eğitimi sırasında kullanacakları grup programlama tekniği hakkındaki görüşleri ve deneyimleri hakkında yüksek lisans çalışmamı yürütmekteyim. Bu mektup size, eğer uygun görürseniz çocuğunuzun bu çalışma kapsamında verilen anketi doldurması ve röportaja katılmasına onay vermeniz için göndermekteyim.

Bu çalışmanın amacı Bilişim Teknolojileri ve Yazılım dersi kapsamınsa programlama eğitimi alan 5. sınıf öğrencilerinin grup programlama yöntemini kullanarak programlama başarılarını ve programlamaya karşı olan özgüvenlerini araştırmaktır. Bu çalışma neticesinde grup programlama tekniğinin öğrencilerin programlama başarısı ve özgüvenine olan etkisi yorumlanacak ve çocukların programlama eğitimi konusunda yapılacak çalışmalara ve güncellemelere dayanak olacaktır.

Çalışmayı gerçekleştirebilmek için çocuklarınızın bazı anketleri doldurmalarına ve röportajlara katılmalarına ihtiyaç duymaktayım. Bu çalışmaya katılmaya izin verdiğiniz takdirde çocuğunuz gerekli anket ve röportajı ders saatinde gerçekleştirecektir. Sizin onayınızdan sonra çalışmaya başlamadan önce mutlaka çocuğunuzdan da sözlü onayı alınacaktır. Size gönderilen bu izin belgelerinin tarafınızca doldurulup öğrenci aracılığıyla bana ulaştırılması rica olunur.

Çocuğunuzdan alınacak olan bilgiler sadece bilimsel araştırma amacıyla kullanılacaktır. Çocuğunuz vereceği cevaplar kesinlikle gizli tutulacak ve üçüncü şahıslarla paylaşılmayacaktır. Veli olarak sizin ve çocuğunuzun kimlik bilgileri kesinlikle gizli tutulacaktır, kimseyle paylaşılmayacaktır**.**

Çocuğunuzun dolduracağı bilgiler sadece bilgisayar programlama deneyimleri ve görüşleriyle ilgili olup çocuğunuzu olumsuz etkileyecek bir durum kesinlikle söz konusu değildir. Siz veya çocuğunuz eğer bu çalışamaya devam etmekten vazgeçerseniz hiçbir olumsuzlukla karşılaşmayacaksınız. Çalışmadan

çocuğunuz rahatsızlık duyarsa ve devam etmek istemezse durumu şahsıma bildirmeniz yeterli olacaktır.

Çalışmayla ilgili daha fazla bilgi almak isterseniz aşağıdaki iletişim bilgilerini kullanarak bana sorularınızı iletebilirsiniz.

Teşekkür ederim.

Habibe KARAOĞLU
Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü
Orta Doğu Teknik Üniversitesi, Ankara
Tel: 05342572789
e-posta: habibe.krgll@gmail.com

Yukarıdaki açıklamaları okudum ve çocuğum ----------' nin bu çalışmada yer almasına izin veriyorum. Velinin:

Adı-soyadı: _____ İmzası: _____

Tarih:_____

(Lütfen bu formu doldurup imzaladıktan sonra çocuğunuz aracılığıyla bana ulaştırınız.)

APPENDIX I

VOLUNTARY PARTICIPATION APPROVAL FORM (TURKISH)

ARAŞTIRMAYA GÖNÜLLÜ KATILIM FORMU

Sevgili Katılımcı,

Bu araştırmada, 5. Sınıf öğrencilerine yönelik uygulanan programlama eğitimi sırasında kullanılacak olan grup programlama tekniği hakkındaki görüşleri ve deneyimleri incelemek amaçlanmaktadır.

Araştırma bilişim teknolojileri sınıfında, bilişim dersi esnasında yürütülecektir. Öğrencilere verilecek olan anket ve diğer formlar programlama başarısı ve programlama özgüvenine yönelik sorular içermektedir.

Çalışmaya katılmak tamamen gönüllülük esasına dayalı olup katılımcı herhangi bir olumsuz durum ile karşılaştığında hiçbir ceza ve yaptırıma maruz kalmadan çalışmayı terkedebilir. Araştırma esnasında cevaplamak istemediğiniz soruları bos bırakabilirsiniz. Sizi rahatsız eden bir soru olursa cevap vermeme hakkınız vardır.

Araştırma için paylaşacağınız görüşler ve bilgiler tamamen gizli tutulacaktır. Bu bilgiler sadece yüksek lisans tezi için kullanılacak olup sonuçları bilimsel ve eğitim amaçlı yayınlarda kullanılabilir. Katılımcıların kişisel bilgileri üçüncü kişilerle asla paylaşılmayacaktır ve gizli tutulacaktır.

Çalışmayla ilgili daha fazla bilgi almak isterseniz aşağıdaki iletişim bilgilerini kullanarak bana sorularınızı iletebilirsiniz.

Habibe KARAOĞLU
Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü
Orta Doğu Teknik Üniversitesi, Ankara
e-posta: habibe.krgll@gmail.com

Yukarıdaki bilgileri okudum ve bu çalışmaya tamamen gönüllü olarak katılıyorum.

(Formu doldurup imzaladıktan sonra uygulayıcıya geri veriniz).

Ad Soyad Tarih İmza

---/----/-----

APPENDIX J

ETHICS COMMITTEE APPLIED ETHICS APPROVAL FORM OF MIDDLE EAST TECHNICAL UNIVERSITY RESEARCH CENTER (TURKISH)

UYGULAMALI ETİK ARAŞTIRMA MERKEZİ
APPLIED ETHICS RESEARCH CENTER

ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

DUMLUPINAR BULVARI 06800
ÇANKAYA ANKARA/TURKEY
T: +90 312 210 22 91
F: +90 312 210 79 59
ueam@metu.edu.tr
www.ueam.metu.edu.tr

Sayı: 28620816 / 31

02 OCAK 2017

Konu: Değerlendirme Sonucu

Gönderen: ODTÜ İnsan Araştırmaları Etik Kurulu (İAEK)

İlgi: İnsan Araştırmaları Etik Kurulu Başvurusu

Sayın Yrd. Doç. Dr. Gülfidan CAN;

Danışmanlığını yaptığınız yüksek lisans öğrencisi Habibe KARAOĞLU'nun ***"Scratch Programlama Ortamında Ortaokul Öğrencilerinin Grup Programlama Tekniğini Kullanarak Programlama Başarısı ve Özgüveninin Geliştirilmesi"*** başlıklı araştırması İnsan Araştırmaları Etik Kurulu tarafından uygun görülerek gerekli onay **2016-EGT-158** protokol numarası ile **28.11.2016 - 31.12.2017** tarihleri arasında geçerli olmak üzere verilmiştir.

Bilgilerinize saygılarımla sunarım.

Prof. Dr. Canan SÜMER
İnsan Araştırmaları Etik Kurulu Başkanı

Prof. Dr. Mehmet UTKU
İAEK Üyesi

Prof. Dr. Ayhan SOL
İAEK Üyesi

Prof. Dr. Ayhan Gürbüz DEMİR (4.)
İAEK Üyesi

Doç. Dr. Yaşar KONDAKÇI
İAEK Üyesi

Yrd. Doç. Dr. Pınar KAYGAN
İAEK Üyesi

Yrd. Doç. Dr. Emre SELÇUK
İAEK Üyesi

APPENDIX K

APPROVAL FORM OF MINISTERY OF EDUCATION (MEB)

T.C.
ANKARA VALİLİĞİ
Milli Eğitim Müdürlüğü

Sayı : 14588481-605.99-E.6826495 12.05.2017
Konu : Araştırma İzni

ORTA DOĞU TEKNİK ÜNİVERSİTESİNE
(Öğrenci İşleri Daire Başkanlığı)

İlgi: a) MEB Yenilik ve Eğitim Teknolojileri Genel Müdürlüğünün 2012/13 nolu Genelgesi.
b) 27/04/2017 tarihli ve 2169 sayılı yazınız.

Fen Bilimleri Enstitüsü Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü Yüksek Lisans öğrencisi Habibe KARAOĞLU'nun **"Scratch Programlama Ortamında Ortaokul Öğrencilerinin Grup Programlama Tekniğini Kullanarak Programlama Başarısı ve Özgüveninin Geliştirilmesi"** konulu tez kapsamında uygulama talebi Müdürlüğümüzce uygun görülmüş ve uygulamanın yapılacağı İlçe Milli Eğitim Müdürlüğüne bilgi verilmiştir.

Görüşme formunun (7 sayfa) araştırmacı tarafından uygulama yapılacak sayıda çoğaltılması ve çalışmanın bitiminde bir örneğinin (cd ortamında) Müdürlüğümüz Strateji Geliştirme (1) Şubesine gönderilmesini rica ederim.

Vefa BARDAKCI
Vali a.
Milli Eğitim Müdürü

Güvenli Elektronik İmzalı
Aslı İle Aynıdır.
12 Mayıs 2017

Konya yolu Başkent Öğretmen Evi arkası Beşevler ANKARA
e-posta: isiatistik06@meb.gov.tr

Ayrıntılı bilgi için
Tel: (0 312) 221 02 17/135-134

Bu evrak güvenli elektronik imza ile imzalanmıştır. http://evraksorgu.meb.gov.tr adresinden 4a43-478c-32c8-b095-60a3 kodu ile teyit edilebilir.