

DEVELOPING KNOWLEDGE OF COMPUTATIONAL THINKING CONCEPTS
IN STUDENTS WITH LEARNING DISABILITIES: A CASE STUDY

MUKADDES ASLI PEHLİVAN ŞAHİNOĞLU

BOĞAZİÇİ UNIVERSITY

2024

DEVELOPING KNOWLEDGE OF COMPUTATIONAL THINKING CONCEPTS
IN STUDENTS WITH LEARNING DISABILITIES: A CASE STUDY

Thesis submitted to
the Institute for Graduate Studies in Social Sciences
in partial fulfillment of the requirements for the degree of

Master of Arts
in
Educational Technology

by
Mukaddes Aslı Pehlivan Şahinoğlu

Boğaziçi University

2024

Developing Knowledge of Computational Thinking Concepts
in Students with Learning Disabilities: A Case Study

The thesis of Mukaddes Aslı Pehlivan Şahinoğlu

has been approved by:

Prof. Diler Öner
(Thesis Advisor)

Assoc. Prof. Günizi Kartal

Assist. Prof. Anastasia Misirli
(External Member)

May 2024

DECLARATION OF ORIGINALITY

I, Mukaddes Aslı Pehlivan Şahinoğlu, certify that

- I am the sole author of this thesis and that I have fully acknowledged and documented in my thesis all sources of ideas and words, including digital resources, which have been produced or published by another person or institution;
- this thesis contains no material that has been submitted or accepted for a degree or diploma in any other educational institution;
- this is a true copy of the thesis approved by my advisor and thesis committee at Boğaziçi University, including final revisions required by them.

Signature.....

Date.....

ABSTRACT

Developing Knowledge of Computational Thinking Concepts in Students with Learning Disabilities: A Case Study

This instrumental case study aimed to examine the effectiveness of a specialized computational thinking curriculum for students with diverse learning needs. The curriculum was developed using the TIPP&SEE metacognitive strategy, which stands for Title, Instructions, Purpose, Play, and Sprite, Events, Explore. This strategy has been successfully implemented for teaching computer science to elementary students using Scratch, a free online block-based programming tool. During a six-lesson curriculum implementation, the development of computational thinking concepts such as events, sequences, parallelism, and loops was closely examined in two middle school students with specific learning disabilities. Data were collected through semi-structured interviews, tests on computational thinking concepts, session journals, and TIPP&SEE worksheets. Findings indicated that participants developed their knowledge of computational thinking concepts at different rates, likely due to varying levels of engagement and individual disabilities. The analysis also revealed that participants struggled with checkbox table questions involving more than two sprites and options. Additionally, students developed misconceptions about a sprite's costumes, and presenting the costumes in multiple ways helped address this issue. Qualitative data suggested that using the OpenDyslexic font improved participants' reading fluency. This study provided recommendations for designing and implementing computational thinking curricula for students with diverse needs, which is essential to promote equity in education.

ÖZET

Özgöl Öğrenme Güçlüğü Olan Öğrencilerde Bilgi İşlemsel Düşünme

Kavramları Bilgisinin Geliştirilmesi: Bir Vaka Çalışması

Bu enstrümantal vaka çalışması, çeşitli öğrenme ihtiyaçları olan öğrenciler için özel olarak tasarlanmış bir bilgi işlemsel düşünme müfredatının etkinliğini değerlendirmeyi amaçlamıştır. Müfredat, Başlık, Talimatlar, Amaç, Oyun ve Sprite, Olaylar, Keşfet anlamına gelen TIPP&SEE üstbilişsel stratejisi kullanılarak geliştirilmiştir. Bu strateji, Scratch adlı ücretsiz çevrimiçi blok tabanlı programlama aracı kullanılarak başarıyla uygulanmıştır. Altı derslik müfredat süresince, olaylar, diziler, paralellik ve döngüler gibi bilgi işlemsel düşünme kavramlarının gelişimi, öğrenme güçlüğü olan iki ortaokul öğrencisinde yakından incelenmiştir. Veriler, yarı yapılandırılmış görüşmeler, bilgi işlemsel düşünme ile ilgili testler ve TIPP&SEE çalışma sayfaları ile toplanmıştır. Bulgular, katılımcıların bu kavramları farklı hızlarda geliştirdiğini, bunun katılım düzeyleri ve bireysel engellerden kaynaklanmış olabileceğini göstermiştir. Analiz, katılımcıların, iki veya daha fazla karakter ve seçenek içeren onay kutusu sorularında zorlandıklarını ortaya koymuştur. Bulgular, öğrencilerin karakterin kostümleri hakkında yanlış kavramsallaştırmalar geliştirdiklerini ve kostümleri çeşitli şekillerde sunmanın bu sorunu çözdüğü görülmüştür. Nitel veriler OpenDyslexic yazı tipinin okuma akıcılığını artırmaya yardımcı olduğunu göstermektedir. Bu çalışma, çeşitli öğrenme ihtiyaçları olan öğrenciler için bilgi işlemsel düşünme müfredatları tasarlama ve uygulama önerileri sunmuştur, bu da eğitimde eşitliği teşvik etmek için önemlidir.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to several individuals who have supported me throughout this thesis.

First and foremost, I extend my deepest gratitude to my thesis advisor, Prof. Diler Öner, for her unwavering guidance and mentorship throughout this journey. Without her patience, wisdom, and endless support, this work would not have been possible.

I am also grateful to my family - Ayvaz, Hatice, Pelin Su Pehlivan - my lovely husband, Tuğberk Şahinoğlu, and our cherished cat, Puding, for their encouragement and unconditional love during this adventure. Your support and motivation have been the cornerstone of my perseverance throughout this study. All of you were there even when I lost my motivation and confidence.

I am thankful to all the esteemed academics who have shaped my academic journey. Your knowledge and assistance were essential to the success of this research. Additionally, I would like to thank my undergraduate study group friends - Yusuf Mert Bal, Mert Kaymaz, and Beyza Şişmanoğlu - from whom I have learned a lot.

To my sisters and brothers from another family, my dearest friends - Selin Altındağ, Esin Kılıç, Damla Nisa Çevik, Ege Su Erden, Ayşin Durgut, and Utku Elagöz - I appreciate your presence and support throughout the entire process.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION	1
CHAPTER 2: LITERATURE REVIEW	6
2.1 Definition of computational thinking.....	6
2.2 General pedagogical approaches for special education students	13
2.3. Pedagogies for computational thinking.....	17
2.4 Summary	24
CHAPTER 3: METHOD.....	26
3.1 Research design.....	26
3.2 The case selection	28
3.3 Data sources	36
3.4 Data collection	40
3.5 Trustworthiness.....	43
3.6 Researcher's position	44
3.7 Data analysis	45
CHAPTER 4: FINDINGS	51
4.1 RQ1a) How do eleven-year-old children with diagnosed specific learning disabilities develop the knowledge of computational thinking concepts of events, sequence, loops, and parallelism?	51
4.2 RQ1b) What are the differences and similarities between the two cases (Fatih and Kaan) in terms of developing the knowledge of computational thinking concepts of events, sequence, loops, and parallelism?.....	99
4.3 RQ2a) How do 11-year-olds with specific learning disabilities respond to TIPP&SEE worksheets' Observe, Predict, Explore, and Debugging Questions?	105

4.4 RQ2b) What are the differences and similarities between the two cases (Fatih and Kaan) in terms of their performance in the Observe, Predict, Explore, and Debugging Questions of the TIPP&SEE worksheets?	123
4.5 Summary of the findings.....	125
CHAPTER 5: DISCUSSION.....	128
5.1 Development of the knowledge of the computational thinking concept of events.....	128
5.2 Development of the knowledge of the computational thinking concept of sequences.....	129
5.3 Development of the knowledge of the computational thinking concept of parallelism	131
5.4 Development of the knowledge of the computational thinking concept of loops	131
5.5 Development of the Performance in Observe, Predict, Explore Questions and Their Debugging Practice	132
5.6 The specialized curriculum effectiveness	134
5.7 Practical implications and suggestions for further research.....	142
APPENDIX A: 16 ELEMENTS OF EXPLICIT INSTRUCTION.....	148
APPENDIX B: THE SPECIALIZED COMPUTATIONAL THINKING CURRICULUM.....	151
APPENDIX C: SESSION JOURNAL QUESTIONS FOR THE RESEARCHER ..	153
APPENDIX D: SESSION JOURNAL QUESTIONS FOR THE RESEARCHER (TURKISH)	154
APPENDIX E: INTRODUCTION INTERVIEW QUESTIONS.....	155
APPENDIX F: INTRODUCTION INTERVIEW QUESTIONS (TURKISH)	156

APPENDIX G: THE EVENTS AND SEQUENCES TEST.....	157
APPENDIX H: THE PARALLELISM AND LOOPS TEST	160
APPENDIX I: THE TIPP&SEE WORKSHEET EXAMPLE	164
APPENDIX J: ETHICS COMMITTEE APPROVAL	170
APPENDIX K: CONSENT FORM	171
REFERENCES.....	175



LIST OF TABLES

Table 1. The Universal Design for Learning Guidelines	14
Table 2. Demographic Information of the Cases	30
Table 3. TIPP&SEE Worksheet Question Types and Example Questions from the Study	32
Table 4. The Flow of Data Collection.....	41
Table 5. Timetable of Data Collection.....	42
Table 6. The Rubric for the Events and Sequences Test.....	47
Table 7. The Rubric for the Parallelism and Loops Test	48
Table 8. Total Scores of the Students on the Computational Thinking Concepts of the Events, Sequences, Parallelism, and Loops and the Maximum Scores of Each ..	99
Table 9. The Total Scores of the Students on the TIPP&SEE Worksheets and the Maximum Scores of Each	100
Table 10. Student's Knowledge Development in Computational Thinking Concept of Events through the Program	101
Table 11. Student's Knowledge Development in Computational Thinking Concept of Sequences through the Program	102
Table 12. Student's Knowledge Development in Computational Thinking Concept of Parallelism through the Program	103
Table 13. Student's Knowledge Development in Computational Thinking Concept of Loops through the Program	104
Table 14. The Total Scores of the Students on the Observe Questions in Each TIPP&SEE Worksheet and the Maximum Scores of Each.....	107

Table 15. Total Scores of the Students on the Predict Questions in Each TIPP&SEE Worksheet and the Maximum Scores of Each	113
Table 16. Total Scores of the Students on the Explore Questions in Each TIPP&SEE Worksheet and the Maximum Scores of Each	115
Table 17. Fatih’s Answers to the Debugging Questions.....	118
Table 18. The Total Points of the Students on The Scoreable Questions from the Worksheets and the Maximum Scores of Each.....	123



LIST OF FIGURES

Figure 1. The screen of the fourth activity	58
Figure 2. The debugging question from the sixth TIPP&SEE worksheet	60
Figure 3. The third question from the Events and Sequence Test	61
Figure 4. The 4a and 4b questions from the Events and Sequence Test.....	62
Figure 5. The last question of the second TIPP&SEE worksheet.....	64
Figure 6. The question from the third TIPP&SEE worksheet	66
Figure 7. The sixth question from the Events and Sequences Test.....	68
Figure 8. The seventh question from the Events and Sequences Test	69
Figure 9. The seventh question from the Parallelism and Loops Test	70
Figure 10. The screen of the fourth activity	71
Figure 11. The screen of the fifth activity	72
Figure 12. The first question of the fifth TIPP&SEE worksheet	73
Figure 13. The sixth question of the Parallelism and Loops Test.....	74
Figure 14. Example code blocks whose values' can be changed.....	76
Figure 15. The predict question from the fourth TIPP&SEE worksheet	77
Figure 16. The example of an expanded version of a loop code command.....	78
Figure 17. The second question of the Parallelism and Loops Test.....	79
Figure 18. The fourth question of the Parallelism and Loops Test.....	80
Figure 19. The first question of the Parallelism and Loops Test	80
Figure 20. The seventh question of the Parallelism and Loops Test	81
Figure 21. The introduction of costumes of a sprite from the fifth TIPP&SEE worksheet	86
Figure 22. The first question of the second TIPP&SEE worksheet.....	87

Figure 23. The first question of the third TIPP&SEE worksheet	87
Figure 24. The question from the third TIPP&SEE worksheet	90
Figure 25. The sixth question from the Events and Sequences Test.....	91
Figure 26. The seventh question from the Events and Sequences Test	91
Figure 27. The fifth question of the Events and Sequences Test	92
Figure 28. The seventh question from the Parallelism and Loops Test	93
Figure 29. The sixth question of the Parallelism and Loops Test.....	94
Figure 30. The seventh question from the Parallelism and Loops Test	97
Figure 31. An observe question example from the third TIPP&SEE worksheet...	106
Figure 32. An observe question example from the fourth TIPP&SEE worksheet..	106
Figure 33. The second question of the sixth TIPP&SEE worksheet	108
Figure 34. The third Observe question from the fourth TIPP&SEE worksheet	110
Figure 35. The fourth Observe question from the fourth TIPP&SEE worksheet ...	110
Figure 36. The first Observe question from the sixth TIPP&SEE worksheet	111
Figure 37. The statement before the Predict Questions	112
Figure 38. A Predict Question example from the fifth TIPP&SEE worksheet.....	112
Figure 39. An example Explore Question from the fourth TIPP&SEE worksheet	114
Figure 40. An example Explore Question from the fifth TIPP&SEE worksheet ...	114
Figure 41. An example statement Before the Explore Question from the first TIPP&SEE worksheet.....	115
Figure 42. The Debugging Question from the sixth TIPP&SEE worksheet.....	120
Figure 43. The font selections and their appearances in Scratch	137

CHAPTER 1

INTRODUCTION

In recent years, integrating computational thinking (CT) into K12 computer science (CS) curricula has become prevalent in many countries, such as Ireland, Serbia, and Singapore. There are many reasons for integrating computational thinking into curricula; the most outstanding one is fostering problem-solving and logical-thinking skills (Bocconi et al., 2022). However, equity among children with different needs cannot be achieved by only integrating computational thinking into the computer science (CS) curriculum. The curriculum should meet each type of student's needs, especially those who have special educational needs, considering that computational thinking is an essential skill that is needed by everyone (Wing, 2006). Thus, students who need special education should be provided with a specialized computational thinking curriculum (Salac et al., 2020).

Diversity in students might stem from their socioeconomic status, diagnosed learning disabilities (dyslexia, dyscalculia, or dysgraphia), bilingualism, diagnosed conditions (such as autism), disabilities (such as hearing impairment), reading incompetency, and ethnicity (Hansen et al., 2016; Hall et al., 2015; Salac et al., 2021a). The needs of diverse learners may vary; these needs could be satisfied by rearranging the way of instruction. Various elements and pedagogies arose to meet the varied needs, such as Universal Design for Learning (UDL) and Explicit Instruction (EI). UDL aims to reach out to all students by providing them with multiple ways of representation, engagement, and expression. EI helps to meet the

need to follow complex multistep tasks by providing scaffolding. These pedagogies can be implemented into many subjects, and computer science is one of them.

There are several reasons for adopting these pedagogies to the computer science curriculum, such as reaching out to more students, making education accessible, and making students active in their learning process. According to Ray and colleagues (2018), typical computer science instruction needs some alterations to make all students participate and be active in class; thus, some strategies are integrated into computer science classes. The adjustments in computer science instruction could be achieved by integrating a particular strategy into lessons, such as using the Universal Design for Learning (UDL) framework or Explicit Instruction (Ray et al., 2018) or adopting the combination of more than one strategy, such as the Scratch Act1 and Scratch Act2 curricula, which aim to improve students' computational thinking skills (ScratchEd, n.d.).

Scratch Act1 and Scratch Act2 curricula are modified from the Creative Computing Curriculum, designed by the Creative Computing Lab at the Harvard Graduate School of Education. The Creative Computing Curriculum consists of activity ideas for Scratch to make students familiar with and fluent in computational thinking and has seven units (The Creative Computing Curriculum, n.d.). The modified versions, Scratch Act1 and Scratch Act2, adopt the Use-Modify-Create, 5E Instructional Model, the metacognitive learning strategy called TIPP&SEE, and Universal Design for Learning strategies together (Intro to ECforall Curriculum, 2022).

While there are several proposed computational thinking curricula in the literature, to the best of our knowledge, there is no such curriculum in Turkish for disadvantaged students. Although many schools have computer science lessons,

students who need special education still need to receive the course according to their needs. The Turkish Directorate of Special Education and Guidance Services provides free books on their website for different levels: preschool, first stage, and second stage, and other courses, such as functional reading and writing, Turkish, sports and physical activities, social skills, mathematics, music and games, communication skills, and daily life skills (Özel Eğitim Öğrencileri için Ders Kitapları, n.d.). An appropriate computational thinking curriculum should be added to these books to foster the computational thinking skills of disadvantaged students.

The other deficiency is the lack of a disability-specific special computational thinking curriculum, such as a computational thinking curriculum for autism or dyslexia, in Turkey. The Creative Computing Lab (n.d.) has prepared special curricula, Scratch Act1 and Scratch Act2, and Salac et al. (2020; 2021a; 2021b). Franklin et al. (2020) applied the Scratch Act1 to their studies of students who need special education due to their disabilities. The target audience of the curricula also includes students with limited English proficiency, below grade level in reading, below grade level in mathematics, and economically disadvantaged students. In addition, the literature has limited studies investigating the support or guidance of curricular approaches to struggling learners, including students with disabilities or economic disadvantages (Salac et al., 2021a).

In light of this information, this study aims to propose and examine the effectiveness of a specialized computational thinking curriculum designed for Turkish middle school students with dyslexia. The study is designed as a case study to enable the researcher to work more closely with students with special needs. The following research questions guide the study:

General Question: How do eleven-year-old children with diagnosed specific learning disabilities develop knowledge of computational thinking concepts as they follow a specialized computational thinking curriculum?

1a) How do eleven-year-old children with diagnosed specific learning disabilities develop the knowledge of computational thinking concepts of events, sequence, loops, and parallelism?

1b) What are the differences and similarities between the two cases (Fatih and Kaan) in terms of developing the knowledge of computational thinking concepts of events, sequences, loops, and parallelism?

2a) How do 11-year-olds with specific learning disabilities respond to TIPP&SEE worksheets' Observe, Predict, Explore, and Debugging Questions?

2b) What are the differences and similarities between the two cases (Fatih and Kaan) in terms of their performance in the Observe, Predict, Explore, and Debugging Questions of the TIPP&SEE worksheets?

The research should make participants reflect on their experiences to delve deeply into examining how diverse learners' knowledge of computational thinking concepts of events, sequences, loops, and parallelism develop by using a specialized computational thinking curriculum. Thus, a qualitative approach is adopted for the study. Since this research is bounded by the specialized computational thinking curriculum with six lessons, the case study is selected among the qualitative research approaches. Because case studies are the type of design that is an in-depth description and analysis of a bounded system (Merriam & Tisdell, 2015; Stake, 1995), they can be used to reflect human experience, refine theory and establish limits on generalizability, as well as outline nuanced areas for more research (Stake, 2000); this design is suitable for the study.

Many people can use this research's findings: special education teachers, ICT teachers with diverse learners in their schools or classes, curriculum planners, people who design special education programs, and policymakers. Special education teachers could use this study's results and adopt the findings into their lessons to help students develop computational skills. The findings may inspire ICT teachers, and if they have or will have diverse students, they could adopt the strategies of this study. The curriculum planners could consider the results of this research and create curricula that could address the needs of all learners. Special education program designers could work with ICT teachers and design CS classes for all levels to sustain equity in CS instruction by using the results of this research. Since policymakers have the power to decide what curricula should cover, they could add specialized CS curricula for students to this study's results impact.

CHAPTER 2

LITERATURE REVIEW

This section provides a summary of the related literature. Firstly, computational thinking and its assessment are defined, then the general pedagogical approaches for special education students are introduced, and then the pedagogies for computational thinking are presented.

2.1 Definition of computational thinking

Before the term Computational Thinking arose, Papert tried to develop the computational thinking skills of young learners by using the LOGO Programming Language (Papert, 1980). Within the prevalence of integration of technology into education in the early 2000s, Jeannette Wing (2006) proposed the term Computational Thinking and defined it as “Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science” (p. 33). Then, Wing (2008) advocated that computational thinking should be expanded to science, technology, and society; thus, Wing revised her definition to be compatible with analytical, mathematical, scientific, and engineering thinking. Wing (2011) supported the idea that computational thinking is a kind of mental activity that has a commonality with logical and systems thinking. She revised the definition as “the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent” (Wing, 2011, p. 20).

CSTA and ISTE (2011) define the term computational thinking as “an approach to solving a problem that empowers the integration of digital technologies with human ideas” (p. 8). Mannila et al. (2014) define computational thinking as an encompassing term that includes a combination of CS concepts and thought processes that facilitate formulating problems and their solutions in various fields.

While there has yet to be a consensus on the definition of computational thinking to date, the existing definitions use common concepts. According to Kalelioğlu et al. (2016), CT definitions primarily include “problem solving” at 22% and “abstraction” at 13%, and “tools” and “analyzing” is the least used at 4%; and computational thinking scope is mainly embraced by “abstraction” at 22%, “algorithmic thinking” at 14%, “test and debugging” at 4%.

Brennan and Resnick (2012) interpret the term computational thinking as a tool to conceptualize learning and development by using Scratch. Scratch is a free programming tool for block-based programming. It enables users to share their projects globally thanks to its more than seventy language support features. Scratch’s interface and navigation bar are not in a complex form; thus, it can be manageable for all levels.

Looking from a broader window, Brennan and Resnick (2012) proposed a framework for computational thinking with three dimensions, (a) computational concepts, (b) computational practices, and (c) computational perspectives, based on their work with Scratch. This free online programming tool encourages peer collaboration.

a. Computational Concepts

Brennan and Resnick (2012) identified seven relevant computational thinking concepts in programming contexts: Sequences, Loops, Events, Parallelism, Conditionals, Operators, and Data, which are explained below.

a.1. Sequences. The task's codes should be written in a series of steps to make the program execute the codes in order. This concept is important because the computer must take the codes individually to run them properly.

a.2. Loops. The repeated code blocks create a pattern that could be written as loops. The loop consists of code blocks that are repeated in a sequence.

a.3. Events. Events are widely used in most programming languages and platforms. It helps to trigger the action to cause another.

a.4. Parallelism. This concept enables the program to execute more than one sequence of instructions simultaneously.

a.5. Conditionals. The conditionals are the decision-making component of a program according to a specific condition.

a.6. Operators. Sometimes, a program needs logical and mathematical expressions to run. The operators enable the programmer to communicate with the program to express these expressions.

a.7. Data. The data keeps, retrieves, and updates the values the programmer wants to use again. The data could be in the form of a variable or a list.

b. Computational Thinking Practices

This dimension emphasizes designers'/programmers' thinking practices when coding. There are four practices: being incremental and iterative, testing and debugging, reusing and remixing, and abstracting and modularizing. These practices explain how the designers think and how they learn while they create a project.

b.1. Being Incremental and Iterative. The design of a project needs some iteration process; in these times, the programmers should find new solutions according to their own experiences.

b.2. Testing and Debugging. The programmer may face some problems or errors in their codes when they test the program. The process of fixing it involves looking for other projects, seeking help from knowledgeable others, or trying different things on their own.

b.3. Reusing and Remixing. These practices require and improve code reading ability because these practices occur when a programmer wants to use and alter someone's work.

b.4. Abstracting and Modularizing. These practices are like putting together the work that the programmer achieves in small steps as stacks.

c. Computational Thinking Perspectives

The computational thinking perspectives, expressing, connecting, and questioning, refer to explanations of changes in the viewpoints of the designer/programmer throughout the use of Scratch. These changes could be about who they are, how they relate to others, and how the technological environment affects them.

- c.1. Expressing. This perspective focuses on the idea that people can produce a product rather than only consume it, and the production step is also a way of expressing themselves.
- c.2. Connecting. From the connecting perspective, when the designers realize they have access to other designers through different platforms such as forums or the Scratch Explore page, the value of creating with others and designing for others arises.
- c.3. Questioning. This perspective helps students question the world presented to them and create artifacts for it.

2.1.1 Assessment of computational thinking

Brennan and Resnick (2012) asserted that a combination of approaches for assessing computational thinking could be more appropriate than a single assessment approach. Brennan and Resnick (2012) proposed three approaches to assess the students' computational thinking development: (a) Project portfolio analysis, (b) Artifact-based interviews, and (c) Design scenarios.

- a. Project Portfolio Analysis: In this approach, the Scrape Tool or User Analysis tool is used to evaluate the development of computational thinking skills (Brennan & Resnick, 2012). This tool visualizes the frequencies of used and unused blocks by specifying the color to which the block group belongs. However, since this approach is “entirely product-oriented”, the process of developing computational thinking skills and the project content could not be assessed by only adopting this type of analysis (Brennan & Resnick, 2012, p. 15). Therefore, if this

approach is followed for evaluation, it would be more beneficial to use another approach besides this to evaluate a project.

- b. **Artifact-Based Interviews:** The other way to assess computational thinking skills is artifact-based interviews, which have four main headings protocol: (1) Background, (2) Project Creation, (3) Online Community, and (4) Looking Forward (Brennan & Resnick, 2012). The interviews last approximately one hour. The Project Creation section poses questions about the project idea and evolution during the time, which is of great importance in evaluating the development of computational thinking concepts and practices (Brennan & Resnick, 2012).
- c. **Design Scenarios:** In this approach, two projects adopt the same concept and practices but with different complexity levels and design elements. The students are asked to choose a project, then explain what it does, how it could be extended, fix the bug or bugs in the project, and remix the project by adding a new feature (Brennan & Resnick, 2012).

In addition to the approaches for assessing the computational thinking of Brennan and Resnick's (2012), the scales and tests were devised to evaluate the computational thinking of middle schoolers in different languages and emphasized different subheadings. One of the Turkish sources, Bilgi İşlemsel Düşünme Becerisine Yönelik Öz Yeterlik Algısı Ölçeği (BİDBÖA), is a 3-point Likert type scale that measures self-efficacy perceptions of 10-14-year-old students (Gülbahar et al., 2018). This scale consisted of 36 questions in the categories of Algorithm Design Competency, Problem-Solving Competency, Data Processing Competency, Basic

Programming Competence, and Self-Confidence Competence, with a reliability score of 0.943 Cronbach's alpha (Gülbahar et al., 2018).

Yağcı (2019) devised a valid and reliable 5-point Likert scale in Turkish with 42 questions that can be grouped under 4 factors: Problem-Solving, Cooperative Learning & Critical Thinking (CL&CT), Creative Thinking, and Algorithmic Thinking.

The other scale in Turkish, the Computational Thinking Levels Scale (CTLS), was created by Korkmaz and her colleagues in 2015. This 5-point Likert-type scale has 22 questions, categorized under these headings: Creativity, Algorithmic Thinking, Collaboration, Critical Thinking, and Problem-Solving, and its reliability score is 0.809. (Korkmaz et al., 2015). There were no tests to assess computational thinking in Turkish; however, Polat and her colleagues (2021) conducted a study to propose a "comprehensive assessment" of this area by combining the Computational Thinking Test (CTt) (González, 2015) as the performance measurement and the Computational Thinking Levels Scale (CTLS) (Korkmaz et al., 2015) as the perception measurement. Since the CTt is in Spanish, not Turkish, the researcher translated and adapted this test to the Turkish population. CTt has 28 questions in different formats: visual arrows, visual blocks, and textual (González, 2015).

The other instrument for assessing computational thinking is the Events and Sequence Test devised by Salac and her colleagues in 2020, comprised of 9 questions in multiple-choice, open-response, and fill-in-blank formats. This test has 5 items focused on events with Cronbach's alpha value of 0.72 and four items on sequences with Cronbach's alpha value of 0.70 (Salac et al., 2020). The same group of researchers developed another instrument for evaluating computational thinking, named The Parallelism and Loops Test. The test has 9 items in multiple-choice,

open-response, and fill-in-blank formats (Salac et al., 2020). It covers the computational thinking concepts of parallelism with one question with Cronbach's alpha value of 0.82 and loops with 8 questions with Cronbach's alpha value of 0.85 (Salac et al., 2020) These two tests are designed for assessing specific concepts, but they do not cover all of the computational concepts. Also, the items of these tests include visuals from the Scratch platform.

2.2 General pedagogical approaches for special education students

This section introduces the Universal Design for Learning and Explicit Instruction as the general pedagogical approaches to reach out to students with special needs.

2.2.1 Universal Design for Learning

Israel et al. (2015) define Universal Design for Learning (UDL):

UDL is an instructional planning and delivery framework intended to increase meaningful access and reduce barriers to learning for students with diverse learning needs, including, but not limited to, students with disabilities, English language learners, and those from diverse cultural and socioeconomic backgrounds. (p. 6)

UDL should highlight students' learning barriers and enhance their learning through effective instructional plans emphasizing engagement, flexible use of materials, and meaningfully accessible instruction by teachers and curriculum developers (Israel et al., 2014). The medium of education is not a factor in UDL; it can be delivered in all forms of education, whether online or hybrid (Rao et al., 2021). UDL has three principles. The first is multiple means of representation (what we learn), in which the content is delivered through various channels, such as discussions, multimedia presentations, and readings. The second principle is multiple means of action and expression, which means providing students with different ways to demonstrate their

understanding, such as traditional tests or papers, art, multimedia presentations, and digital recordings. The last principle of UDL is multiple means of engagement, which requires considering the use of collaborative learning, instructional games and simulations, and real and virtual tours to engage students in learning. Israel and her colleagues (2015) suggested guidelines and checkpoints for implementing UDL into teaching practices for providing effective instruction to students with diverse needs. These guidelines are provided in Table 1.

Table 1. The Universal Design for Learning Guidelines

Multiple Means of Representation	Provide options for perception
	Provide options for language, mathematical expressions, and symbols
	Provide options for comprehension
Multiple Means of Action and Expression	Provide options for physical action
	Provide options for expression and communication
	Provide options for executive functions
Multiple Means of Engagement	Provide options for recruiting interest
	Provide options for sustaining effort and persistence
	Provide options for self-regulation

Note. Adapted from “Universal Design for Learning: Recommendations for Teacher Preparation and Professional Development” (p. 9) by M. Israel, C. Ribuffo, and S. Smith, 2014.

2.2.2 Explicit Instruction

Explicit Instruction (EI) is an effective, structured, and systematic methodology for teaching skills with a direct and unambiguous approach to teaching and includes instructional design and delivery procedures (Archer & Hughes, 2010). Israel and colleagues (2015) define explicit instruction as an approach to education that is

effective for students with learning disabilities and who struggle with following multistep directions. Explicit instruction provides support or scaffolding to students with feedback that guides them through the learning process with straightforward explanations about the aim, the rationale for learning the new skill, and the instructional target until students can achieve mastery by themselves (Archer & Hughes, 2010).

When students cannot solve a problem, they get frustrated and seek help from teachers or peers (Israel et al., 2015). Explicit instruction can be adopted to prevent frustration. Each task step is explained in detail, which may help reduce students' frustrations or make them work collaboratively. Although explicit instruction allows students to work in groups, the way of grouping is an essential element. According to Archer and Hughes (2010), the pupils for instruction should be grouped according to their instructional needs and current functioning level. By doing so, students have more opportunities to interact with their peers and also improve soft skills such as turn-taking (Archer & Hughes, 2010).

Israel et al. (2015) asserted that the term explicit computing education helps students learn about thinking processes and discrete skills related to computing and makes them use these skills in authentic instructional contexts. They also highlighted the idea of using instructional practices such as modeling and peer collaboration, which are commonly found effective in other content areas, in computing education to support struggling learners (Israel et al., 2015).

Explicit Instruction has 16 elements that can be considered as the methods of designing and delivering effective instruction (Archer & Hughes, 2010). These elements are shown in Appendix A.

Israel and her colleagues (2015) suggest that seven of these EI elements can be applied to CS education, and they further provide examples to give an idea for implementation. These elements are #1, 5, 7, 8, 9, 12, and 13.

Considering these elements, an explicit lesson has three parts: opening, body, and closing. In the opening, the first thing is gaining and maintaining students' attention to the lesson by adopting a procedure. The procedure must be repeated throughout the program to make it "ritualized." When attention is ensured, the next step is to introduce the lesson's aim to the students to let them know what they will learn. Then, the target skill's relevance is explained. At the end of the opening part to ensure all students know how to perform the prerequisite skills.

The body section of an explicit lesson has three steps: modeling or demonstrating the skill (I do it), providing prompted or guided practice (we do it), and providing unprompted practice (you do it). The first step involves demonstrating the skill and describing what is being done. While describing the performance, thinking aloud is one of the most helpful strategies to make students internalize the new skill and recall the steps and the decisions involved in using the new skill (Archer & Hughes, 2010). When thinking aloud, the description should be clear, and the wording should be concise and consistent. Although the teacher is the performer of the new skill, the students can be involved in the process by asking questions about the content without performing the new skill. The second step is "we do it." The teacher provides prompts while the teacher and students both perform the new skill. Prompts can be physical, verbal, and visual. Physical prompts are mostly preferred when the students have developmental delays or when the learning unit requires only motor tasks (Archer & Hughes, 2010). Verbal prompts can be in three forms: telling, asking, and reminding. The highest level of the verbal prompt is

telling, then asking, then reminding (Archer & Hughes, 2010). The final step of the body part is “you do it.” The students are asked to perform the new skill without any prompt. They are assigned to complete tasks similar to the former steps in the lesson.

In the closing part, a short review, preview, and assigning appropriate independent practice to occur. Reviewing what was learned should be interactive rather than static (Archer & Hughes, 2010). Independent practice with feedback plays a crucial role in the learning process of becoming fluent in and acquiring a new skill, especially with students who suffer from learning problems (Archer & Hughes, 2010). There are several practice formats, and using different types of practices is suggested by Archer and Hughes (2010). These formats are (1) paper-pencil tasks, (2) practice activities during instructional groups, (3) peer-focused practice activities, and (4) using computer programs.

In their book, Archer and Hughes (2010) asserted that providing demonstrations and completed models of problems to students can be a way to scaffold explicit instruction. Since the Use-Modify-Create learn-by-example approach in which students first examine the existing code and make changes in that code (Salac et al., 2021), EI can be implemented in CS education by adopting the Use-Modify-Create approach.

2.3 Pedagogies for computational thinking

This section presents Predict-Run-Investigate-Modify-Make (PRIMM) and Use-Modify-Create pedagogies to guide the design of lessons to teach computational thinking.

2.3.1 Predict-Run-Investigate-Modify-Make

Predict, Run, Investigate, Modify, Make (PRIMM) is a teaching strategy that aims to structure the programming lessons and sequence of these lessons and is mainly adopted by primary and secondary school teachers (Sentance et al., 2019; National Centre for Computing Education, 2019).

A PRIMM lesson has three parts: (a) Predict and Run, (b) Investigate, and (c) Modify and Make. In the first part of a PRIMM lesson, (a) Predict and Run, “Predict what code will do,” and “Run the code to test predictions” steps are implemented. Students discuss an existing code block and try to guess the outcome; then, they check whether their answer is correct by running the code. In the prediction part, the students can work in groups of two or three. The investigation part of the lesson, “Investigate the structure of code,” happens through the teacher's questions. These questions must require comprehension of the algorithm of the code (Sentance et al., 2019). The last section of a PRIMM class aims to “Modify the code to add functionality” and “Make a new program using the same/modified structures.” In the making part, the students are not asked to build a program from scratch; they are asked to develop a problem along with a problem description provided by the teacher (Sentance et al., 2019).

2.3.2 Use-Modify-Create

Use-Modify-Create is a three-stage progression and a curricular approach that aims for students to learn from preexisting codes by first modifying and then creating a new one (Franklin et al., 2020; Lee et al., 2011). In this approach, first, students were expected to explore the provided structured code by examining it to understand how the concept is applied to it (Franklin et al., 2020). The Modify step of this approach

requires students to make small changes in the existing code without writing code from scratch (Salac et al., 2020). The Create part is about producing computational artifacts by applying the newly learned knowledge to their projects. Owing to the Modify and Create steps' requirements, this approach has implications for Apply and Create levels of Bloom's Taxonomy (Franklin et al., 2020).

2.3.2.1 TIPP&SEE

Title, Instructions, Purpose, Play, and Sprite, Events Explore, TIPP&SEE is a metacognitive learning strategy used for Use→Modify activities in the Use-Modify-Create pedagogy, which provides scaffolding to elementary students for computer science learning by discovering the existing program (Salac et al., 2020; Salac et al., 2021b). This strategy is specialized for Scratch. The TIPP&SEE metacognitive strategy can make students with disabilities perform better than their peers with no disabilities by using Scratch (Salac et al., 2021a). Salac and her colleagues (2021b) asserted that the TIPP&SEE strategy works "very well" for students aged 9 to 10.

Metacognitive strategies inspired the development of the TIPP&SEE strategy as metacognitive strategies help struggling learners achieve learning goals by engaging in, practicing, and internalizing the ways of regulating their thinking, behaviors, and motivation (Franklin et al., 2020). TIPP&SEE strategy was also inspired by reading comprehension strategies: Previewing and Text Structure. The first part of the TIPP&SEE strategy, TIPP, is related to the previewing strategy. The previewing method is adapted to TIPP by scanning the code to point out familiar and unfamiliar concepts and predict the new one, then examining its syntax to activate prior knowledge. The other part, SEE, is designed along with text structure strategy from reading comprehension strategies. The adapted form of text structure strategy in

the SEE part can be a “roadmap” to find out the related code blocks by clicking the sprites so that the students can establish a procedure of the process by tinkering deliberately (Salac et al., 2021a).

The first part of the mnemonic TIPP&SEE, TIPP, stands for Title, Instructions, Purpose, and Play. The Title section is for answering these questions: “What is the title of the project?” and “Does it tell you something about the project?”. The Instructions part guides the students to answer, “What do the instructions tell you to do?”. The Purpose section expects answers to “What is the purpose of this activity?” and “What will this code teach you?”. The Play part of the process has a motto: “Run the project and see what it does!” and seek an answer to “Which sprites are doing the actions?”.

The last part of the strategy, SEE, stands for Sprites, Events, and Explore. The Sprites part is to explore the sprites by clicking on the sprite that the students want to learn from or change. The Events section is for looking at the events blocks starting to script and requires a response to “Which scripts are most useful?”. The Explore part drives the students to “Try different changes to the scripts and observe what happens.”

The TIPP&SEE strategy overlaps with the requirements of an explicit lesson. As in the opening part of an explicit lesson and the TIPP&SEE lesson, the students first get to know the aim before learning something new. Then, in the explicit lesson, the skills they get and the relevance of the task are introduced, while in the TIPP&SEE classroom, they learn the skills they will gain.

2.3.2.1.1 TIPP&SEE worksheets

A typical TIPP&SEE worksheet consists of three types of questions. These are (a) Observe Questions, (b) Predict Questions, and (c) Explore Questions.

a. Observe questions

Observe questions are included in each worksheet, and it asks students to keep a record of their observations during the run project.

b. Predict questions

Predict questions require students to examine the code and determine which blocks are responsible for the actions they observe. This type of question is a subset of the worksheet.

c. Explore questions

Explore questions consist of two parts. Firstly, the student is expected to make changes in the code and run the program; the other part is recording the program's response according to the change. Explore questions are also a subset of the worksheet.

One of the proposed curricula that apply all features of TIPP&SEE is Scratch Act1 and Scratch Act2 curricula created by the Creative Computing Lab. The Scratch Act1 Curriculum has five units, and three of these units emphasize computational thinking concepts. Each of these three units focuses on only one computational thinking concept. These computational thinking concepts are sequences, events, and loops. The number of lessons may vary, and each unit has only one TIPP&SEE-based lesson plan. The target audience of this curriculum is fourth-grade students, whereas the Scratch Act2 curriculum targets fifth-grade students. The Scratch Act2 Curriculum has four units, and each unit has five lessons. Also, except for Unit 2, each unit has only one TIPP&SEE-based lesson plan. Unlike ScratchAct1, Scratch

Act2 has a unit that combines two computational thinking concepts, loops and conditionals. The units focus on Animation, Loops and Conditionals, Parallelism and Synchronization, and Variables. Salac and her colleagues (2020) examine the Scratch Act1 curriculum with different diverse learners; the results might be early proof that adopting the TIPP&SEE strategy in the computational thinking curriculum supports their computational thinking development.

Salac and her colleagues' (2021a) work, which employed the Scratch Act1 curriculum, demonstrated that adopting the TIPP&SEE metacognitive strategy for the computational thinking curriculum makes diverse learners, students with low income, students with disabilities, and students who are low achievers in terms of math and reading, perform as well as their typically developed peers. In the knowledge of improving computational thinking concepts, employing the TIPP&SEE strategy helps students have an advanced comprehension of the concepts of events, sequences, and loops (Salac et al., 2020a).

Franklin et al. (2020) conducted a study to discover the behavior of struggling students aged between 9 to 10 using the TIPP&SEE learning strategy. This study revealed that the students who used TIPP&SEE worksheets applied their knowledge better than those who did not. Also, TIPP&SEE students achieved the highest "production/create" level in Bloom's Taxonomy and were also better able to develop more complicated projects.

Adopting the TIPP&SEE strategy enables disadvantaged students to perform statistically significantly in their knowledge of computational thinking concepts of events, sequences, and loops (Salac et al., 2021a). Also, providing TIPP&SEE embedded instruction for diverse learners helps them catch up with their peers in terms of the rate of project completion of diverse learners, which is equal to or higher

than their peers provided by traditional CS instruction (Salac et al., 2020b). Besides the learning of events, in nested loops and parallelism subjects, students taught by TIPP&SEE could not outperform their peers taught by traditional CS education (Salac et al., 2020a).

The students who taught a curriculum with the TIPP&SEE strategy focused more on programming elements; thus, they stayed on task compared to students who taught without the TIPP&SEE strategy. (Salac et al., 2020b). Using the TIPP&SEE strategy in the curriculum also enables the students to perform better in their ability to apply their knowledge of events and create more complex projects that involve objectives at the production level in Matrix Taxonomy, which is a CS-specific taxonomy that focuses on the cognitive domain (Salac et al., 2020b; Fuller et al., 2007). The production level of the Matrix Taxonomy is about the ability to design and build a product that realizes the Apply and Create stages of Bloom's taxonomy (Franklin et al., 2020; Fuller et al., 2007).

The Matrix Taxonomy, in other words, the Two-Dimensional Adaptation of Bloom's Taxonomy, is based on the revised version of Bloom's taxonomy (Fuller et al., 2007). The dimensions of Matrix Taxonomy encompass two abilities: Interpreting and Producing. The interpreting step is realized through the TIPP&SEE worksheets and the end-of-module assessments; the producing dimension is achieved by creating code artifacts thanks to the Create step of the Use-Modify-Create approach (Franklin et al., 2020). The production level of the Matrix Taxonomy is about the ability to design and build a product that realizes the Apply and Create stages of Bloom's taxonomy (Franklin et al., 2020; Fuller et al., 2007).

2.4 Summary

According to the literature, computational thinking is a skill that everyone should have, and it is related to problem-solving. Computational thinking can be assessed through scales and tests or by employing different instruments and approaches.

These instruments may be different in terms of their factors or adopted computational thinking definition since there is no consensus on the definition of the term computational thinking. This study adopted Brennan and Resnick's (2012) definition that covers concepts, perspectives, and practices of computational thinking.

Computational thinking has been widely integrated into Computer Science curricula all over the world. It should be considered that there are students with special needs, which can stem from different origins, such as economic, diagnosed disabilities, or bilingualism, and it is important to sustain equity in education in the integration process. A specialized computational thinking curriculum should be prepared according to needs to ensure equity.

The various needs of the students with disadvantages can be met by adopting different pedagogies to adapt computational thinking lessons. These pedagogies that integrate into Computer Science pedagogies, such as PRIMM or Use-Modify-Create, are UDL and EI. Furthermore, there is a metacognitive learning strategy called TIPP&SEE, which fosters higher application-level outcomes.

This case study aims to propose and examine the effectiveness of a specialized mini computational thinking curriculum designed for Turkish middle school students with specific learning disabilities and understand the development of the knowledge of computational thinking concepts (events, sequence, and parallelism).

This research was developed with the guidance of a general research question and its sub-questions. The questions are listed as:

General Question: How do eleven-year-old children with diagnosed specific learning disabilities develop knowledge of computational concepts as they follow a specialized computational thinking curriculum?

1a) How do eleven-year-old children with diagnosed specific learning disabilities develop the knowledge of computational thinking concepts of events, sequence, loops, and parallelism?

1b) What are the differences and similarities between the two cases (Fatih and Kaan) in terms of developing the knowledge of computational thinking concepts of events, sequence, loops, and parallelism?

Data Sources: TIPP&SEE Worksheet Questions, The Session Journals, The Events and Sequences Test, and The Parallelism and Loops Test

2a) How do 11-year-olds with specific learning disabilities respond to TIPP&SEE worksheets' Observe, Predict, Explore, and Debugging Questions?

2b) What are the differences and similarities between the two cases (Fatih and Kaan) in terms of their performance in the Observe, Predict, Explore, and Debugging Questions of the TIPP&SEE worksheets?

Data Sources: TIPP&SEE Worksheets

CHAPTER 3

METHOD

The method section explains the research design, the case, the computational thinking curriculum designed for special education students, data sources, data collections, trustworthiness, the researcher's position, and data analysis.

3.1 Research design

The purpose of this multiple case study is to propose and examine the effectiveness of a mini-specialized computational thinking curriculum designed for Turkish middle school students with specific learning disabilities. It aims to examine the development of the knowledge of computational thinking concepts (events, sequences, loops, and parallelism), the first component of Brennan and Resnick's (2012) CT framework.

The curriculum employs Scratch, a free block-based programming tool, as the information-processing agent. The qualitative approach is embraced since this study aims to work with students with special needs. Qualitative research aims to fully comprehend how things are, why and how they are, and how participants in the context see them and aim to delve deeply into the research environment (Gay et al., 2011).

In order to achieve this study's purpose, a specialized curriculum is prepared for Turkish middle school students by adopting the TIPP&SEE strategy. This strategy is developed to enhance diverse learners' computational thinking learning experience. The curriculum also supports the computational thinking practices of

Testing and Debugging and Reusing and Remixing (Brennan & Resnick, 2012).

However, since the curriculum's duration and scope are limited, the main goal of the present study is to examine the development of the knowledge of computational concepts.

To study the development of computational thinking of diverse learners, this curriculum has six lessons and six worksheets, which are prepared based on the TIPP&SEE strategy and Explicit Instruction pedagogy, and interviews were conducted with the participants.

A case study is an in-depth description and analysis of a bounded system that aims to understand how people make sense of their lives and experiences (Merriam & Tisdell, 2015; Stake, 1995). This definition is also suitable for the present study. A multiple case study is an approach that more than one case is examined and provides an opportunity to explore the differences and similarities within a case itself and across multiple cases (Baxter & Jack, 2008; Stake, 1995).

The cases examined in this study are two middle school students with specific learning disabilities. Each student is considered as a case and is bounded by their experiences with the designed curriculum for six lessons. This curriculum is developed for diverse learners with no prior coding instruction experience or coding knowledge and consists of six lessons targeting the computational thinking concepts of events, sequences, parallelism, and loops. This study focuses on the knowledge of computational thinking concepts.

In this study, the cases are an instrument for learning and understanding the effectiveness of the proposed specialized curriculum. Thus, this study employed an instrumental case study approach according to Stake's (2000) definition. He defines instrumental case studies as those examined to provide insight into an issue or redraw

generalization. Also, he sees the case in the instrumental case study as of secondary importance. However, its context is examined, and its activities are detailed to pursue an external interest. This study requires more than quantitative data to reveal how diverse learners' computational thinking is developed, so qualitative data is needed. The qualitative data were gathered through interviews and session journals. The interviews were conducted with the participants to get their opinions about the process of learning computational thinking concepts. The interviews were held at different times: before the curriculum implementation and during the implementation. In addition to interviews, the session journals kept by the researcher did provide qualitative data. The researcher either wrote or voice-recorded journal entries at the end of each lesson. These data enabled the researcher to explain the progress of knowledge of computational thinking concepts and their development. Data were provided for the researcher not only through the interviews and session journals but also through the TIPP&SEE worksheets. All these data made the researcher examine how diverse learners' knowledge of computational thinking concepts developed using the proposed computational thinking curriculum and its effectiveness.

3.2 The case selection

Since the universities switched to online education due to a massive earthquake in Türkiye in February 2023, the researcher had to return to her hometown. For this reason, the research site was determined as a small town in the Aegean region of Türkiye. The population of this town is approximately twenty-five thousand. There are fifty-seven villages in this district. Therefore, transported education is widespread. In addition to other schools in this town, there is one special education

and rehabilitation center named A. This center provides services for individuals with hearing impairments, intellectual disabilities, and physical disabilities of all ages and transported education students.

A Special Education and Rehabilitation Center with twelve teachers was selected after a few visits. The first visit was made with the institution's director, founder, and education coordinator. Information was given about the purpose and content of the research, the target participants of the study, the data collection methods, and the duration of the curriculum. Then, other meetings were held with the education coordinator of the center. Ten days after the first meeting, the education coordinator shared the list of students who could fit the study's target audience with the researcher. The cases were selected by employing a typical case sampling strategy which enables the research to select and study with a typical group and introduce this group to those unfamiliar with the setting (Patton, 2014; Suri, 2011). Creswell (2007) defines a typical case sampling strategy's purpose as "highlighting what is normal or average" (p.127). Seven students were on this list, but five were eliminated because of their grade level. Also, the transported education students of this institution were not included in the study because the return hours of these students to the villages did not match with the hours provided by the center for the researcher. The parents of the other three students and students were interviewed. They have been informed about the purpose of the research and the overall procedure of the program, and the data will only be used for scientific purposes. Then, their permission was obtained. At the end of all these, two students were included in the study. The demographic information of the selected students is given in Table 2.

Table 2. Demographic Information of the Cases

Pseudonym	Sex	Age	Grade	Type of Disability	Disability Rate (%)	Diagnosed Institution	Date of Diagnosis
Fatih*	Male	11	5	SLD (Tendency to Dyslexia)	20	Hospital	2016
Kaan*	Male	11	5	SLD (Tendency to Dyslexia) Epilepsy	30	Hospital	2016

Note. Names are pseudonyms.

a. The first case: Fatih

Fatih is an eleven-year-old male student who was diagnosed with specific learning disabilities. He attends 5th grade in a public school and also a special education and rehabilitation center twice a week. He was diagnosed by the hospital in 2016. He had a personal cell phone and smart TV at home. He has not taken any coding training and has never used Scratch before.

b. The second case: Kaan

Kaan is an eleven-year-old male student who was diagnosed with specific learning disabilities and epilepsy. He attends 5th grade in a public school and a special education and rehabilitation center twice weekly. He was diagnosed by the hospital in 2016. He did not have any personal technological devices (tablet, cell phone, laptop). He has not taken any coding training and has never used Scratch before.

3.2.1 The computational thinking curriculum designed for special education students

The curriculum developed and investigated in this study has six lessons, each of which is planned to last approximately sixty minutes. The curriculum details can be seen in Appendix B. The whole program is designed to be held face-to-face and be

delivered individually to each student. Each lesson focuses on selected computational thinking concepts and has different activities based on Scratch projects prepared by the researcher.

Scratch is a free online block-based programming tool developed by Scratch Foundation. The platform provides an environment for a community to share projects across nearly two hundred countries. It also supports Turkish, besides more than seventy languages, and it has a simple navigation and interface design, which are critical features to selecting Scratch as a tool for this study. Also, the Scratch website asserts that “Scratch promotes computational thinking and problem solving skills; creative teaching and learning; self-expression and collaboration; and equity in computing” (Scratch, n.d.).

Every lesson has a worksheet about the lesson activity. These worksheets are designed along the TIPP&SEE metacognitive strategy, which has been shown to have statistically significant effects on the computational skills of diverse learners of nine to eleven years old (Salac et al., 2021b). The classic version of the TIPP&SEE worksheet has three types of questions: Observe Questions, Predict Questions, and Explore Questions; but as this research tries to support the debugging practice too, a new type of question is added as an open-ended type, Debugging Questions. The Debugging questions may have some storytelling to provide students with insight or include scaffolding. The question types in a TIPP&SEE worksheet of this study and their examples are listed in Table 3.

Table 3. TIPP&SEE Worksheet Question Types and Example Questions from the Study

Question Type	Example Question
Observe Question	When you click the green flag icon, [...]
Predict Question	Mark the code block you predict is making the letter L grow.
Explore Question	What happens when I replace the ballerina's "forever" code block with a "repeat 10 times" code block?
Debugging Question	There is a sequencing error in Dory's speech. Let's debug this error and fix the problem. The sequencing error in Dory's speech may be due to a mistake in her code block. Let's examine the code block for this. "I found you!" says. How do you think we can fix this?

This curriculum focuses on the following computational thinking concepts: events, sequence, loops, and parallelism. The selection of the computational thinking concepts of events, sequence, and loops is based on the proposed specialized computational thinking curriculum by The Creative Computing Lab, Scratch Act1. Scratch Act1 is an introductory curriculum for computational thinking developed for fourth graders. Since the target participants of the study have no prior training in coding, this study's curriculum should be in an introductory format. Thus, the concepts of events, sequence, and loops are suitable for teaching this study's target audience. This is why this study's curriculum includes the concepts of events, sequence, and loops.

The concept of parallelism is not covered in the Scratch Act1 curriculum but is covered in the Scratch Act2 curriculum, devised for fifth graders. Nevertheless, the literature suggests that a study on the concept of parallelism is needed for

improvement (Salac et al., 2020a). Thus, parallelism is covered in this study to elaborate the literature.

Israel and her colleagues (2015) point out the seven elements of Explicit Instruction that can be applied in Computer Science education. These are focusing instruction on critical content (first element), beginning lessons with a clear statement of the lesson's goals and expectations (fifth element), providing step-by-step demonstrations (seventh element), using clear and concise language (eighth element), providing an adequate range of examples and non-examples (ninth element), monitoring student performance closely (twelfth element), and providing immediate affirmative and corrective feedback (thirteenth element).

When the curriculum was developed, these suggested elements of Explicit Instruction were considered, except for providing an adequate range of examples and non-examples element which did not overlap with the TIPP&SEE strategy.

Besides the suggested ones, this study also covered the other elements of sequencing skills logically (second element), providing guided and supported practice (tenth element), requiring frequent responses (eleventh element), and providing distributed and cumulative practice (sixteenth element).

The first element, focusing instruction on critical content, is integrated by teaching vocabulary terms and concepts about computational thinking. The second element, sequencing skills logically, is realized by including the concepts in the curriculum in a way and order that is suitable for novice learners. The fifth element, which asserted that lessons should begin with a clear statement of the lesson's goals and expectations, was realized through the adopted strategy, TIPP&SEE's Purpose heading in all the lessons. The seventh one, providing step-by-step demonstrations, was utilized in debugging questions, especially when the cases hesitated to solve the

bug. The eighth element applied throughout the study which focused on keeping the language clear and concise. In this manner, the researcher avoided using redundant words and out-of-context speech. Also, the tenth element, providing guided and supported practice, is employed by adopting the TIPP&SEE strategy, which provides scaffolding to the students. The eleventh one requires frequent responses, which occurred by assigning the cases with worksheets or test questions. As the lessons were undertaken one-to-one and provided immediate feedback when needed, the cases are monitored closely, as the twelfth and thirteenth elements offer. Lastly, the sixteenth element provides distributed and cumulative practice, covered by handling concepts in multiple lessons that include new and already taught ones and also with various types of questions.

3.2.2 The content of the worksheets

Each worksheet accompanying each lesson has the same type of questions: Observe, Predict, Explore, and Debugging questions. The total number of questions may vary in each lesson. Every worksheet begins with Observe questions, and these questions ask students to mark the checkboxes or options among multiple choices according to their observations when they click on the green flag or any Event code block. The Observe questions aim to assess the TIPP part. Meanwhile, the SEE part is evaluated by the Predict and Explore questions. The Predict questions ask students to select among the options that are made of the code blocks that cause a specific action of a sprite. The Explore questions require some modifications in the code, and the student should make these alterations and then monitor the differences that come with the changes in the codes. The original form of a TIPP&SEE worksheet does not involve

Debugging questions. This type of question is added for this research because of the idea of getting students' views of solving bugs in the codes in their own words.

However, every worksheet has a minimum of one of each question type, except for the first; first worksheet has not any Debugging questions. The question items are in types of multiple choice, checkboxes, checkbox tables, and open-ended. The variety of item types may give an idea about the preferred orientations of or familiarity of the item types with the students. Also, these worksheets enabled the researcher to monitor the student's progress.

3.2.3 The format of the worksheets

The TIPP&SEE worksheets were in a paper pen format, and their design, like font selection and font size, was based on the findings from the literature. When the worksheets were designed, some factors from the literature played a role in the process. According to a study by Ismail and Jaafar (2011), the small font size is found to be hard to read; the highlighted, moving, or crowded texts are not preferred by students with dyslexia. Evett and Brown (2005) asserted that if a text needs to be highlighted, the text should be put in a box or make text bold rather than the format italics or underlined. Thus, the highlighted text was in bold or in a box to prevent students from struggling. Evett and Brown (2005) added that the font size should be a minimum of twelve points, the color should be in contrast with the background like black on yellow, lightweight font should not be used, and the typeface should be sans serif. Thus, the texts in the worksheet were typed in OpenDyslexic font, the font size was 13 points, the colors of the texts contrast with the white and each other, and the

important words were in bold format or put in a box to ease the reading for people with specific learning disabilities.

3.3 Data sources

This research has three data sources: The Session Journals that the researcher kept, The Semi-Structured Interviews, and the TIPP&SEE Worksheets to be filled out by the participants.

3.3.1 Session journals

The session journals are observational reflective notes that the researcher recorded or wrote depending on the time the researcher had after each instructional session with the student. For these, I, as the researcher, answered the questions in Appendix C, for the Turkish version in Appendix D.

These journals aimed to keep a fresh record of the whole session and evaluate the effectiveness of the activities and students' computational thinking progress. Since each worksheet had the same types of questions, these session journals further shed light on the improvement of students' computational thinking.

3.3.2 Semi-structured interviews

This study employed semi-structured interviews, the introduction interview, and the knowledge of computational thinking concepts tests. All the interviews conducted in the classroom.

3.3.2.1 The introduction interview. This type of semi-structured interview conducted before the instruction begins. In this interview (see Appendix E, for the Turkish

version see Appendix F), the cases asked about some demographic information and whether they had used the Scratch program before. The information about the experiences in Scratch and coding instruction also gathered by parents while obtaining permission.

3.3.2.2 The Knowledge of Computational Thinking Concepts Tests. The Knowledge of Computational Thinking Concepts Tests was for evaluating the knowledge of computational thinking concepts covered in the proposed curriculum, events, sequences, parallelism, and loops across the program. There are two Knowledge of Computational Thinking Concepts Tests in terms of covered concepts of computational thinking. In these interviews, two tests developed by Salac and her colleagues (2020) were used. The tests were originally developed in English, they are concept-specific tests, and reliability and validity studies have been done. The tests were translated into Turkish by the researcher with the consultation of one English teacher whose undergraduate degree is from a Foreign Language Education Department and who is currently in an English Language Education MA program and one professor in the Computer Education and Educational Technology department. Also, the reliability of the adopted tests was calculated based on data obtained from Turkish students (see below). The tests were administered under the researcher's surveillance to prevent misreading the questions as the cases suffered from reading from time to time.

3.3.2.2.1 The Events and Sequence Test. This test (See Appendix G) was conducted after the third lesson, in which the lessons about computational concepts of events and sequences were completed. The Events and Sequences Test (see Appendix G)

was used in the interview. The test consists of nine questions in multiple-choice, fill-in-the-blank, and open-response formats. Five of the items are about the computational thinking concept of events, four of which are about the sequences. The original version of the events part's Cronbach alpha value was calculated as 0.72; the sequences part's Cronbach alpha value was calculated as 0.70 (Salac et al., 2020a).

The reliability study of the translated version of this test was made with 174 students who were 9 to 11 years old. The students were enrolled in a Technology Workshops organized in a large city. The students participated in these workshops once a week, and they lasted four hours a day. Unplugged computer science activities, computer game design and development, and graphic design and production were covered in those lessons. The Events and Sequence Test's Cronbach alpha value is calculated as 0.606; for the events factor, it is calculated as 0.558, and 0.604 for the sequences factor.

3.3.2.2.2 The Parallelism and Loops Test. This test interview (see Appendix H) was conducted after the sixth lesson, in which the lessons about computational concepts of parallelism and loops were completed. In the test administration, the Parallelism and Loops Test was used. The test has nine questions; two integrate the concepts of parallelism and loops, and the other questions are about the concept of the loops. The test's overall score of Cronbach's alpha was calculated as 0.85 (Salac et al., 2020a). The question that combines loops and parallelism is excluded, and the calculated reliability score again, and Cronbach's alpha was found 0.82 (Salac et al., 2020a). The reliability study of the translated version of this test was made with 174 students who were 9 to 11 years old. The students were enrolled in Technology Workshops

organized in a large city. The students participated in these workshops once a week, and they lasted four hours a day. The lessons in that workshops covered unplugged computer science activities, computer game design and development, and graphic design and production. The Parallelism and Loops Test's Cronbach alpha value is calculated as 0.536; for the parallelism factor, it is calculated as 0.277 and 0.508 for the loops factor.

When the Cronbach alpha value was low, a measurement and evaluation expert was consulted. The expert emphasized that reliability is influenced by two main factors: the number of items and the factor structure. In addition, there were others like outliers or translations. From this study perspective, the number of questions remained constant, the test had two factors, parallelism and loops, and the translation effect was minimized by consultation in translating procedure of the tests. To overcome the low Cronbach alpha value, tests were implemented under the researcher's surveillance, as mentioned before.

3.3.3 TIPP&SEE worksheets

There are six TIPP&SEE worksheets, each of which is different in terms of content and created along the TIPP&SEE strategy (See Appendix I for an example). These worksheets are developed for the Scratch activities, each focusing on one or more computational thinking concepts. These concepts are events, sequence, parallelism, and loops. Each lesson has one TIPP&SEE worksheet with different numbers of questions designed for the belonged lesson's activity. The worksheets included four types of questions, Observe, Predict, Explore, and Debugging, in different formats, like multiple-choice, checkbox table, and open-ended.

The participants filled out the TIPP&SEE worksheets during class hours. Although these worksheets are also suitable for working alone, in this study, the researcher guided the participants in filling in if they asked for help or were struggling, especially when reading the questions or options. When the students needed help, the researcher provided probes, feedback, or recalls. Sometimes, the students had difficulty reading the Scratch code blocks in the worksheet questions; in those times, the researcher read the piece for them. The researcher facilitated the students filling out these worksheets since the lesson had a limited time, and the worksheet should be filled out in class. If the worksheet cannot be completed during the lesson, the researcher took the worksheet with her, which will be completed first in the next class.

3.4 Data collection

The data collected after the ethical approval was obtained from the Ethics Committee in Social Sciences and Humanities (SOBETIK) of Bogazici University (See Appendix J), and the students' caregivers signed the permission letter (See Appendix K). The children and their caregivers were informed before the study about the purpose of the research, the overall procedure of the program, the anonymity of the participants, and the usage of data that only be used for scientific purposes.

The flow of data collection can be seen in Table 4.

Table 4. The Flow of Data Collection

Lesson	Interview	TIPP&SEE Worksheet	Test
First	Introduction	I am introducing myself	
Second		Finding Nemo	
Third		Name Animation	The Events and Sequences Test
Fourth		The Race of Rabbit and Hedgehog	
Fifth		Dancing Ballerina	
Sixth		Abilities of Wizard	The Parallelism and Loops Test

The data collection procedure started with an introduction interview (see Appendix E). After conducting this, the instruction began. The researcher was the instructor of the program, which lasted six lessons. The instruction was provided in a half-semester break, so the students could attend the lessons during the day. In the instruction, the specialized computational thinking curriculum, which is developed in the light of the TIPP&SEE strategy, was used. The activities and the instructional objectives were designed based on the guide from the literature, Scratch Act1 curriculum, Scratch Act2 curriculum, and Scratch Encore. In the instruction, a computer, a pencil, and that week's TIPP&SEE worksheet were used.

Fatih began attending classes two days after Kaan, and except for Sundays, he attended classes consecutively on other days. However, Kaan could only participate in the sixth class five days after the 5th class. The reason for this was the scheduling of classes by the educational institution in this manner. A timetable for the data collection can be seen in Table 5.

Table 5. Timetable of Data Collection

13.11.2023 Monday	14.11.2023 Tuesday	15.11.2023 Wednesday	16.11.2023 Thursday	17.11.2023 Friday	18.11.2023 Saturday
- Kaan Activity: I am introduc ing myself (1hour 5 minutes)	- Kaan Activity: Finding Nemo (1 hour 11 minutes)	- Kaan Activity: Name Animation and The Events and Sequences Test (1 hour 10 minutes)	- Fatih Activity: Finding Nemo (46 minutes)	- Fatih Activity: Name Animation (46 minutes)	- Fatih Activity: Name Animation (44 minutes) and The Race of Rabbit and Hedgehog (46 minutes)
		- Fatih Activity: I am introduc ing myself (43 minutes)	- Kaan Activity: The Race of Rabbit and Hedgehog (48 minutes)	- Kaan Activity: Dancing Ballerina (37 + 38 minutes)	
20.11.2023 Monday	21.11.2023 Tuesday	22.11.2023 Wednesday	23.11.2023 Thursday	24.11.2023 Friday	25.11.2023 Saturday
- Fatih Activity: Dancing Ballerina (42 + 37 minutes)	- Fatih Activity: Abilities of Wizad and The Parallelism and Loops Test (51 + 45 minutes)	- Kaan Activity: Abilities of Wizad and The Parallelism and Loops Test (41 + 44 minutes)			

After the third session interview, the Events and Sequences Test (Appendix G) was conducted within the scope of the knowledge of computational thinking concepts (events and sequence) interview. However, Fatih faced a problem due to the extension of class hours as he commuted to the educational institution by school bus, unlike Kaan. The bell rang just two questions before the end of our worksheet,

marking the end of the class time. Consequently, he didn't complete the test during the class hour. The researcher gave him the Events and Sequences Test to complete at home and kept the worksheet to be completed in the next lesson. The next day, during our fourth class, he mentioned that he had solved the test and brought it with him. Since he found some questions challenging and marked them for review, the lesson started by going through the test together. Afterward, we completed the worksheet for the third class and moved on to the fourth.

Since Kaan always waits for his mother at the educational institution until the end of the dismissal time, we did not face any time-related issues. Although our class hours had ended, Kaan had solved the test in the classroom, allowing him to receive immediate feedback.

After the third class with Fatih, the researcher recognized the potential for classes to be extended, specifically for the fourth, fifth, and sixth classes. Thus, the researcher requested an additional hour for each of these classes, and the educational institution accommodated this request.

When the sixth lesson ended, the Parallelism and Loops Test (Appendix H) was administered for the knowledge of computational thinking concepts (parallelism and loops).

3.5 Trustworthiness

Lincoln and Guba (1986) converted the terms into social sciences as “credibility as an analog to internal validity, transferability as an analog to external validity, dependability as an analog to reliability, and confirmability as an analog to objectivity” (pp. 76-77). They asserted that the combination of these four criteria,

credibility, transferability, dependability, and confirmability, make up the trustworthiness of a study.

There are strategies to ensure credibility: prolonged engagement, persistent observation, and triangulation that aim to make the findings more credible; peer debriefing that is for external checking; negative case analysis; referential adequacy; and member checking (Lincoln & Guba, 1986). Among these techniques, triangulation is applied in this study to ensure the trustworthiness of the study.

Triangulation is a technique used to record and report on multiple perspectives to ensure the credibility of research (Patton, 2002). It is based on the idea that there is no one method or perception to look for a single truth or explanation, so using a variety of data collection and analysis techniques gives researchers more information to work with (Patton, 2002; Patton, 2014; Stake, 2000). According to Patton (2014), there are four types of triangulations: triangulation of qualitative sources, mixed qualitative-quantitative methods triangulation, analyst triangulation, and theory/perspective triangulation. The triangulation of qualitative sources is implemented by employing multiple data sources, session journals (like observation notes), interviews, and TIPP&SEE worksheets. Besides using multiple data sources and applying triangulation of data sources, triangulation of methods involves collecting different data types. The interviews will provide both qualitative verbal and quantitative data, and the worksheets will provide written qualitative data.

3.6 Researcher's position

The researcher had more than one position in this study: the instructor, the curriculum designer, and the researcher. Since I, as the instructor of this study, graduated from the Computer Education and Educational Technology department,

which is under the Faculty of Education, my profession is suitable for this role. Also, the researcher's experiences as an instructor of 4th and 5th graders at a technology workshop in a large city, which covered unplugged computer science activities, computer game design and development, and graphic design and production, improved my teaching skills. These teaching experiences taught me to address and meet the needs of nine to eleven-year-old students as well as communicate with them. As the curriculum designer of this research, the courses that I, as the researcher, took in my undergraduate degree and my ongoing M.A. degree in Educational Technology at the same university and taking place in devising the Technology Workshops curricula allowed me to develop this curriculum. In addition, I had an experience in data collection via cognitive interviews in one of my graduate courses. This experience gave me an insight into the process of the interviews and preparing the interview protocols.

These different roles enabled me to be more active in the data collection procedure and have different perspectives on data analysis. Being the instructor, the curriculum designer, and the researcher of the program at the same time made me present in the teaching sessions, criticize the curriculum's effectiveness, and point out its weaknesses and strengths, and observe the students during the instruction; those gave me more comprehensive insight into the data analysis.

3.7 Data analysis

The organization and analysis of data began during the data collection process by classifying the documents after the introduction interview and highlighting the possible key points of the interview. By doing so, the researcher had an idea for the

analysis and further interviews. When all the data were organized, the session journals were documented, the audio records of the interviews were transcribed, and the tests (Events and Sequences Test and Parallelism and Loops Test) and TIPP&SEE worksheets used in the interviews were scored.

3.7.1 Analysis of quantitative data

First, students' scores on two tests were calculated as follows.

The Events and Sequences Test has nine questions in the form of multiple choice, fill-in-the-blank, and open response. This test has questions with subquestions (Q4, Q6, and Q7) or needs two or three markings or typing for the same question (Q3 and Q7b). The rubric of this test (See Table 6) was provided by Jean Salac, one of the researchers who developed the tests. The rubric for the test can be seen in Table 6.

Table 6. The Rubric for the Events and Sequences Test

The Events and Sequences Test	Points	How Points Were Assigned
Q1	1	1 point for correct answer
Q2	1	1 point for correct answer
Q3	1	1 point for correct answer
Q4a	1	1 point for correct answer
Q4b	1	1 point for correct answer
Q5	1	1 point for correct answer
Q6	6	2 points per blank (1 point for the correct verb & 1 point for correct parameter)
Q7a	1	1 point for correct answer
Q7b	6	2 points per blank (1 point for the correct verb & 1 point for correct parameter)
Total	19	

Besides the maximum score of the test, the maximum score for the concepts of events and sequence will be calculated by adding relevant questions to those concepts. The possible maximum score for the events part of the test will be 5, and for the sequence part will be 14.

$$\text{Events} = \text{Q1} + \text{Q3} + \text{Q4a} + \text{Q4b} + \text{Q7a}$$

$$\text{Sequence} = \text{Q2} + \text{Q5} + \text{Q6} + \text{Q7b}$$

The Parallelism and Loops Test consists of nine questions that are in the form of multiple choice, fill-in-the-blank, and open response, as in the Events and Sequences Test. This test also has some questions with subquestions (Q5, Q6, and Q7) or needs two or three markings or typing for the same question (Q3, Q5a, Q5b,

Q5c, and Q7). The rubric of this test (See Table 7) was provided by Jean Salac. The rubric can be found in Table 7.

Table 7. The Rubric for the Parallelism and Loops Test

The Parallelism and Loops Test	Points	How Points Were Assigned
Q1	1	1 point for correct answer
Q2	1	1 point for correct answer
Q3	4	2 points for each of 2 correct options; -1 point for each incorrect option selected
Q4	1	1 point for correct answer
Q5a	4	2 points for each of 2 correct options; -1 point for each incorrect option selected
Q5b	1	1 point for correct answer
Q5c	1	1 point for correct answer
Q6a	1	1 point for correct answer
Q6b	1	1 point for correct answer
Q7	7	2 points per blank (1 point for the correct verb & 1 point for the correct parameter); 1 point for the correct number of loop iterations
Q8	0	excluded
Extra Credit	0	excluded
Total	22	

In addition to the test's maximum score, the maximum score for the parallelism and loops concepts of computational thinking will be determined by adding the points of relevant questions. The possible maximum score for the parallelism part of the test will be 2, and for the loops part will be 20. As there are only two questions on

parallelism, the worksheets have more questions about that concept to assess students' knowledge of that concept.

$$\text{Parallelism} = Q6a + Q6b$$

$$\text{Loops} = Q1 + Q2 + Q3 + Q4 + Q5a + Q5b + Q5c + Q7$$

After calculating the students' test scores, tables for each test were created. These tables indicated how many points students received from each question, the maximum scores of each question, and the total scores of the students. Then, another table was created to determine the students' scores on each concept. To do that, the scores obtained by the students from the questions in the relevant concept were added and written on the table.

Then, the worksheets were scored. The students got 1 point for each correct answer in the Observe, Predict, and Explore questions; Debugging questions were not scored. Then, tables were created for each worksheet. These tables indicated the scores on each question type, Observe, Predict, and Explore, and the students received from each question, the maximum scores of each question, and the total scores of the students. Then, another table was created to demonstrate the students' total scores for each question type by adding the received scores of the students in each type of question.

3.7.2 Analysis of qualitative data

The qualitative data sources of this study were the session journals and interviews which were in recorded voice formats. First of all, the folders were created for each case with subfolders for each lesson. Each case folder had 6 subfolders for classes, and it included the session journal voice record or written document, the worksheet used in that class, and the recording of the interview. Firstly, the session journal

recordings were transcribed, and if there was a mentioned question from the tests or worksheets, this question was checked to see whether the student answered correctly or not to discuss in the findings section. Then, the interview data were listened to gain insight, and the key conversations were transcribed. After that, the researcher listed the points for each concept for the findings and pieces of conversations to exemplify these points.



CHAPTER 4

FINDINGS

This section aims to reveal the findings of this research. First, the first case, Fatih, is introduced, and the findings about Fatih's knowledge development in computational thinking concepts of events, sequences, parallelism, and loops were examined one by one. Then, the second case, Kaan, is introduced, and Kaan's knowledge development in each computational thinking concept of events, sequences, parallelism, and loops is examined. After that, the comparison of students' overall development of the knowledge of the computational thinking concepts of events, sequences, parallelism, and loops is made in light of the findings of this study. Finally, the findings on the comparison of students' overall development in terms of observe, predict, explore questions, and their debugging practice were mentioned.

4.1 RQ1a) How do eleven-year-old children with diagnosed specific learning disabilities develop the knowledge of computational thinking concepts of events, sequence, loops, and parallelism?

4.1.1 The first Case: Fatih

4.1.1.1 Description of the first case

Fatih is an eleven-year-old boy who was diagnosed with a specific learning disability. He was diagnosed in 2016 with a 20% disability rate. He was born in 2012 in a city in the Aegean region (Introduction Interview, 15.11.2023). When he was 5, Fatih and his family moved to the current district where he lives (Introduction

Interview, 15.11.2023). A short time ago, but there is no precise date of it, he lost his father and brother. Now, he is living with her mother. He talked about his father during the lessons without mentioning the loss. His sentences about his father were not in the past tense. So, the researcher assumed that his father was alive. Then, one day, he said, "My father died." (Third Lesson Interview, 17.11.2023). The researcher got confused, and it was hard to continue the lesson. The researcher gave him some stickers in another class and told him he could share them with his sisters or brothers. Then, he said that he lost his brother, too. Again, it was not easy to move on to study with the worksheet.

He is attending one of the public schools in this district and is enrolled in 5th grade. He also attends forty-five minutes of special education at a special education and rehabilitation center once a week by using the center's transportation service, which is provided free of charge by the education center. While this is a great opportunity, it eliminates the possibility of spontaneously extending a lesson if needed. Recognizing the potential for classes to be extended, specifically for the fourth, fifth, and sixth classes, the researcher requested an additional hour for each of these classes, and the educational institution accommodated this request. However, the researcher had not foreseen this for the third class.

Fatih was a slow reader. He struggled to read the texts on the computer screen compared to the paper. Rarely he complete the sentences from his mind, and if he struggled to read a word, like a trumpet, he did not hesitate to ask for help (Fourth Session Journal, 18.11.2023). His writing was meaningful and conveyed an answer to the question, and he did not refuse to write in any lessons.

His favorite subject is math; however, his score on math is low. He asked the researcher whether he would pass math class with a point of 50 when the passing

score was 50. Then, the researcher explained that he would pass if the passing condition were 50 and above. Then, he shared the Turkish score with the researcher, 38. He interpreted this note as “very low” (çok düşük) and added that this course requires some memorization, like the first two stanzas of the Turkish national anthem and a poet. He asserted that he had struggled to memorize the first verse of the national anthem and had been working for a month to do so.

He took a compulsory Computer Science class in his school. The ICT curricula of The Ministry of Education include the topics of digital citizenship, ethics, spreadsheet programs, and debugging and programming. Since Fatih's school needs computers, the ICT lessons only covered topics that do not require computers, like ethics. Thus, Fatih has no coding experience and has not heard of and used Scratch.

In terms of having technological devices, he has his own phone with an internet connection and a smart TV in their home. He can connect the TV to the internet and surf on it, mostly YouTube. He has a YouTube channel, and he aims to reach 100 subscribers. In our first class, he asked the researcher to subscribe to his channel and added that he gave a like it if the researcher wrote a comment. When he wanted to upload a new video, he asked for help from his neighbor's son. The neighbor's son created a video for him in CapCut, and he shared it on his channel. He also said he would make a video about Scratch when he starts a new project to make a game and share it on his page. In the fourth lesson, his mother bought him a new phone after going to the dentist, and he told the researcher that he put much effort into memorizing his and his parent's phone numbers. After his father died, he started using his father's phone line.

Having his cell phone and smart TV at home may increase Fatih's familiarity with the Internet and how the websites work. He often wanted to click the buttons, drag the code blocks to the code screen, change the value of a code block, and widen the play screen, which enabled him to be active in learning. He made the changes in the code, restarted the project if he had missed a thing in the scene, and had control over the project. He quickly found the buttons on the screen. Especially in the beginning of the Predict questions, there was a saying like "Let's click on the 'See inside' button. Then, let's find the events by clicking on the sprites. After that, let's answer the questions here.". In our first lesson, the researcher showed him the button and mentioned that he could see this button by the icon of it. Fatih found, showed, and clicked this button in all other lessons. After showing the location of this button in the 3rd lesson, he said, "I have memorized it now, teacher" (ezberledim artik öğretmenim) (Third Lesson Interview, 17.11.2023).

Also, since he had personal or shared technological devices, he had a chance to explore Scratch on his own. Somedays, at the beginning of the class, he mentioned what he did last night in Scratch, asked the researcher about how to save a Scratch project, or showed her which background he used in his project. His interest was aroused, and it was good for his development of computational thinking to have a chance to maintain it while he had technological devices at home.

Although he had technological devices, his favorite games were Rain, which he and his friends played in a park with a ball, and football. Even though he brings up playing mobile games in later discussions, the preference for street games as the first favorites that come to mind could suggest that technology is not primarily used for gaming purposes.

The researcher gave him a pencil in our first session and told him to bring it to every class. Additionally, the researcher mentioned that at the end of each class, we would put a sticker on that pen to count how many courses we completed together through these stickers. He brought the pen with him for the first three classes and proudly showed it as soon as he entered the classroom. However, after the third class, he forgot the pen at home. In the last lesson, the researcher put the other stickers she had promised him on the back of the certificate the researcher gave him.

4.1.1.2 Fatih's development of the knowledge of the computational thinking concept of events

The computational thinking concept of events is about triggering the code blocks. The development of Fatih's knowledge about the concept of events can be seen throughout the lessons since all the classes covered this concept. This part will examine Fatih's knowledge of the computational thinking concept of events.

Fatih encountered the concept of events for the first time in the first lesson. The researcher explained how the Events code blocks work by trying each of them in an example project and said,

"We have another, and I think the most important, code category called Events. The code blocks in Events are crucial because if I want the game to start a certain way or the character to move in a specific manner, for example, if I want it to move when the green flag is clicked, I need to place this code block first. Or, if you like using the keyboard, you can use the 'when space key pressed' code block, and from here, you can select any letter or number on the keyboard. This way, the code blocks you place underneath will start working based on this event." (First Lesson Interview, 15.11.2023)

By saying so, the importance of events, such as their mission, example of the Events code blocks, and where their location should be in the code command, were

emphasized. Although he remembered where the Events code category takes place in the Scratch screen in the third lesson, he sometimes forgot significant information about this concept, like the position of an Event code block, it must be included in each code command, and numbers also can be selected in 'when space key pressed' code block. Thus, it became important to recall those in the other lessons.

For instance, the researcher recalled the information about the Events code blocks in the fifth lesson after she realized that Fatih had difficulty finding the Events code block when asked to show that. Then, she reminded Fatih that all the code commands need an Events code block to be triggered, and these Events code blocks should be positioned as the first code block of a code command. After this reminder, Fatih identified all the Events code blocks correctly.

The third lesson's activity required him to explore the 'when this sprite clicked' Events code block. When this question was examined, he opened the tab of the Events code category on the Scratch and found this code block. Then, he tried to drag it under another command sequence. When it did not stick into the code block, the researcher used his analogy to recall; "It (when this sprite clicked code button) does not stick into here (next costume code block). You know, as you said, it will stick here like a puzzle piece, but it doesn't have any place to stick. It doesn't have any suitable indentation for this. But of course, I can add a code block underneath this." (Third Lesson Interview, 17.11.2023). Then, he testified to the researcher's saying by trying to drag a random code block into the 'next costume' code block and said, "For example, let us do something random, teacher. Let's say motion, looks. For example, 'hello.' For instance, it sticks here (in the next costume). But it doesn't stick in this one." (Third Lesson Interview, 17.11.2023). His active engagement in trying out different scenarios enhanced his understanding of the concept of Events. Through

this instance, he rediscovered that the Events code block should be the first in the ordered sequence and cannot be attached to another code block within its category. Also, the importance of the Events code blocks comes first and is emphasized in another lesson besides the first class. For example, in the third class activity, the researcher reemphasized its importance by saying, “I have two code blocks for my sprite, I. The first one is the event code block, as always.” (Third Lesson Interview, 17.11.2023).

In the third class, while clicking on ‘See inside’ to examine the events of the sprites, he said, “Teacher, I have never placed a flag like this.” (Third Lesson Interview, 17.11.2023), which he meant for his project. Here, the researcher realized that the concept of events was not fully understood, so the researcher explained it again. Initially, the researcher explained that the Events code block should be placed at the top of a code command to run the other code blocks under it. While saying this, the researcher thought that emphasizing the need for an Event code block at the beginning would help him understand that the code block underneath would be triggered and executed. Later, he interrupted the researcher and asked, “How else could it be, for instance?” (Third Lesson Interview, 17.11.2023). To answer this, the researcher opened the Events code category in Scratch and reintroduced the Events code blocks to him.

Else, I have a space bar. I can change this space to any key. It could be left or right, yes, it could be letters, and it could be numbers. You can choose whatever you want. For example, every code you write under this event block will now run when you press four. Or you could say, ‘When this sprite clicked’. The code blocks I wrote below will now run when you click A on the game screen. (Third Lesson Interview, 17.11.2023)

He used that knowledge in the fourth activity, which mainly focused on the concept of parallelism. This activity was about a race between a rabbit and a hedgehog (See Figure 1).

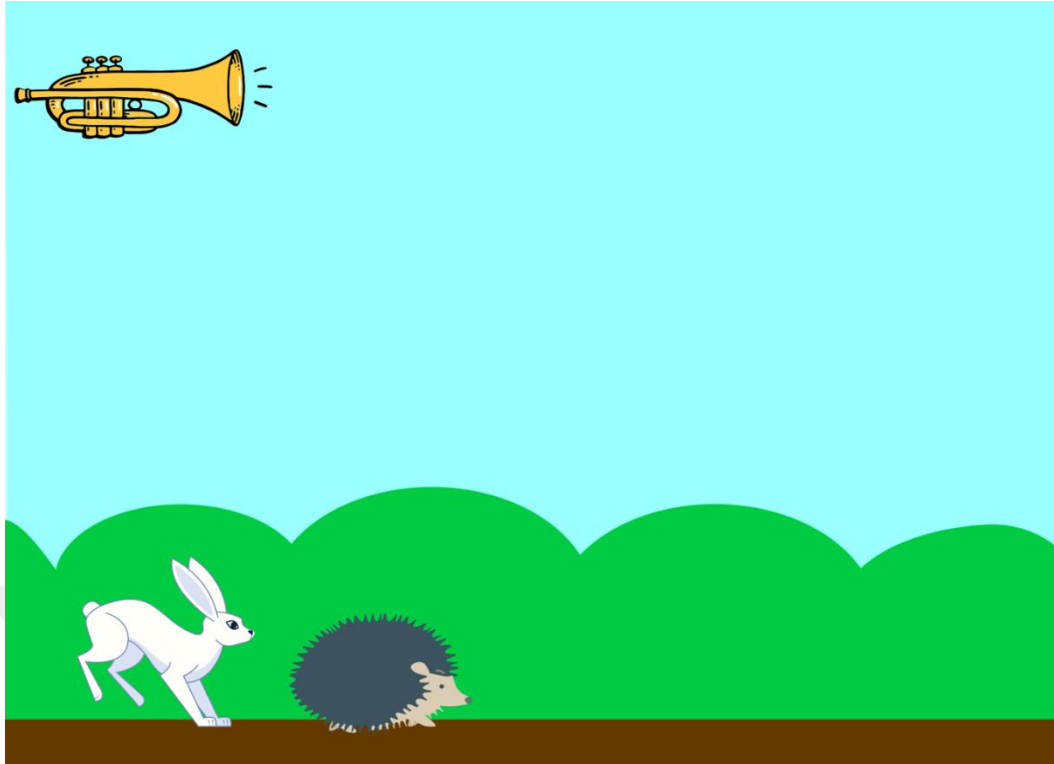


Figure 1. The screen of the fourth activity

He not only used his knowledge but also extended his knowledge about the concept of events by changing the Events code blocks of the sprites, which was upon his suggestion.

F: Well, teacher, what happens if we do something like this? For example, we say the T key, we can edit it, (...) For example, what if we say (...) 64?

Researcher: Here, I can say either 6 or 4, not 64.

F: Oh, only a digit.

Researcher: Yes, only a digit

F: Hmm, I was wondering about that.

Researcher: You can only write a one-digit number.

F: Teacher, let's make numbers once. We didn't do with numbers. (Fourth Lesson Interview, 18.11.2023)

By making this suggestion, it can be understood that he knew that the “when the space key is pressed” Event code block could be changed by another key button from the keyboard, like a letter into a number, but he did not know the format of the number. Then, he wanted to change the rabbit’s Event code block, “when the T key is pressed”, into “when the 7 key is pressed” and the hedgehog’s “when the key K is pressed”, into “when the 8 key is pressed”. After making the changes, the researcher reminded him that pressing either T or K would no longer make the sprites move due to the event code blocks for sprites. The researcher also demonstrated that by pressing the T and K keys. Then, he continued by saying, “But if we press 7, the rabbit starts to move.” (Fourth Lesson Interview, 18.11.2023)., and he pressed 7 to move the rabbit. After a while, when the rabbit was no longer visible on the screen, the researcher asked him how we could bring the rabbit back to the starting point. He replied, “(It) comes back if we click the flag again” (Fourth Lesson Interview, 18.11.2023)., which was a correct answer considering the relevant code block is the event block ‘when green flag clicked’.

A similar occurrence took place during the third class, in the Name Animation activity. After making changes to the code blocks of a sprite for a question on the worksheet, he wanted to see what would happen when he ran the codes. For this, he said in a poor sentence structure, “We click this to be bigger ... we clicked the green flag.” (Third Lesson Interview, 17.11.2023).

In studying the fourth worksheet, examining the code blocks of the sprites, the researcher explained the code command that makes the sprite go back to its initial positions. Then, she asked Fatih what he should do to change the sprite’s position to its initial position, and he answered correctly and verified his answer by clicking the green flag.

Those instances where he changed an Events code block to another and found the Events code block that triggered the desired action demonstrate his ability to think from different perspectives regarding the computational thinking concept of Events, even though he sometimes needed reminders.

In the later lessons, he was able to show the Events code blocks on his own, or he could add those blocks to the code screen to solve the Debugging questions, as he did in the sixth worksheet's Debugging question (See Figure 2).

Büyücünün son dediği şey benim de kafamı karıştırdı. Büyücünün evine geldiğimizde kutup ayıları da bizimle geldi. Bunu nasıl çözebiliriz? Kendi çözümünü deneyip alttaki kısma yazabilirsin.

Figure 2. The debugging question from the sixth TIPP&SEE worksheet

When he proposed his solution to the question (See Figure 2), he confused the categories of Events and Control, but Fatih explained what he thought. He wanted to create a code command to hide the polar bear sprite when the left key is pressed, which also triggers the décor to change into Wizard's house, but he said Control rather than Events. He could not explain what he wanted to make but showed by creating the code command. He tried to explain his idea by saying, "If we put left

button control (he means when left key pressed code block) of the polars (polar bears), (Kutupların kodlarına sol tuşunu kontrol dersek)” and say hide there (under the ‘when left key is pressed’ code block), (oraya gizle dersek olur?)” (Sixth Lesson Interview, 21.11.2023). His saying can demonstrate that he has the knowledge that the events code block must be placed at the top and the others placed underneath it, and also, he developed his knowledge on an events code block, such as ‘when space key pressed’ can have different input besides the ‘space’.

Knowledge development in this concept can be assessed through the student's overall score in the Events and Sequences Test’s events questions. This test includes five questions specifically focused on the Events concept, with a maximum score of 5. Fatih achieved a score of 3, making two mistakes.

One of his false answers was to the third question (See Figure 3). This question asked the students to select all the code commands that were executed by clicking on the sprite. The students were expected to mark two code commands, while Fatih did not select any correct answers. His selection had the ‘when green flag clicked’.

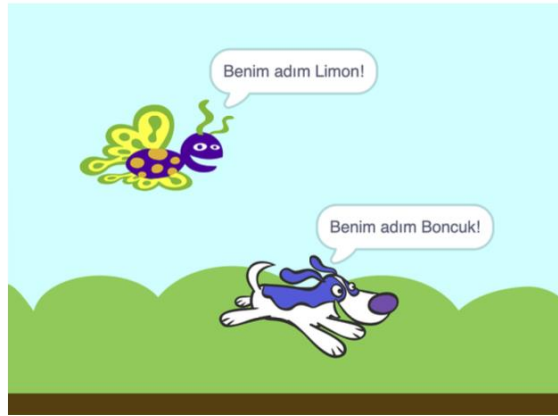
3. Aşağıdaki komut dizileri bir kuklaya aittir. Kuklaya tıkladığınızda çalışan tüm komut dosyalarını **yuvarlak** içine al.



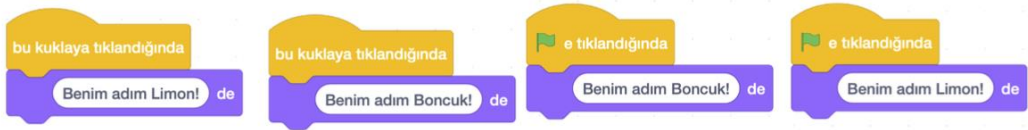
Figure 3. The third question from the Events and Sequence Test

The other mistake he made, which was the 4a question (See Figure 4), is confusing since this question has a common root with question 4b (See Figure 4), which he answered correctly. The question probed, ‘The scene will be like this when the green flag is clicked’, and 4a asked to mark the option that the butterfly has. Whereas the student was expected to select the code command with green flag Events block and say code block with the butterfly’s name on as a value, he did not select the correct form of the Events block, but his answer covered the correct ‘say’ code block.

4. Yeşil Bayrak’a tıkladığında, sahne şöyle görünür:



4a. Kelebek için çalışan komut dizisini **yuvarlak** içine al.



4b. Köpek için çalışan komut dizisini **yuvarlak** içine al.



Figure 4. The 4a and 4b questions from the Events and Sequence Test

Fatih acquired and expanded his knowledge of the computational thinking concept of Events throughout the study. Although he needed reminders from time to time or was

confused in using the terms, like Events and Control, at the end of the classes, he demonstrated the knowledge that the Events should be included in all code commands and placed at the beginning of it, and some Events code block's input can be changed to specify the triggering key.

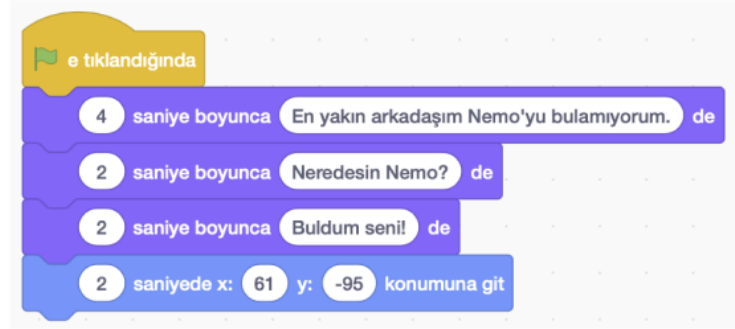
4.1.1.3 Fatih's development of the knowledge of the computational thinking concept of sequences

The computational thinking concept of sequences is mainly about the order of the code blocks that are executed by the computer. In the curriculum, this concept is first discussed in the context of Finding Nemo, then Name Animation. In the other lessons, this concept was studied and evaluated in both tests, and it is critical to determine which code block is executed in which order.

Due to its importance, this concept is studied for the first time in the second lesson. In Finding Nemo's worksheet, which is the second class activity's worksheet, the last question (See Figure 5), which is a debugging question, is about this concept. Since this question is an open-ended format, it has no numerical value. Fatih, when he first examined the question, said, "Teacher, there is nothing wrong here." (Second Lesson Interview, 16.11.2023). Later, after restarting the project, the researcher showed him that he said, 'I found you' before the fish reached Nemo, that is, before it reached its friend. Then, the researcher explained this situation to him with a real-life example.

1. Dory'nin konuşmasında bir sıralama hatası var. Bu hatayı ayıklayıp sorunu çözelim.

- Bu sıralama hatası, onun kod bloğundaki bir yanlışlıktan ötürü olabilir. Hadi kod bloğunu inceleyelim.



- Daha Nemo'nun yanına gitmeden "Buldum seni!" diyor. Sence bunu nasıl düzeltebiliriz? Aşağıdaki kutuya yazabilirsin.

Figure 5. The last question of the second TIPP&SEE worksheet

Researcher: For example, let's say you lost your eraser, right? You can't see your eraser anymore. The eraser is far away.

F: Yes.

Researcher: Exactly. When you go to it like this and pick it up, you say I found you.

F: (Nods)

Researcher: Exactly. You should go to where the eraser is, pick it up, and then say, 'I found you.'

F: Yes, yes.

Researcher: In this case, he actually told us that. He says, 'I found you before even going to Nemo.' How do you think we can do that? I mean, I understand what he's asking for. First, go to Nemo, the red fish, then say, 'I found you.' (Second Lesson Interview, 16.11.2023)

He talked about other subjects in between; later, the researcher brought the conversation back to the topic. After that, Fatih suggests restarting the game and watching it again. Then, he explains the solution, again in a poor sentence structure: "Teacher, like this, uhh, let's get to 'I found you.' Let's do the opposite of the movement seconds with 'I found you.'" (öğretmenim, şöyle, ııı, gelelim, buldum seni. Buldum seniyle hareket saniyesini tam tersini yapalım) (Second Lesson Interview, 16.11.2023). His suggestion included changing the code block's positions, which was evidence that he developed his knowledge of the concept of sequences in the first lesson that covered this concept. He built a link between the computational thinking concept of sequences and the positions of the code blocks.

Then, the researcher made the code changes based on his suggestions and clicked on the green flag to observe and verify. Before moving on to the writing part, the researcher once summarized the solution process for him. After that, he asked "Shall we say, 'switch the code blocks' places?" (Second Lesson Interview, 16.11.2023) and, upon the researcher's approval, he wrote the answer. The researcher offered him a suggestion for writing, such as specifying which code blocks he moved, but he wrote it in the way that came to his mind. Even though he may not spell the word 'block' correctly, his response is satisfactory; he wrote, "We change the position of the code block." (Kod buluyuna yerini değiştirdik.) (Second TIPP&SEE worksheet, page 4). He was aware of the solution he found and could

write it down in his own words, even though the researcher suggested other options, such as “You can also write which code blocks you changed the position of.” (Second Lesson Interview, 16.11.2023).

In the Name Animation task, the third class activity, there was a Predict question about the concept of sequences (See Figure 6). The question asks, "When you transform the codes of the letter 'L,' which are the codes in the first visual, into the codes in the second visual, what happens?" Since he misread the question while reading it, the researcher reread the question, progressing while explaining it, "Hmm, so it seems I need to make a change. How do you think I can transform these code blocks into this state? What difference do you see between them?" (Third Lesson Interview, 17.11.2023) Then, he read the code blocks, starting the Events one by one, “Now, the flag, okay. Next costume. Wait 1 second. Change size by 1.” Then, he offered to make the changes on the Scratch screen.

1. L harfinin kodlarını, yani 1 numaralı görseldeki kodları 2. görseldeki kodlar haline getirdiğinde ne olur?

e tıkladığında

sonraki kostüm

1 saniye bekle

boyutu 10 birim değiştir

1

e tıkladığında

boyutu 10 birim değiştir

1 saniye bekle

sonraki kostüm

2

☐ L harfinin önce rengi değişir. 1 saniye sonra büyür.

☐ L harfi önce büyür. 1 saniye sonra rengi değişir.

☐ Bir değişiklik olmaz.

Figure 6. The question from the third TIPP&SEE worksheet

While the researcher expected him to change the order of the code blocks on the screen, he deleted all of them except the Event code block. Then, he said, “Now, first the size of it”, which is the first code block after the Events code block. Afterward, he tried to find the ‘change size by 10’ code blocks among the codes and asked the researcher if this code block belonged to the movement category, and then the researcher helped him search. When he found the code block, he put that under the Events code block. Then, he read the next code block, ‘wait 1 second’. The researcher directed him to the control category to help him find the relevant code category by explaining that the ‘wait 1 second’ code block allows us to control the game. After he found the code block, he placed it under the ‘change size by 10’ code block. Then, again, he read the following code block, ‘next costume’. Later, the researcher reminded him that the costume is related to the sprite's appearance and explained the location of the Looks code category based on the Motion code category. Afterward, he dragged the ‘next decor’ code block onto the screen. After cautioning by saying, “But ‘next decor’ says here”, he found the correct code block and placed it under the ‘wait 1 second’ code block. Then, he executed the code block. He described his observations with the words, “It got bigger. It changed color.”, which was true. While converting the code from the first image to the code in the second image, he almost rewrote the entire command sequence from scratch. While constructing the code process, he found and arranged the code blocks one by one, following the sequence of commands in the desired order. His ability to place these code blocks in the requested order and interpret the differences in the command sequence was an important step in developing his knowledge of this concept. However, in marking the correct answer, he was very excited. When he read the first option, he wanted to mark it, but the researcher directed him to read the other

options. After that, he was indecisive between the first and second options. Later, the researcher read the two options to him again, and he marked the correct option.

The Events and Sequence Test is crucial for assessing the understanding of this concept. It is comprised of four questions focused on sequences with a total maximum score of 14. Fatih scored 11 on this concept. One of his mistakes is finding the last executed ‘say’ code block, the second question of the Events and Sequences Test. Despite successfully arranging the complexly ordered code blocks into the specified order, he struggled to find the correct response. The researcher explained the question during the review and circled the right answer. Due to time constraints, Fatih completed the test at home, resulting in delayed feedback. Had he done the test in the classroom, he might have sought assistance and potentially identified the correct choice. The other mistakes stem from the sentence structure in the last items of the 6 (See Figure 7) and 7b (See Figure 8) questions. While he wrote full sentences in the first and second items of the 6 and 7b questions, he did not do the same thing in the last items of those questions. He only wrote the correct parameters and the code blocks, like ‘move 20 steps’ and ‘next costume’. Since he did not complete his sentences, he got partial credits for correctly writing the parameters.

6. Yeşil bayrağa tıklandığında kukla ne yapacak?



İlk olarak, _____

Sonra, _____

Son olarak, _____

Figure 7. The sixth question from the Events and Sequences Test

7.

Soru 7a ve 7b soruları aşağıdaki komut dizisi ile ilgili.



7a. Komut dizisini çalıştırmak için ne yapmak lazım?

- A. Yeşil bayrağa tıklamak
- B. Kuklaya tıklamak
- C. Boşluk tuşuna basmak

7b. Komut dizisi çalıştığında kukla ne yapar?

İlk olarak, _____

Sonra, _____

Figure 8. The seventh question from the Events and Sequences Test

The development of the knowledge of the computational concept of sequences can also be monitored by the seventh question's first three items of the Parallelism and Loops Test (See Figure 9). Since this question requires students to write what happens when the loop code block is executed in sequential order, the answer he gave enabled the researcher to monitor the development of this concept after three lessons from the Events and Sequence Test. As in the Events and Sequence Test's answers to 6 (See Figure 7) and 7b (See Figure 8) questions, he wrote true parameters in each item of the seventh question of the Parallelism and Loops Test. Although his mistakes about the sentence structure were only in the last items of 6 (See Figure 7) and 7b (See Figure 8) of the Events and Sequence Test, the first three items of the seventh question (See Figure 9) of the Parallelism and Loops Test were in all items of the questions about this concept. So, he gained partial credit for

writing the correct parameters. It can be interpreted as he knows the concept of sequences, but his ability to write full sentences has declined.



7. Bu komut dizisinin kuklaya ne yaptıracağını açıkla.

Döngü her çalıştığında:

İlk olarak, _____

Sonra, _____

Son olarak, _____

Bunların hepsini _____ kez yapar.

Figure 9. The seventh question from the Parallelism and Loops Test

Overall, Fatih's knowledge of the sequences has developed as he progressed in the curriculum. Constructing a connection between the positions of the code blocks and the Sequence concept was helpful for him in learning this concept.

4.1.1.4 Fatih's development of the knowledge of the computational thinking concept of parallelism

One of the computational thinking concepts the program covers is parallelism, in which more than one sequence of commands can be executed simultaneously. This concept was initially introduced in the fourth lesson after studying the Events and Sequences concepts as they are prerequisite knowledge for comprehending it.

In the fourth lesson's activity (See Figure 10), there was a race between a rabbit and a hedgehog, and each sprite had more than one code command that

included different Events code blocks. The rabbit moves forward when 'T' is pressed on the keyboard; the hedgehog moves forward when 'K' is pressed, and each of them has a code command that changes their positions to their initial position with a 'when green flag clicked' Events code block.

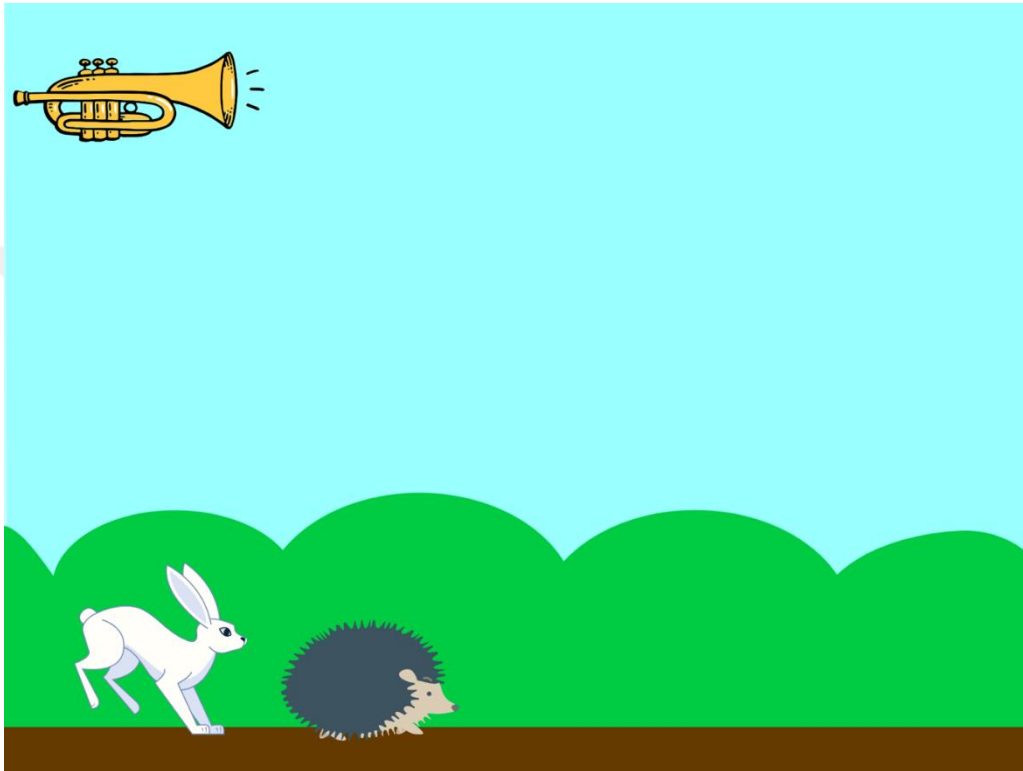


Figure 10. The screen of the fourth activity

When he started to analyze the project according to the worksheet, firstly, he only pressed the 'T' key, then only pressed the 'K' key. After changing their keys from the code upon his suggestion, he offered the researcher to press 8, and he to press 7. He realized that if we pressed those buttons at the same time, the sprites would move simultaneously and independently from each other. This realization was the first and the most important step in developing knowledge of the computational thinking concepts of parallelism. To ensure that he knew the reason for moving

simultaneously and independently from each other, the researcher showed him the code blocks in the examination of the Predict question's statement and emphasized their shared and distinct events code blocks of the code commands of the sprites.

In the fifth lesson, there was a Ballerina who danced when the space key was pressed, and music played when the green flag was clicked (See Figure 11).



Figure 11. The screen of the fifth activity

In the fifth TIPP&SEE worksheet's first question (See Figure 12), which is an Observe question, he marked a wrong choice. When he started the project, he did what the Ballerina said and pressed the space button. Thus, he heard music playing and did not comprehend which action was triggered by clicking on the green flag and which was triggered by pressing the space key, as the actions happened

simultaneously. After immediate feedback was provided by the researcher that involved restarting the game, this conversation happened:

"Researcher: Look, my question was, 'when I clicked the green flag,' but here we also marked 'dancing.' Did it dance when I clicked the green flag? Let's look at it again together."

F: No, it didn't. Press the space key. How about we try something like this? For instance, let's make it dance right away. Pressing the space key... I'm pressing it. There's no sound. It comes later." (Fifth Lesson Interview, 20.11.2023)

1. Yeşil bayrağa tıkladığımda



Balerin

Kostüm değiştiriyor.	☐
Ses başlıyor.	☐
Konuşuyor.	☐
Bir şey değişmiyor.	☐
Dans ediyor.	☐

Figure 12. The first question of the fifth TIPP&SEE worksheet

Trying to restart the game and examining the actions enabled him to reach the correct answer and realize the actions' triggers.

Fatih's knowledge about the computational thinking concepts of parallelism can be the Parallelism and Loops Test's 6a and 6b questions (See Figure 13). In the Parallelism and Loops Test, Fatih answered all questions related to the concept of parallelism correctly. Then the researcher explained it to him, she said, "The dog has two green flag commands, so when I click the green flag, will both of them happen simultaneously?" he interrupted the researcher and said, "No." Later, the researcher continued her question: "Will they happen at different times?" Then he interrupted the researcher again and said, "Yes, they happen at the same time. Because why? Both of them have a green flag." From this, one can conclude that Fatih perceives the concept of parallelism as code blocks starting with the same event that will run simultaneously.

6.



Kedi'nin
Kodları



Köpek'in
Kodları



6a. Yuvarlak içine al: Yeşil bayrağa tıklandığında Kedi ne yapacak?

- A. Kedi 7 kez "miyav" der, sonra 4 kez kostüm değiştirir.
- B. Kedi "miyav" der ve aynı anda kostüm değiştirir.

6b. Yuvarlak içine al: Yeşil bayrağa tıklandığında Köpek ne yapacak?

- A. Köpek 7 kez "hav" der, sonra 4 kez kostüm değiştirir.
- B. Köpek "hav" der ve aynı anda kostüm değiştirir.

Figure 13. The sixth question of the Parallelism and Loops Test

The concept of parallelism was introduced in the fourth lesson. His knowledge of parallelism was expanded in terms of specifying the triggerings of the actions by restarting the project. For the first play, he could not identify the triggering of specific actions that happened at the same time. He developed a strategy himself to differentiate the actions' triggering, which was restarting the project and looking at the codes of the sprite.

4.1.1.5 Fatih's development of the knowledge of the computational thinking concept of loops

The concept of loops is about the code blocks that are repeated in a sequence. Thus, knowing the concept of sequences is essential for developing knowledge of the concept of loops. Since the computational thinking concept of loops was first introduced in the fifth lesson, the development of the knowledge of this concept was examined through the fifth and sixth TIPP&SEE worksheets and the Parallelism and Loops Test.

Besides its relation with the concept of the sequences, he had the knowledge of some code blocks' values and comprehended the significance of these values can change. He demonstrated this by manipulating the code block with white spaces in it. This knowledge helped him to develop the knowledge of the concept of loops by using Scratch as the 'repeat 10' Loops code block requires a value from the user. This knowledge was not developed only while working with the loops concept because the change in the value inside code blocks is not just a feature of loop code blocks. Code blocks belonging to categories such as Motion and Looks can also have values that can be changed (See the examples in Figure 14). Not only this information but also understanding what the values are for is significant for

developing knowledge on this concept. Fatih acquired and applied this knowledge in the fourth lesson and demonstrated a consistent application in subsequent lessons.

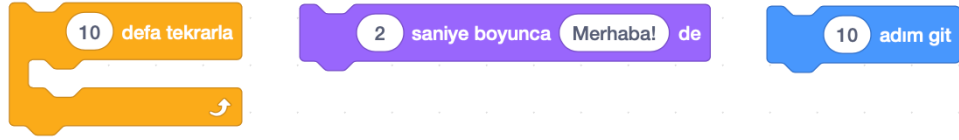


Figure 14. Example code blocks whose values' can be changed

Fatih's comprehension of this concept was notably influenced by his proactive approach to using the computer independently, such as pressing buttons or making changes to code blocks, allowing him to gain practical experience. Through his active role in the learning process, he experimented to identify which code blocks accepted values and accurately predicted the outcomes of these inputs. Importantly, he independently applied this knowledge beyond worksheet questions, creating scenarios and devising solutions.

He explained one of his ideas while studying the fourth worksheet's Predict question. This question (See Figure 15) was about finding the code block that makes the rabbit move. When he found the correct answer, which was 'move 10 steps', he started to explain his thought by saying, "For example, if we make this (10) 100, we can go faster" (mesela oraya 100 yaparsak daha hızlı gidebiliriz) (Fourth Lesson Interview, 18.11.2023). He also predicted the outcome of the value's changing. Having and being able to apply this knowledge had an impact on the development of knowledge of the computational thinking concept of loops.

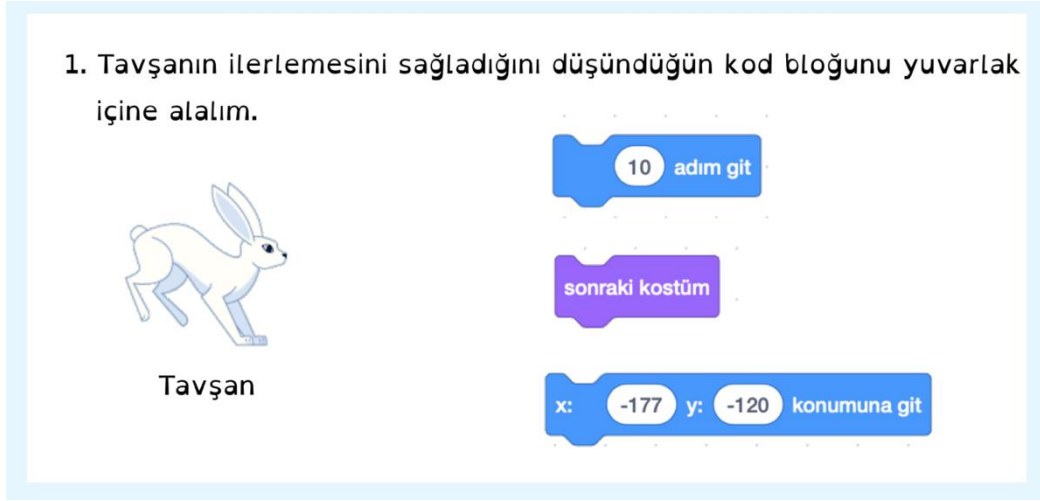


Figure 15. The predict question from the fourth TIPP&SEE worksheet

The Parallelism and Loops Test evaluates the knowledge of both concepts of parallelism and loops. The total score for the loops concept is 21, and Fatih got 16. Fatih's mistakes in the test questions related to this concept primarily appear in two distinct ways. The first is incorrect sentence structure, similar to the concept of the sequences, for which he got partial credit. The second incorrectly identifies the expanded, i.e., long version (See Figure 16) of a loop code command from the options.

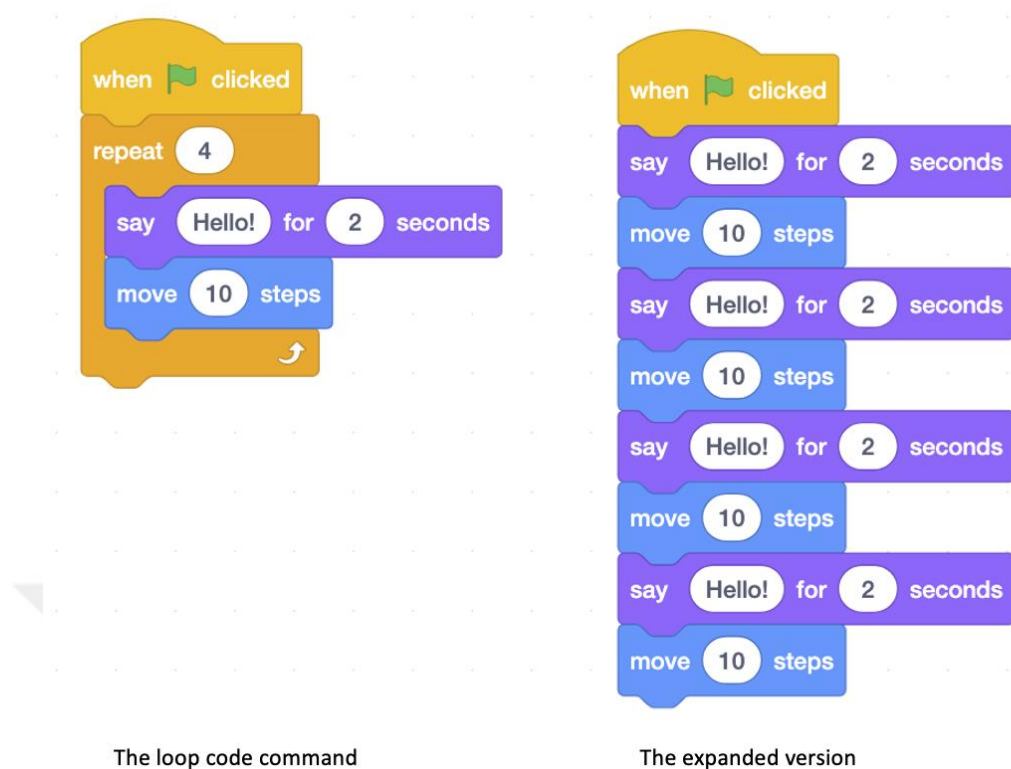


Figure 16. The example of an expanded version of a loop code command

He successfully answered the questions that required specifying how many times a loop code block would repeat by looking at the loop code block; he did not miss any. Although he correctly indicated how many times a loop would repeat, he was not successful in choosing the command sequence that would achieve the same without a loop code block, the expanded version. For instance, in the second test question (See Figure 17), he stated that the given command sequence would repeat four times; he did not choose the right option. Instead of choosing the option where the code blocks inside the loop code block were repeated four times, he initially selected the option with only the code blocks inside the loop code block, totaling two lines of code. After immediate feedback, he then chose the answer with the code blocks inside the loop code block repeated twice, resulting in a four-line command sequence.



2. Kuklanın yukarıdaki döngüyle aynı şeyi yapmasını sağlayan komut dizisini **yuvarlak** içine al.



Figure 17. The second question of the Parallelism and Loops Test

A similar situation occurred in the fourth question of the test (See Figure 18). In this question, he correctly expressed the number of repetitions in the loop code block, which is three, but chose the code blocks inside the loop code block from the options. In the second question, although he selected a command sequence with the number of repetitions, perhaps he couldn't make such a mark because there might not be a three-line command sequence in this question. Therefore, it cannot be determined whether he constructed a relationship between the number of repetitions and the number of lines of code.



4. Kuklanın yukarıdaki döngüyle aynı şeyi yapmasını sağlayan komut dizisini yuvarlak içine al.



Figure 18. The fourth question of the Parallelism and Loops Test

The test data that ensures he correctly expresses the number of repetitions of a loop code block is the first question (See Figure 19) and the seventh question's fourth item (See Figure 20). These questions aim to measure the student's ability to find the number of repetitions of a loop code block. Fatih's correct answers to these two questions support that he possesses this knowledge of the concept of loops.



1. Döngüdeki bloklar kaç kez tekrar eder?

Figure 19. The first question of the Parallelism and Loops Test



7. Bu komut dizisinin kuklaya ne yaptıracağını açıkla.

Döngü her çalıştığında:

İlk olarak, _____

Sonra, _____

Son olarak, _____

Bunların hepsini _____ kez yapar.

Figure 20. The seventh question of the Parallelism and Loops Test

The test also includes an open-ended question about this concept, and it asks, "How do you know when you need to use a loop?" Since we had limited time to conclude the lesson, Fatih answered this question orally rather than in writing. He said, "Now, teacher, a loop, we write something, for example, we write twelve, it repeats all of it twelve times and gathers the crowd." (The Parallelism and Loops Test interview, 21.11.2023). This saying can be interpreted as he perceived the computational concept of loops as a concept that enables writing long code blocks in a shorter form, thus emphasizing the importance of composing a shorter code block rather than a long and "crowded" one.

Fatih developed his knowledge of the concept of loops in terms of indicating the number of repetitions and when loops were used thanks to knowing some code blocks' values can be changed, comprehending the significance of these values, and his knowledge of the sequences concept of computational thinking.

4.1.2 The second case: Kaan

4.1.2.1 Description of the second case

Kaan is an eleven-year-old boy who was diagnosed in 2016 with a specific learning disability (with a tendency to dyslexia) and epilepsy, with a 30% disability rate. For his epilepsy, Kaan is prescribed a medication called Keppra. The education coordinator informed the researcher about Kaan's condition. Following epileptic seizures, Kaan may experience memory loss, and the frequency of these seizures is unstable. The most recent episode occurred about a month before our classes started, and he was at school then. Subsequently, there was no neurologist in their district, so they sought treatment at a private hospital in a nearby metropolitan city. Now, his treatment will be monitored by the doctor there. The coordinator also mentioned that he might have an epileptic attack in classes with the researcher and could forget about previous lessons and their coverage. Overall, during the data collection process, neither in our classes nor at school or home did Kaan experience any epileptic seizures.

He is attending one of the public schools in this district and is enrolled in 5th grade. He took a compulsory Computer Science class in school. While the Ministry of Education's ICT curricula for the fifth grade focuses on digital citizenship, ethics, spreadsheet programs, and debugging and programming, Kaan's school lacks computers. Therefore, the subjects that did not require technological devices were taught to students. Even though the curricula include debugging and programming, the students could not teach these topics. So, Kaan had no prior training in coding instruction and had not heard of and used Scratch.

He also attends special education at the A Special Education and Rehabilitation Center. His mother is a janitor in the A special education and

rehabilitation center. Since he was diagnosed with epilepsy, his mother does not want to leave him alone in the home, so Kaan goes there after school and on Saturdays. He does homework, sleeps in an empty classroom, or drinks tea with the other institution staff. All the teachers there know Kaan and are aware of his diagnosis. The hours for Kaan's classes at the institution are not predetermined. Kaan takes classes there based on the availability of the teachers at that time. Thanks to his flexibility in being present in the special education and rehabilitation center, there were no time issues during the lessons. This prevented any situations like the lesson being cut short or not completing a test and not getting immediate feedback.

Kaan had problems with writing throughout the study. Occasionally, he refused to write, or his writing had no space between words. Mostly, his writing did not answer the question. He also struggled to read, especially from the computer. His reading was more fluent in the worksheets. When he had difficulty reading or if the text was longer than a line, instead of reading the rest of the sentence, he generally used a word that he knew or a word with similar pronunciation or completed the word from his mind.

He has no personal cell phone, tablet, or computer. He said “Well, teacher, we had an old computer. My mother bought it and we broke it. We also had a small tablet like this that my auntie gave me. My teacher, it broke too. So I don't have a tablet or phone.” (Introduction interview, 13.11.2023). This situation may have influenced his skills in using those devices. In class, he was very hesitant about using the computer. Since his technological belongings broke, he might also feel anxious about breaking mine. When it came to tasks such as clicking buttons, dragging and dropping code blocks onto the coding screen, zooming in on the screen, and making changes in the code blocks, he would step back and ask the researcher to do it; he

passively participated in the classes. Although he occasionally changed some code blocks himself, such situations were rare.

I also gave Kaan a green pencil and told him that we could keep track of the number of classes through stickers. Like Fatih, Kaan started forgetting his pen after the third class. However, to cover up this situation, Kaan claimed to have bought a new eraser and offered to bring it to our future classes. Unfortunately, he only brought the eraser to the fourth class and forgot to bring it to the others. In the last lesson, the researcher put the other stickers she had promised him on the back of the certificate the researcher gave him, as she did for Fatih.

When Kaan heard that the researcher started to have classes with Fatih, too, he started to ask questions about the lessons with Fatih. He was curious about how Fatih did in the classes, what we covered in those lessons, and whether we concluded the lessons with Kaan or not since the researcher studied with Fatih. He was relieved when he heard the lessons with him would continue. However, he often asked questions about Fatih's lessons, and he wondered which one was better at class, him or Fatih. He said they met years ago, and they were classmates until this semester.

4.1.2.2 Kaan's development of the knowledge of the computational thinking concept of events

Events, as one of the computational thinking concepts, is about triggering the code blocks. His events concept knowledge development can be monitored through the program since all the lessons have covered this concept.

Kaan moved along the worksheets' questions, and he did not ask further questions. Thus, his knowledge development of this concept can only be monitored

by his answers to those questions in the worksheets and the Events and Sequences Test.

Like in the researcher's Session Journal notes on Kaan, he had difficulty expressing his observations in words. This resulted in some mistakes in the questions about the concept of events since the questions related to this concept required him to make selections about what the sprites do when a specific action is taken in worksheets. However, he realized that each worksheet had a question about clicking on the flag or pressing some keys and the results of that. Owing to that, he knew what should be done to solve those questions.

In the first worksheet, he struggled to point out the change in the sprite's costume. This struggle can be seen in the different lessons like in the fourth and sixth lessons. In the first lesson, I am introducing myself activity, he did not realize that the sprite changed the costume. In the fourth activity, The Race of Rabbit and Hedgehog, he did not observe the movement in the feet of the animals, so he did not observe that the triggering caused a change in the costumes. In the sixth lesson, the Abilities of the Wizard activity, he did not infer the movement in the arms of the wizard as a costume change. This situation can be named a misconception about sprite's costumes. According to the researcher's Session Journal notes on Kaan, he perceived the costume of a sprite as only a change in the color of the sprite, not the movement of the arms or legs or rotation of the sprite. However, in the Dancing Ballerina activity, which was the fifth lesson, he named costume change by looking at the actions made by the sprite when triggering occurred. The structure of the fourth worksheet might cause this inconsistency. This activity and the worksheet differed from others regarding the number of sprites. While the others had two or more sprites, the costumes of the sprites were only introduced orally and assisted by

a computer. However, as this activity had only one sprite, the sprite's costume was introduced both orally and assisted by computer and also in paper format (See Figure 21). Presenting a sprite's costumes through multiple means enhanced his understanding of it. But this understanding did not embark on the other lessons.



Figure 21. The introduction of costumes of a sprite from the fifth TIPP&SEE worksheet

The other mistakes that Kaan made in the worksheets can be seen in the second worksheet, the Finding Nemo activity. The question (See Figure 22) asked the students to select the options and the consequences of clicking the green flag. One of the options was about swimming. Since the sprites of the activity were fish, he marked this incorrect option.

1. Yeşil bayrağa tıkladığımda





Dory

Nemo

Konuşuyor.		
Yüzüyor.		

Figure 22. The first question of the second TIPP&SEE worksheet

In the first question of the third worksheet (See Figure 23), the Name Animation activity, he poorly observed the changes by clicking the green flag, although the researcher navigated him by examining the sprites' actions one by one. However, Kaan missed that the sprite L changed color when the green flag was clicked.

1. Yeşil bayrağa tıkladığımda







A

S

L

I

Renk değiştiriyor.				
Büyüyor.				
Dönüyor.				

Figure 23. The first question of the third TIPP&SEE worksheet

The development of knowledge in this concept can be evaluated by the student's overall score in the Events and Sequences Test's events questions. This test covered five questions specifically focused on the Events concept, with a maximum score of 5. Kaan gained a score of 5 by marking all the questions correctly.

When his achievements on the worksheets and the test were compared, his mistakes in the worksheets might have been caused by the checkbox table question form or the crowdedness of the page. Since he did not make any inferences, making interpretations based on his knowledge development of the computational thinking concept of events is hard and not observable. He showed his knowledge with his answers to the questions on the test and worksheet questions.

4.1.2.3 Kaan's development of the knowledge of the computational thinking concept of sequences

The concept of sequences of computational thinking is covered in the order of the code blocks executed by the computer. This concept was first introduced in the activity of Finding Nemo, then Name Animation in the curriculum. In the other lessons, this concept was studied and assessed by both tests, and it is significant to indicate which code block is executed in which order.

When he met the first question related to this concept, which was in the worksheet for the Finding Nemo activity, he was asked to solve a sequence issue by changing the positions of the code blocks. The researcher encouraged him to solve the question by reading and explaining questions and also what the problem is. The researcher put the code blocks separately in the code screen and explained each code block's aims. Then, the researcher wanted him to navigate her to replace the code blocks. The solution process went like this: firstly, the researcher told him what he

should do with the sprite, and then Kaan showed the code block that made the sprite do that action. On the first try, he correctly navigated the researcher and solved this question with this method. After that, since this question also required the student to write the solution in his own words, he expected to write the solution or his thinking process while solving it. However, Kaan wrote an answer out of context. The researcher tried to motivate him to write an answer related to the solution, but he refused to write, even though the researcher summarized the solving process.

His passive involvement in the class and being distracted influenced the development of the computational thinking concept of sequences. In the TIPP&SEE worksheet of the Name Animation task, the third lesson interview, a question (See Figure 24) required him to change the code command's sequential order and create a new one. He did not make the changes himself, even though the researcher encouraged him. Then, the researcher made all the changes and asked him what happened with these changes. He said, "That's correct, teacher." (Third Lesson Interview, 15.11.2023), which was not the answer; yet the researcher asked him twice, and he insisted on his answer. He might be distracted or could not put his observations into words. Then, the researcher offered to examine code blocks to find the answer. The researcher explained how each code block works and read each option to him. He did not pick any of them; then she suggested that he observe the sprite when the researcher clicked the green flag. He only realized that the sprite got bigger, and when the researcher read the options again, he selected the option that he heard 'gets bigger' at first. This shows that Kaan cannot distinguish what changes when he compares two code commands that contain the same code blocks in different orders on the paper, nor when he makes the necessary changes on the computer screen and starts the game.

1. L harfinin kodlarını, yani 1 numaralı görseldeki kodları 2. görseldeki kodlar haline getirdiğinde ne olur?



- ☐ L harfinin önce rengi değişir. 1 saniye sonra büyür.
- ☐ L harfi önce büyür. 1 saniye sonra rengi değişir.
- ☐ Bir değişiklik olmaz.

Figure 24. The question from the third TIPP&SEE worksheet

The Events and Sequence Test is for evaluating the understanding of this concept and has four questions focused on this concept with a total maximum score of 14. Kaan scored 1 on this concept. Unlike Fatih, he completed the test during class hours since there were no time constraints. Thus, he provided immediate feedback when he needed it.

His correct answer was to the second question of the test, which is about finding the last executed 'say' code block in a line of commands. Since his false answers to questions 6 (See Figure 25) and 7b's items (See Figure 26) were required to write the sequential order of the executed code blocks, it cannot be said that he knew which code block was executed last.

6. Yeşil bayrağa tıklandığında kukla ne yapacak?



İlk olarak, _____

Sonra, _____

Son olarak, _____

Figure 25. The sixth question from the Events and Sequences Test

7.

Soru 7a ve 7b soruları aşağıdaki komut dizisi ile ilgili.



7a. Komut dizisini çalıştırmak için ne yapmak lazım?

- A. Yeşil bayrağa tıklamak
- B. Kuklaya tıklamak
- C. Boşluk tuşuna basmak

7b. Komut dizisi çalıştığında kukla ne yapar?

İlk olarak, _____

Sonra, _____

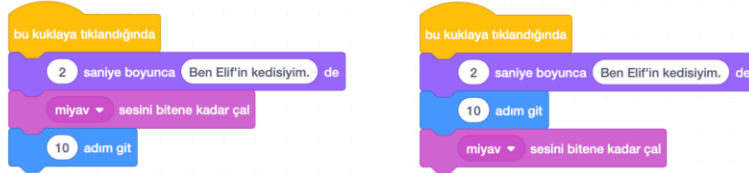
Son olarak, _____

Figure 26. The seventh question from the Events and Sequences Test

The other mistake he made was the fifth question (See Figure 27). This question emphasized the importance of different sequential orders causing different results by requiring students to compare two code commands. However, Kaan did not show any evidence of the importance of order. When the question required him to compare

two command sequences containing the same actions but in different order, he argued that since the actions were the same, the command sequences were also the same.

Aşağıdaki iki komut dizisini karşılaştır.



5. Doğru olanı **yuvarlak** içine al.

- A. Farklı eylemlerde bulunurlar.
- B. Aynı eylemleri farklı bir sırada yaparlar.
- C. Fark yoktur.

Figure 27. The fifth question of the Events and Sequences Test

The knowledge of the sequences concept development can be tracked not only by the Events and Sequences Test but also by the Parallelism and Loops Test, owing to its seventh question's first three items (See Figure 28). These items are related to writing the sequential order of the executed given loop code block. He answered correctly only the third item of the seventh Parallelism and Loops Test question, the correct parameters, and sentence structure. The other related items regarding the sentence structure and the parameters were answered incorrectly.



7. Bu komut dizisinin kuklaya ne yaptıracağını açıkla.

Döngü her çalıştığında:

İlk olarak, _____

Sonra, _____

Son olarak, _____

Bunların hepsini _____ kez yapar.

Figure 28. The seventh question from the Parallelism and Loops Test

To develop Kaan's knowledge of the computational thinking concept of sequences, the researcher tried different methods, like separating the code blocks and making him direct the researcher to put those. This method worked well on the second worksheet, but he did not apply this strategy on his own, so his score on the test was

1. It can be said that his development was minimal without any help or feedback.

4.1.2.4 Kaan's development of the knowledge of the computational thinking concept of parallelism

Parallelism, one of the computational thinking concepts, emphasizes that more than one sequence of commands can be executed simultaneously. In this study, this concept was handled for the first time in the fourth lesson after studying the events and sequences concepts. The knowledge development on this concept can be seen in both worksheets and the Parallelism and Loops Test.

In the worksheets, Kaan did not develop his knowledge. In the fourth lesson, Kaan did not show any interest in the reasons for moving the sprites at the same time.

He did not even try to press the keys 'T' and 'K,' which were the buttons to move the sprites. It can be explained that he did not think of this situation. In the fifth lesson, he moved on with the worksheet's questions, so he was not confused about which action was triggered by which button or key.

His knowledge of this concept can be seen in the Parallelism and Loops Test. In the test, there was only two questions related to this concept, the 6a and 6b questions (See Figure 29). He asked for help in solving 6a and 6b and did 6a correct and 6b false.

6.



Kedi'nin
Kodları





Köpek'in
Kodları



6a. Yuvarlak içine al: Yeşil bayrağa tıklandığında Kedi ne yapacak?
A. Kedi 7 kez "miyav" der, sonra 4 kez kostüm değiştirir.
B. Kedi "miyav" der ve aynı anda kostüm değiştirir.

6b. Yuvarlak içine al: Yeşil bayrağa tıklandığında Köpek ne yapacak?
A. Köpek 7 kez "hav" der, sonra 4 kez kostüm değiştirir.
B. Köpek "hav" der ve aynı anda kostüm değiştirir.

Figure 29. The sixth question of the Parallelism and Loops Test

His mistake in 6b, the researcher read the for him and showed the related code blocks, then she asked, "When I click on the green flag, how does this work?". He replied, "It moves, teacher." This response was completely wrong, as the code blocks did not involve any motion code blocks. Therefore, this answer may indicate that he

did not thoroughly examine the code blocks and provided a memorized response.

When the researcher realized this, she asked, “Which one makes it move?” After the researcher’s question, he started examining the code blocks and corrected his saying,

“I mean, it barks teacher.” (The Parallelism and Loops Test Interview, 22.11.2023).

He still could not provide the correct response despite attempting to correct his answer. This is because he had only examined the code block begins with the left one of the ‘when green flag is clicked’ of the sprite. Then, the researcher guided him to examine the other code command, “What else can it do?”, but he did not examine it and said the code block under the one that makes it bark, ‘wait 1 second’. Afterward, to ensure at least this code block was thoroughly examined, the researcher asked how many times the sprite does these actions, and he replied correctly.

To direct him to explore the other code command, she said, “The event code block of this is ‘when green flag clicked’ too when I click on the green flag, the dog will do this too.” and interpreted the code command to him. Then, posed a question to Kaan, “Which code command executed when I clicked on the green flag?” he replied incorrectly by showing the dog’s code block on the left and saying, “Since you put this first, this one will be executed first. Then the other will be executed.” His saying showed that he did not have the knowledge that the code blocks having the same Events code block will be executed simultaneously.

Lastly, the researcher read the question and the options; he chose the wrong option without thinking about or examining it. When he heard the “at the same time” from the correct option, he did not show any reactions. He selected the option that supported his idea that code commands have shared Events code blocks executed by the order of left to right. This can be explained by the fact that he was unaware of the computational thinking concept of parallelism.

Kaan's knowledge of the computational thinking concept of parallelism did not progress throughout the program. His solving process of the 6b question from the Parallelism and Loops Test enlightened the researcher to explore his understanding. In terms of this concept, he neither understood that having the same Events code block makes code commands executed simultaneously, nor he realized that a sprite could have more than one code command, which could be executed at the same time.

4.1.2.5 Kaan's development of the knowledge of the computational thinking concept of loops

The computational thinking concept of loops focuses on the code blocks that are repeated in a sequence. This concept was introduced in the fifth lesson at first, and the concept of sequences is important in the development of this concept.

Kaan had hardships in detecting what code blocks' value can be manipulated. Since some of the important loops code blocks in the Scratch, like 'repeat [...]', require a value from the user, understanding that the values of certain code blocks can be changed, and the significance of these values play a key role in developing the knowledge of the computational thinking concepts related to loops. While Kaan did not fully acquire this knowledge, his knowledge development of concept of loops had a different pace.

In the third lesson, when the researcher asked what the 'change size by [...]' code block does, he said, "Something else was replacing it" (onun yerini başka şey değiştiriyordu) (Third Lesson Interview, 15.11.2023). He knew this code block's numerical value could be changed, but he did not remember this information in another lesson. This lack of understanding may be attributed to his passive participation during the class, in contrast to Fatih's active engagement. If he were

active in the lessons, he would know the values' significance and aim and give more correct answers. For instance, he asserted that 10 in the 'change size by [...]' would make the sprite shorten the sprite, which is wrong. A similar situation happened in the loops code block. Consequently, there has been minimal development in Kaan's understanding of the concept of loops, resulting in errors in test questions.

The maximum score for the loops concept in the test is 21, and Kaan scored 12. Particularly in questions related to choosing the command sequence that achieves the same outcome without a loop code block, he did not earn any points.

His mistakes in questions about the number of repetitions of a loop code block, which are the first question and the fourth item of the seventh, were inconsistent; some were correct, and others were incorrect. Although he received credit for providing the correct answer to the first question, he received none for the fourth item in the seventh question. A similar inconsistency can be observed in the responses to the items of the seventh question, especially regarding sentence structure.



7. Bu komut dizisinin kuklaya ne yaptıracağını açıkla.
Döngü her çalıştığında:

İlk olarak, _____

Sonra, _____

Son olarak, _____

Bunların hepsini _____ kez yapar.

Figure 30. The seventh question from the Parallelism and Loops Test

The seventh question (See Figure 30) required him to describe sequentially what would happen each time the loop ran for the first three items, and the fourth item expected him to specify how many times each of these actions would occur, i.e., the number of repetitions. While responding to the first three items, Kaan made mistakes in both sentence structure and specifying the correct code block, the parameter, in the first two items. However, he succeeded in both sentence structure and specifying the correct code block in the third item, earning only points in the seventh question. As there were no time constraints in class with Kaan, he answered the open-ended question by writing. Although he mostly refused to write or wrote out of context, this was an improvement in terms of writing, especially considering it was the last lesson and the last activity. The motivation for his writing might be attributed to the nature of the test; perhaps he took the tests more seriously than the worksheets. The open-ended question was framed as "How do you know when to use a Loop?". His answer was, "It means to repeat, that is, to repeat. To repeat means to do something again. To repeat is a good and beautiful thing." (defa tekrarlamak yani. Tekrarlamak yani bişeyi yeniden yapmak demektir. Tekrarlamak iyi ve güzel bişeydir.) (The Parallelism and Loops Test Interview, 22.11.2023). It was positive to see that he established a connection between the computational concept of loops and "repeating," linking two related ideas. Since the Scratch Loops code blocks include the term "repeat," using Scratch likely helped him make these connections. However, he had difficulty clearly recognizing the need for a loop or identifying the circumstances in which loops are used. Therefore, his answer was not fully satisfactory.

His knowledge of the concept of loops was developed by making a connection between the word 'repeat' and this concept. He became aware of which

code blocks belonged to the loops concept, but it could not be interpreted as he knows how they work, as his answers were inconsistent through the data sources.

4.2 RQ1b) What are the differences and similarities between the two cases (Fatih and Kaan) in terms of developing the knowledge of computational thinking concepts of events, sequence, loops, and parallelism?

This part aims to compare students' knowledge development of the events, sequences, parallelism, and loops concepts of computational thinking, considering their scores on each concept on tests and worksheets.

Table 8 indicates the students' test scores regarding each computational thinking concept. Fatih outperformed Kaan, and the most significant score gap is seen in the concept of the sequence. While Kaan had the full score on the concept of the events, Fatih had a full score on the concept of parallelism, whereas Kaan had one point on this concept.

Table 8. Total Scores of the Students on the Computational Thinking Concepts of the Events, Sequences, Parallelism, and Loops and the Maximum Scores of Each

Student	Events	Sequences	Parallelism	Loops
Fatih	3/5	11/14	2/2	15/21
Kaan	5/5	1/14	1/2	11/21

Note. The Events scores were calculated by adding the students' points on 1, 3, 4a, 4b, and 7a questions of the events and Sequence Test, and the sequences scores were calculated by adding the students' points on 2, 5, 6, and 7b questions of the Events and Sequence Test. The parallelism scores were calculated by adding the students' points on the 6a and 6b questions of the Parallelism and Loops Test, and the loops scores were calculated by adding the students' points on 1, 2, 3, 4, 5a, 5b, 5c, and 7 questions of the Parallelism and Loops Test.

The comparison of students in terms of their knowledge development in the computational thinking concepts of events, sequences, parallelism, and loops could not be made by only looking at their scores on the tests; their successes on the worksheets and in-class activities should also be examined.

Table 9 points out the students' overall scores in each of the TIPP&SEE worksheets. Fatih outscored Kaan except for the first worksheet, in which both had the same score and on the fifth worksheet, Kaan had a higher score than Fatih.

Table 9. The Total Scores of the Students on the TIPP&SEE Worksheets and the Maximum Scores of Each

Student	TSW1	TSW2	TSW3	TSW4	TSW5	TSW6
Fatih	8/9	4/4	9/9	14/15	18/19	14/15
Kaan	8/9	3/4	8/9	13/15	19/19	12/15

Table 10, Table 11, Table 12, and Table 13 were created to make it easy to take a look at the students' knowledge development in each concept. These tables are created by combining the notes on the Session Journals, the in-class worksheet interviews, and the tests' outcomes.

Table 10. Student's Knowledge Development in Computational Thinking Concept of Events through the Program

Student	Before the instruction	After the instruction	Final Evaluation
Fatih	· Could not find correctly the position of an Events code block. · Could not show where to find an Events code block in the Scratch. · Confused about which triggering causes a specific action.	· Able to place the Events code block at the top of a code command. · Can show the Events code blocks category in the Scratch · Able to change the values of some Events code blocks and identify the restrictions on these changes	· Able to manipulate the Events code block whenever necessary.
Kaan	· Could not show where to find an Events code block in the Scratch. · Confused about which triggering causes a specific action.	· Able to indicate the specific triggering causes a specific action.	·

Table 11. Student's Knowledge Development in Computational Thinking Concept of Sequences through the Program

Student	Before the instruction	After the instruction	Final Evaluation
Fatih	Was not aware of that changing the code block's position matters.	Recognized that the code block's order in a code command means its order in execution.	Knows that the change in the position of a code block in a code command resulted in a change in the execution order of the actions
Kaan	Was not aware of that changing the code block's position matters	Could not indicated which code block in a code command will be executed in which order.	

Table 12. Student's Knowledge Development in Computational Thinking Concept of Parallelism through the Program

Student	Before the instruction	After the instruction	Final Evaluation
Fatih	· Could not indicate the Events code block of the actions that happened simultaneously. · Predicted that two different code commands can be executed at the same time. · Predicted that two different code commands that have the same Events code block can be executed simultaneously.	· Recognized that two different code commands that have the same Events code block can be executed simultaneously. · Recognized that two different code commands can be executed at the same time.	· Knows that if a sprite has more than one code command with the same Events code block, the sprite will do both code commands simultaneously.
Kaan	· Was not being aware that two different code commands can be executed at the same time. · Was not being aware that two different code commands that have the same Events code block can be executed simultaneously.	· Could not recognize that two different code commands can be executed at the same time. · Was not being aware that two different code commands that have the same Events code block can be executed simultaneously.	· Could not indicate that if a sprite has more than one code command with the same Events code block, the sprite will do both code commands at the same time.

Table 13. Student's Knowledge Development in Computational Thinking Concept of Loops through the Program

Student	Before the instruction	After the instruction	Final Evaluation
Fatih	· Could identify that the 'repeat 10' code block's value can be changed.	· Stated that the value in the 'repeat 10' is the repetition number of the code command. · Could not find the expanded version of a loop code block. · Can explain that when the Loops were used	· While he can correctly indicate the repetition number of a code block, he cannot identify the expanded version of a code block.
Kaan	· Could not state the value in the 'repeat 10' as the repetition number of the code command.	· Related the word 'repeat' and this concept. · Cannot find the expanded version of a loop code block. · Often stated that the value in the 'repeat 10' is the repetition number of the code command.	· Even though he rarely correctly indicates the repetition number of a code block, he cannot identify the expanded version of a code block

When the students' total scores on each computational thinking concept of events, sequences, parallelism, and loops in the tests and on each TIPP&SEE worksheet were compared, it can be said that Fatih usually had higher points than Kaan and showed more progress in terms of developing the knowledge of CT concepts. One can explain this difference by considering the different levels of engagement level in

class, having access to the technological devices at home or not, and having different disability rates.

4.3 RQ2a) How do 11-year-olds with specific learning disabilities respond to TIPP&SEE worksheets' Observe, Predict, Explore, and Debugging Questions?

This section aims to compare the students' general development in each worksheet question types of Observe, Predict, and Explore and their practice of Debugging.

Observe, Predict, and Explore Questions are polytomous items, so they can be scored. However, the Debugging Question is in an open-ended format, and no rubric was developed because Debugging questions are meant to examine students' debugging practices.

4.3.1 Comparison of students' overall development in terms of the Observe questions

Observe questions are the questions required to start the project, monitor the changes in the game screen, and select the correct options. This question type usually comprises polytomous items, primarily presented in a checkbox table format and occasionally in a multiple-choice form (See Figure 31 and Figure 32).

1. Yeşil bayrağa tıkladığımda







A
S
L
I

Renk değiştiriyor.	☐	☐	☐	☐
Büyüyor.	☐	☐	☐	☐
Dönüyor.	☐	☐	☐	☐

Figure 31. An observe question example from the third TIPP&SEE worksheet

2. Trompete tıkladığımda






Trompet
Tavşan
Kirpi

☐	Trompet yok oluyor.
☐	Ses çıkıyor.
☐	Kirpi hareket ediyor.

Figure 32. An observe question example from the fourth TIPP&SEE worksheet

Observe questions can be numerically scored. To compare students' development in Observe Questions, their scores of this question type can be helpful. Table 14 was generated to include the cumulative scores of the students' Observation Questions from the TIPP&SEE worksheets.

Table 14. The Total Scores of the Students on the Observe Questions in Each TIPP&SEE Worksheet and the Maximum Scores of Each

Student	TSW1	TSW2	TSW3	TSW4	TSW5	TSW6
Fatih	5/6	2/2	5/5	5/8	9/10	8/9
Kaan	5/6	1/2	4/5	4/8	6/10	7/9

They only received the same score in the Observe Questions of the first worksheet. In the second, third, fourth, and sixth worksheets, Fatih scored one point higher than Kaan. According to the researcher's Session Journal notes, Fatih successfully expressed his observations of the game screen in his own words. The following is a note from the journal. In the second question of today's worksheet (the sixth TIPP&SEE Worksheet) (See Figure 33), which is an observation question, the checkbox table format misled him. Despite making correct observations while expressing his observations of the game screen, he marked the wrong character and noticed and corrected his mistake. This could be because of the high number of characters and questions. (Sixth Session Journal, 21.11.2023).

2. Boşluk tuşuna bastığımda







	Büyücü	Büyücü'nün kızı	Kutup Ayıları	Gemi
Konuşuyor.	☐	☐	☐	☐
Kostüm değiştiriyor.	☐	☐	☐	☐
Büyüyor.	☐	☐	☐	☐
Küçülüyor.	☐	☐	☐	☐
Hareket ediyor.	☐	☐	☐	☐

Figure 33. The second question of the sixth TIPP&SEE worksheet

According to the Session Journal, Kaan often had difficulty in expressing his observations. The score differences on Observe questions might stem from Kaan's challenges in focusing on relevant information on the screen while Fatih competently expressed his observations.

Fatih generally outperformed Kaan's total score on the Observation Questions. Since Observe questions usually ask what happened when clicking on the green flag or pressing some keys, which are kind of an Events code block, this question type is related to the computational thinking concept of Events. However, there was a controversy in this situation. When the students' overall points for Observe questions and events concepts points from the Events and Sequence Test were compared, Kaan outperformed in the test questions related to events. This sheds light on the fact that Observe questions are not only about the concept of events and

emphasizes that the test itself cannot be the only thing that provides information about the development of the concept of events and that worksheets should also be given importance for concept development.

The largest point difference in the Observe Questions occurred in the fifth worksheet. Fatih scored 3 points higher than Kaan. The fifth worksheet was related to the activity introducing the concept of parallelism. Considering Kaan's development of the knowledge of the concept of parallelism, the reason for this point difference may be related to comprehending this concept. Also, the Parallelism and Loops Test data ensures this situation where Kaan got 0, and Fatih got 1 point out of 1 in terms of parallelism questions.

Fatih's misconception about the costume of a sprite detected via Observe questions. most of the TIPP&SEE worksheets had a question like 'when the green flag is clicked [sprite] changes costume'. He answered incorrectly at first, especially where the sprite does not change color. In the fourth activity's third (See Figure 34) and fourth (See Figure 35) Observe questions, he failed to indicate which sprites change costume. In this activity, he asserted that if the sprite did not change color, it did not change costume (Fourth Lesson Interview, 18.11.2023). Then, the researcher considered this as a misconception and tried to overcome it with explanations of what the costume of the sprite is and what color is a feature of a sprite by using the computer. These explanations and reminders were made in the other lessons.

3. T tuşuna bastığımda



Trompet

Tavşan

Kirpi

Hareket
ediyor.

☐☐☐

Kostüm
değiştiriyor.

☐☐☐

Figure 34. The third Observe question from the fourth TIPP&SEE worksheet

4. K tuşuna bastığımda



Trompet

Tavşan

Kirpi

Kostüm
değiştiriyor.

☐☐☐

Hareket
ediyor.

☐☐☐

Figure 35. The fourth Observe question from the fourth TIPP&SEE worksheet

Fatih's misconception was corrected through intervention, this can be seen in the last lesson. In the sixth activity's first Observe question (See Figure 36), he identified the sprite's (wizard) costume change without any help. He said, "The wizard moved his arms. Is this considered as a costume change, did it, teacher?" (Sixth Lesson Interview, 21.11.2023). Correctly indicating the costume change of the sprite without any guidance showed that he corrected his misconception and mastered that skill.

1. Yeşil bayrağa tıkladığımda







	Büyücü	Büyücü'nün kızı	Kutup Ayıları	Gemi
Konuşuyor.	☐	☐	☐	☐
Kostüm değiştiriyor.	☐	☐	☐	☐

Figure 36. The first Observe question from the sixth TIPP&SEE worksheet

4.3.2 Comparison of students' overall development in terms of Predict questions

Predict questions require students to examine the code blocks of the sprites and make selections among the options based on their predictions on which code block is responsible for a specific action.

Before moving on to the Predict questions on the worksheet, there is a statement (See Figure 37) that directs students to click on the 'See inside' button, then click on the sprites and find their Events code block. This statement is the same across

all the worksheets, and a worksheet has only one of those statements. With this statement, students are enabled to find the project's sprites and their event code blocks. Then, they find the opportunity to examine the code blocks of these sprites for the first time here. In this way, they can inspect the code blocks that the sprites have before moving on to predict questions (See Figure 38). So, they did not see the sprite's code blocks the first time in the worksheet.

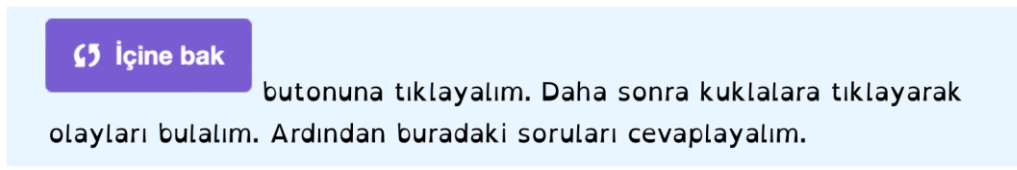
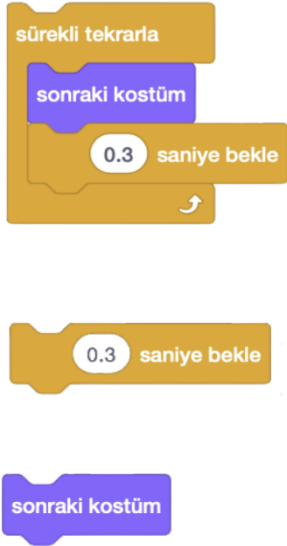


Figure 37. The statement before the Predict Questions

1. Balerinin birden fazla kez kostüm değiştirerek onun dans ediyormuş gibi olmasını sağladığını düşündüğün kod bloğunu yuvarlak içine alalım.



Balerin



The code blocks are as follows:

- A yellow "sürekli tekrarla" (Repeat Forever) loop block.
- Inside the loop, a purple "sonraki kostüm" (Next Costume) block.
- Below that, a yellow "0.3 saniye bekle" (Wait 0.3 seconds) block.
- Below the loop, another yellow "0.3 saniye bekle" (Wait 0.3 seconds) block.
- At the bottom, a purple "sonraki kostüm" (Next Costume) block.

Figure 38. A Predict Question example from the fifth TIPP&SEE worksheet

Since the Predict questions are in multiple-choice form, they can be numerically scored. Analyzing their scores for this question type can be beneficial to assess students' progress in Predict Questions. Table 15 was created to include the total scores of the students' Predict Questions from the TIPP&SEE worksheets.

Table 15. Total Scores of the Students on the Predict Questions in Each TIPP&SEE Worksheet and the Maximum Scores of Each

Student	TSW1	TSW2	TSW3	TSW4	TSW5	TSW6
Fatih	2/2	1/1	2/2	3/3	1/1	2/2
Kaan	2/2	1/1	2/2	3/3	1/1	2/2

This table indicates that Fatih and Kaan had the same score across all worksheets for the Predict question type. While score differences exist in the Observe questions, both students obtained the same score in the Predict questions. The reason for that might be the different forms of question format and their familiarity with those. The Observe questions mostly consisted of a checkbox table, while the Predict questions comprised multiple-choice forms. The students were more familiar with the multiple-choice format than the checkbox table, which was understood by their reactions to it. Both students asked the researcher how they would mark in the checkbox table in the first and second lessons, but they did not in multiple-choice formatted questions.

4.3.3 Comparison of students' overall development in terms of Explore questions

Explore questions require students to make changes in the code screen and mark the correct option upon these modifications. This question type consists of multiple-choice and checkbox table formats (See Figure 39 and Figure 40).

1.



'ın  kod bloğunu 5,  'nin

 kod bloğunu 20 ile doldurduğunda değiştirirsem ne olur?

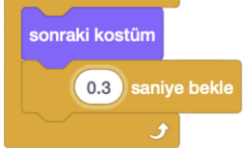
	Tavşan	Kirpi
Hızlanır.	☐	☐
Yavaşlar.	☐	☐

Figure 39. An example Explore Question from the fourth TIPP&SEE worksheet

2.



sürekli tekrarla



çıkarılırsa ne olur?

☐	Bir değişiklik olmaz.
☐	Boşluk tuşuna basıldığında Balerin bir kez kostüm değiştirir.
☐	Boşluk tuşuna basıldığında Balerin sürekli ve çok hızlı kostüm değiştirir.

Figure 40. An example Explore Question from the fifth TIPP&SEE worksheet

Like in the Predict question, there is a statement before the Explore questions (See Figure 41). However, these statements are not the same all over the worksheets, like in the Predict questions, since those statements are to indicate which code block will

be explored. Also, the number of those statements differs based on the number of to-be-explored code blocks, where the number of Predict question statements is one for each worksheet.



Figure 41. An example statement Before the Explore Question from the first TIPP&SEE worksheet

Explore questions can be scored numerically. To check students' development in Explore Questions, their scores of this specific question type can be revealing. Table 16 was drawn to include the sum scores of the students' Explore Questions from the TIPP&SEE worksheets.

Table 16. Total Scores of the Students on the Explore Questions in Each TIPP&SEE Worksheet and the Maximum Scores of Each

Student	TSW1	TSW2	TSW3	TSW4	TSW5	TSW6
Fatih	1/1	1/1	2/2	2/4	8/8	4/4
Kaan	1/1	1/1	2/2	2/4	8/8	3/4

The students shared the same points except for the sixth worksheet's Explore questions. Kaan scored one point less than Fatih on the Explore question, which focused on the concept of loops.

As Explore questions require making changes in the codes, active participation played a key role in monitoring students' progress on these questions. According to the Session Journal notes on Fatih, he independently made almost all

the code changes himself. He had hardships when finding the to-be-explored code block among the code categories. When his observations and the option overlapped, he marked the option. On the other hand, the Session Journal notes on Kaan emphasized that Kaan tended to step back or refuse when it came to changing the code blocks and asked the researcher to do it and read the options.

This question type encouraged Fatih to make more changes. In the Session Journal notes on Fatih, the researcher wrote:

The Explore questions in previous worksheets must have encouraged him because, even before reaching the Explore questions, he wanted to make some changes himself and see what would happen. To maintain his curiosity and interest, we occasionally tried his suggestions. Sometimes, he not only suggested changes but also directly predicted the outcome and wanted to try it out. In some cases, the changes he proposed were already present in subsequent questions. (Fifth Session Journal, 20.11.2023)

4.3.4 Comparison of students' overall development in terms of Debugging practice

Debugging practice is an essential skill to seek and find a solution to a problem. In the worksheets, except the first one, each of them has one question about Debugging practice. Those questions are in an open-ended form to obtain the students' own words for the solution. There is no rubric for evaluation because this study aims to explore understanding their way of solving the problem and thinking through the process.

Fatih's practice of Debugging developed gradually across the program. The first time he faced the Debugging question, he said, "There were no errors, teacher." (Second Lesson Interview, 16.11.2023) and the researcher explained where the error was. On the other hand, when he read the Debugging question of the last worksheet, he said he realized there was a problem. So, not only has his Debugging practice progressed, but also his ability to detect if there is a problem has expanded. His

answers to the Debugging questions mostly included the solution or the required changes of the code blocks to overcome the problem or inferences that he made through solving the problem. None of his answers were out of context or had wrong solutions. He double-checked his solutions by making changes in the codes and observing those. Some of his writings consisted of wrong-spelled words or incomplete sentence forms, but the contexts were related to his oral explanations of his solutions.

His answers to the Debugging questions listed in Table 17 and the wrong-spelled words are highlighted by asterisks.

Table 17. Fatih's Answers to the Debugging Questions

The TIPP&SEE Worksheet Number	Debugging Question	His answer
2	She (Dory) says 'I found you!' before even going to Nemo. How do you think we can fix this? You can write in the box below.	We change the position of the code block*. (Kod buluyuna yerini değiştirdik.)
3	What I want to do is, after clicking on the green flag, first change the costume, and then increase its size by 7 units. How do you think I can do that? You can write below.	When green flag* clicked/Next costume/Change size by 7 (Yeşil bayrı tıklıyınca/Sonraki kostüm/boyut 7 birim değiştir)
4	If I wanted the trumpet to speak 5 seconds after the rabbit and 3 seconds after the hedgehog, what would I do?	We wrote wait 6 seconds to the hedgehog. We wrote wait 9 seconds to the trumpet. We wrote wait 4 seconds to the rabbit.
5	How can we make the music start and the ballerina dance simultaneously? You can write below."	You should use the same code block* for the events I want to happen at the same time. (Aynı anda olması istediğim olaylar için aynı olan kodbilonu kullanmalısın.)
6	The last thing the wizard said also confused me. When we arrived at the wizard's house, polar bears came with us. How can we solve this? You can try your own solution and write it in the section below.	It happens when left key pressed (and) hide. (Sol ok tuşuna basılınca gizle oluyor.)

He developed strategies to tackle the problems through the study. Fatih used two main strategies when solving debugging questions. One of them is the trial and error method, which he often used when filling in the code blocks' value, such as 'wait for

a second'. His trying is like, "Let's try with 5. It did not. Let's try with 6. Yes, it does now.". Another strategy for him to find a solution was to restart the game, pause it when the error occurred, carefully observe the screen, examine the code blocks, and then provide a solution. These strategies and his active participation in class enabled him to find the correct solutions consistently. Also, he applied these strategies without the researcher's suggestion, which shows that he knew these before the instruction and applied them in this field, too. In the last lesson, he even found a solution the researcher had not considered. The sixth session journal notes on Fatih emphasized his development:

In fact, there was a light in him from the very first question. Over time, as he learned Scratch, this skill increased through practice, and in today's question, he found a solution that the researcher hadn't thought of. In the process of finding this solution, he was as active as in the others and used his strategies, like restarting the game. (Sixth Session Journal, 21.11.2023)

An instance of using his restart-the-game strategy worked well in the sixth lesson's activity, Abilities of the Wizard. Not only this strategy but also his familiarity with the Scratch and active involvement in the class made him solve the problem himself independently. The question (See Figure 42) asked how to hide the polar bears on the screen when transitioning from the Arctic decor to the wizard's house decor.

Büyücünün son dediği şey benim de kafamı karıştırdı. Büyücünün evine geldiğimizde kutup ayıları da bizimle geldi. Bunu nasıl çözebiliriz? Kendi çözümünü deneyip alttaki kısma yazabilirsin.

Figure 42. The Debugging Question from the sixth TIPP&SEE worksheet

First, he wanted to examine the code blocks and click the sprites. Then, he said, “Wait! I guess I caught something.” and restarted the game. Afterward, he asked the researcher to pause the game when he said ‘stop’. The conversation is an example of using his strategy to find a solution to the problem and his Debugging practice, and also, it was also evident in his active participation:

F: I will show something here. Teacher, you press, teacher. Now, there are arctics here, right?

A: Yes, there are arctics.

F: We pressed the left button.

A: Yes.

F: If we put left button control (he means ‘when left key pressed’ code block) of the arctics, (Kutupların kodlarına sol tuşunu kontrol dersek)

A: Yes.

F: And say hide there, what will happen?

A: Wow, you came up with a great idea. The polar bears, hmm let's look polar bears- There are when the decor is Arctics code block

F: There was a control, you know, when we pressed. (kontrol vardı ya hani bastığımızda)

A: Yes.

F: When we press, we show the polar bears, and when we say hide underneath, they disappear too. (Sixth Lesson Interview, 21.11.2023)

First, he explained how to get to this point by stopping the game where the error occurred, which was pressing the left arrow key, allowing the transition from the Arctic decor to the wizard's house decor. Then, he described his idea for the solution to the researcher. The solution included the idea of adding a new code block. He suggested adding the event code block after specifying the code block that triggers the transition to the wizard's house when the left arrow key is pressed. He used the wrong term at this point, mentioning a control code block instead of an event code block. Meanwhile, as the researcher did not have Fatih's solution in mind, she tried to guide him towards her own solution. Her solution was to hide the polar bears when the wizard's house decor was displayed. When Fatih noticed the researcher's effort to guide him, he directed the researcher back to his own solution, but again, he used the wrong term, saying Control instead of Events category. After placing the desired code block, he applied the targeted solution by putting the new code block he discovered in this worksheet, the 'hide' code block, under the events code block. To see if it worked, he restarted the game and realized that the new code block he wrote worked, solving the problem.

His efforts to handle the problem and create solutions for it demonstrated that his practice of Debugging has increased and progressed, though he used the wrong terms sometimes. This wrong use might stem from the excitement to express his

solution or confusion. However, his active engagement and willingness to express himself enabled the researcher to understand his thinking process in a Debugging practice, his strategies, and his application of those.

Kaan's development in debugging practice occurred more in terms of being able to write something related to the lesson rather than providing a solution to the questions. When Kaan was faced with a Debugging question type for the first time in the second lesson, where the sequences concept was introduced, his answer was totally out of context. This question required him to write the solution for the sequence issue, but his writing is like a letter to the teacher. He wrote his wishes for her. It can be understood that he lost his attention to the class or, was distracted, or did not comprehend what the question asked. When the researcher read his writing, she was shocked and offered him to write what he did to solve this question, but he refused to write. Then, the researcher offered to explain the solution process orally, but he refused again. He did not keep his attitude to writing in other lessons. However, his answers were related to the process of the Scratch project rather than what the question was asked.

Kaan's answers to the Debugging questions were mostly about how the game works and the sprites' actions throughout the game. None of his writings or oral expressions on those questions gave a clue about his thinking process for finding a solution or what the solution is. In the third session journal note on Kaan, the researcher noted: "He didn't make much effort to come up with a solution. When the researcher gave him a hint for the solution, he had reactions like 'What are we going to do now?' or 'What should we do now?'" This would be an outcome of his passive involvement in the lessons.

4.4 RQ2b) What are the differences and similarities between the two cases (Fatih and Kaan) in terms of their performance in the Observe, Predict, Explore, and Debugging Questions of the TIPP&SEE worksheets?

The scorable question types and students' points are given in Table 18. The students had the same points for Predict questions; Fatih got one point more than Kaan in Explore questions, which means he did all correctly and got three points more than Kaan in Observe questions. The most outstanding gap occurred in the Observe question type.

Table 18. The Total Points of the Students on The Scorable Questions from the Worksheets and the Maximum Scores of Each

Student	Total Score of Observe Questions	Total Score of Predict Questions	Total Score of Explore Questions
Fatih	36/40	11/11	21/21
Kaan	33/40	11/11	20/21

When the students compared their development in terms of the Observe question type, Fatih outscored Kaan in total. As indicated in the Session Journals of the researcher, Kaan had difficulties expressing his observations. Since this question type emphasizes what happens when an action is taken, like pressing the space key or clicking the green flag, and requires interpreting the game screen and marking the correct option upon these, being able to make and say observations is the key to success in Observe questions. The reason for Kaan's low performance in those questions than Fatih might be this. Another reason for mistakes in the Observe question might be the familiarity with the question format. As most of these questions are in the checkbox table, students are often confused in marking, especially if there are more than three sprites. Having different levels of development

in the concept of events and parallelism of the students may cause differences between the scores since the Observe questions are related to those concepts.

The students' development in Predict questions seems to be equal as they had the same score in each worksheet, so the total score and the maximum score as well.

When comparing students in terms of their development in the Explore question type, it can be said that their participation level in the class and their knowledge of the concept of loops may have an impact. Since the students are asked to modify the code blocks in Explore questions, their involvement in the classes is important to find the code blocks and make those changes with these. The one-point difference between the students' scores in this question type came from a question related to the concept of loops. The Parallelism and Loops Test's results for the loops concept verified this difference. Fatih had a higher score than Kaan, as in the total points for the Explore questions.

To compare students' development in Debugging practice, it can be observed that their levels of engagement with the class play a key role in improving this practice. As the level of engagement in the class increases, the development of Debugging practice becomes more visibly noticeable. Additionally, skills such as understanding the question and expressing observations in words, which are essential in answering every question, can be considered prerequisites for developing this practice.

All in all, in terms of scorable question types, Fatih got higher points than Kaan in Observe and Explore questions while they had equal scores on Predict questions. The biggest score gap occurred in Observe questions, which might be affected by Kaan's low performance in expressing his observations in words or his attention to the class.

Their Debugging practice developed to various degrees. Even though Kaan refused to write the first debugging question, he improved his performance by writing the flow of the Scratch project in the later lessons. His writing did not involve the solution of the question or his thinking process for solving the question, but at least he tried to write related to the question, even if it was not the solution. Fatih developed his practice of Debugging by applying some strategies that he had, such as trial and error or pausing the activity and examining the codes.

4.5 Summary of the findings

Throughout the study, Fatih actively participated in the classes by creating scenarios and devising solutions or predicting the outcomes of these scenarios. At the same time, Kaan moved along with the worksheet and test questions only.

Both students needed to be constantly reminded of the importance of the order of an Events code block in a code command throughout the program and provided immediate feedback and probes to which triggering causes which actions. If the recall included the student's analogy, the comprehension of this knowledge was enabled and become more meaningful for the student.

Regarding the events concept knowledge development, Kaan outscored Fatih in the test, whereas his engagement with the class was minimal. Both worksheet and test scores should be compared to evaluate students' computational thinking concept development, especially the events concept. There might be a controversial situation, such as in this study, where the score could not be parallel. The Observe questions in the worksheets are mostly related to the computational thinking concept of events, so while assessing students' knowledge development of the concept of the events, scores

on the Observe questions and events scores on the Events and Sequences Test should be considered when using TIPP&SEE strategy.

The students' knowledge development in sequences concepts has differed. Fatih acquired the knowledge that the position of the code block in a code command change means that the sequential order of the executed code blocks will change too. Kaan did not demonstrate that he had such knowledge even though the researcher adopted various teaching strategies.

In terms of the parallelism concept of computational thinking, Fatih developed his knowledge of simultaneously executed code commands and had difficulties in differentiating the triggering of a specific action, while Kaan could specify the triggering but had hardships in understanding two code commands that have the same Events code block executed at the same time.

The knowledge development of the loops concept of the students varied. Fatih successfully identified the number of repetitions of a loop code block, but Kaan did not do that consistently. In these instances, knowing that some code blocks can be manipulated, what this manipulation caused or gets input from the user, and the meaning of this input in that context played a key role. However, correctly indicating the repetition number of a code block did not provide evidence for finding the expanded version of a loop code command. While Kaan failed to indicate the repetition number occasionally, he found the expanded version of a loop code command, as Furkan could not.

The students had equal and maximum scores on Predict questions while they differed in the other question types of the worksheets, Observe and Explore. The most significant gap in the worksheet scores occurred in the Observe questions, which Fatih outscored by three points. Faith had a maximum score on the Explore

question, whereas Kaan had one point lower than him. The other question type in the worksheet was Debugging, which was in an open-ended form. The students varied in writing; Fatih's answers were about the solution, but Kaan's were about what happened through the activity and never included any part of the solution.

There were differences among Kaan's achievements on the worksheets and tests. One of the reasons for this variation in his performance could be the administration formats of the tests and worksheets. While the worksheets were implemented by both using the computer to see the Scratch activity and the worksheet in paper format, the tests were administered only in paper format.

The two cases demonstrated misconceptions about the sprite's costume at different times during the study. Both cases first perceived the costume of a sprite as a change in their colors. The researcher employed various strategies for an intervention to overcome their misconceptions. The interventions covered reminders and explanations for the sprite's features, and the recalls' language was kept simple and consistent. The recalls were assisted by a computer to promote its efficiency by visualizing the sprite's features. While the intervention helped Fatih to correct the misconception, it did not work for Kaan.

CHAPTER 5

DISCUSSION

This study is conducted to propose and examine the effectiveness of a mini-specialized computational thinking curriculum for Turkish students with specific learning disabilities that can be delivered in a technology-supported environment and understand their development of conceptual knowledge of the computational thinking process. Thus, this section aims to connect the research questions and findings, examine practical implications, make suggestions for further studies, and state the limitations.

5.1 Development of the knowledge of the computational thinking concept of events

Learning the concept of events was not linear; some points needed to be recalled during the study, like the position of an Events code block in a code command and where to find the Events code category in Scratch. Students often needed to remember the fact that the first ordered code block in a code command should be an Events code block. Constantly reminding students of information at different times as feedback helped them progress and acquire this knowledge. In contrast, Salac and her colleagues (2021a) asserted that reminding of domain-specific terms to students might result in a load on students' working memory and long-term retrieval. In addition, Israel and her colleagues (2015) advocated that the feedback's language should be clear and if the feedback involves a definition of terms, this definition should be consistent all over the instruction. In this study, the researcher tried to be

consistent about the definitions and explanations of the terms and observed that this has been helpful for the students.

Besides using reminders, the researcher used the analogy that participant's told in former lessons to recall the position of an Events code block. Israel and her colleagues (2015) point out the significance of providing students with different support structures, for guided practice. Fatih had associated the code blocks with puzzle pieces. When Fatih dragged the Events code block under a code block, he realized that it did not stick. Then, the researcher exemplified this situation as the indentation of the puzzle pieces must be overlapped to stick. After this explanation, Fatih acknowledged that the Events code blocks must be placed at the top of a code command to put the other code blocks under it.

Also, the students have differed in finding the triggering of a specific action. While Kaan could specify the Events code block that caused an action among the options, Fatih needed guidance or feedback. Researchers indeed stated that providing immediate feedback in computing education when adopting an explicit instruction approach is important to lessen the chance of making mistakes (Archer & Hughes, 2010; Israel et al., 2015)

5.2 Development of the knowledge of the computational thinking concept of sequences

The students did not develop their knowledge of the computational thinking concept of sequences at the same pace. While Fatih recognized that the code block order in a code command affects the outcomes, Kaan did not completely demonstrate this knowledge on the tests and worksheet questions. The curriculum materials consisted of different types of questions which enabled students to demonstrate their

knowledge as Israel and her colleagues (2015) identified as one of the key principles that can be applicable to computing education. Providing multiple methods of action helps the study to monitor students' knowledge development in the sequence concept to prevent mistakes stemming from the question type, like open-ended and multiple-choice formats. While some questions required students to write the sequential order of a given code command, another asked the student to select the correct option by comparing two code commands in terms of the order of code blocks' outcomes.

To emphasize the significance of the order of the code blocks, the researcher tried out different methods to achieve the UDL's first principle, which is Multiple Means of Representation (Israel et al., 2014), such as creating two code commands that included the same code blocks with different orders and executed and made the student write what happens in a sequential order and changing a code block's position and made the student observe the changes. While this strategy worked with Furkan, it did not affect Kaan's knowledge development in the sequence concept of computational thinking; he demonstrated low performance on sequences-related questions. At the same time, Fatih displayed his knowledge of this concept in both test and worksheet questions about sequences. He expanded his knowledge of this concept throughout the study and became known that the code block's order in a code command means its order in execution.

Since both tests that the study included as a data source had questions related to the computational thinking concept of sequences, the knowledge development of this concept can be observed whether it enhanced or not.

Overall, monitoring the knowledge development of sequences was enabled by adopting different teaching methods, having various questions, and evaluating at different times in this study. The learning pace differed among the students taught by

the same curriculum. Some reasons for this situation could be the students' engagement level in the class and having different disability rates.

5.3 Development of the knowledge of the computational thinking concept of parallelism

The students' knowledge development of the computational thinking concept of parallelism also varied. Fatih had difficulty differentiating the triggerings of the actions that happened simultaneously, but he overcame this hardship by trying to trigger at different times during the project. Also, he became capable of identifying that more than one code command can be executed simultaneously, whether they had the same Events code block or not at the end of the study. However, Kaan did not develop any of that knowledge. Kaan failed to indicate that the two code commands with the same Events code block would execute simultaneously; he asserted that the code command was placed to the left. This might be because of how reading in Turkish works, left to right.

5.4 Development of the knowledge of the computational thinking concept of loops

When the two cases were compared, having the knowledge of some code blocks' value and comprehending the significance of these values can change could be two of the most important prior knowledge for the development of the knowledge of the concept of loops, especially when using 'repeat [...]' loop code block in Scratch. It can be seen by the data of this study that having this prior knowledge had an impact on the students' knowledge development in the computational thinking concept of loops. Students who had this prior knowledge, like Fatih, may expand their knowledge of the concept of loops by consistently identifying the repetition number

of the code command by looking at the codes or changing the repetition number of the codes.

However, correctly indicating the loop code command's repetition number did not mean that the student could specify the expanded version of a loop code block. While Fatih successfully and consistently detected the repetition number by looking at the codes, he failed to find the expanded version of a code block among the options. More exercises about the extended version of a loop code block should be included in the programs to support students' knowledge development.

In knowing where to use loops, the students demonstrated their knowledge at different levels. While Fatih expressed the loops' aim by diminishing the crowdedness of a code command, Kaan displayed his knowledge by linking the word "repeat" and the concept of loops. Having different levels of primary knowledge on that some code blocks' can get input from the user and the meaning of the input might result in differences in knowledge development of the computational thinking concept of loops.

5.5 Development of the Performance in Observe, Predict, Explore Questions and Their Debugging Practice

Fatih and Kaan's performance in the worksheet's Predict question type was the same; they had equal scores. However, in the Observe question type, Fatih got three points higher than Kaan and one point more than in Explore questions.

Regarding their Debugging practice, their performance on those questions developed in different ways. Fatih's development was devising strategies to solve the problem, like stopping the project and inspecting the codes to handle the bug. Kaan's development of the Debugging practice was limited by his writing and the context of

the activity. While he first refused to write, in time, he wrote about what happened in the project but not any piece of the solution.

The cases demonstrated equal success on the Predict questions on the worksheets. The main difference between the Predict questions and the other question types of the worksheet is the format of the questions. While all the Predict questions are in multiple-choice format, the others are in a checkbox table, multiple-choice, and open-ended forms. This can be interpreted as the cases being familiar with the multiple-choice formats or providing them with different options might help to determine the answer. Also, multiple-choice questions may have eliminated the effects of having different disability rates.

Kaan's worksheet scores were better than his test scores. Since the implementation of the worksheets enabled him to try out the different scenarios that the question asked, he had a chance to manipulate the code blocks by using the computer and see the results of the executed code blocks. According to Chen, Lu, and Lien (2021), learning environments with different levels of technological engagement in the class materials show variations in student learning outcomes in fourth graders. In this study, worksheet administration had higher levels of technological engagement than the tests' administration since the tests did not require any computer usage other than the worksheets. So, the reason for the better performance on the worksheets than the test could be attributed to this. If the students' test performance is aimed to increase, the tests could be integrated into technology. In addition, higher LTE enhanced students' learning scientific knowledge in low achievers (Chen, Lu, & Lien, 2021). Providing students with more technology-integrated environments helps them extend their knowledge.

The implementation of the worksheets, providing the code blocks on both paper and the computer, enabled the provision of contextual cues to the students. The contextual cue might help them to link or remember the code block's features. However, in the test, he had no opportunity to test the code blocks to see the changes. For students like Kaan, who have a higher disability rate or a low level of engagement in class, the test's structure might be changed or integrated into the technology. So, students could try out the options, or see them in the computer environment, which might improve their test performance. Even though Öz and Özturan (2018) found out the test scores for participants who took the computer-based test and those who took the paper-based test did not differ significantly, they also said that computer-based tests could be an alternative to paper-pen tests. In addition, Nalbantoglu Yılmaz (2021) asserted that the difference between these two administration formats can be neglected. Although these studies did not suggest that computer-based or assisted tests predicted better learning outcomes, they did not say that those tests negatively influenced students' learning. In addition, the two studies mentioned above were not conducted with students with specific learning disabilities. To broaden the results, further study is needed.

5.6 The specialized curriculum effectiveness

The conceptual knowledge development of the computational thinking concepts of events, sequences, parallelism, and loops was promoted through a specialized curriculum that included six lessons that adopted the TIPP&SEE metacognitive strategy. This strategy provides scaffolding to elementary students for computer science learning by discovering the existing program in Scratch, a free block-based programming tool (Salac et. al., 2020). Brennan and Resnicks's (2012) definition of

the computational thinking concepts of events, sequences, parallelism, and loops are adopted in this study. The concepts of events, sequences, and loops were selected along with the literature suggested for beginner-level programming instruction. The computational thinking concept of parallelism was included in the curriculum by the recommendation of Salac and her colleagues (2020), who said that the instruction of this concept needs to be improved.

The curriculum was delivered in a technology-supported environment utilizing the combination of hardcopy format worksheets and Scratch, a block-based programming platform for free. Each worksheet was created for a specific project on Scratch. All the worksheets had the same layout, which was designed along with the four basic design principles: Proximity, Alignment, Repetition, and Contrast. The Proximity principle is obtained by grouping the related items, like presenting the sprites in a chunk rather than presenting them separately. The Alignment Principle is obtained by avoiding using more than one text alignment. The Repetition Principle is applied by framing each question type in different colors. By doing so, all the Observe questions were put in an orange colored box, the Predict questions in a blue, Explore questions in green and the Debugging questions in purple boxes; these were applied to all worksheets to sustain consistency.

The cognitivist instructional strategies are also used in the curriculum and its material development. One of these strategies is Advance Organizer from the Bridging Strategies. Except for the first lesson, all lessons started with an advance organizer to build a bridge between what is known and unknown. Another one is Rehearsal from General Purpose Strategies that facilitates the processing of information into short-term memory by keeping the material active. The Rehearsal strategy was applied through practicing and predicting with the Predict questions.

Also, Keller's ARCS Model of Motivation, stands for Attention, Relevance, Confidence, and Satisfaction, is adopted in developing activities and worksheet questions to foster the motivation of the cases. As the cases required to be active in their learning process via the activity and its worksheets, the Active Participation element in the Attention part. All the Debugging questions were posed in a surprising context, the Perceptual Arousal is obtained, too.

The worksheets were written in OpenDyslexic font, which was created to meet the needs of people with specific learning difficulties. The letter's size and the text's characteristics were determined along with the literature. Since Ismail and Jaafar (2011) asserted that if the font size is small, the reading would be difficult, and Evett and Brown (2005) suggested that the minimum font size should be twelve to prevent students from reading, the minimum font size determined as 13. Also, according to Ismail and Jaafar (2011), students with specific learning disabilities did not like crowded texts, so the texts were tried to be kept simple. The texts were in a bold format and put in a box, and the colors and the boxes were also white in contrast to highlight the texts, as Evett and Brown (2005) suggested.

The one of the most outstanding differences between the original version of a TIPP&SEE worksheet and the worksheets used in this study is the OpenDyslexic font written worksheets with a size of 13. In the original version, Comic Sans MS is used. This study only used this font in worksheets as it cannot be integrated into Scratch. This situation allowed us to compare the easiness of reading different fonts.

According to the researcher's Session Journals, this font utilization in the study enables the students to read the worksheets more fluently than the texts on the computer. However, the different formats of the texts might affect fluent reading in worksheets. The students were provided the OpenDyslexic font only in the

worksheets in a hard copy format and the other fonts in the computer as a soft copy format. Further research is needed to have more information on the effect of the font selection and the presentation format, such as having two groups of participants and providing one group with a worksheet written in OpenDyslexic font in a soft copy and the other group with a worksheet written in OpenDyslexic font in a hard copy.

Integrating fonts into the systems is essential to provide more user access. Scratch has various font selections (See Figure 43), which are Sans serif, Serif, Handwriting, Marker, Curly, and Pixel, only for creating costumes or decors, but the system itself has only one font. Since the data from this study revealed that the OpenDyslexic font made reading easier for students with specific learning disabilities, integrating this font into systems is essential to reach more users, especially a program such as Scratch that offers such a service free of charge to schoolers all around the world. If Scratch enables the user to add a new font or the OpenDyslexic font into the system, it makes the user read fluently and expands their target user.

Sans Serif

Serif

Handwriting

Marker

Curly

Pixel

中文

日本語

한국어

Figure 43. The font selections and their appearances in Scratch

The worksheets consisted of the same types of questions, which are Observe, Predict, Explore, and Debugging with different numbers, except for the first worksheet, which did not have a Debugging question. The worksheets had various questions in different formats: multiple-choice, checkbox tables, and open-ended questions. The Debugging questions were open-ended to enable the students to express themselves in their own words, which Kaan had struggles with writing. As a result, the student's knowledge development in computational concepts of events, sequences, parallelism, and loops can be monitored, as well as their Debugging practice.

The number of Explore questions in the worksheets determined the class hours. These questions took more time than the Observe and Predict questions. The researcher realized this after the fourth lesson with Kaan. Then, the researcher requested extra hours for the fourth, fifth, and sixth lessons from the institution. Thus, the number of Explore questions in the worksheets should be considered when estimating the class hours with students with specific learning disabilities while employing the TIPP&SEE strategy.

The students were faced with different questions they were unfamiliar with, such as checkbox tables. While filling these boxes, especially when there were more than two sprites and three questions, students needed clarification and needed help or intervention. In those times, the researcher provided immediate feedback by navigating the students to draw lines from the box to the sprite and the option. Thanks to this feedback, the expressions of students that they observed on the screen and their selected options were consistent.

Immediate feedback was provided not only in worksheets but also in the tests, which mostly covered multiple-choice questions. However, Fatih could not get immediate feedback for the Events and Sequence Test since there was no time to do

it during class hours, so he solved it alone at home. He did not have a chance to ask the researcher when he was struggling, but he put a sign on the questions he had difficulty with and asked the researcher in later class. When the researcher explained those questions, he found the correct answers. However, as Kaan had no issues with the time, he studied for the test in class and asked for help from the researcher when he had hardships. Eventually, he got total points from the events questions on the Events and Sequence Test, unlike Fatih. As Stephen and Rollings (1976) asserted in multiple-choice exam questions, students who received immediate and item-by-item feedback demonstrated better performance compared to those who didn't receive any feedback. This highlights the importance of timely and specific feedback in improving student learning outcomes.

Immediate feedback was used in the intervention to handle misconceptions about the costume of a sprite. As the cases sent signals for this misconception at different times, their intervention was provided right after their signals. Fatih's intervention mainly included explaining the sprite's features orally and assisted by a computer. Kaan's intervention was provided through the fifth worksheet. This worksheet presented the sprite costumes, and the researcher supported this introduction with her explanations. The cases also differed in terms of correcting their misconception. Fatih dispelled his misconception after the intervention he got, and he consistently correctly indicated whether a happening was considered a change in the costume of a sprite or not without any guidance. Although Kaan showed progress in the fifth lesson, he did not in the sixth. The differences between those lessons' worksheets were the number of sprites and their introduction details. The fifth one had only one sprite, and the worksheet covered the costumes of a sprite's introduction; the others did not. According to this study's findings, the sprite's

costume should be presented in various formats, such as in the digital environment, in paper format, and in oral explanations about its features in simple and consistent language to overcome this misconception in students with specific learning disabilities.

This curriculum differed from the Scratch Act1 and Scratch Act2 curricula from the related literature regarding adopted strategies and question types. The Scratch Act1 and Scratch Act2 were created by employing the Use-Modify-Create, 5E Instructional Model, and Universal Design for Learning strategies besides the TIPP&SEE strategy (ECforALL Curriculum, n.d.). Thus, the outcomes of using Scratch Act1 and Scratch Act2 curricula may represent the other strategies, not only the TIPP&SEE. This study's curriculum has adopted only the TIPP&SEE strategy to emphasize this particular strategy's effects on one's knowledge development of the computational thinking concepts of events, sequences, parallelism, and loops.

However, as the TIPP&SEE strategy is specifically created for the Use-Modify steps of the Use-Modify-Create pedagogy of computational thinking and the nature of this strategy, it does not cover how to create a new project. Since this study adopted this strategy as the main, the students had yet to experience the Create step of the Use-Modify-Create pedagogy.

From time to time, Fatih proposed opening an existing Scratch project and examining its codes to implement those in his project. The researcher explained to him that there was no time for doing this during class hours and suggested that he do what he offered at home. However, if Fatih had no personal technological devices at home, like Kaan, he would not have had an opportunity to do that. Regarding access to technological devices, the students should be encouraged to start a new project at the school or after-school programs where students can access those devices. Since

all students have no access, covering the Create step of the Use-Modify-Create pedagogy is significant.

Students who seek to explore the Scratch have a high engagement level in the class or are willing to create a unique project on their own, like Fatih, may not have a possibility to try it out during class hours if the only adopted strategy is TIPP&SEE. However, the students who are not curious about the subject, pay less attention, and are willing or have a low engagement level in class, like Kaan, may not feel a need to try to devise and create a new project by only adopting the TIPP&SEE strategy. To sustain the willingness of curious students to create new projects and arouse the other students' attention and willingness, adding another strategy to the curriculum for the Create step might be beneficial for the learning process besides TIPP&SEE.

In terms of the contexts of the project activities of the curriculum, they were created upon the researcher's experiences with teaching 9-11-year-old students and the Scratch Act1 and Scratch Act2. However, in the second activity, Finding Nemo, the researcher faced with a surprising situation. In this activity, there were two sprites, Nemo and Dory, and Dory trying to find Nemo, like in the movie. Her former students lived in a metropolitan city and had no specific learning disabilities, and all of them were informed about Nemo, the cartoon character. However, the participants of this study did not hear about Nemo. Thus, Dory's aim needed to be explained by the researcher; the activity itself did not evoke any ideas of Fatih and Kaan. When the participants did not recognize these sprites, the researcher briefly mentioned the movie and the project and used 'red fish' for Nemo and 'blue fish' for Dory. Thus, the context should be carefully chosen while conducting a study in rural or low-development areas, considering that participants may not know or have not heard

about what the researcher expected them to know. It makes the context more meaningful and accessible for the target group.

5.7 Practical implications and suggestions for further research

This study's qualitative data revealed that there were some points to improve in the curriculum besides its features that enhance students' learning process. Some of the features that need room for improvement stem from the curriculum or the nature of the adopted strategy. Also, the students' conditions affected their learning process. In light of this study's findings, practical implications and suggestions for educational designers, instructors with students with specific learning disabilities, policymakers, and future research ideas will be discussed in this part.

5.7.1 Practical implications for teachers

Since the students often sought reminders for the order of an Events code block in a code command, diagnostic assessments can be beneficial in determining whether the student knows or not. This knowledge is significant in indicating that the Events code blocks are positioned in a code command, and the relevant Events code block of action can be found on the screen to enhance students' process of events concept knowledge development. The recalls can be shaped according to the diagnostic assessment across the program. After the diagnostic assessment, the reminders' language should be consistent throughout the instruction. While some researchers advocated avoiding using domain-specific words, this study found that using those words consistently makes students with specific learning disabilities develop their knowledge of computational thinking concepts of events.

This study's findings supported the idea that using OpenDyslexic font in a paper format document eased the reading of students with specific learning disabilities. When the class materials were designed, teachers with students with specific learning disabilities could use this font with a minimum of 12 points to meet the needs of those students.

Teachers who teach students with specific learning disabilities should provide multiple means of representing a new learning subject and provide different types of questions to give students a chance to demonstrate their knowledge. If open-ended questions are selected to assess students' learning, students may refuse to write or get frustrated. At those times, students can be asked to answer the question orally.

5.7.2 Practical implications for curriculum designers

When preparing TIPP&SEE worksheets, font selection and the use of contrasting colors are important for readability. The findings of this study have shown that using the OpenDyslexic font with a minimum size of 13 points makes it easier to read compared to the computer's default font. Additionally, since it has been demonstrated that students are more successful with question formats they are familiar with, the selection of question types is emphasized. If, as in this study, checkbox tables are to be used, they should be prepared with a maximum of 3 sprites and 3 questions to prevent confusion. Furthermore, the findings of this study indicate that the number of Explore questions on the worksheets affects the duration of the lesson. Therefore, when preparing worksheets, the number of Explore questions should also be considered when determining lesson durations. Also, the context choice of the activities should be set considering the environment of the students to broaden the comprehensiveness.

Since the TIPP&SEE strategy does not cover the Create step of Use-Modify-Create pedagogy, students do not have an opportunity to build or plan a project from scratch. They cannot face bugs that they generate or choose the context of a project as they reuse and remix an existing one. Not covering the Create step also eliminates the chance to assess students' projects and building processes. If the goal of the curriculum includes seeing the students' projects, another strategy must be adopted besides TIPP&SEE.

5.7.3 Practical implications for policymakers

According to this study's findings, a specialized curriculum for students with specific disabilities enhances those students' knowledge development of computational thinking concepts. While many countries integrated computational thinking into computer science curricula, specialized computational thinking curriculum is needed to sustain equity in education.

A specialized curriculum integration might not be sufficient to maintain equity in education; the materials and tools must also be provided to follow that curriculum and foster student engagement with the technology.

Additionally, there is little research on developing specialized curricula for students with diverse needs. The government should support and sponsor this research field to develop curricula, investigate its effectiveness, do teacher training, and integrate it into computer science curricula.

a. Having a personal technological device

Access to technological devices is vital in our modern world, enabling people's lives since technology is everywhere. Having a personal technological device also

influences one's learning process by integrating technology into education. As this curriculum was provided in a technology-supported environment, prior knowledge of using technological devices affects and supports students' classroom engagement level and knowledge development.

Regarding having those devices, the students differed; Fatih has a personal cell phone and smart TV at home, but Kaan has no phone, tablet, or smart TV. The effects of this difference among the students on their learning process can be seen throughout the study in terms of their active participation and curiosity about Scratch and programming, as well as their knowledge development in computational thinking concepts of events, sequences, parallelism, and loops.

Fatih had an opportunity to explore Scratch himself outside the class and try out what he learned in different projects at home with his technological tools. Thus, his enthusiasm aroused and sustained, and he felt more comfortable using the researcher's laptop in class. His familiarity with those devices made him also familiar with the interfaces. He knew that dragging an object on the screen, clicking a button to see what happens, scrolling the page, and using the keyboard to make changes to the value of a code block. All these abilities enabled him to be active in his learning process. Thanks to his active participation in the lessons, his knowledge development in the computational thinking concepts of events, sequences, parallelism, and loops was enhanced.

Kaan hesitated to use the researcher's laptop during the classes; he mostly drew back when modifying the codes or clicking the buttons. His hesitation made him a passive learner in his learning process. He could not try what he learned in the classes at home because of poor access to technological devices. As a result, his

knowledge development in the computational thinking concepts of events, sequences, parallelism, and loops was minimal.

Students lacking technological devices, like Kaan, must be provided with the necessary technological support. Supporting those students with tablets or an environment with computers could be one of the solutions. Another idea to assist students without technological devices might be integrating unplugged programming into the extracurricular curriculum by utilizing a kit that looks like Scratch. The kit may be in the form of a puzzle and made of materials with velcro, so the students stick the pieces to a surface and place them under each other. Also, this material can be presented in different font options. In that case, this study's font and text selection findings can be beneficial and specific learning disability friendly. Thus, students can learn and develop their knowledge of computational thinking concepts by experience without computers or tablets.

5.7.4 Future research

In this study, the font used in the worksheets cannot be embedded into Scratch. So, the cases faced different fonts in different environments, such as paper-pen and computer environments, during the instruction. All the paper-based materials were written in OpenDyslexic font, except for the Scratch code blocks in the worksheets, and all the computerized materials, like Scratch activities, in different fonts. This situation affected students' reading and made this study fail to detect whether this font caused more fluent reading on paper or computer. It should be studied to determine how OpenDyslexic font influences one's reading fluency across the different levels of technological engagement environments.

5.7.5 Limitations

As this study only adopted the TIPP&SEE metacognitive learning strategy, the cases did not create a project themselves from scratch. Their knowledge developments on computational thinking concepts were only evaluated by tests and worksheets. If the students are asked to develop artifacts, the artifacts can be evaluated, and their knowledge development on those concepts and debugging processes can be monitored by different assessment techniques.

As this study was conducted during a semester break, the curriculum's effects on in-school and after-school activities were not observed. Also, since the cases did not participate in any school lessons during the study, the impact of learning from the school on this study's knowledge gain could not be monitored.

APPENDIX A

16 ELEMENTS OF EXPLICIT INSTRUCTION

#	16 Elements of Explicit Instruction	Explanations
1	Focus instruction on critical content.	Teach students skills, strategies, vocabulary terms, concepts, and rules that will empower them in the future while also meeting their educational needs.
2	Sequence skills logically.	Consider several curricular variables, such as teaching high-frequency skills before less frequent skills and ensuring mastery of prerequisites to a skill before teaching the skill itself.
3	Break down complex skills and strategies into smaller instructional units.	Teach in small steps. Segmenting the learning material into smaller units can be helpful in addressing the issues of cognitive overloading, processing demands, and the capacity of students' working memory.
4	Design organized and focused lessons.	Organized lessons are on-topic, well-sequenced, and contain no irrelevant digressions. So, the instructional time is used at optimum.
5	Begin lessons with a clear statement of the lesson's goals and expectations.	Tell students clearly what is to be learned and why it is important.
6	Review prior skills and knowledge before beginning instruction.	Make sure that all students have the prerequisite skills and knowledge to learn the skill being taught in the lesson. By doing so, an opportunity arises to connect the new skill and other related skills
7	Provide step-by-step demonstrations.	Modeling the skill, explaining the decision-making process, or performing the skill by thinking aloud are to show students a model of proficient performance.

8	Use clear and concise language.	The language should be consistent, with unambiguous wording and terminology.
9	Provide an adequate range of examples and non-examples.	Provide a wide range of examples and non-examples to help students understand the limits of when and when not to apply a skill, strategy, concept, or rule in an effective way
10	Provide guided and supported practice.	Adjusting the level of task difficulty and guidance helps to promote success and confidence. However, the level of guidance and task difficulty are in inverse ratio when the student shows success.
11	Require frequent responses.	Having students respond frequently helps them focus on the lesson content. Plan for a high level of student–teacher interaction via the use of question-and-end-of-the-year tests, quizzes, and other activities.
12	Monitor student performance closely.	Careful monitoring allows one to make timely adjustments if students are making mistakes or missing key points in a lesson. It also gives you the chance to provide them with feedback about how well they are doing.
13	Provide immediate affirmative and corrective feedback.	Follow up on students’ responses as quickly as you can. Providing immediate feedback to students about the accuracy of their responses helps ensure high success rates and decreases the likelihood of practicing errors.
14	Deliver the lesson at a brisk pace.	Instructions should be delivered at an appropriate pace to maximize instructional time and on-task behavior. The desired pace is neither too slow nor too fast for the students to keep up with.

15	Help students organize knowledge.	Since many students struggle to see how certain skills and concepts relate to one another, it is critical to employ teaching methods that make these connections more obvious or explicit. Students will find it easier to retrieve information and integrate it with new material if it is well-organized and connected.
16	Provide distributed and cumulative practice.	Distributed practice indicates multiple opportunities to practice a skill over time. Cumulative practice is a method of giving distributed practice that includes practice opportunities for both already and newly acquired skills. Provide learners with several practice efforts to overcome retention and automaticity concerns.

Note. Sixteen elements of explicit instruction are defined. Adapted from “Explicit Instruction: Effective and Efficient Teaching” (p. 2-3) by A. L. Archer and C. A. Hughes, 2010, Guilford Publications.

APPENDIX B

THE SPECIALIZED COMPUTATIONAL THINKING CURRICULUM

Lesson	CT Concepts	CT Practice	Instructional Objectives	Scratch Activity
First	· Events		· Create a Scratch account. · Reuse a Scratch project. · Use events. · Use event handlers. · Modify existing code blocks.	I am introducing myself
Second	· Events · Sequence	· Debugging	· Differentiate between the concepts of algorithm, program, and sequence. · Recognize how sequences are used in a Scratch program. · Recognize the importance of the order of blocks in Scratch. · Debug the codes.	Finding Nemo
Third	· Events · Sequence	· Debugging	· Use events. · Use event handlers. · Practice the importance of the order of blocks in Scratch. · Debug the codes.	Name Animation

Fourth	· Events · Parallelism	· Debugging	· Explain the meaning of parallelism in Scratch context. · Practice that the scripts run parallel by using the same event block. · Practice the synchronization between sprites is achieved by “wait 1 second“ code block. · Debug the codes.	The Race of Rabbit and Hedgehog
Fifth	· Events · Parallelism · Loops	· Debugging	· Define loops. · Give examples of loops. · Animate a sprite using a repeat loop. · Use multiple actions in a loop. · Compare scripts with and without loops. · Set the speed of the animation by using code blocks. · Use repeat blocks. · Debug the codes.	Dancing Ballerina
Sixth	· Events · Sequences · Parallelism · Loops	· Debugging	· Change the background by using code blocks. · Use repeat blocks. · Animate a sprite using a repeat loop. · Set the speed of the animation by using code blocks. · Debug the codes.	Abilities of Wizard

APPENDIX C

SESSION JOURNAL QUESTIONS FOR THE RESEARCHER

What did we cover in this lesson?

What do I, as a teacher, think that the things that the student understood in this lesson? What is the evidence for that?

Were there any parts of the class that the students struggled with? If yes, what are those points?

Were there any questions on the worksheet that the students struggled with? If yes, what are those questions?

Were there any things that unexpectedly happened during the lesson, and if so, what happened?

Were there any points where the student's attention was distracted, and if so, where were they?

APPENDIX D

SESSION JOURNAL QUESTIONS FOR THE RESEARCHER (TURKISH)

Bu derste neler işledik?

Ben bu derste öğretmen olarak öğrencinin neleri anladığını düşünüyorum?

Bunun göstergeleri neler?

Öğrencinin derste zorlandığı yerler oldu mu? Olduysa bunlar neler?

Çalışma kağıdı sorularında öğrencinin zorlandığı oldu mu? Olduysa bunlar hangileri?

Derste beklenmedik şekilde gerçekleşen bir şey oldu mu? Olduysa ne oldu?

Öğrencinin dikkarini dağıtan bir yer oldu mu? Olduysa nerelerde oldu?

APPENDIX E

INTRODUCTION INTERVIEW QUESTIONS

What is your name?

How old are you?

Which school are you going to?

Which grade are you in?

Do you have a computer or tablet of your own?

Do you like to play games?

What kind of games do you like?

What do you do when you encounter a problem?

Have you ever taken coding training before?

Have you ever used Scratch before?

APPENDIX F

INTRODUCTION INTERVIEW QUESTIONS (TURKISH)

Adın ne?

Kaç yaşındasın?

Hangi okula gidiyorsun?

Kaçıncı sınıfa gidiyorsun?

Kendine ait bir bilgisayarın veya tabletin var mı?

Oyun oynamayı sever misin?

Ne tarz oyunlar seversin?

Bir problemle karşılaştığında ne yaparsın?

Daha önce kodlama eğitimi aldın mı?

Daha önce Scratch kullandın mı?

APPENDIX G

THE EVENTS AND SEQUENCES TEST

Olaylar ve Diziler

Scratch Kullanıcı Adı: _____

Aşağıdaki komut dizileri, Kedi adlı bir kuklaya aittir:



1. Yuvarlak içine al: Kedi'ye "Merhaba!" dedirtmek için ne yapmalısın?

- A. Boşluk tuşuna basmak
- B. Yeşil bayrağa tıklamak
- C. Aşağı oka basmak
- D. Kuklaya tıklamak

2. En son çalıştırılacak De bloğunu yuvarlak içine alın.



3. Aşağıdaki komut dizileri bir kuklaya aittir. Kuklaya tıkladığınızda çalışan tüm komut dosyalarını yuvarlak içine alın.



Yeşil Bayrak'a tıkladığında, sahne şöyle görünür:



4a. Kelebek için çalışan komut dizisini **yuvarlak** içine alın.



4b. Köpek için çalışan komut dizisini **yuvarlak** içine alın.



Aşağıdaki iki komut dizisini karşılaştır.



5. Doğru olanı **yuvarlak** içine al.

- A. Farklı eylemlerde bulunurlar.
- B. Aynı eylemleri farklı bir sırada yaparlar.
- C. Fark yoktur.

6. soru için lütfen aşağıdaki boşlukları doldurunuz.

6. Yeşil bayrağa tıklandığında kukla ne yapacak?



İlk olarak, _____

Sonra, _____

Son olarak, _____

7a ve 7b soruları aşağıdaki komut dizisi hakkında sormaktadır.



7a. Komut dizisini çalıştırmak için ne yaparsın?

- A. Yeşil bayrağa tıklamak
- B. Kuklaya tıklamak
- C. Boşluk tuşuna basmak

7b. Komut dizisi çalıştığında kukla ne yapar?

İlk olarak, _____

Sonra, _____

Son olarak, _____

APPENDIX H

THE PARALLELISM AND LOOPS TEST

Paralellik ve Döngüler

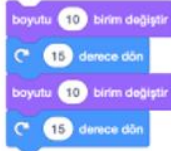
Scratch Kullanıcı Adı: _____



1. Döngüdeki bloklar kaç kez tekrar edecek?

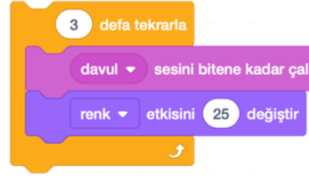


2. Kuklanın yukarıdaki döngüyle aynı şeyi yapmasını sağlayan komut dizisini yuvarlak içine alın.



3. Kuklanın tam olarak 3 kez kostüm değiştirmesini sağlayan **TÜM** komut dizilerini yuvarlak içine alın.





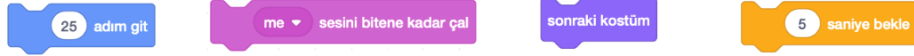
4. Kuklanın yukarıdaki döngüyle aynı şeyi yapmasını sağlayan komut dizisini **yuvarlak** içine al.



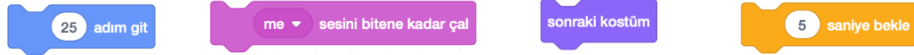
5.



5a. Yukarıdaki komut dizisinde 6 kez çalışan **TÜM** bloklarını **yuvarlak** içine al.



5b. Yukarıdaki komut dizisinde tekrar döngüsünden önce çalışan **TÜM** blokları **yuvarlak** içine al.



5c. Yukarıdaki komut dizisinde tekrar döngüsünden sonra çalışan **TÜM** blokları **yuvarlak** içine al.

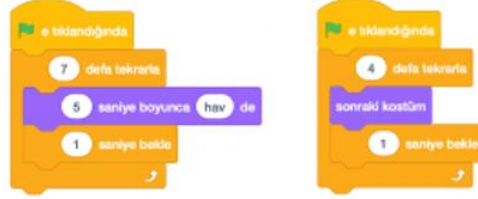




Kedi'nin
Kodları



Köpek'in
Kodları



6a. Yuvarlak içine al: Yeşil bayrağa tıklandığında Kedi ne yapacak?

- A. Kedi 7 kez "miyav" der, sonra 4 kez kostüm değiştirir.
- B. Kedi "miyav" der ve aynı anda kostüm değiştirir.

6b. Yuvarlak içine al: Yeşil bayrağa tıklandığında Köpek ne yapacak?

- A. Köpek 7 kez "hav" der, sonra 4 kez kostüm değiştirir.
- B. Köpek "hav" der ve aynı anda kostüm değiştirir.



7. Bu komut dizisinin kuklaya ne yaptıracağını açıklayın.

Döngü her çalıştığında:

İlk olarak, _____

Sonra, _____

Son olarak, _____

Bunların hepsini _____ kez yapar.

Daha fazla soru bir sonraki sayfada!

8. Döngü kullanman gerektiğini nasıl anlarsın?



Ekstra Zorluk: "vak" sesi kaç kez çalacak?

APPENDIX I

THE TIPP&SEE WORKSHEET EXAMPLE

• İSİM ANİMASYONU

İSİM:

1. Projenin **başlığını** okuyalım.

2. Bu projeyi **nasıl** kullanacağımızı öğrenmek için projenin **yönergelerini** okuyalım.

3. Bu projeden öğreneceğimiz **yetenekleri** öğrenmek için “**Amaç**” kısmını okuyalım.

Hadi şimdi aşağıdaki yönergeleri takip edelim ve gördüğümüz eylemlere göre işaretleme yapalım.

Kuklalarımızla tanışalım.



A



S



L



I

1. Yeşil bayrağa tıkladığımda



A



S



L



I

Renk
değiştiriyor.

☐
☐
☐
☐

Büyüyor.

☐
☐
☐
☐

Dönüyor.

☐
☐
☐
☐

İçine bak

butonuna tıklayalım. Daha sonra kuklalara tıklayarak olayları bulalım. Ardından buradaki soruları cevaplayalım.

1. L harfinin büyümesini sağladığını düşündüğün kod bloğunu yuvarlak içine alalım.



L

sonraki kostüm

1 saniye bekle

boyutu 10 birim değiştir

1. L harfinin kodlarını, yani 1 numaralı görseldeki kodları 2. görseldeki kodlar haline getirdiğinde ne olur?



1



2

☐

L harfinin önce rengi değişir. 1 saniye sonra büyür.

☐

L harfi önce büyür. 1 saniye sonra rengi değişir.

☐

Bir değişiklik olmaz.

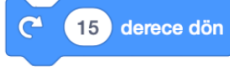
15 derece dön

kod bloğunu kullanarak keşfedelim.

1.

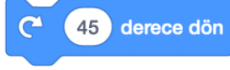


harfinin

15 derece dön

kod bloğunu 45 ile

doldurduğunda, yani

45 derece dön

haline getirdiğinde ne olur?

☐

I harfi döner ve yandaki gibi olur.

☐

I harfi döner ve yandaki gibi olur.

☐

I harfi yandaki gibi olur.



bu kuklaya tıklandığında ve Pop sesini bitene kadar çal

kullanarak keşfedelim.

1.



harfine

bu kuklaya tıklandığında Hey sesini bitene kadar çal

kod bloklarını

eklediğinde ne olur?

☐

Yeşil bayrağa tıklandığında A harfi hem renk değiştirir hem de ses çalmaya başlar.

☐

Klavyeden A tuşuna basıldığında ses çalmaya başlar.

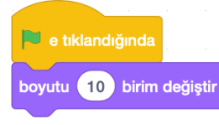
☐

A harfi kuklasına tıklandığında ses çalmaya başlar.

1. Şu an fark ettim ki bu oyunda S harfi için eksik ve hatalı kod yazmışım. O yüzden istediğim gibi çalışmıyor. Yapmak istediğim şeyi sana anlatayım, sen de bana ne yapmam gerektiğini anlat. Bunun için oyun kodlarında değişiklik yapabilirsin.



S kuklası



S kuklasının kod blokları

- Yapmak istediğim şey S harfinin yeşil bayrağa tıklandıktan sonra önce kostüm değiştirmesini, daha sonra ise boyutunun 7 birim artmasıydı. Sence bunu nasıl yapabilirim? Aşağıya yazabilirsin.

APPENDIX J

ETHICS COMMITTEE APPROVAL

Evrak Tarih ve Sayısı: 12.04.2023-122519

T.C.
BOĞAZİÇİ ÜNİVERSİTESİ
SOSYAL VE BEŞERİ BİLİMLER YÜKSEK LİSANS VE DOKTORA TEZLERİ ETİK İNCELEME
KOMİSYONU
TOPLANTI KARAR TUTANAĞI

Toplantı Sayısı : 43
Toplantı Tarihi : 12.04.2023
Toplantı Saati : 14:00
Toplantı Yeri : Zoom Sanal Toplantı
Bulunanlar : Prof. Dr. Feyza Çorapçı, Doç.Dr. Arhan S. Ertan, Dr. Öğr. Üyesi Yasemin Sohtorik İlkmen,
Dr. Öğr. Üyesi Ayşegül Metindoğan
Bulunmayanlar : Doç. Dr. Senem Yıldız

Mukaddes Aslı Pehlivan
Bilgisayar ve Eğitim Teknolojileri Eğitimi Bölümü

Sayın Araştırmacı,

"Examining the Development of Knowledge of Computational Thinking Concepts of Students with Specific Learning Disabilities: A Case Study" başlıklı projeniz ile ilgili olarak yaptığımız SBB-EAK 2023/30 sayılı başvuru komisyonumuz tarafından 12 Nisan 2023 tarihli toplantıda incelenmiş ve uygun bulunmuştur.

Bu karar üyelerin toplantıya çevrimiçi olarak katılımı ve oy birliği ile alınmıştır. Onay mektubu üye ve raportör olarak Yasemin Sohtorik İlkmen tarafından toplantıya katılan bütün üyeler adına e-imzalanmıştır.

Saygılarımızla, bilgilerinizi rica ederiz.

Dr. Öğr. Üyesi Yasemin
SOHTORİK İLKMEN
Öğretim Üyesi

e-imzalıdır
Dr. Öğr. Üyesi Yasemin Sohtorik
İlkmen
Öğretim Üyesi
Raportör

SOBETİK 43 12.04.2023

Bu belge, güvenli elektronik imza ile imzalanmıştır.

APPENDIX K
CONSENT FORM

KATILIMCI BİLGİ ve ONAM FORMU

Araştırmayı destekleyen kurum: Boğaziçi Üniversitesi

Araştırmanın adı: Özgül Öğrenme Güçlüğü Olan Öğrencilerin Bilgi İşlemsel
Düşünmeye Dair Kavram Bilgisi Gelişiminin İncelenmesi: Bir Vaka Çalışması

Proje Yürütücüsü: Prof. Dr. Diler Öner

E-mail adresi:

Telefonu:

Araştırmacının adı: Mukaddes Aslı Pehlivan

E-mail adresi:

Telefonu:

Sayın Veli,

Boğaziçi Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü öğretim
üyesi Prof. Dr. Diler Öner “Özgül Öğrenme Güçlüğü Olan Öğrencilerin Bilgi
İşlemsel Düşünmeye Dair Kavram Bilgisi Gelişiminin İncelenmesi: Bir Vaka
Çalışması” başlıklı bir bilimsel araştırma projesi yürütmektedir. Bu çalışmanın amacı
özgül öğrenme güçlüğü olan ortaokul öğrencilerinin bilgi işlemsel düşünme
kavramları bilgisinin gelişimini incelemektir. Çalışma sonunda öğrencilere
kodlamanın temel kavramlarının öğretilmesi ve öğrencilerin bilgi işlemsel düşünme
becerilerinin geliştirilmesi hedeflenmektedir. Özel eğitim merkezi müdürünüz
kurumun bu çalışmaya katılması için izin verdi. Bu araştırmaya yardımcı olmanız
için siz velilerin öğrencilerini projemize davet ediyoruz ve araştırmaya katılması için

onayınızı bekliyorum. Kararınızdan önce araştırma hakkında sizi bilgilendirmek istiyorum. Bu bilgileri okuduktan sonra araştırmaya katılmak isterseniz lütfen bu formu imzalayıp kapalı bir zarf içinde bize ulaştırınız.

Bu araştırmaya velisi olduğunuz öğrencinin katılmasını kabul ettiğiniz takdirde öğrenci ile yüz yüze gerçekleşecek altı özel ders işleyeceğiz. Derslerin başlamasından üç gün önce bir tanışma görüşmesi, derslerin bitiminden üç gün sonra da program sonu görüşmesi yapacağız. Bu altı ders esnasında ise bilgisayar, İnternet bağlantısı, çalışma kağıdı, kalem ve silgi kullanacağız ve her dersin sonunda öğrenci ile dersin değerlendirmesine yönelik birer görüşme yapacağız. Görüşmelere ek olarak üçüncü ve altıncı dersin sonunda dokuz soruluk bir öğrenme testini doldurmasını isteyeceğiz. Bu testler öğrencilerin bilgi işlemsel düşünme kavramları bilgisinin gelişimini inceleyebilmemize yardımcı olacaktır. Öğrenciyle yapılacak olan görüşmelerde ses kaydı alınacaktır. Katılımcı çocuklara bu çalışmaya katıldıklarını belgeleyen bir takdirname verilecektir. Katılımcı özel eğitim ve rehabilitasyon merkezi müdürü ve öğretmenlerine de bir teşekkür belgesi vereceğiz. Çocuğunuzun ders esnasında sıkılması riskine karşın derslerde sık ara verilecek ve bu aralarda ek etkinlikler yapılacaktır. Motivasyon risklerine yönelik olarak ise her bir tamamlanan etkinlik için öğrencileri teşvik edebilecek ödülleri verilebilecektir (sticker, kalem veya silgi gibi).

Çocuğunuzun katılımı sayesinde elde edeceğimiz araştırma sonuçları öğrenme güçlüğü çeken öğrencilere özgü müfredatın oluşmasına katkı yapacaktır. Bu araştırma bilimsel bir amaçla yapılmaktadır ve katılımcı bilgilerinin gizliliği esastır. Ses kayıtlarının yazıya dökülmüş halinde çocukların ismi yerine başka bir isim kullanılacaktır. Ses kayıtları araştırma projemiz süresince kilitli bir dolapta muhafaza edilip araştırma sona erdiğinde bir süre daha saklandıktan sonra imha

edilecektir. Kayıtlara sadece proje ekibinin erişimi olacaktır. Elde edilen veriler bilimsel tez veya yayınlarda öğrencinin kimliği tamamen gizli tutularak kullanılabilir.

Bu araştırmaya katılmak tamamen isteğe bağlıdır. Velisi olduğunuz öğrenci bu çalışmaya katıldığı takdirde çalışmanın herhangi bir aşamasında herhangi bir sebep göstermeden çalışmadan ayrılabilir. Katılmadığı takdirde ise özel eğitim ve rehabilitasyon merkezinde araştırma sırasında ve sonrasında alacağı hizmetlerle ilgili herhangi bir problem yaşanmayacaktır. Araştırma projesi hakkında ek bilgi almak istediğiniz takdirde lütfen Boğaziçi Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü Öğretim Üyesi Prof. Dr. Diler Öner ile temasa geçebilirsiniz Ayrıca araştırmaya katılımı ile ilgili haklarınız konusunda Boğaziçi Üniversitesi Sosyal ve Beşeri Bilimler Yüksek Lisans ve Doktora Tezleri Etik İnceleme Komisyonu'na (SOBETİK) danışabilirsiniz.

Eğer bu araştırma projesine katılmasını kabul ediyorsanız, lütfen bu formu imzalayıp kapalı bir zarf içerisinde bize geri yollayın.

Ben, (katılımcının velisinin adı), yukarıdaki metni okudum ve katılmam istenen çalışmanın kapsamını ve amacını, gönüllü olarak üzerime düşen sorumlulukları tamamen anladım. Çalışma hakkında soru sorma imkanı buldum. Bu çalışmayı istediğim zaman ve herhangi bir neden belirtmek zorunda kalmadan bırakabileceğimi ve bıraktığım takdirde herhangi bir olumsuzluk ile karşılaşmayacağımı anladım.

Bu koşullarda söz konusu araştırmaya kendi isteğimle, hiçbir baskı ve zorlama olmaksızın katılmayı kabul ediyorum.

Formun bir örneğini aldım.

Formun bir örneğini almak istemiyorum (bu durumda araştırmacı bu kopyayı saklar).

.....

Katılımcının VELİSİNİN Adı-Soyadı:.....

İmzası:.....

Tarih (gün/ay/yıl):...../...../.....

☐ Ses kaydı alınmasını kabul ediyorum.

Araştırmacının Adı-Soyadı: Mukaddes Aslı Pehlivan

İmzası:.....

Tarih (gün/ay/yıl):...../...../.....

REFERENCES

- Archer, A. L., & Hughes, C. A. (2010). *Explicit instruction: Effective and efficient teaching*. Guilford Publications.
- Baxter, P., & Jack, S. (2008). Qualitative case study methodology: Study design and implementation for novice researchers. *The Qualitative Report*, 13(4), 544-559.
- Bloom, B. S., Engelhart, M. D., Furst, E. J., Hill, W. H., & Krathwohl, D. R. (1956). *Handbook I: cognitive domain*. New York: David McKay.
- Chen, Y. C., Lu, Y. L., & Lien, C. J. (2021). Learning environments with different levels of technological engagement: a comparison of game-based, video-based, and traditional instruction on students' learning. *Interactive Learning Environments*, 29(8), 1363-1379.
- Creswell, J. W. (2007). *Qualitative inquiry and research design: Choosing among five approaches* (2nd ed.). Sage Publications
- CSTA and ISTE (2011). *Computational Thinking in K–12 Education leadership toolkit*. Retrieved from <http://csta.acm.org/Curriculum/sub/CurrFiles/471.11CTLeadershipToolkit-SP-vF.pdf>.
- ECforALL Curriculum, n.d., *Act1 Curriculum Overview*. Retrieved from <https://impactconectar.wixsite.com/website/act-1>
- Evett, L., & Brown, D. (2005). Text formats and web design for visually impaired and dyslexic readers—Clear text for all. *Interacting with computers*, 17(4), 453-472.
- Franklin, D., Salac, J., Crenshaw, Z., Turimella, S., Klain, Z., Anaya, M., & Thomas, C. (2020). Exploring student behavior using the TIPP&SEE learning strategy. In *Proceedings of the 2020 ACM Conference on International Computing Education Research* (pp. 91-101).
- Fuller, U., Johnson, C. G., Ahoniemi, T., Cukierman, D., Hernán-Losada, I., Jackova, J., Lahtinen, E., Lewis, T. L., McGee Thompson, D., Riedesel, C., & Thompson, E. (2007). Developing a computer science-specific learning taxonomy. *ACM SIGCSE Bulletin*, 39(4), 152-170.
- Gay, L. R., Mills, G. E., & Airasian, P. W. (2011). *Educational research: Competencies for analysis and applications*. Pearson Higher Ed.
- Graue, M. E., & Walsh, D. J. (1998). *Studying children in context: Theories, methods, and ethics*. Sage.

- Gülbahar, Y., Kert, S. B. ve Kalelioğlu F. (2018). Bilgi işlemsel düşünme becerisine yönelik öz yeterlik algısı ölçeği (BIDBÖA): Geçerlik ve güvenirlik çalışması. *Türk Bilgisayar ve Matematik Eğitimi Dergisi*. Advance online publication. doi:10.16949/turkbilmat.385097
- Hansen, A. K., Hansen, E. R., Dwyer, H. A., Harlow, D. B., & Franklin, D. (2016). Differentiating for diversity: Using universal design for learning in elementary computer science education. In *Proceedings of the 47th ACM technical symposium on computing science education* (pp. 376-381).
- Hall, T. E., Cohen, N., Vue, G., & Ganley, P. (2015). Addressing learning disabilities with UDL and technology: Strategic reader. *Learning Disability Quarterly*, 38(2), 72-83.
- Intro to ECforall Curriculum. (2022). Retrieved from https://drive.google.com/file/d/1PyXgNaQoKVou_HbaXFCebW3qgzaZ4xhZ/view
- Ismail, R., & Jaafar, A. (2011). Interactive screen-based design for dyslexic children. In *2011 International Conference on User Science and Engineering (i-USer)* (pp. 168-171). IEEE.
- Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, 82, 263-279.
- Israel, M., Ribuffo, C., & Smith, S. (2014). Universal design for learning: Recommendations for teacher preparation and professional development. *Document No. IC-7 CEEDAR Centre*. Retrieved from http://cedar.education.ufl.edu/wp-content/uploads/2014/08/IC-7_FINAL_08-27-14.pdf:(accessed on 22 March 2018).
- Johnson, C. G., & Fuller, U. (2006). Is Bloom's taxonomy appropriate for computer science?. In *Proceedings of the 6th Baltic Sea conference on Computing education research: Koli Calling 2006* (pp. 120-123).
- 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.
- Kasalak, İ., & Altun, A. (2018). Blok temelli programlamaya ilişkin öz-yeterlik algısı ölçeği geliştirme çalışması: Scratch örneği. *Eğitim Teknolojisi Kuram ve Uygulama*, 8(1), 209-225.
- Korkmaz, Ö., Çakır, R., Özden, M. Y. (2015). Bilgisayarca Düşünme Beceri Düzeyleri Ölçeğinin (BDBD) ortaokul düzeyine uyarlanması. *Gazi Eğitim Bilimleri Dergisi*, 1(2), 67-86.
- Krathwohl, D. R. (2002). A revision of Bloom's taxonomy: An overview. *Theory into practice*, 41(4), 212-218.

- Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads*, 2(1), 32-37.
- Lincoln, Y. S., & Guba, E. G. (1986). But is it rigorous? Trustworthiness and authenticity in naturalistic evaluation. *New directions for program evaluation*, 1986(30), 73-84.
- Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014). Computational thinking in K-9 education. In *Proceedings of the working group reports of the 2014 on innovation & technology in computer science education conference* (pp. 1-29).
- Merriam, S. B., & Tisdell, E. J. (2015). *Qualitative research: A guide to design and implementation*. John Wiley & Sons.
- Nalbantoglu Yilmaz, F. (2021). Comparing paper-pencil and computer-based tests: A meta-analysis study in the sample of Turkey. *Eurasian Journal of Educational Research*, 93, 279-300.
- National Centre for Computing Education (n.d.) *Pedagogy Quick Reads Using PRIMM to structure programming lessons*. Retrieved from <https://raspberrypi-education.s3-eu-west-1.amazonaws.com/Quick+Reads/Pedagogy+Quick+Read+11+-+PRIMM.pdf>
- Öz, H., & Özturan, T. (2018). Computer-based and paper-based testing: Does the test administration mode influence the reliability and validity of achievement tests?. *Journal of Language and Linguistic Studies*, 14(1), 67-85.
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.
- Patton, M. Q. (2002). *Qualitative research and evaluation methods* (3rd ed.). Thousand Oaks, CA: Sage Publications.
- Patton, M. Q. (2014). *Qualitative research & evaluation methods: Integrating theory and practice*. Sage publications.
- Polat, E., Hopcan, S., Kucuk, S., & Sisman, B. (2021). A comprehensive assessment of secondary school students' computational thinking skills. *British Journal of Educational Technology*, 52(5), 1965–1980. doi:10.1111/bjet.13092
- Rao, K., Torres, C., & Smith, S. J. (2021). Digital tools and UDL-based instructional strategies to support students with disabilities online. *Journal of Special Education Technology*, 36(2), 105-112.
- Ray, M. J., Israel, M., Lee, C. E., & Do, V. (2018). A cross-case analysis of instructional strategies to support participation of K-8 Students with disabilities in CS for all. In *Proceedings of the 49th ACM technical symposium on computer science education* (pp. 900-905).

- Salac, J., Thomas, C., Butler, C., Sanchez, A., & Franklin, D. (2020). TIPP&SEE: A learning strategy to guide students through Use-Modify Scratch activities. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 79-85).
- Salac, J., Thomas, C., Butler, C., & Franklin, D. (2021a). Supporting diverse learners in K-8 computational thinking with TIPP&SEE. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (pp. 246-252).
- Salac, J., Thomas, C., Butler, C., & Franklin, D. (2021b). Investigating the role of cognitive abilities in computational thinking for young learners. In *Proceedings of the 17th ACM Conference on International Computing Education Research* (pp. 2-17).
- Scratch. (n.d.). *About Scratch*. Retrieved from <https://scratch.mit.edu/about>
- ScratchEd. (n.d.). *Curriculum*. Retrieved from <https://scratched.gse.harvard.edu/guide/curriculum.html>
- Sentance, S., Waite, J., & Kallia, M. (2019). Teachers' experiences of using PRIMM to teach programming in school. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 476-482).
- Suri, H. (2011). Purposeful sampling in qualitative research synthesis. *Qualitative research journal*, 11(2), 63-75.
- Stake, R. E. (1995). *The art of case study research*. Sage.
- Stake, R. E. (2000). Case Studies. In N. K. Denzin, & Y. S. Lincoln (Eds.), *Handbook of Qualitative Research* (pp. 435-453). Thousand Oaks, CA: Sage.
- Stephen, F. R., & Rollings, H. E. (1976). Item-by-item feedback and multiple choice test performance. *The Journal of Experimental Education*, 44(4), 30-32.
- The Creative Computing Curriculum. (n.d.). *Curriculum*. Retrieved from <http://creativecomputing.gse.harvard.edu/guide/curriculum.html>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717-3725.
- Wing, J. (2011). Research notebook: Computational thinking—What and why. *The link magazine*, 6, 20-23.
- Yağcı, M. (2019). A valid and reliable tool for examining computational thinking skills. *Education and Information Technologies*, 24(1), 929-951.