REPUBLIC OF TURKEY

YILDIZ TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

IMPROVING MIDDLE SCHOOL STUDENTS' COMPUTATIONAL THINKING SKILLS AND ATTITUDES TOWARDS COMPUTER SCIENCE THROUGH STEM-BASED ROBOTICS EDUCATION

Hatice VARLIK

MASTER OF SCIENCE THESIS

Department of Mathematics and Science Education

Science Education Program

Advisor

Prof. Dr. Mustafa Sami TOPÇU

February, 2021

REPUBLIC OF TURKEY

YILDIZ TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

IMPROVING MIDDLE SCHOOL STUDENTS' COMPUTATIONAL THINKING SKILLS AND ATTITUDES TOWARDS COMPUTER SCIENCE THROUGH STEM-BASED ROBOTICS EDUCATION

A thesis submitted by Hatice VARLIK in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE is approved by the committee on 05.02.2021 in Department of Mathematics and Science Education, Science Education Program.

Prof. Dr. Mustafa Sami TOPÇU

Yıldız Technical University

Advisor

Approved by the Examining Committee

Prof. Dr. Mustafa Sami TOPÇU, Advisor

Yıldız Technical University _____

Prof. Dr. Hasan ÜNAL, Member

Yıldız Technical University _____

Prof. Dr. İbrahim ERDOĞAN, Member

Muş Alparslan University _____

I hereby declare that I have obtained the required legal permissions during data collection and exploitation procedures, that I have made the in-text citations and cited the references properly, that I haven't falsified and/or fabricated research data and results of the study and that I have abided by the principles of the scientific research and ethics during my Thesis Study under the title of Improving Middle School Students' Computational Thinking Skills and Attitudes towards Computer Science through STEM-Based Robotics Education supervised by my supervisor, Prof. Dr. Mustafa Sami TOPÇU. In the case of a discovery of false statement, I am to acknowledge any legal consequence.

Hatice VARLIK

Signature

Dedicated to my family

ACKNOWLEDGEMENTS

First of all, I would like to thank you my family for their consistent support. They prayed me continously and showed me patience at times during the whole thesis process, especially my little sister Neslihan. I really appreciate your support.

I would like to thank you my dear friends Kader KOÇ and Gülsüm BAYER. Whenever I feel stuck, they gave me valuable advices and helped me solve problems. Besides, I would like to thank you my classmate and dear friend İlayda KILIÇ for giving me moral support and helping me every time I need assistance.

Finally, I would like to thank my thesis advisor Prof. Dr. Mustafa Sami TOPÇU for his great support. His contributions to my thesis was so important and valuable.

Hatice VARLIK

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

CS	Computer Science
CTS	Computational Thinking Skills
ICT	Information and Communication Technologies
ISTE	International Society for Technology in Education
NGSS	Next Generation Science Standards
NRC	National Research Council
STEM	Science, Technology, Engineering, Mathematics

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Improving Middle School Students' Computational Thinking Skills and Attitudes towards Computer Science through STEM-Based Robotics Education

Hatice VARLIK

Department of Mathematics and Science Education

Master of Science Thesis

Advisor: Prof. Dr. Mustafa Sami TOPÇU

In the future, computer technologies are expected to become an integral part of different professions and working fields. Therefore, it is very important for the students living in today to become competent in using computer technologies and solving computational problems. As a result, schools across different countries started to include computer science lessons in their educational programs to teach students how to use computer technologies efficiently and to make them gain computational thinking skills. However, students may develop a negative attitude towards computer science if computer science lessons are delivered without establishing a context related to real life. Consequently, students may not gain computational thinking skills. To overcome this attitude problem, some research suggests that integrating STEM approach to computer science lessons can help educators to establish a meaningful context. On the other hand, previous studies argue that educational robotics is an effective tool to integrate STEM into computer science lessons for the purpose of enhancing students' computational thinking skills. However, there is not enough research in the literature that investigates this issue. Therefore, the purpose of this study is to develop a STEM-based robotics unit and to

investigate its effects on students' attitude towards computer science and computational thinking skills. The unit consisted of 6 lessons each of which includes a STEM-related real-life problem. In each lesson, students are expected to solve the problem that belongs to that lesson by preparing models with the robotics set. At the beginning and end of the study, students were given computational thinking levels scale and attitude towards computer programming scale. The analysis of the pre-test and post-test scores showed that STEM-based robotics education resulted in a statistically significant difference in students' computational thinking skills and their attitudes towards computer science. In this way, this study aims to contribute to the literature by providing a basis for further research that will propose other ways of integrating STEM approach to computer science.

Keywords: STEM, computational thinking, robotics, programming, computer science

ÖZET

STEM Tabanlı Robotik Eğitimi ile Öğrencilerin Bilgi İşlemsel Düşünme Becerilerinin ve Bilgisayar Bilimlerine Karşı Tutumlarının Geliştirilmesi

Hatice VARLIK

Matematik ve Fen Bilimleri Eğitimi Programı

Yüksek Lisans Tezi

Danışman: Prof. Dr. Mustafa Sami TOPÇU

Günümüz öğrencileri, bilgi iletişim araçlarının çeşitli iş dallarının ve çalışma disiplinlerinin ayrılmaz bir parçası olduğu bir geleceğe doğru büyümektedirler. Bilgi işlemsel problemleri çözebilmeleri ve bilgi işleme araçlarını kullanmak için gerekli becerileri kazanmaları oldukça önem kazanmıştır. Bu doğrultuda öğrencilerin bilgi iletişim araçlarını etkili bir şekilde kullanabilmeleri ve bilgi işlemsel düşünme becerilerini kazanmaları amacıyla okullarda bilgisayar bilimleri dersi verilmeye başlanmıştır. Ancak gerçek yaşamdan kopuk bir bağlamda verilen bilgisayar bilimleri dersleri öğrencilerin bu disipline karşı olumsuz bir tutum geliştirmesine sebep olabilmekte ve onların bilgi işlemsel düşünme becerilerinin gelişimine yeterince katkı sağlamayabilmektedir. Yapılan araştırmalar göstermiştir ki STEM ile ilişkili bir şekilde işlenen bilgisayar bilimleri dersleri öğrencilerin hem bu disipline karşı olan tutumlarını iyileştirmekte hem de onların bilgi işlemsel düşünme becerilerini kazanmalarına katkı sağlamaktadır. Öte yandan, STEM yaklaşımını bilgisayar bilimleri dersine entegre etmenin en iyi yolu olarak eğitsel robotik

setlerinin kullanımı olduğu öne sürülmüştür. Bu sebeple 6 haftalık bir STEM tabanlı robotik ünitesi hazırlanmış ve robotik dersinde uygulanmıştır. Ünitedeki her bir aktivitede STEM ile ilişkili bir gerçek yaşam problemi bulunmakta ve öğrencilerin bu problemi robotikten faydalanarak çözmeleri istenmiştir. Ünitenin başında ve sonunda bilgi işlemsel düşünme ölçeği ve bilgisayar bilimlerine karşı tutum ölçeği uygulamıştır. Öntest ve sontest sonuçları analiz edildiğinde öğrencilerin bilgi işlemsel düşünme becerileri ve bilgisayar bilimlerine karşı tutumları arasında anlamlı bir pozitif fark olduğu tespit edilmiştir. Bu araştırma, STEM yaklaşımını bilgisayar bilimleri dersine entegre etmenin yollarını önerecek çalışmalara bir temel teşkil etmesi yönüyle literatüre önemli bir katkı sunmayı hedeflemiştir.

Anahtar Kelimeler: STEM, bilgi işlemsel düşünme, robotik, bilgisayar bilimleri, programlama

1
INTRODUCTION

1.1 Literature Review

Computational thinking is generally described as developing solutions to problems that can be carried out by information processing agents (Cuny, Snyder & Wing, 2010; Wing, 2011). These agents include humans, computers or both. The essential idea of computational thinking includes breaking down problems into smaller ones (decomposition), writing a sequence of steps to solve each problem (algorithms), focusing on important information (abstraction), deciding how the solution applies to similar problems (pattern recognition) and having computers do repetitive tasks (automation) (CSTA, 2011; Yadav, Hong & Stephenson, 2016). In addition, computational thinking consists of critical thinking, algorithmic thinking, creativity, collaborative learning and use of digital tools to solve problems (Yünkül, Durak & Çankaya, 2017). As a result, computational thinking is considered as 21st century skills (Mohaghegh & McCauley, 2016). Students should develop computational thinking skills since the future they have been growing up to is heavily shaped by computer technologies. Wing (2006) also claims that computational problem-solving methods would become basis for all disciplines as a result of the advancements in computing. Therefore, every child should learn to think computationally along with learning howto read, write and make calculations (International Society for Technology in Education (ISTE), 2015; Wing, 2006).

In the recent years, different ways of incorporating computational thinking into K-12 classrooms were put into practice. Most of them mainly focused on computer science curriculum (Yadav, Hong & Stephenson, 2016). However, there are some criticisms about the way computer science curriculum is applied. Goode, Estrelle and Margolis (2005) found that computer science courses in copy-paste format solely focus on basic programming skills and prevent students from truly understanding the language and how it can be used to solve a problem. As a result,

students miss the connection between computer science and the issues related to everyday life; hence, perceive computer science-related works as less creative and unappealing. In addition, there are some studies claiming that negative attitude towards computer programming can be an obstacle to learning process (Yiğit, 2016). Results of a study conducted by Başer (2013a) demonstrate that students' achievement in programming is influenced by their attitudes towards the subject. As a result, negative attitude towards computer science lessons might have a negative effect on the development of computational thinking skills. Therefore, it is very important to design computer science lessons related to real-life problems that will engage students and help them develop a positive attitude towards the lesson.

To overcome the engagement and motivation problem, Burbaitė, Drąsutė and Štuikys (2018) claims that computer science education must be enriched with new learning tools, methods and content. To that end, they introduce what they called "STEM-driven computer science education" as a context related to real life issues to support students' computational thinking skills (Burbaitė, Drąsutė & Štuikys, 2018). On the other hand, the tools used in computer science education highly matters for student motivation and learning (Burbaitė, Drąsutė & Štuikys, 2018; Cooper & Cunningham, 2010). Research suggest that educational robotics makes learning activities more engaging and motivating for students (Burbaitė, Drąsutė & Štuikys, 2018; Cooper & Cunningham, 2010; Eguchi & Uribe, 2017; McKay, Lowes, Tirthali & Camins, 2015). Besides, it is an effective tool to integrate STEM and computational thinking (Burbaitė, Drąsutė & Štuikys, 2018; Eguchi & Uribe, 2017; Kim, Kim, Yuan, Hill, Doshi & Thai, 2015; Kopcha et al., 2017) along with possibility to improve students' computational thinking skills (Burbaitė, Drąsutė & Štuikys, 2018). Therefore, in this study students are given STEM-based robotics education to improve their computational thinking skills and their attitude towards computer science.

1.2 Objective of the Thesis

The purpose of this study is to improve middle school students' computational thinking skills and their attitude towards computer science through STEM-based robotics education. For this purpose, a 6-week STEM-based robotics unit was

prepared. LEGO® WeDo 2.0 Core Set was used as the educational robotics set. To make the robotics unit STEM-based, the lessons were built upon STEM-related real world problems. Those problems were related to the science and mathematics objectives in the middle school curriculum published by Ministry of Education in Turkey. In the first 4 weeks, students were asked to construct robots to solve a STEM-based real world problem. In the fifth and sixth weeks, they were given an open-ended problem and asked to make an appropriate robot design and code it appropriately as a final project. By that means, the whole process included engineering and technology.

1.2.1 Research Questions

This study deals with the following research questions:

1. What are the effects of STEM-based robotics education on middle school students' computational thinking skills?

What are the effects of STEM-based robotics education on middle school students' attitude towards computer science?

1.3 Original Contribution

As mentioned above, students should gain computational thinking skills at early ages. Computer science education helps students gain computational thinking skills since it could be gained by acquiring the knowledge and skills that belongs to computer science (Hsu, Chang & Hung, 2018). On the other hand, the development of students' computational thinking skills might be negatively affected if students exhibit negative attitude towards computer science lessons (Baser, 2013a). Hence, improving students' attitude towards computer science may help them gain computational thinking skills better.

There are some studies conducted to decide the factors that affect students' attitude towards computer science. Those factors generally focus on gender, parental role models and self-efficacy (Başer, 2013a; Korkmaz & Altun, 2013; Moorman & Johnson, 2003; Özyurt & Özyurt, 2015; Peters & Pears, 2012; Teo, 2006). However, the connection between computer science and real-life issues might be an important factor that affects students' attitude as well. The number of studies on this issue is

very limited; therefore, there is a need to investigate students' computational thinking skills and their attitude towards computer science and offer a way to improve them. For this study, a STEM-based robotics unit was developed to close the gap between computer science and real-life issues for the sake of improving students' attitude towards computer science and their computational thinking skills. On the other hand, our STEM-based robotics unit will be an example for teachers who would like to give robotics education in a meaningful context since similar materials about this topic are very limited.

2
BACKGROUND

2.1 Computational Thinking

2.1.1 Definition and Components of Computational Thinking

One of the oldest uses of computational thinking term was mentioned by Seymour Papert (1980) in his book, entitled "Mindstorms: Children, computer and powerful ideas". Papert (1980) proposes that computers allow people to solve problems with a better analysis and explanation of them. Papert and colleagues developed the software LOGO that allow children to engage in programming in the late 1960's. The main purpose of LOGO was to help children think logically through a programming language (Cansu & Cansu, 2019). Children were supposed to move a *floor turtle* connected to a computer with a cord by giving special commands such as forward, back, left and right. For example, children would type *back 50* to make the turtle move backward by 50 steps or *left 90* to turn it 90 degrees left. Later, the LOGO software was included a screen turtle, so children could program the small graphic image of turtle on the screen by giving the same commands. Papert (1980) argued that children can reflect their own cognitive process in programming the turtle; thus, the turtle becomes an object that enables children to think concretely about thinking itself (Resnick, Ocko & Papert, 1988). Also, Papert (1980, 1991). He believed that children who manipulate the computer develop procedural thinking through programming. These are the ideas that the roots of computational thinking go back to.

The term was popularized by Wing in 2006 and draw the international community's attention. Wing (2006) argued that computational thinking includes skills applicable to everyone to solve problems and to design systems by applying techniques fundamental to computer science. Upon some improvements, Wing redefined the term as the thinking skill required to define solutions to problems that can be executed by agents that process information (2011). It focuses on 1) solving

problems by applying concepts from computer science; 2) organizing and analyzing data; 3) using abstractions to understand and solve problems in a more efficient way; and 4) thinking algorithmically to develop more efficient solutions; 5) automating solutions; 6) generalizing solutions to similar problems (Grover & Pea, 2013; Wing, 2006). What is important is that computational thinking does not require humans to think like a computer (Wing, 2006). Instead, it is about developing mental tools that leads to using computing to solve complex human problems (Reges, 2008, as cited in Lu & Fletcher, 2009). Lee et al. (2011) adopts a similar definition for computational thinking. They state that computational thinking is a skill that involves conceiving a problem, explaining it and providing solutions, reasoning it with abstraction, using automation and implementing solutions. As a result, computational thinking can be considered as a way of thinking and problem-solving technique that involves using digital technologies to find the most efficient solutions to problems (Aho, 2012).

Since computational thinking is a newborn term, there are other definitions suggested by academia and institutions as shown in Table 2.1. Although they may seem to be different, all those definitions approach computational thinking as a problem-solving method that uses tools and techniques from computer science and that can be applied across different disciplines other than computer science.

Table 2.1 Computational thinking definitions

Authors, Year	Definition
Aho (2011)	Computational thinking is the taught process required to solve problems through algorithms.
Barr and Stephenson (2011)	Computational thinking is to develop computational solutions to problems across different subjects.
The Royal Society (2012)	Computational thinking helps to recognize the computational aspects in the world. It involves tools and techniques from computer science that allows us to understand systems around us.

Table 2.1 Computational thinking definitions (devamı)

ISTE (2015)	Computational thinking is a problem-solving method whose characteristics can be listed as below: • Define problems and develop computational solutions that can be carried out by information-processing tools. • Organizing and analyzing data • Using abstraction to represent data • Using algorithmic thinking to automate solutions and generalize them to different problems across subjects • Finding the most appropriate solution in terms of efficiency and resources by evaluating each possible solution
Angeli et al. (2016)	Computational thinking involves the use of abstraction, generalization, decomposition, algorithmic thinking and debugging.
Kalelioglu, Gülbahar, and Kukul (2016)	Computational thinking includes different tools and methods from computer science but can be applicable to other disciplines other than computer science. That is to say, many disciplines and subjects can benefit from computational thinking to infer what can be computed.
Bocconi et al. (2016)	Computational thinking is a problem-solving method independent from technology. It allows to find solutions to problems in a way that can be implemented by a computer or human.
Csizmadia and Boulton (2017)	Computational thinking entails a problem-solving method using the concepts and practices from computer science.

What these definitions have in common that computational thinking basically includes breaking down problems into small ones (decomposition), removing

irrelevant details in those small problems (abstraction), writing a sequence of steps to solve each problem (algorithms) and using digital tools to make problem solutions mechanized (automation). Similarly, Wing (2006) proposes that automation, abstraction and algorithms are the most important three constructs computational thinking includes.

Algorithms are the sequence of steps written to solve problems or complete tasks. They are actually part of our lives (Yadav, Hong & Stephenson, 2016). For example, when someone follows a cooking recipe or gives directions from one place to another use algorithms. One way of learning algorithms for students is to identify the steps taken to complete daily tasks such as brushing teeth or cooking recipes (Deschryver & Yadav; 2015; Yadav, Mayfield, Zhou, Hambrusch & Korb, 2014).

Abstraction is the idea of removing irrelevant details to make an artifact more understandable (Cansu & Cansu, 2019). Besides, abstraction includes developing physical models of real world such as the one that represents the solar system (Yadav, Hong & Stephenson, 2016). Yadav, Hong and Stephenson (2016) claim that students can learn abstraction by analyzing data to form opinions and to deduce a principle.

Automation is about making computers or machines do repetitive tasks without human power by using the digital tools and simulations (Lee, 2011). With automation, computers are instructed to execute tasks quickly and efficiently. Lee et al. (2011) suggests that modeling and simulation, robotics and game design and development are the three ways of teaching students the idea of automation. For example, sensors on a robot always check the environment and return values to monitor the conditions. In game design, users' actions are immediately responded by the game itself.

In addition to these three key constructs, the Computer Science Teachers Association and ISTE proposed a definition for computational thinking that includes nine core computational thinking concepts to make computational thinking more applicable to K-12 (Barr and Stephenson, 2011). Those concepts and the short definitions of them is shown in Table 2.2.

Table 2.2 Computational thinking vocabulary chart (ISTE, 2011)

Concept	Definition
Data Collection	The process of collecting data
Data Analysis	Understanding data to detect patterns and draw conclusions
Data Representation	Organizing and demonstrating data with graphs, charts, words or images
Problem Decomposition	Breaking down problems to smaller ones
Abstraction	Removing irrelevant details to define main idea
Algorithms & Procedures	Series of steps written to solve a problem
Automation	Having computers do repetitive tasks
Simulation	Imitation or model of a process with which one can run experiments
Parallelization	Organizing resources in a way that carry out tasks simultaneously to achieve a common goal

2.1.1.1 Conceptual Framework

The present study relies on the computational thinking definition that is proposed by ISTE (2015). ISTE (2015) states that computational thinking refocuses on creativity, critical thinking and reasoning while emphasizing to solve problems using a computer. ISTE (2015) also highlights collaborative learning and communication as an important attribute a computational thinker has. The studies in the literature also depicts computational thinking as the combination of creativity, algorithmic thinking, problem solving, critical thinking and collaboration (Basogain et al., 2012; Binkley et al., 2012; ISTE, 2015; Sarıtepeci & Durak, 2017). In conclusion, computational thinking is the manifestation of creative thinking, algorithmic thinking, critical thinking, problem solving, collaborative learning and

communication (ISTE, 2015, as cited in Korkmaz, Çakır and Özden, 2015). Besides, research shows that 21st century learners should have innovation, creativity, research, collaboration, problem solving, critical thinking, social, technological, cognitive and communication skills (Günüç, Odabaşı & Kuzu, 2013). Therefore, the computational thinking definition proposed by ISTE (2015) can said to be very inclusive. As a result, the computational thinking definition of ISTE guided design and implementation of the current study.

2.1.2 The Importance of Computational Thinking

Today, computer technologies are used in different fields of study and has an important impact on the way work is done (Barr, Harrison & Conery; 2011; Yadav et al., 2014). They are part of our daily lives and work already and make their usage necessary to complete a great number of tasks (Barr, Harrison & Conery; 2011; Czerkawski, 2015; Wing, 2014). Computers and other technological tools empower the human thought and extend our ability of problem solving (Barr, Harrison & Conery, 2011). Consequently, we are supposed to be aware of how, when and where digital technologies and computers can be applied to solve problems (Barr, Harrison & Conery, 2011). In this sense, students must develop some key skills such as problem solving, decomposing problems into smaller ones, using digital technologies regardless of the profession they will choose in the future (Bubica & Boljat, 2018). As a reflection of being an increasingly information-based society, computational thinking appears to be an essential skill that individuals must develop (Yadav et al., 2014).

Computational thinking is a key competency for todays' students since they will work in fields heavily affected by computing and need to be able to deal with computing in their daily lives already (Barr & Stephenson, 2011; Grover & Pea, 2013). It is a skill that can be applied to a variety of disciplines other than computer science and informatics (Yadav et al., 2014). Computational thinking is accepted as a twenty-first century skill since non-computer scientists can also adopt a computational approach to solve problems (Cuny, Snyder & Wing, 2010). Those who have the ability to think computationally can use this skill to solve problems from different disciplines (Bundy, 2007). To illustrate, revealing trends in a population

(analysis) and making general principles from data (abstraction) are the examples of key computational thinking concepts embedded in social studies (Barr & Stephenson, 2011). Similarly, people in language arts can benefit computational thinking by making linguistic analysis of sentences and demonstrating patterns among different sentence structure (Yadav et al., 2014). Moreover, computational thinking meets the need of making students media information literate which consists of understanding how the representation of information and data affects the meaning (Wilson et al., 2013). Students' creativity is augmented with computational thinking since it allows students to become tool builders that leave an impact on society instead of just consuming the existing tools (Mishra & Yadav, 2013; Phillips, 2009). Even if a student prefers to work in a field other than computing, he/she will continue to benefit the skills developed through computational thinking (Mohaghegh & McCauley, 2016).

2.1.3 Computational Thinking in K-12 Curriculum

According to Wing, it is very important to teach students to think computationally in K-12 education in addition to teaching reading, writing and arithmetic (Wing, 2006). Similarly, National Research Council (NRC) suggests that almost everyone should gain computational thinking skills (NRC, 2010). As a result, the idea of integrating computational thinking to K-12 curriculum has been accepted by countries; however, this integration comes in different forms.

Different ways of incorporating computational thinking into K-12 classrooms were tried. Most of them relied on delivering computer science education through programming tools such as block-based programming environment (Yadav, Hong & Stephenson, 2016). Previous studies propose that students can acquire computational thinking skills through programming education (Garcia-Penalvo & Mendes, 2017; Koorsse et al., 2015; Yadav et al., 2011). Although it is possible to include computational thinking in various subjects, teachers generally prefer programming languages to teach it (Basogain et al. 2017; Lye & Koh, 2014; Zhong, Wang, Chen, & Li, 2016) since they believe that the most appropriate way of teaching computational thinking is through programming (Hsu, Chang & Hung, 2018).

Accordingly, most of the European countries integrate computational thinking to curriculum within computer science lessons (Bocconi, Chioccariello, Dettori, Ferrari & Engelhardt, 2016). For example, Poland has been delivering computer science and informatics education for a long time. With the new curriculum implemented in 2017, it is aimed to motivate students to apply computational thinking in different subjects (Bocconi et al., 2016). In Portugal, 7th and 8th graders are taught algorithm and programming concepts as part of a compulsory subject called information and communication technology (ICT) (Bocconi et al., 2016). England included different computational thinking courses in the school curriculum such as computer science, information technology and digital literacy (Brown, Sentence, Crick & Humphreys, 2014). In Turkey fifth and six graders are required to take computer technologies and software lesson which includes topics related to problem solving and programming, information technologies, communication, research and cooperation, making artefacts and internet ethics and security (Ministry of National Education [MONE], 2018a). Although those courses have different names, they all include computer science education. On the other hand, some European countries integrate computational thinking through courses other than computer science such as information technology, mathematics, information and communication technologies. To illustrate, France and Finland embed computational thinking to the curriculum via mathematics lessons while Austria, Poland, Italy, Portugal and Lithuania integrate computational thinking into informatics (Bocconi et al. 2016).

2.2 Attitude Towards Computer Science

Attitude is defined as how people evaluate objects through their cognitive, affective and behavioral information (Maio & Haddock, 2010). Researchers generally agree on the idea that attitude is very important for student learning (Bain et al., 2010; Kuhlemeier, Van Den Bergh & Melse, 1997; Mantle-Bromley, 1995; as cited in Liu, 2014). It is very difficult to engage someone in a subject without positive attitude regardless of what the subject is (Phillips & Brooks; 2017). Accordingly, Başer (2013a) claims that students' achievement in computer programming is determined by their attitudes towards programming. Hence, students' attitude towards computer science might affect their acquisition of computational thinking.

Therefore, students' attitude towards computer science should be investigated to contribute their computational thinking skills.

Teaching context is one possible factor that have influence on students' attitude towards computer science. It is very important for students' learning and attitude (Cooper & Cunningham, 2010). Cooper and Cunningham (2010) claim that context is necessary for learning and can be used to support student learning. They state that a powerful motivation for learning can be achieved with context since they believe it is a technique that helps students work with the content of computer science courses (Cooper & Cunningham, 2010). Furthermore, there are some criticisms about the way computer science curriculum is applied which include putting a great emphasis on programming skills rather than problem solving (Jona et al., 2014). Computer science lessons who solely focus on teaching programming skills prevent students from comprehending how the language can be used to solve a problem (Goode, Estrelle & Margolis, 2005). As a result, students might consider computer science-related works as less creative and unattractive, which may cause them to develop a negative attitude towards the discipline. In the light of these arguments, it clearly appears that designing computer science lessons in an engaging context is very important in terms of helping students develop a positive attitude towards computer science. In this sense, Burbaite, Drasute and Stuikys (2018) proposes STEM-paradigm as a means to provide a concrete context that motivates and engages students in computer science education. Moreover, some researchers argue that a concrete context can be achieved with educational robotics by appealing students who like to build and manipulate something physically (Bers, 2008; Cooper & Cunningham, 2010). When students apply STEM concepts to solve real world problems in their robotic projects, they experience authentic amazement moments (Eguchi, 2016), which might help students develop a positive attitude towards computer science.

2.3 Teaching Computational Thinking with STEM-Based Robotics Education

Robotics provides a possibility to motivate and engage students in learning activities (Burbaite, Drasute & Stuikys, 2018). Andic et al. (2015) claims that

educational robotics help students process information faster. It is an important learning tool to improve students' computational thinking skills (Eguchi, 2016). It provides a concrete context that allows students understand abstract concepts since students get instant feedback when they test their programs (Bers, 2008). However, working with robots in a lesson does not assure that students will learn to think computationally (Kopcha et al., 2017). To that end, students must work with robotics in a meaningful environment which includes real world problems and explorations that allow students experience authentic STEM skills (Pea, 1987). When students apply STEM concepts to solve real world problems in their robotic projects, they experience authentic amazement moments (Eguchi, 2016).

Choi et al. (2016) presents a theoretical framework for a STEM-integrated robotics curriculum for fifth grade that is consistent with science and mathematics standards. The aim of the STEM-integrated robotics curriculum is to teach computer programming in a meaningful context where students write a program to solve a problem. The researchers developed an eight-lesson module in each of which students were given a driving question related to a real world scenario and expected to understand the problems, write the necessary programs to solve them and optimize those programs in the last two weeks.

Eguchi and Uribe (2017) conducted a study about an educational robotics unit they have developed. The unit included different programming challenges based on the 4th grade science curriculum. Researchers define educational robotics as using robotics as a learning tool and suggest that STEM and computational thinking come together in a project through educational robotics. In this unit, students are provided a challenge to build a moving fans. Later, students are provided ten programming challenges that will improve their programming skills. At the end, students are told to begin working on their final robotics project based on their ideas. Students' progress is followed via engineering journals. While working on the programming challenges and final project, students make practice about how to define problems in a way that can be solved by developing a programmable solution. This study provides a good rationale as to integrating robotics and STEM education through project-based learning. However, the science standards targeted in this

study are linked to engineering design, not science content. For example, one of those standards says that students will be able to "define a simple design problem reflecting a need or a want that includes specified criteria for success and constraints on materials, time, or cost" (Next Generation Science Standards [NGSS], 2013, p. 46).

Burbaite, Drasute and Stuikys (2018) conducted a study for the purpose of showing how STEM-driven computer science education helps to develop high school students' computational thinking skills. In addition, the researchers introduce a model by connecting computational thinking skills with Bloom's taxonomy to evaluate students' success. They propose a programming curriculum at high school based on STEM-driven approach and computational thinking skills. For this purpose, they introduce two case studies. In the first one, the researchers analyze the functionality and programming principles of ultrasonic sensor which calculates the distance to the nearest object by sending a sound wave that cannot be heard by humans. By that means, students talk about the physical principles that allows the sensor to measure the distance between objects. In the second case, students develop robot control programs. They develop robotic models with LEGO® NXT robotics set and program their models with a software.

Yang, Swasnon, Chittoori, and Baek (2018) developed a project-based, STEM+C (Computing) curriculum for 4th to 6th graders as a part of NSF (National Science Foundation)-funded STEM+C project. During the development of curriculum, the researchers focused on two aspects of computational thinking: 1) using abstraction to solve problems; 2) discussing and sharing ideas in computational terms (Yang et al., 2018). The curriculum includes two different projects: Life on Mars and Building Earthquake Resistant Bridges. In the Life on Mars project, students are required to build a robot using LEGO® Mindstorms robotic kits which detects if there is life on a simulated Mars built by researchers. In this project, students apply computational thinking and STEM subject knowledge to build a bridge that can resist earthquake forces. Both projects require students to integrate computational thinking and STEM using computational thinking components of vocabulary, abstraction, decomposition, algorithms, automation, conditional logic, heuristics, data collection,

data analysis and representation, simulation and modeling, and communication to solve the driving questions.

The studies above are examples of integrating STEM approach and computational thinking through educational robotics. However, these studies do not provide any empirical data as to the development of students computational thinking skills. In addition, students' attitude towards computer science should be investigated along with developing strategies to improve their attitude since attitude is very important for student learning (Bain et al., 2010; Kuhlemeier, Van Den Bergh & Melse; 1997; Mantle-Bromley, 1995, as cited in Liu, 2014). As a result, attitude towards computer science can play an important role on the development of students' computational thinking skills. However, the number of studies conducted in this issue is very limited. Therefore, there is a need to investigate the effects of STEM-based computer science education on students' computational thinking skills and their attitude towards computer science.

3
METHOD

3.1 Research Design

This study aimed to improve students' computational thinking skills and their attitudes towards computer science through STEM-based robotics education, as measured by the Computational Thinking Levels Scale (Korkmaz, Çakır & Özden, 2015) and Computer Programming Attitude Scale (Başer, 2013b). This study uses one-group pre-test post-test research design (Allen, 2017). In this research design, a single group of participants is given a pre-test at the beginning of the study. Upon implementing of the study, the same group were given a post-test to determine if there is a change between the beginning and end of the treatment (Allen, 2017). Accordingly, the participant students answered the Computational Thinking Levels Scale (Korkmaz, Çakır & Özden, 2015) and Computer Programming Attitude Scale (Başer, 2013b) at the beginning of the study. Later, 6-week STEM-based robotics education was delivered. Upon finishing the implementation, students answered Computational Thinking Levels Scale (Korkmaz, Çakır & Özden, 2015) and Computer Programming Attitude Scale (Başer, 2013b) as a post-test.

3.2 Participants

To determine the participants, first the school where the study would be conducted was specified. The researcher of this study was working at a private school located at Başakşehir district of Istanbul at the time the research was planned to carry out, so the study was conducted at there in the spring term of academic year of 2019-2020. The number of students who participated in the study were 12. They are all male students from different grades (6 students from eighth grade, 2 from sixth grade and 4 from fifth grade) and voluntarily participated in this research. Their ages varied between 11 and 13. Those students chose robotics class as elective course, so they were all in the same class at the robotics lesson hour. All students

took computer technologies and programming lessons in the previous years of their education either as elective or compulsory course. They learned programming with Scratch software developed by Massachusetts Institute of Technology (Resnick et al., 2009). However, it was the first time for them to work with robotics.

3.3 Role of Researcher

The researcher saw students questioning the usefulness of programming so many times and not appreciating the value of computational thinking skill development. After reviewing the literature, she thought integrating STEM with robotics might improve students' attitude towards computer science and their computational thinking skills in a meaningful context. Therefore, she chose to conduct a research about STEM-integrated robotics education.

As mentioned in the participants section above, the researcher of this study was working as a computer science teacher at the school where the study was conducted and gave robotics education in robotics club throughout the year. The study was conducted with the students in robotics club. Therefore, the researcher also was the teacher who delivered the STEM-based robotics unit.

3.4 Development of STEM-Based Robotics Unit

The definition of STEM integration can be made as combining science, technology, engineering and mathematics to enhance both student understanding of those disciplines and interest in STEM disciplines to motivate students to take part in STEM fields (Moore, 2008). There are five different characteristics of a STEM-integrated instruction: 1) a content that brings science and mathematics disciplines under previously-specified learning goals; 2) engineering practices as the context of the content that will be thought; 3) including relevant technologies that require design justification using of scientific and mathematical concepts; 4) an emphasis on 21st century skills; 5) a context including a real-world problem to be solved through teamwork (Bybee, 2013; Moore et al., 2014; NRC, 2014; NRC, 2012; Partnership for 21st Century Skills, 2009; Sanders, 2009; as cited by Bryan, Moore, Johnson & Roehrig; 2016). In the present study, the STEM-based robotics education includes all the characteristics of STEM-integrated education with one exception. Rather than

science and mathematics concepts, teaching computational thinking through the use of robotics is the primary goal of STEM-based robotics unit.

The STEM-based robotics unit was developed based on the concepts in science curriculum for primary and secondary schools (MoNE, 2018b) and in the lesson ideas on the LEGO® WeDo website (The LEGO Group, n.d.). Throughout the unit, students are given a real-world problem and asked to design a solution with their groupmates by using robotics. By that means, 21st century skills (collaboration, communication, critical thinking and creativity) are emphasized (Sipayung, Sani & Bunawan, 2018). Those problems were chosen based on the objectives of science and mathematics education specified in the curriculum of Ministry of Education in Turkey. For example, the earthquake-resistant building project is related to the natural disasters unit of fifth grade science curriculum and area calculation unit in fifth and sixth grade mathematics curriculum. Similarly, the mars rover project includes objectives from sixth grade solar system unit and seventh grade space research unit in the science curriculum. Thus, students needed to use those objectives to develop a solution to the problems.

The STEM-based robotics unit includes 5 lesson plans. During the first 4 weeks, students were asked to construct robots to solve a STEM-based real-world problem by following the instructions on the LEGO® WeDo software. Each week a lesson takes 70 minutes and students work with a different project. In the fifth and sixth week, they were given an open-ended problem and asked to make an appropriate robot design without any instruction and code it appropriately as a final project. Lessons are briefly explained in Table 3.1.

Table 3.1 6-week STEM-based robotics unit

Week #	STEM-based real-world problem	Lesson explanation
Week 1, 2, 3	How do scientists explore the places they cannot go?	Students learn how LEGO® WeDo works and to program the smarthub. Students gain an insight about what they can do with LEGO® WeDo robotics set. They design a rover by using tires, motor and other LEGO bricks. Later, students learn motion sensor and add it to the rover they have built the previous week. As a result, the rover senses the objects ahead of it. Lastly, students build a joystick by using tilt sensor to control the rover remotely.
Week 4	What is the good way of building earthquake-resistant structures?	Students build an earthquake simulator by using the various LEGO parts they have used in the previous lessons.
Week 5 & 6	How can we keep oceans clean of plastic debris?	Students will design a device and program it appropriately. They will not be given any instructions.

3.5 Implementation

The STEM-based robotics unit was examined by an expert who have experience with both computational thinking and STEM education. Later, the necessary permissions were taken from the university ethics committee and Provincial Directorate for National Education of İstanbul. Upon completing the permission process, the study was started to be conducted. At the beginning of the study, students were explained the research and asked if they willing to participate to the study. 12 students become volunteer out of 16. Those students were delivered the Computational Thinking Levels Scale (Korkmaz, Çakır & Özden, 2015) and Attitude Scale toward Computer Programming (Başer, 2013b). Students were given 20 minutes to fill the tests. Then,

the researcher collected them all and started to deliver the STEM-based robotics unit.

The implementation took 6 weeks as planned. During each project, students worked with pairs to develop their collaboration skills. They were asked to brainstorm and complete the project without being given a specific role. At the beginning of each project, students were defined the problem situation and asked how that problem can be solved with the help of robotics. At the end of the discussion, students were explained what their robots should accomplish. The design and coding of the robot were left to students. Later, students started to build the robots and finished them until the end of the lesson. The first four weeks were delivered in the same way. In the fifth lesson, students were given an open-ended problem. This time students were not given anything about what their robots should do. They were told to find how their robots should be to solve the given problem. During the fifth and sixth weeks, students work on their final projects and present them to their classmates. Lastly, the participant students were again asked to answer the questions in the Computational Thinking Levels Scale (Korkmaz, Çakır & Özden, 2015) and Attitude Scale toward Computer Programming (Başer, 2013b).

To explain the implementation process in detail, students were given a brief information about what they would do in the next 6 lessons and introduced the first STEM-based real-world problem that would be addressed for the next three lessons in the first week. After making necessary explanations, students were asked how scientists can make investigations in places they cannot go to. Upon some discussion, students were told to build a rover prototype with the robotics kit. In the first lesson, students learned how LEGO® WeDo works and is programmed. They gained insight about what they can do with LEGO® WeDo set and designed a rover by using tires, motor and other LEGO bricks. In the second week, students were asked how autonomous vehicles sense the environment and respond accordingly. By that means, they were acquainted with the concept of sensor which is the electronic devices that gathers information from the environment such as the existence of an object or the temperature of air. Students were asked how the rovers can move by themselves without hitting an object. They were shown motion sensor

that can detect physical objects nearby. Students added the motion sensor to the rover they had built the previous week and make it an autonomous rover. In the third week, students were asked how to control the rover remotely. Then, they were told to build a joystick using tilt sensor and program it in a way that navigates the rover based on the direction coming from the joystick.

In the fourth week, a new STEM-based real-world problem was given to the students. Students were asked what is the good way of building an earthquake-resistant structure. They were shown earthquake-resistant structures videos and discussed the features of those structures. Then, students were told to build an earthquake simulator by using the various LEGO parts they have used in the previous lessons. Later, they build different structures and tested them for the maximum earthquake magnitude they can resist.

In the last two lessons, students were given open-ended tasks. They were presented the problem of plastic pollution in the oceans. Students watched some videos about the problem and asked to build and code a device using robotics to keep the oceans clean of plastic debris. Unlike previous lessons, students were not provided any

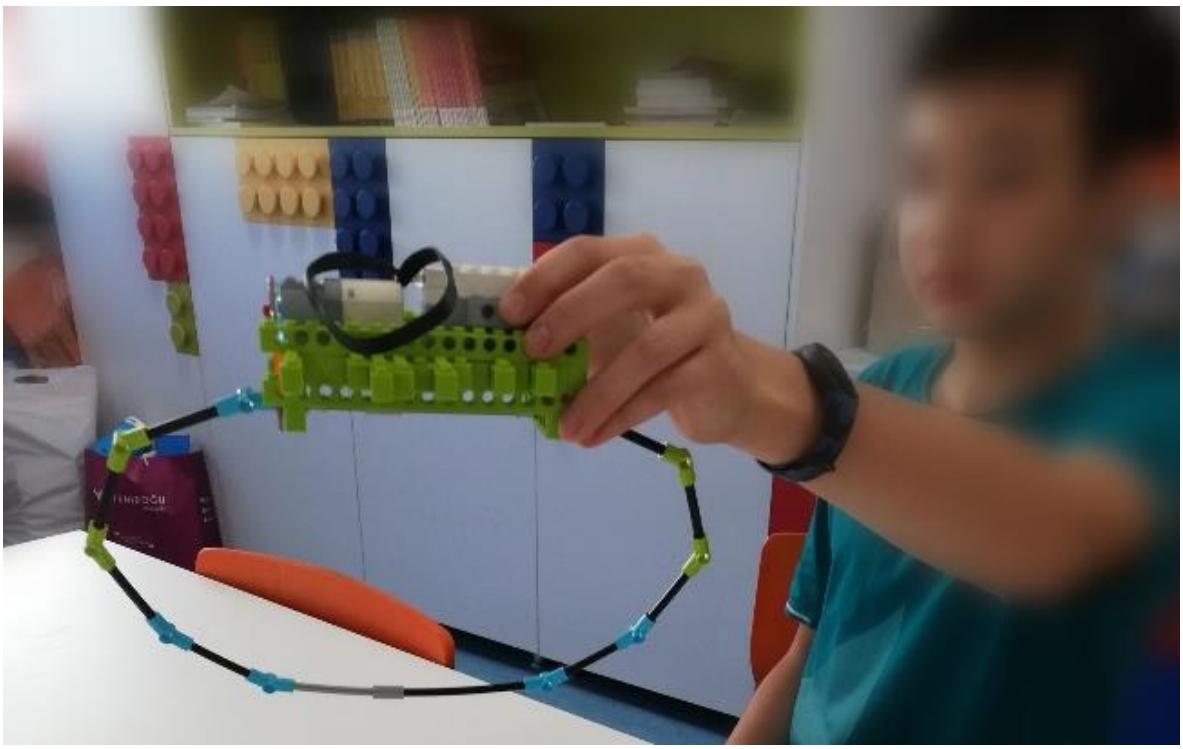

Figure 3.1 Garbage collector with netting

building instructions. Some students made novel designs and build devices that collects garbage from the ocean. For example, one student thought of collecting plastic garbage with a netting system as shown in Figure 3.1. He could not use a netting but explained it in his presentation. Another student made a device that has a dustbin in which garbage was kept as in Figure 3.2.

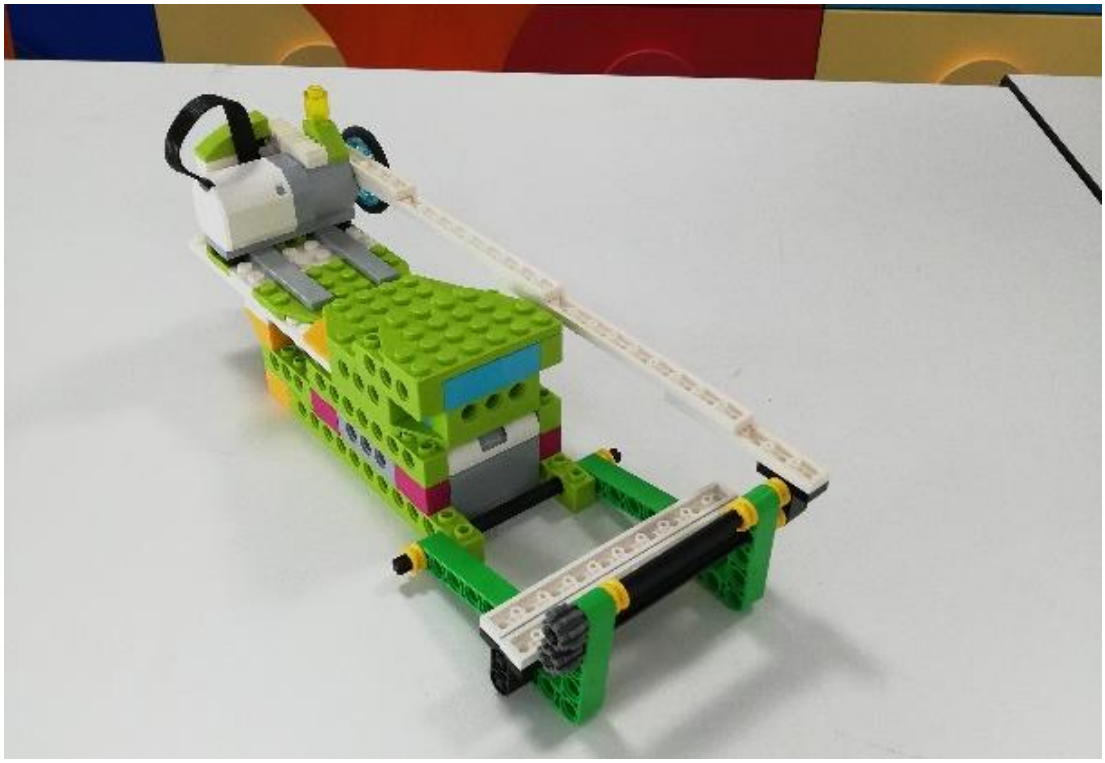

Figure 3.2 Garbage collector with a dustbin

The detailed lesson plans can be seen in Appendix A.

3.6 Data Sources

Data was collected through two different scales. For the first research question, Computational Thinking Levels Scale (Korkmaz, Çakır & Özden, 2015) was used as both pre-test and post-test to measure students' computational thinking skills. For the second research question, Computer Programming Attitude Scale (Başer, 2013b) was used to measure students' attitudes towards computer science.

3.6.1 Computational Thinking Level Scale

Computational Thinking Levels Scale is developed by Korkmaz, Çakır and Özden (2015) to assess students' computational thinking skills based on the definition proposed by ISTE (2015). It is a five-point Likert type scale ranging from strongly agree to strongly disagree with 22 items under five different factors. Those factors are creativity, algorithmic thinking, cooperation, critical thinking and problem solving. Number of items and Cronbach Alpha reliability value were found by the researchers as shown in Table 3.2. The Cronbach Alpha value of the creativity factor is lower than .70. However, since the Cronbach Alpha for the total score is high enough, it can be said that the internal consistency of the scale is high enough, which means the scale can make reliable measurements.

Table 3.2 Cronbach alpha reliability test results for Computational Thinking Levels Scale (Korkmaz et al., 2015)

Factor	Number of Items	Cronbach Alpha
Creativity	4	.640
Algorithmic Thinking	4	.762
Cooperation	4	.811
Critical Thinking	4	.714
Problem Solving	6	.867
Total	22	.809

3.6.2 Computer Programming Attitude Scale

Başer (2013b) developed Computer Programming Attitude Scale (Başer, 2013b) was to determine the change in the students' attitude towards computer programming. To develop this instrument, Başer (2013b) first translated the Computer Science Attitude Survey developed by Wiebe's et al. (2003) to Turkish. The scale developed by Wiebe at al. (2003) is a 5 Likert type scale ranging from

strongly agree to strongly disagree and consisting of 57 items about the attitude towards computer science and computer programming. The "computer science" expression in the original survey was translated as "computer programming" by Başer (2013b). The reason to this might be that computer programming is a term widely used in Turkey compared to computer science. After reliability analysis, 19 items were removed by the researcher. The final version of the scale included 38 items under four subscales. Those factors are self-confidence in programming, benefits of programming, attitude against the success at programming and social perception of success in programming. Cronbach Alpha value for each factor and number of items were found by the researcher as illustrated in Table 3.3.

Table 3.3 Cronbach alpha reliability test results for Attitude Scale toward Computer Programming (Baser, 2013b)

Factor	Number of Items	Cronbach Alpha
Self confidence in programming	17	0.944
Benefits of programming	10	0.920
Attitude against the success at programming	8	0.926
Social perception of success in programming	3	0.618
Total	38	0.953

3.7 Data Analysis Procedure

To determine if there is a statistically significant difference between the pre-test and post-test scores on the Computational Thinking Levels Scale (Korkmaz, Çakır & Özden, 2015) and Attitude Scale toward Computer Programming (Başer, 2013b), a paired sample t-test was determined to be applied since it is used to compare the mean scores of the same group collected at two different points in time (Pallant,

2020). Two important assumptions for a paired sample t-test were checked. Those assumptions are: 1) the dependent variable must be interval or ratio; 2) the data must be normally distributed. To determine the normality, the pre-test and post-test scores were entered to SPSS software. Shapiro-Wilk normality test was conducted on the difference between pre-test and post-test scores since the number of participant students are lower than 50. In case of a normal distribution, it was decided to conduct a paired sample t-test. Otherwise, a Wilcoxon signed rank test was applied which is the non-parametric alternative of a paired sample t-test.

4
RESULTS AND DISCUSSION

4.1 Results

In this section, the effects of STEM-Based robotics education on middle school students' computational thinking skills and the five sub-dimensions under computational thinking skills (creativity, algorithmic thinking, cooperativity, critical thinking and problem solving) were examined.

4.1.1 Effects of STEM-Based Robotics Education on Middle School Students' Computational Thinking Skills

According to the results of Shapiro-Wilk normality test, the difference data coming from Computational Thinking Levels Scale was not normally distributed as shown in Table 4.1. As a result, a Wilcoxon signed rank test was applied instead of a paired sample t-test. To compare the pre-test and post-test scores of each factor in the scale, the same procedure was followed. As shown in Table 4.1, the difference in each subscale was normally distributed except creativity. Therefore, a paired sample t-test was applied to determine if there is any significant different between pre-test and post-test scores of algorithmic thinking, cooperativity, critical thinking and problem solving while a Wilcoxon signed rank test was applied on the data of creativity.

Table 4.1 The results of the normality test of the difference between the pre-test and post-test scores in Computational Thinking Levels Scale

	Statistic	**df**	**Sig.**
Computational Thinking	.769	11	.004
Creativity	.832	11	.022

Table 4.1 The results of the normality test of the difference between the pre-test and post-test scores in Computational Thinking Levels Scale (devamı)

Algorithmic thinking	.949	11	.627
Cooperativity	.946	11	.582
Critical Thinking	.893	11	.128
Problem Solving	.876	11	.079

A Wilcoxon signed rank test was conducted to determine whether the students' computational thinking skills were improved. The results of the Wilcoxon signed rank test in Table 4.2 demonstrated that STEM-based robotics activities resulted in statistically significant increase in student's computational thinking skills; (Z = -2.316, $p < .05$).

Table 4.2 The results of Wilcoxon signed rank test showing pre-test and post-test scores in Computational Thinking Levels Scale

	M	**SD**	**Z**	**p**
Pre-test	78	12.541	-2.316	.021
Post-test	85.42	16.005		

4.1.1.1 Effects of STEM-Based Robotics Education on Middle School Students' Creativity

Since the difference scores do not have a normal distribution (see Table 4.1), a Wilcoxon signed rank test was conducted to determine whether the students' creativity was improved. The results of the Wilcoxon signed rank test in Table 4.3 demonstrated that STEM-based robotics activities resulted in statistically significant increase in student's creativity; (Z=-2.414, $p < .05$).

Table 4.3 The results of Wilcoxon signed rank test showing pre-test and post-test scores for creativity in Computational Thinking Levels Scale

	M	**SD**	**Z**	**p**
Pre-test_C	16.75	12.541	-2.414	.016
Post-test_C	17.83	16.005		

4.1.1.2 Effects of STEM-Based Robotics Education on Middle School Students' Algorithmic Thinking

Since the difference scores have a normal distribution (see Table 4.1), a paired sample t-test was conducted to determine whether the students' algorithmic thinking skill was improved. Although there is an increase in students algorithmic thinking skill, the results of the paired sample t-test was in Table 4.4 demonstrated that there is not a significant increase in student's algorithmic thinking after the STEM-based robotics activities; ($t(12)=2.091$, $p > .05$).

Table 4.4 The results of Wilcoxon signed rank test showing pre-test and post-test scores for algorithmic thinking in Computational Thinking Levels Scale

	M	**SD**	***t*(12)**	***p***
Pre-test_A	13.75	3.888	2.091	.061
Post-test_A	15.67	4.479		

4.1.1.3 Effects of STEM-Based Robotics Education on Middle School Students' Cooperativity

Since the difference scores have a normal distribution (see Table 4.1), a paired sample t-test was conducted to determine whether the students' cooperativity was improved. The results of the paired sample t-test was in Table 4.5 demonstrated that

there is not a significant increase in student's algorithmic thinking after the STEM-based robotics activities; ($t(12)=1.159$, $p > .05$).

Table 4.5 The results of Wilcoxon signed rank test showing pre-test and post-test scores for cooperativity in Computational Thinking Levels Scale

	M	**SD**	***t*(12)**	***p***
Pre-test_O	15.58	3.888	1.159	.521
Post-test_O	15.08	4.479		

4.1.1.4 Effects of STEM-Based Robotics Education on Middle School Students' Critical Thinking

Since the difference scores have a normal distribution (see Table 4.1), a paired sample t-test was conducted to determine whether the students' cooperativity was improved. Although there is an increase in students critical thinking skill, the results of the paired sample t-test was in Table 4.6 demonstrated that there is not a significant increase in student's critical thinking skills after the STEM-based robotics activities; ($t(12)=.914$, $p > .05$).

Table 4.6 The results of Wilcoxon signed rank test showing pre-test and post-test scores for critical thinking in Computational Thinking Levels Scale

	M	**SD**	***t*(12)**	***p***
Pre-test_T	14.67	3.055	.914	.380
Post-test_T	15.67	3.312		

4.1.1.5 Effects of STEM-Based Robotics Education on Middle School Students' Problem Solving

Since the difference scores have a normal distribution (see Table 4.1), a paired sample t-test was conducted to determine whether the students' cooperativity was improved. Although there is an increase in students problem solving thinking skill,

the results of the paired sample t-test was in Table 4.7 demonstrated that there is not a significant increase in student's problem solving skills after the STEM-based robotics activities; ($t(12)=1.697$, $p > .05$).

Table 4.7 The results of Wilcoxon signed rank test showing pre-test and post-test scores for problem solving in Computational Thinking Levels Scale

	M	**SD**	***t*(12)**	***p***
Pre-test_P	17.25	6.690	1.697	.118
Post-test_P	21.17	7.433		

In summary, a Wilcoxon signed rank test was applied to answer the first research question since the data was not normally distributed and there are two measurements of the same group (pre-test and post-test). The result conveyed that there is a significant increase in students' computational thinking skills after STEM-based robotics unit. In addition, the pre-test and post-test data coming from each factor was tested against normality. It was seen that creativity scores were not normally distributed whereas algorithmic thinking, cooperativity, critical thinking and problem-solving scores had a normal distribution. Therefore, a Wilcoxon signed rank test was applied to determine if there is a significant difference between the pre-test and post-test scores of creativity scores. The results showed that STEM-based robotics activities resulted in statistically significant increase in student's creativity; ($Z=-2.414$, $p < .05$). On the other hand, a paired sample t-test was test were used for algorithmic thinking, cooperativity, critical thinking and problem-solving scores since those scores were normally distributed. The results demonstrated that the difference between algorithmic thinking, cooperativity, critical thinking and problem-solving scores were not statistically significant.

4.1.2 Effects of STEM-Based Robotics Education on Middle School Students' Attitude Towards Computer Science

In this section, the effects of STEM-Based robotics education on middle school students' attitude towards computer science with the four sub-dimensions under

(confidence and motivation in learning programming, usefulness of programming, attitude toward success in programming and social perception of success in programming) were examined.

To test if there is a significant difference between the pre-test and post-test scores on the Attitude Scale toward Computer Programming (Başer, 2013b), normality test was conducted. The data was seen to be normally distributed, so a paired sample t-test was conducted to determine whether there is a significant difference between pre-test and post-test scores. Furthermore, the difference between the pre-test and post-test scores of each subscale is tested for normality. The results (see Table 4.8) demonstrated that the data in "confidence and motivation in learning programming" and "usefulness of programming" was normally distributed; therefore, a paired sample t-test was applied to determine if there is any significant difference between the pre-test and post-test scores. On the other hand, the data "attitude toward success in programming" and "social perception of success in programming" did not have a normal distribution.

Table 4.8 The results of the normality test of the difference between the pre-test and post-test scores in Attitude Scale toward Computer Programming

	Statistic	**df**	**Sig.**
Attitude towards Computer Science	.952	9	.695
Confidence and motivation in learning programming	.890	9	.168
Usefulness of programming	.887	9	.157
Attitude toward success in programming	.780	9	.008
Social perception of success in programming	.817	9	.023

Students' pre-test and post-test scores for their attitude towards computer science were analyzed to investigate if there is a significant difference between them. Two students were removed from the data set since they did not want to fill the post-test. Since the difference scores has a normal distribution (see Table 4.8), a paired sample t-test was conducted to determine whether the students' attitudes towards computer science were improved. The results of the paired sample t-test in Table 4.9 demonstrated that STEM-based robotics activities resulted in statistically significant increase in student's attitude towards computer science; ($t(10) = 3.075$, $p < .05$).

Table 4.9 The results of the paired sample t-test showing pre-test and post-test scores in Attitude Scale toward Computer Programming

	M	**SD**	***t*(10)**	***p***
Pre-test	178.3	30.804	3.075	.013
Post-test	194.4	29.444		

4.1.2.1 Effects of STEM-Based Robotics Education on Middle School Students' Confidence and Motivation in Learning Programming

Since the difference scores have a normal distribution (see Table 4.8), a paired sample t-test was conducted to determine whether the students' confidence and motivation in learning programming were improved. Although there is an increase in students' confidence and motivation in learning programming skill, the results of the paired sample t-test was in Table 4.10 demonstrated that there is not a statistically significant increase in student's confidence and motivation in learning programming after the STEM-based robotics activities; ($t(10)=1.814$, $p > .05$).

Table 4.10 The results of the paired sample t-test showing pre-test and post-test scores in confidence and motivation in learning programming

	M	**SD**	***t*(10)**	***p***
Pre-test	63	10.414	1.814	.103
Post-test	67.90	11.338		

4.1.2.2 Effects of STEM-Based Robotics Education on Middle School Students' Views on Usefulness of Programming

Since the difference scores have a normal distribution (see Table 4.8), a paired sample t-test was conducted to determine whether the students' views on usefulness of programming were improved. Although there is an increase in students' views on usefulness of programming, the results of the paired sample t-test was in Table 4.11 demonstrated that there is not a statistically significant increase in student's views on usefulness of programming after the STEM-based robotics activities; ($t(10)= 1.742$, $p > .05$).

Table 4.11 The results of the paired sample t-test showing pre-test and post-test scores in usefulness of programming

	M	**SD**	***t*(10)**	***p***
Pre-test	38.70	10.414	1.742	.115
Post-test	42.30	11.338		

4.1.2.3 Effects of STEM-Based Robotics Education on Middle School Students' Attitude towards Success in Programming

Since the difference scores were not normally distributed (see Table 4.8), a Wilcoxon signed rank test was conducted to determine whether the students' attitudes towards success in programming were improved. The results of the Wilcoxon signed rank test was as shown in Table 4.12 demonstrated that there is a

statistically significant increase in student's attitudes towards success in programming after the STEM-based robotics activities; (Z=-2.527, $p < .05$).

Table 4.12 The results of the Wilcoxon signed rank test showing pre-test and post-test scores in success in programming

	M	**SD**	**Z**	***p***
Pre-test	33	6.766	-2.527	.012
Post-test	36.80	3.994		

4.1.2.4 Effects of STEM-Based Robotics Education on Middle School Students' Social Perception of Success in Programming

Since the difference scores were not normally distributed (see Table 4.8), a Wilcoxon signed rank test was conducted to determine whether the students' social perception of success in programming were improved. Although there is an increase in students' social perception of success in programming, the results of the Wilcoxon signed rank test was as shown in Table 4.13 demonstrated that there is not a statistically significant increase in student's social perception of success in programming after the STEM-based robotics activities; (Z=-1.364, $p > .05$).

Table 4.13 The Results of the Wilcoxon Signed Rank Test Showing Pre-test and Post-test Scores in Success in Programming

	M	**SD**	**Z**	***p***
Pre-test	9.50	4.327	-1.364	.172
Post-test	11.40	3.307		

In summary, a paired sample t-test was applied to answer the second research question since the data was normally distributed and there are two measurements of the same group (pre-test and post-test). The result demonstrated that there is a

significant increase in students' attitudes towards computer science after STEM-based robotics unit. In addition, the pre-test and post-test data coming from each subscale was tested against normality. It was seen that confidence and motivation in learning programming scores and usefulness of programming scores were normally distributed whereas attitude toward success in programming scores and social perception of success in programming scores did not have a normal distribution. Therefore, a paired sample t-test was applied to determine if there is a significant difference between the pre-test and post-test scores of confidence and motivation in learning programming and usefulness of programming. The results showed that STEM-based robotics activities resulted in a non-significant increase in student's confidence and motivation in learning programming ($t(10)=1.814$, $p > .05$) and in usefulness of programming ($t(10)= 1.742$, $p>.05$). On the other hand, a Wilcoxon signed rank test was used for attitude toward success in programming and social perception of success in programming scores since they were not normally distributed. The results demonstrated that there is a significant increase in students' attitudes toward success in programming; ($Z=-2.527$, $p<.05$). However, the difference between the pre-test and post-test scores of social perception of success in programming subscale were seen to be statistically non-significant.

4.2 Discussion

This study aimed to investigate how STEM-based robotics education affects students' computational thinking skills and their attitudes towards computer science. In this chapter, the effects of STEM-based robotics education on students' computational thinking skills were examined with sub-dimensions of creativity, algorithmic thinking, cooperativity, critical thinking, problem solving. Also, it was discussed how STEM-based robotics education changed students' attitude towards computer science. This change was handled under sub-dimensions of confidence and motivation in learning programming, usefulness of programming, attitude toward success in programming and social perception of success in programming.

4.2.1 STEM-Based Robotics to Improve Computational Thinking Skills

The results of the present study demonstrated that STEM-based robotics education made a statistically significant positive change in students' computational thinking

skills. That might be because robotics provided students a concrete context that allow them to build and manipulate something physically. Students felt more motivated when they work with something to tinker, play and share with others. They saw how their actions were ended up immediately and came up with new solutions if there was a problem with the robot they have built. During the projects, students worked with pairs, argue how to solve problems, built a robot and programmed it. Programming a robot might have improved students' computational thinking skills since they needed to decompose problems and write algorithms to achieve a task. Constantinou and Ioannou (2018) who conducted a two-group pre-test-post-test study with 7th, 8th and 9th graders reported similar results. The researchers state that students who worked with educational robotics gained higher computational thinking skills compared to those in the control group. Besides, Chookaew, Howimanporn and Pratumsuwan (2018) investigated if there is a meaningful difference in high school students' computational thinking skills after a three-day workshop with STEM-associated robotics-based learning activities. The study revealed that there is a positive correlation between students' performance in robotics and their computational thinking skills on different dimensions including problem solving, creativity and logical thinking. These studies (Chookaew, Howimanporn & Pratumsuwan, 2018; Constantinou & Ioannou, 2018) and the current study has significant implications for computer science education. First, educational robotics presents students with the opportunity to practice computational thinking in a concrete way. Students apply abstract computational thinking concepts such as decomposition, abstraction and debugging in robotics activities. For example, to build a robot students first make a robot design and decide which sensors and other materials are necessary and write an algorithm that will run the robot. Thus, they practice decomposition concept. Similarly, they test their robots all the time and solve the problems that arise during tests, which is an example of debugging. Second, educational robotics might contribute to students' creativity and imagination since they can design unique robotics projects. All these implications lead to the conclusion that STEM-based robotics applications should take place in learning environments to improve students' computational thinking skills.

4.2.1.1 STEM-Based Robotics to Improve Creativity

The results of the present study demonstrated that STEM-based robotics activities led to a statistically significant increase in student's creativity. The reason of this increase might be related to the fact that students are required to design different robots or manipulate the existing ones during the STEM-based robotics education. A similar conclusion can be found in the study of Khanlari (2013). Khanlari (2013) collected data about the effects of robotics on students' creativity from seven robotics teachers through interview. All participants agreed on the idea that robotics enhances student creativity (Khanlari, 2013). Furthermore, Anwar, Bascou and Menekse (2019) found 53 studies about the relationship between educational robotics and student creativity. The researchers state that these studies demonstrated that working with educational robotics provide the possibility to improve creativity (Anwar, Bascou & Menekse, 2019). In addition, Zawieska and Duffy (2015) state that robotics provides an opportunity for hands-on learning which is the missing part in traditional education; therefore, robotics invoke curiosity in students. Since curiosity is related to creativity (Starko, 2013), including robotics in education support creativity in students (Alimisis, 2013; Botelho, Braz, Rodrigues, 2012; Khanlari, 2013). This result implies that robotics can be used across different subjects to invoke students' curiosity in a particular subject matter and to improve their creativity.

4.2.1.2 STEM-Based Robotics to Improve Algorithmic Thinking

Although not being statistically significant, the results of the current study demonstrated that there is an increase in students' algorithmic thinking after STEM-based robotics education. Therefore, it was determined that STEM-based robotics education improve students' algorithmic thinking skills. Sarıtepeci and Durak (2017) conducted a similar study with ninth graders about how block and robotic programming affect students' computational thinking skills. The study of Sarıtepeci and Durak (2017) included two experimental groups and a control group. Students in the experimental group 1 took block based and robotics programming education and students in the experimental group 2 took only block-based coding education. In the control group, students were given instruction based on the standard

schedule. The results of the study demonstrated that students' computational thinking skills who worked with block and robotic coding improved better than those who took traditional courses in the control group, which coincides with the result of the present study. The reason of this increase might be that students needed to develop algorithm before programming their robots. Therefore, their algorithmic thinking skills are improved lesson by lesson.

4.2.1.3 STEM-Based Robotics to Improve Cooperativity

The results of the current study demonstrated that there is not an increase in students' cooperativity skills after STEM-based robotics education. On the contrary, the mean of post-test scores was lower than that of pre-test scores. On the other hand, Sarıtepeci and Durak (2017) found in their research conducted with ninth graders about the effects of block and robotic coding activities on students' computational thinking skills that students who participated in robot coding activities had higher scores on cooperativity than those who took traditional course. Similarly, Denis and Hubert (2001) conducted a research with 10-years-old students about how collaborative learning occurs in educational robotics environment. Participant students were separated into two couples. Couple 1 was asked to build a LEGO® robot while couple 2 was required to modify an existing model. The researchers observed that students in group 1 worked together and involved in a strong collaboration; hence, their problem-solving and collaboration skills were enhanced (Denis & Hubert, 2001).

The reason why this study did not contribute to students' cooperativity might be that every student loved to work with the robotics kit and wanted to use it individually. Also, they sometimes could not agree on who would build the robot and who would program it. Therefore, their cooperativity skills could not be enhanced. To overcome this problem, students might have been given specific roles for each project. A well-structured task sharing might help students be clearer about their roles in each project and decrease the number of disputes.

4.2.1.4 STEM-Based Robotics to Improve Critical Thinking

Although not being statistically significant, the results of the current study revealed that students had higher scores in critical thinking after the STEM-based robotics activities. Sarıtepeci and Durak (2017) found a similar result. The researchers stated that the students in the experimental groups scored significantly higher in critical thinking sub-dimension than those in the control group.

In STEM-based robotics education, students had the opportunity to develop and test different ideas to solve problematic situations without depending on a single solution. This situation could lead to the improvement of students' critical thinking abilities. Beer, Chiel and Drushel (1999) proposed a similar claim that the learning environments in which students test different ideas using robotics to solve various problems contribute to the development of critical thinking. Based on the results of the present study, it might be concluded that the process of producing and testing different ideas improves students' critical thinking skills.

4.2.1.5 STEM-Based Robotics to Improve Problem Solving

The results of the current study revealed that students had higher scores in problem solving after the STEM-based robotics activities although difference was not statistically significant. Similarly, Saritepeci and Durak (2017) revealed that students who had robotics-based programming experience have a higher level of problem solving skills than those who took merely block-based programming lessons. The reason of this consequence might be that students are not obliged to make a specific robot design. They could try different models and get feedback immediately regardless of having a completely correct solution; therefore, their problem solving skills were improved. On the other hand, studies on improving students' problem solving skills have been implemented in a longer period of time (Argelagos & Pifarre, 2012; Hwang, Hung & Chen, 2013; O'Hearn & Gatz, 2002; Woods et al., 1997), which may imply that it could be difficult to improve problem solving skills in such a short time.

4.2.2 STEM-Based Robotics to Improve Students Attitude towards Computer Science

Results of the current study demonstrated that STEM-based real-world problems provided a meaningful context to students; therefore, made a statistically significant difference in favor of the post-test scores on students' attitude towards computer science. In this study, students had the possibility of applying their programming knowledge for real-world situations thanks to the STEM-based real world problems. By this means, they experienced computer science in a meaningful context and gained a real view of the discipline. Students understood that computer science is more than being an abstract concept, which might be the reason of the improvement of students' attitude towards computer science. Similarly, Burbaite, Drasute and Stuikys (2018) came up with an approach that integrates STEM into computer science education to improve students' computational thinking skills. They prepared a STEM-driven programming module that requires students to use robot control programs to accomplish real world tasks. However, Burbaite, Drasute and Stuikys did not investigate how the programming module affected students' computational thinking skills. The present study might represent a real world application of the approach proposed by Burbaite, Drasute and Stuikys (2018).

Students often want to see the connection between the subjects taught in schools and the real life. If they decide there is nothing useful to them, they lose their motivation to learn. The same situation goes for computer science education as well. Although computer science education has an important role in improving students' computational thinking skills, students may not appreciate this intention if the lessons have no real life connection. In that case, computer science teachers often come across with the question "What am I going to do with this in my life?". If students are not given a satisfying answer, they see computer science and programming-related works as boring. As a result, they develop a negative attitude and teachers have difficulty with engaging those students. The results of our study offer a way to make computer science education more attractive to students. The idea of using STEM-based robotics provides a meaningful context and gave students the possibility to make different designs and share with their classmates

immediately. Students become more engaged and discovered how computer science is useful in their lives.

4.2.2.1 STEM-Based Robotics to Improve Students' Confidence and Motivation in Learning Programming

Although not being statistically significant, the results of the present study demonstrated that STEM-based robotics education led to the increase in students' confidence and motivation to learn programming. Similarly, Lykke et al. (2014) reported in their study that students found working with robots to be very motivating and interesting. Besides, Johnson (2003) argues that robots are one of the most motivating tools thanks to their concrete and human needs-related nature. Lykke et al. (2014) claim that an important factor that makes robots motivating for students is the possibility to build a better robot. Students have the opportunity of improving their robots by adding new features or changing the existing ones. In addition, when working with robots, students feel the determination of finishing the project and work towards that end. Consequently, these might be the aspects that motivate students and make them gain confidence in learning programming.

4.2.2.2 STEM-Based Robotics to Improve Students' Views on Usefulness of Programming

The results of the present study demonstrated that STEM-based robotics education improved students' views on usefulness of programming. However, the difference was not statistically significant. The reason of this increase might be that STEM-based robotics provided students an authentic context in which robotics is used as a tool to solve real-world problems. By that means, students saw how robotics and programming are useful in real life. Therefore, their views were improved in positive way. This results are also related to the ones discussed in the section of STEM-based robotics to improve students' confidence and motivation in learning programming section. Since students have seen real-life applications of robotics and programming throughout the lessons, they appreciated how programming is useful in our lives and got motivated in completing the projects.

4.2.2.3 STEM-Based Robotics Education to Improve Students' Attitude toward Success in Programming

This sub-dimension includes items as to how students feel when they become successful in programming. According to the results of the present study, STEM-based robotics education led to a statistically significant increase in students' attitude toward success in programming.

During the stem-based robotics education, students were challenged with different robotics projects. They tried different models until they reach a solution to the stem-based real-world problem. During this whole process, they compared their projects and started to care being more successful in programming. For example, students entered into competition with each other about whose building was more resistant to higher earthquake intensity in earthquake-resistant structures project. This is also relevant to the motivation students gained while working with robotics. As discussed in the section of STEM-based robotics to improve students' confidence and motivation in learning programming, students become more motivated to finish their projects successfully (Lykke et al., 2014).

4.2.2.4 STEM-Based Robotics Education to Improve Social Perception of Success in Programming

This factor included items related to students' believes about how other people think when they become successful in programming. Although not being statistically significant, the results of the present study demonstrated that STEM-based robotics education improved students' social perception of success in programming. This might be because students started to care being successful in programming more as discussed in the section of STEM-based robotics education to improve students' attitude toward success in programming section. They shared their projects with their classmates, which might be the reason why students' social perception of success in programming improved. Presenting projects to the whole class provided feedback to students about how others think about their projects. Sometimes students get into competition about whose robot was faster or worked better. Those were the signs indicating that students started to care to be seen successful by others in programming.

4.2.3 Implications and Recommendations

In this study, the aim was to develop students' computational thinking skills and their attitude towards computer science through STEM-based robotics education. To that end, a STEM-based robotics unit was developed according to the STEM-integrated instruction features (Bryan, Moore, Johnson & Roehrig; 2016). Each lesson in the unit were shaped by a real-world problem and students were asked to solve them by using developing robotic models with their groupmates. By this means, students got concrete ideas how computer science is useful in real life while improving their computational thinking skills and learning robotics and programming. According to the results of the study, the STEM-based robotics education resulted in a significant difference at students' computational thinking skills and their attitude to computer science in a significant way.

This study has important implications for computer science, science and mathematics teachers. Today, schools try to teach computational thinking to students through computer science lessons. The lessons that have no relationship with real life and solely focus on programming may cause students to develop a negative attitude towards robotics and programming, in general, to computer science. Therefore, the development of their computational thinking skills might be damaged. On the other hand, STEM-integrated education offers educators the possibilities of establishing a meaningful context related to real life and emphasizing 21st century skills. Besides, robotic kits are considered as an engaging tool that motivates students for learning. As a result, teachers may benefit the STEM-based robotics education approach proposed by the present study to help students gain a positive attitude towards computer science develop computational thinking skills. In addition, science and mathematics teachers may use robotics in their lessons to make abstract concepts such as speed, acceleration, friction and force more concrete. For example, students can design a robot that pulls objects in its basket. Students can make their robots move on different surfaces like wood and carpet to observe the force of friction applying to the system in the reverse direction of the movement. Similarly, they can try their robots with different sizes of gears to understand how the speed of the robot is affected. Also, students can learn how

speed is calculated based on the distance and time necessary to move a robot from a point to another.

Although this study provides an approach for the purpose of improving students' computational thinking skills and their attitude towards computer science, further research needs to be done. Observation and interview data need to be collected to better analyze the effects of STEM-based robotics education on students' computational thinking skills and their attitude towards computer science. Besides, working with a larger sample at each level in middle school will increase the generality of the results.

4.2.4 Conclusions

Computing has been changing the way things have been done. It offers new ways to develop innovative solutions to the problems across all disciplines. Todays' students will be living in a life heavily affected by computing. It is not enough to wait students learn computational concepts until they go to a higher education institute. They must learn to work with computational tools and problem solving methods at an early age. Therefore, helping students gain computational thinking skills during K-12 education has become very important.

The most common way of including computational thinking to K-12 curriculum is through computer science lessons. Many countries include computer science education at K-12 level. Since context is very important for learning and attitude, giving computer science education in a meaningful context related to real life makes students more motivated to learn; hence, their computational thinking skills are also improved. In this way, the present study introduces a way to establish an engaging and meaningful context in computer science lessons by integrating STEM-integrated instruction features to computer science lessons through robotics. The results indicate that STEM-integrated robotics education may improve students computational thinking skills and their attitudes towards computer science. Students learn to think computationally throughout lessons, and they develop a positive attitude towards computer science. Hopefully, this study will provide a basis for further research that proposes a framework to integrate STEM and computer science at different grades and topics.

REFERENCES

Aho, A. V. (2011). Ubiquity symposium: Computation and computational thinking. *Ubiquity*, *2011*(January).

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

Alimisis, D. (2013). Educational robotics: Open questions and new challenges. *Themes in Science and Technology Education*, *6*(1), 63-71.

Allen, M. (2017). *The sage encyclopedia of communication research methods* (Vols. 1-4). Thousand Oaks, CA: SAGE Publications, Inc doi: 10.4135/9781483381411

Andic, B., Grujicic, R., & Markuš, M. M. (2015). Robotics and Its Effects on the Educational System of Montenegro. *World Journal of Education*, *5*(4), 52-57.

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Journal of Educational Technology & Society*, *19*(3), 47.

Anwar, S., Bascou, N. A., Menekse, M., & Kardgar, A. (2019). A systematic review of studies on educational robotics. *Journal of Pre-College Engineering Education Research (J-PEER)*, *9*(2), 2.

Argelagós, E., & Pifarré, M. (2012). Improving information problem solving skills in secondary education through embedded instruction. *Computers in Human Behavior*, *28*(2), 515-526.

Bain, S. K., McCallum, R. S., Bell, S. M., Cochran, J. L., & Sawyer, S. C. (2010). Foreign language learning aptitudes, attitudes, attributions, and achievement of postsecondary students identified as gifted. *Journal of Advanced Academics*, *22*(1), 130-156.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?. *Acm Inroads*, *2*(1), 48-54.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?. *Acm Inroads*, *2*(1), 48-54.

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology*, *38*(6), 20-23.

Basogain, X., Olabe, M. Á., & Olabe, J. C. (2017, November). Transition to a modern education system through e-learning. In *Proceedings of the 2017 international conference on education and e-learning* (pp. 41-46).

Basogain, X., Olabe, M. A., Olabe, J. C., Maiz, I., & Castaño, C. (2012). Mathematics education through programming languages. In *21st annual world congress on learning disabilities* (pp. 553-559).

Başer, M. (2013a). Attitude, gender and achievement in computer programming. *Online Submission*, *14*(2), 248-255.

Başer, M. (2013b). Bilgisayar programlamaya karşı tutum ölçeği geliştirme çalışması. *The Journal of Academic Social Science Studies*, *6*(6), 199-215.

Beer, R. D., Chiel, H. J., & Drushel, R. F. (1999). Using autonomous robotics to teach science and engineering. *Communications of the ACM*, *42*(6), 85-92.

Bers, M. U. (2008). *Blocks to Robots Learning with Technology in the Early Childhood Classroom*. Teachers College Press.

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M., & Rumble, M. (2012). Defining twenty-first century skills. In *Assessment and teaching of 21st century skills* (pp. 17-66). Springer, Dordrecht.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education-Implications for policy and practice* (No. JRC104188). Joint Research Centre (Seville site).

Botelho, S. S., Braz, L. G., & Rodrigues, R. N. (2012). Exploring creativity and sociability with an accessible educational robotic kit. In *Proc. 3rd Int. Conf. on Robotics in Education (RiE 2012), Prague, Czech Republic* (pp. 55-60).

Bryan, L.A., Moore, T.J., Johnson, C.C., & Roehrig, G.H. (2016). Integrated STEM Education. In C.C. Johnson, E.E. Peters-Burton & T.J. Moore (Eds.), *STEM ROAD MAP: A Framework for Integrated STEM Education* (pp. 23-37). Routledge.

Bubica, N., & Boljat, I. (2018). Assessment of computational thinking. In *Proceedings of the International Conference on Computational Thinking Education 2018* (p. 121).

Bundy, A. (2007). Computational thinking is pervasive. *Journal of Scientific and Practical Computing*, *1*(2), 67-69.

Burbaitė, R., Drąsutė, V., & Štuikys, V. (2018, April). Integration of computational thinking skills in STEM-driven computer science education. In *2018 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1824-1832). IEEE.

Bybee, R. W. (2013). *The case for STEM education: Challenges and opportunities*. NSTA press.

Cansu, S. K., & Cansu, F. K. (2019). An Overview of Computational Thinking. *International Journal of Computer Science Education in Schools*, *3*(1), n1.

Choi, I., Kopcha, T., Mativo, J., Hill, R., Hodge, E., Shin, S., ... & Bae, Y. (2016, March). Learning Computer Programming in Context: Developing STEM-integrated Robotics Lesson Module for 5th Grade. In *Society for Information Technology & Teacher Education International Conference* (pp. 68-74). Association for the Advancement of Computing in Education (AACE).

Chookaew, S., Howimanporn, S., Pratumsuwan, P., Hutamarn, S., Sootkaneung, W., & Wongwatkit, C. (2018, July). Enhancing High-School Students' Computational Thinking with Educational Robotics Learning. In *2018 7th International Congress on Advanced Applied Informatics (IIAI-AAI)* (pp. 204-208). IEEE.

Constantinou, V., & Ioannou, A. (2018). Development of Computational Thinking Skills through Educational Robotics. In *EC-TEL (Practitioner Proceedings)*.

Cooper, S., & Cunningham, S. (2010). Teaching computer science in context. *Acm Inroads*, *1*(1), 5-8.

Csizmadia, A., & Boulton, H. (2017). Computational thinking–back to the future. In *Conference Proceedings. The Future of Education* (p. 108). libreriauniversitaria. it Edizioni.

CSTA, ISTE. (2011). Computational thinking teacher resources. *National Science Foundation under Grant No. CNS-1030054*.

Cuny, J., Snyder, L., & Jeannette, M. (2010). Wing. *Demystifying Computational Thinking for Non-Computer Scientists, work in progress*.

Czerkawski, B. (2015, October). Computational thinking in virtual learning environments. In *E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education* (pp. 1227-1231). Association for the Advancement of Computing in Education (AACE).

Denis, B., & Hubert, S. (2001). Collaborative learning in an educational robotics environment. *Computers in Human Behavior*, *17*(5-6), 465-480.

DeSchryver, M. D., & Yadav, A. (2015). Creative and computational thinking in the context of new literacies: Working with teachers to scaffold complex technology-mediated approaches to teaching and learning. *Journal of Technology and Teacher Education*, *23*(3), 411-431.

Eguchi, A., & Uribe, L. (2017, March). Robotics to promote STEM learning: Educational robotics unit for 4th grade science. In *2017 IEEE Integrated STEM Education Conference (ISEC)* (pp. 186-194). IEEE.

García-Peñalvo, F. J., & Mendes, A. J. (2018). Exploring the computational thinking effects in pre-university education.

Goode, J., Estrella, R., & Margolis, J. (2006). Lost in translation: Gender and high school computer science. In J.M.

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

Günüç, S., ODABAŞI, H., & Abdullah, K. U. Z. U. (2013). 21. yüzyıl öğrenci özelliklerinin öğretmen adayları tarafından tanımlanması: bir twitter uygulaması/the defining characterıstıcs of students of the 21st century by student teachers: a twitter activity. *Eğitimde Kuram ve Uygulama*, *9*(4), 436-455.

Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education*, *126*, 296-310.

Hwang, G. J., Hung, C. M., & Chen, N. S. (2014). Improving learning achievements, motivations and problem-solving skills through a peer assessment-based game development approach. *Educational Technology Research and Development*, *62*(2), 129-145.

International Society for Technology in Education (ISTE). (2015). *Computational Thinking leadership toolkit*. https://id.iste.org/docs/ct-documents/ct-leadershipt-toolkit.pdf?sfvrsn=4.

Johnson, J. (2003). Children, robotics, and education. *Artificial Life and Robotics*, *7*(1-2), 16-21.

Jona, K., Wilensky, U., Trouille, L., Horn, M. S., Orton, K., Weintrop, D., & Beheshti, E. (2014). Embedding computational thinking in science, technology, engineering, and math (CT-STEM). In *future directions in computer science education summit meeting, Orlando, FL*.

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

Khanlari, A. (2013). Effects of robotics on 21st century skills. *European Scientific Journal*, *9*(27).

Kim, C., Kim, D., Yuan, J., Hill, R. B., Doshi, P., & Thai, C. N. (2015). Robotics to promote elementary education pre-service teachers' STEM engagement, learning, and teaching. *Computers & Education*, *91*, 14-31.

Koorsse, M., Cilliers, C., & Calitz, A. (2015). Programming assistance tools to support the learning of IT programming in South African secondary schools. *Computers & Education*, *82*, 162-178.

Kopcha, T. J., McGregor, J., Shin, S., Qian, Y., Choi, J., Hill, R., ... & Choi, I. (2017). Developing an integrative STEM curriculum for robotics education through educational design research. *Journal of Formative Design in Learning*, *1*(1), 31-44.

Korkmaz, Ö., & Altun, H. (2013). Engineering and ceit student's attitude towards learning computer programming. *The Journal of Academic Social Science Studies International Journal of Social Science*, *6*(2), 1169-1185.

Korkmaz, Ö., Çakır, R., & Özden, M. Y. (2015). Bilgisayarca düşünme beceri düzeyleri ölçeğinin (bdbd) ortaokul düzeyine uyarlanması. *Gazi Eğitim Bilimleri Dergisi*, *1*(2), 143-162.

Kuhlemeier, H., Van Den Bergh, H., & Melse, L. (1996). Attitudes and achievements in the first year of German language instruction in Dutch secondary education. *The Modern Language Journal*, *80*(4), 494-508.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., ... & Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads*, *2*(1), 32-37.

Liu, Y. (2014). Motivation and attitude: two important non-intelligence factors to Arouse students' potentialities in learning English. *Creative Education*, *2014*.

Lu, J. J., & Fletcher, G. H. (2009, March). Thinking about computational thinking. In *Proceedings of the 40th ACM technical symposium on Computer science education* (pp. 260-264).

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.

Lykke, M., Coto, M., Mora, S., Vandel, N., & Jantzen, C. (2014, April). Motivating programming students by problem based learning and LEGO robots.

In *2014 IEEE Global Engineering Education Conference (EDUCON)* (pp. 544-555). IEEE.

Maio, G. & Haddock, G. (2010). *The Psychology of Attitudes and Attitude Change*. Sage, London.

Mantle-Bromley, C. (1995). Positive attitudes and realistic beliefs: Links to proficiency. *The Modern Language Journal*, *79*(3), 372-386.

McKay, M. M., Lowes, S., Tirhali, D., & Camins, A. H. (2015). Student learning of STEM concepts using a challenge-based robotics curriculum. In *Proceedings of 2015 ASEE Annual Conference & Exposition, Seattle, Washington. https://doi. org/10.18260* (p. 24756).

Ministry of National Education. (2018b). *FEN BİLİMLERİ*. https://mufredat.meb.gov.tr/ProgramDetay.aspx?PID=325

Ministry of National Education. (2018a). *BİLİŞİM TEKNOLOJİLERİ VE YAZILIM*. http://mufredat.meb.gov.tr/ProgramDetay.aspx?PID=374

Mishra, P., & Yadav, A. (2013). Of art and algorithms: Rethinking technology & creativity in the 21st century. *TechTrends, 57*(3), 11.

Mohaghegh, D. M., & McCauley, M. (2016). Computational thinking: The skill set of the 21st century.

Moore, T. J., Stohlmann, M. S., Wang, H. H., Tank, K. M., Glancy, A. W., & Roehrig, G. H. (2014). Implementation and integration of engineering in K-12 STEM education. In *Engineering in pre-college settings: Synthesizing research, policy, and practices* (pp. 35-60). Purdue University Press.

Moore, T. J. (2008). STEM integration: Crossing disciplinary borders to promote learning and engagement. *Invited presentation to the faculty and graduate students of the UTeachEngineering, UTeachNatural Sciences, and STEM Education program area at University of Texas at Austin*, 1-13.

Moorman, P., & Johnson, E. (2003). Still a stranger here: Attitudes among secondary school students towards computer science. *ACM SIGCSE Bulletin*, *35*(3), 193-197.

National Research Council (NRC). (2014). *STEM integration in K-12 education: Status, prospects, and an agenda for research*. National Academies Press.

National Research Council (NRC). (2012). *Education for life and work: Developing transferable knowledge and skills in the 21st century*. National Academies Press.

National Research Council (NRC). (2010). *Report of a workshop on the scope and nature of computational thinking*. National Academies Press.

Neil, C., Brown, N. C. C., Sentence, S., Crick, T., & Humphreys, S. (2014). Restart: The resurgence of computer science in UK schools. ACM Transactions on Computing Education, 14, 1.

Next Generation Since Standards (NGSS). (2013). *DCI Arrangements of the Next Generation Science Standards*. https://education.lego.com/en-us/lessons?products=WeDo+2.0+Core+Set

O'Hearn, T. C., & Gatz, M. (2002). Going for the Goal: Improving youths' problem-solving skills through a school-based intervention. *Journal of Community psychology, 30*(3), 281-303.

Özyurt, Ö., & Özyurt, H. (2015). Learning style based individualized adaptive e-learning environments: Content analysis of the articles published from 2005 to 2014. *Computers in Human Behavior*, *52*, 349-358.

Pallant, J. (2020). *SPSS survival manual: A step by step guide to data analysis using IBM SPSS*. Routledge.

Papert, S. (1980). Mindstorms; Children, Computers and Powerful Ideas. *New York: Basic Book*.

Papert, S., & Harel, I. (1991). Situating Constructionism. Constructionism. *Norwood, NJ: Ablex Publishing*.

Partnership for 21st Century Learning (2009). Framework for 21st century learning. Retrieved from http://www.p21.org/our-work/p21-framework

Pea, R. D. (1987). Cognitive technologies for mathematics education. *Cognitive science and mathematics education*, 89-122.

Peters, A. K., & Pears, A. (2012, October). Students' experiences and attitudes towards learning computer science. In *2012 Frontiers in Education Conference Proceedings* (pp. 1-6). IEEE.

Phillips, R. S., & Brooks, B. P. (2017). The hour of code: Impact on attitudes towards and self-efficacy with computer science.

Phillips, P. (2009). Computational thinking: A problem-solving tool for every classroom. *Communications of the CSTA*, *3*(6), 12-16.

Reges, S. (2008). The mystery of" b:=(b= false)". *ACM SIGCSE Bulletin*, *40*(1), 21-25.

Resnick, M., Ocko, S., & Papert, S. (1988). LEGO, Logo, and design. *Children's Environments Quarterly*, 14-18.

Sanders, M. (2009). STEM, STEM education, STEMmania. *The Technology Teacher*. Retrieved from http://esdstem.pbworks.com/f/TTT+STEM+Article_1.pdf

Sarıtepeci, M., & Durak, H. (2017). Analyzing the effect of block and robotic coding activities on computational thinking in programming education. *Educational research and practice*, 490-501.

Sipayung, D. H., Sani, R. A., & Bunawan, H. (2018, December). Collaborative inquiry for 4C skills. In *3rd Annual International Seminar on Transformative Education and Educational Leadership (AISTEEL 2018)*. Atlantis Press.

Starko, A. J. (2013). *Creativity in the classroom: Schools of curious delight*. Routledge.

Teo, T. (2006). Factors affecting gender differences in attitudes towards computers among students. *Educational Research and Reviews*, *1*(1), 7-12.

The Lego Group. (n.d.). *Get ready with Lessons*. Lego Education. https://education.lego.com/en-us/lessons?products=WeDo+2.0+Core+Set

The Royal Society. (2012). *Shut Down or Restart? The Way Forward for Computing in UK Schools*.

http://royalsociety.org/uploadedFiles/Royal_Society_Content/education/policy/compu ting-in-schools/2012-01-12-Computing-in-Schools.pdf

Wiebe, E., Williams, L. A., Yang, K., & Miller, C. S. (2003). *Computer science attitude survey*. North Carolina State University. Dept. of Computer Science.

Wilson, C., Grizzle, A., Tuazon, R., Akyempong, K., & Cheung, C. K. (2013). *Media and information literacy curriculum for teachers*. UNESCO. http://www.unesco.org/new/fileadmin/MULTIMEDIA/HQ/CI/CI/pdf/media_and_information_literacy_curriculum_for_teachers_en.pdf.

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

Wing, J. (2011). Research notebook: Computational thinking—What and why. *The link magazine*, *6*.

Wing, J. M. (2014). Computational thinking benefits society. *40th Anniversary Blog of Social Issues in Computing*, *2014*, 26.

Woods, D. R., Hrymak, A. N., Marshall, R. R., Wood, P. E., Crowe, C. M., Hoffman, T. W., ... & Bouchard, C. K. (1997). Developing problem solving skills: The McMaster problem solving program. *Journal of Engineering Education*, *86*(2), 75-91.

Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T. (2011, March). Introducing computational thinking in education courses. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 465-470).

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, *14*(1), 1-16.

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: pedagogical approaches to embedding 21st century problem solving in K-12 classrooms. *TechTrends*, *60*(6), 565-568.

Yang, D., Swasnon, S. R., Chittoori, B. B., & Baek, Y. (2018). *Board 70: Work in Progress: Integrating Computational Thinking in STEM Education through a Project-based Learning Approach.* Paper presented at the 2018 ASEE Annual Conference & Exposition.

Yiğit, M. F. (2016). *Görsel programlama ortamı ile öğretimin öğencilerin bilgisayar programlamayı öğrenmesine ve programlamaya karşı tutumlarına etkisinin incelenmesi* (Master's thesis, Ondokuz Mayıs Üniversitesi, Eğitim Bilimleri Enstitüsü).

Yünkül, E., Durak, G., Çankaya, S., & Mısırlı, Z. (2017). The effects of Scratch software on students' computational thinking skills. *Necatibey Eğitim Fakültesi Fen ve Matematik Eğitimi Dergisi*, *11*(2), 502-517.

Zawieska, K., & Duffy, B. R. (2015). The social construction of creativity in educational robotics. In *Progress in Automation, Robotics and Measuring Techniques* (pp. 329-338). Springer, Cham.

Zhong, B., Wang, Q., Chen, J., & Li, Y. (2016). An exploration of three-dimensional integrated assessment for computational thinking. *Journal of Educational Computing Research*, *53*(4), 562-590.

A
STEM-BASED ROBOTICS UNIT LESSON PLANS

Lesson 1: Building a Mars rover

Time: 60 minutes

STEM-Based Real World Problem: How do scientists make discoveries in the places they cannot go to?

Lesson Objectives:

- Students will learn how LEGO® WeDo works.
- Students will learn the parts of the LEGO® WeDo robotics set.
- Students will learn how to code a robot.
- Students will learn different ways of exploring places in remote places.

Lesson Assessment: Students' products and the codes they wrote

Instruction Sequence:

Students are given a brief information about what they would do in the next 6 lessons and introduced the first STEM-based real world problem that will be addressed for the next three lessons in the first week. After making necessary explanations, students are asked how scientists can make investigations in places they cannot go to. Upon some discussion, students are told to build a rover prototype with the robotics kit. They are described how to use LEGO® WeDo software to write code and upload it to the robot they have built. Students are given some time to build their robot and upload the following codes to their robots.

Students are asked to discover what each code block is used for. They are given some time to make experiments to understand each code block. Later, the whole class is explained the codes used to move the rover by emphasizing the importance of algorithms.

Lesson 2: Building a Mars rover that runs autonomously

Time: 60 minutes

STEM-Based Real World Problem: How do scientists make discoveries in the places they cannot go to?

Lesson Objectives:

- Students will understand how autonomous systems work.
- Students will build an autonomous robot by using motion sensor.
- Students will learn conditional structure in programming.

Lesson Assessment: Students' products and the codes they wrote

Instruction Sequence:

Students are asked how autonomous vehicles sense the environment and respond accordingly. By that means, they are acquainted with the concept of sensor which is the electronic devices that gathers information from the environment such as the existence of an object or the temperature of air. Students are asked how the rovers can move by themselves without hitting an object. They are shown motion sensor that can detect physical objects nearby. Students add the motion sensor to the rover they had built the previous week and make it an autonomous rover. Then, they are shown the codes below and infer which code block is responsible of detecting objects with motion sensor. After some discussions and trials, students are explained the whole code.

Lesson 3: Building a joystick to control the Mars rover

Time: 60 minutes

STEM-Based Real World Problem: How do scientists make discoveries in the places they cannot go to?

Lesson Objectives:

- Students will understand what tilt sensor is used for.
- Students will build a robot that responds to the data coming from tilt sensor.
- Students will learn the loops in programming.

Lesson Assessment: Students' products and the codes they wrote

Instruction Sequence:

Students are asked how to control the rover remotely. Then, they are told to build a joystick using tilt sensor and program it in a way that navigates the rover based on the direction coming from the joystick. While writing codes, students are explained the loop structure in programming and how to use it in the current project.

Lesson 4: Building an earthquake simulator

Time: 60 minutes

STEM-Based Real World Problem: What is the good way of building earthquake-resistant structures?

Lesson Objectives:

- Students will learn how earthquake happens.
- Students will explore the features that make a structure earthquake-resistant.
- Students will build a device to test structure designs.

Lesson Assessment: Students' products and the codes they wrote

Instruction Sequence:

Students were asked what is the good way of building an earthquake-resistant structure. They were shown earthquake-resistant structures videos and discussed the features of those structures. Then, students were told to build an earthquake simulator by using the various LEGO parts they have used in the previous lessons. Later, they build different structures and tested them for the maximum earthquake magnitude they can resist. The following codes could be used in this project.

Lesson 5 & 6: Cleaning the oceans

Time: 120 minutes

STEM-Based Real World Problem: How can we keep oceans clean of plastic debris?

Lesson Objectives:

- Students will learn the importance of keeping oceans clean.
- Students will develop a sense of clean environment.

Lesson Assessment: Students' final projects

Instruction Sequence:

In this lessons, students are given open-ended tasks. They are presented the problem of plastic pollution in the oceans. Students watch some videos about the problem and are asked to build and code a device using robotics to keep the oceans clean of plastic debris. Unlike previous lessons, students are not provided any building instructions. In the second lesson, students present their models to the class and get feedback from their classmates.

PUBLICATIONS FROM THE THESIS

Contact Information: haticevarlik343@gmail.com

Conference Papers

Varlık, H., & Topçu, M. S. (2020). Improving Middle School Students' Computational Thinking Skills and Attitudes towards Computer Science through STEM-Based Robotics Education. *VIIth International Eurasian Educational Research Congress: EJERCongress 2020 ABSTRACTS* (pp. 1695-1697). Anı Yayıncılık. https://drive.google.com/file/d/113VBIiiGd7T6AsplakovGBzB1oggYJ-3/view