EXPLORING ELEMENTARY STUDENTS’ COMPUTATIONAL THINKING
LEVERAGING AFFORDANCES OF LEARNING TRAJECTORIES

By

FEIYA LUO

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2020
To my family:
My parents, who have unconditional love and support for me;
my aunt, who gave me the very first English lessons of my life and who has been an inspiration ever since;
my cousins, who look up on me, giving me constant momentum to excel so I can continue having the pleasure of being their role model;
my grandfather, who would be happily in tears had he not have Alzheimer’s;
and everyone else, who have loved me in their own ways.

And in loving memory of my grandmother,
Zhang Heying (1936 - 2015),
who would not hesitate to sacrifice for the ones she loved ever so dearly.
ACKNOWLEDGMENTS

I dedicate this section to all those who have helped me in all possible ways throughout my entire PhD journey. I would like to first thank Drs. Pasha Antonenko and Maya Israel, my dissertation chair and co-chair, for giving me generous guidance and mentorship and opportunities to thrive in the academia. Through working with the two of them, I learned to become a better student to my teachers, colleague to my teammates, and teacher to my students. I cannot be happier and more content now that I am graduating with my PhD degree and transitioning to my new faculty life.

Thinking back, I am grateful for every experience I have had in the Educational Technology program. I thank Dr. Albert Ritzhaupt for recommending me to work as the web developer for the grant project, GoFlag, where I worked with Pasha on two scholarly papers. I thank the hiring committee at UF that reached out to Maya and all those who facilitated Maya’s moving from UIUC to UF. Maya’s joining our program was a Jackpot prize for me.

I thank Drs. Thomasenia Adams, Christina Gardner-McCune, and Brian Gane for serving as members on my doctoral committee and providing me with critical comments and feedback that made this dissertation study stronger.

I would also like to thank Mr. Todd Lash and Ms. Gakyung Jeong for helping me collect the data for this dissertation study. I am also thankful to Dr. Rob Ore, Mrs. Diane Ore, and their lovely dog, Dooley, for providing me shelter in freezing wind chill weather and an amazing collection of teas during my data collection trips.

Last but not the least, I thank my parents and all the family members for their support and love. This dissertation would not have been completed without the support from all of them.
**TABLE OF CONTENTS**

ACKNOWLEDGMENTS .......................................................................................................................... 4

LIST OF TABLES .................................................................................................................................. 9

LIST OF FIGURES ................................................................................................................................ 10

LIST OF ABBREVIATIONS .................................................................................................................. 11

DEFINITIONS OF KEY TERMS ......................................................................................................... 12

ABSTRACT ............................................................................................................................................. 13

CHAPTER

1 INTRODUCTION ............................................................................................................................... 15

   Background ....................................................................................................................................... 15
   Statement of the Problem .................................................................................................................. 17
   Context of the Study .......................................................................................................................... 17
   Purpose of the Study .......................................................................................................................... 19
   Research Questions .......................................................................................................................... 19
   Significance ...................................................................................................................................... 20

2 LITERATURE REVIEW ..................................................................................................................... 21

   Defining Computational Thinking (CT) ............................................................................................. 21
      CT Definition Overview .................................................................................................................. 21
      CT Concepts vs. CT Skills ............................................................................................................. 23
   CT in Elementary Settings ................................................................................................................ 23
      Challenges of Implementing CT in Elementary Schools ............................................................... 23
      CT Instruction with Plugged and Unplugged Activities ................................................................. 24
   Theoretical Perspectives in CT ........................................................................................................ 27
      Influences of Constructionism ....................................................................................................... 27
      Influences of Explicit Instruction .................................................................................................. 30
      Arguments for Combining Constructionism and Components of Explicit Instruction ... 32
      Learning Trajectories (LTs) as the Theoretical Foundation ........................................................... 33
         Definition of LT ............................................................................................................................ 33
         LTs in CT instruction .................................................................................................................... 36
   Elementary CT Assessment ............................................................................................................... 38
      Assessing CT Using Interviews ..................................................................................................... 38
      Assessing CT Using Portfolios ....................................................................................................... 40
      Assessing CT with Selected- or Constructed-Response Tests ....................................................... 42
      Gaps in the Elementary CT Assessment Literature ................................................................. 43
3 METHOD ........................................................................................................................................... 47

Method Overview ................................................................................................................................. 47
Participants and Settings ......................................................................................................................... 47
Qualitative Methodology ....................................................................................................................... 48
Design .................................................................................................................................................. 49
  Integrated Math-CT Instruction Based on LTs ..................................................................................... 50
  Integrated Math-CT Assessment ......................................................................................................... 52
  Cognitive Interview with the Think-Aloud Protocol .......................................................................... 54
The Pilot Study .................................................................................................................................... 55
Item Selection and Grouping ............................................................................................................... 57
  Item selection criteria .......................................................................................................................... 57
  Item grouping method – Phase 1 ......................................................................................................... 57
  Item grouping method – Phase 2 ......................................................................................................... 58
Data Collection Procedures ................................................................................................................ 59
  Phase 1 ............................................................................................................................................. 59
  Phase 2 ............................................................................................................................................ 60
Data Analysis ....................................................................................................................................... 60
  A priori Codes ................................................................................................................................. 61
  Interpretation Criteria ........................................................................................................................ 61
  Constant Comparison ....................................................................................................................... 62
Analytical Process ............................................................................................................................... 64
  Analysis using a priori codes ............................................................................................................. 64
  Analysis using constant comparison ................................................................................................. 64
  Mapping the correspondence ........................................................................................................... 66
Rigor .................................................................................................................................................... 66

4 FINDINGS ......................................................................................................................................... 76

Overall Data ......................................................................................................................................... 76
Themes of Participants’ CT Articulation ............................................................................................... 77
Sequence: Grade 3 (G3) Participants Generally Demonstrated Understanding and Ability in Using Complete, Precise, and Ordered Instructions .......................................................... 77
Repetition ............................................................................................................................................ 80
  G3 and G4 participants generally showed evidence of understanding and recognizing repetition in the coding item ............................................................................................................. 80
  G3 and G4 participants showed varied ability in recognizing and using repeat instructions in word problems ..................................................................................................................... 81
Conditionals ......................................................................................................................................... 83
  G4 participants showed inconsistent understanding of evaluating conditions involving number comparisons ................................................................................................................................. 83
  G4 participants generally showed no understanding of evaluating conditions in the coding item ................................................................................................................................. 83
Decomposition ....................................................................................................................................... 84
  G3 and G4 participants could identify components of a number ..................................................... 84
  G3 and G4 participants used decomposition in arithmetic but demonstrated varied ability in decomposing the word problem ................................................................................................. 85
Corresponding Participants’ Articulated CT to LTs ................................................................. 86
   Sequence .............................................................................................................................. 87
   Repetition ............................................................................................................................ 88
   Conditionals ...................................................................................................................... 88
   Decomposition .................................................................................................................. 89
Summary of Findings ............................................................................................................. 90
5  DISCUSSION ....................................................................................................................... 96

LT-based Targeted Instruction of CT .................................................................................... 96
   Sequence for G3 .................................................................................................................. 97
   Repetition for G3 and G4 ................................................................................................ 98
   Decomposition for G3 and G4 ........................................................................................ 100
   Conditionals for G4 .......................................................................................................... 101
Different Perspectives of CT Assessment ............................................................................. 103
   Affordances of Scratch .................................................................................................... 104
   Word Problem .................................................................................................................. 104
Interpreting the Correspondence between Articulated CT and the LTs .............................. 109
   Context of Assessment Items ......................................................................................... 109
   Knowledge, Skills, and Abilities (KSAs) .......................................................................... 110
Implications .......................................................................................................................... 111
Limitations ............................................................................................................................ 114
Conclusion ............................................................................................................................. 115

APPENDIX

A THINK-ALOUD PROTOCOL ............................................................................................ 119
B SAMPLE INTEGRATED MATH-CT LESSON ................................................................. 121
C CURRICULUM MAPPING OF CT CONCEPTS ............................................................... 126
D KSA OF ASSESSMENTS ................................................................................................. 127
E LEARNING TRAJECTORIES WITH INSTRUCTION MAPPING ..................................... 128
F KSA-ITEM MAPPING ....................................................................................................... 130
G APPROVED IRB PROTOCOL ......................................................................................... 132
H CONSENT LETTER ........................................................................................................... 137
I ASSENT LETTER ................................................................................................................ 139
J ITEM LIST .......................................................................................................................... 141

LIST OF REFERENCES ......................................................................................................... 147
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>Grade 3 lesson coverage of CT concepts</td>
<td>68</td>
</tr>
<tr>
<td>3-2</td>
<td>Grade 4 lesson coverage of CT concepts</td>
<td>69</td>
</tr>
<tr>
<td>3-3</td>
<td>Distribution of items by CT concept</td>
<td>69</td>
</tr>
<tr>
<td>3-4</td>
<td><em>A priori</em> codes for Sequence</td>
<td>70</td>
</tr>
<tr>
<td>3-5</td>
<td><em>A priori</em> codes for Repetition</td>
<td>70</td>
</tr>
<tr>
<td>3-6</td>
<td><em>A priori</em> codes for Conditionals</td>
<td>71</td>
</tr>
<tr>
<td>3-7</td>
<td><em>A priori</em> codes for Decomposition</td>
<td>71</td>
</tr>
<tr>
<td>4-1</td>
<td>Participant articulation - LT correspondence (Sequence)</td>
<td>91</td>
</tr>
<tr>
<td>4-2</td>
<td>Participant articulation - LT correspondence (Item R.03.c)</td>
<td>91</td>
</tr>
<tr>
<td>4-3</td>
<td>Participant articulation - LT correspondence (Item R.01.a)</td>
<td>92</td>
</tr>
<tr>
<td>4-4</td>
<td>Participant articulation - LT correspondence (Conditionals)</td>
<td>92</td>
</tr>
<tr>
<td>4-5</td>
<td>Participant articulation - LT correspondence (Item DC.02.a)</td>
<td>92</td>
</tr>
<tr>
<td>4-6</td>
<td>Participant articulation - LT correspondence (Item DC.06.c)</td>
<td>93</td>
</tr>
<tr>
<td>4-7</td>
<td>Participant articulation - LT correspondence (Item DC.02.b)</td>
<td>93</td>
</tr>
<tr>
<td>5-1</td>
<td>Suggested targeted instructional activities for pattern recognition</td>
<td>116</td>
</tr>
<tr>
<td>5-2</td>
<td>Suggested explicit instructional activities for condition evaluation</td>
<td>117</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>2-1</td>
<td>The sequence learning trajectory</td>
<td>45</td>
</tr>
<tr>
<td>2-2</td>
<td>The repetition learning trajectory</td>
<td>45</td>
</tr>
<tr>
<td>2-3</td>
<td>The conditionals learning trajectory</td>
<td>46</td>
</tr>
<tr>
<td>2-4</td>
<td>The decomposition learning trajectory</td>
<td>46</td>
</tr>
<tr>
<td>3-1</td>
<td>The design framework</td>
<td>72</td>
</tr>
<tr>
<td>3-2</td>
<td>Example math-CT integrated lesson</td>
<td>73</td>
</tr>
<tr>
<td>3-3</td>
<td>Item S.01.a.</td>
<td>74</td>
</tr>
<tr>
<td>3-4</td>
<td>Analysis using <em>a priori</em> codes.</td>
<td>74</td>
</tr>
<tr>
<td>3-5</td>
<td>Analysis using constant comparison for each CT concept</td>
<td>75</td>
</tr>
<tr>
<td>3-6</td>
<td>Direct correspondence between articulated CT and CT LTs.</td>
<td>75</td>
</tr>
<tr>
<td>4-1</td>
<td>Item R.03.c.</td>
<td>94</td>
</tr>
<tr>
<td>4-2</td>
<td>Item R.01.b.</td>
<td>94</td>
</tr>
<tr>
<td>4-3</td>
<td>Item C.02.e.</td>
<td>95</td>
</tr>
<tr>
<td>4-4</td>
<td>Item C.03.b.</td>
<td>95</td>
</tr>
</tbody>
</table>
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCA</td>
<td>Constant comparison analysis</td>
</tr>
<tr>
<td>CS</td>
<td>Computer Science</td>
</tr>
<tr>
<td>CT</td>
<td>Computational thinking</td>
</tr>
<tr>
<td>G3</td>
<td>Grade 3</td>
</tr>
<tr>
<td>G4</td>
<td>Grade 4</td>
</tr>
<tr>
<td>KSA</td>
<td>Knowledge, skill, and ability</td>
</tr>
<tr>
<td>LT</td>
<td>Learning trajectory</td>
</tr>
<tr>
<td>PD</td>
<td>Professional development</td>
</tr>
</tbody>
</table>
DEFINITIONS OF KEY TERMS

**Computational thinking**  
The thought processes involved in expressing solutions as computational steps or algorithms that can be carried out by a computer” (K–12 Computer Science Framework, 2016, p.68) and that it “requires understanding the capabilities of computers, formulating problems to be addressed by a computer, and designing algorithms that a computer can execute” (K–12 Computer Science Framework, 2016, p.69).

**Learning trajectories**  
Learning trajectories consist of learning goals students are expected to meet as they engage in instruction, learning progressions (i.e., hypothetical developmental paths that students take as they progress toward increasingly sophisticated understanding during learning), and the instructional activities that support the learning progressions (Simon, 1995; Clements & Sarama, 2004; Clements & Sarama, 2004; Maloney et al., 2014).
Abstract of Dissertation Presented to the Graduate School of the University of Florida in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

EXPLORING ELEMENTARY STUDENTS’ COMPUTATIONAL THINKING LEVERAGING AFFORDANCES OF LEARNING TRAJECTORIES

By

Feiya Luo

August 2020

Chair: Pasha Antonenko
Cochair: Maya Israel
Major: Curriculum & Instruction

There is little empirical research related to how elementary students develop computational thinking (CT) and how students apply CT in problem-solving. To address this gap in knowledge, this study made use of learning trajectories (LTs; hypothesized learning goals, progressions, and activities) in CT concept areas such as sequence, repetition, conditionals, and decomposition to better understand students’ understanding of CT. This basic interpretive, qualitative study aimed at gaining a deeper understanding of elementary students’ CT by having students express and articulate their CT in assessment-based problem-solving scenarios. This study implemented eight math-CT integrated lessons aligned to both the Common Core State Standards for Math (CCSS-M) and the LTs throughout the school year 2019-2020 with third- and fourth-grade students. The current study specifically examined the students’ CT in the areas of sequence, repetition, conditionals, and decomposition. By using the cognitive interview technique with a think-aloud protocol, this study revealed the CT thought processes of 22 elementary students while they worked on CT assessment items. Participants’ CT articulation was examined using a priori codes translated verbatim from the learning goals in the LTs and mapped to the learning goals and progressions in the LTs.
Results suggested that students demonstrated a more consistent articulation of CT in the area of sequence and varied understanding in repetition, conditionals, and decomposition, with some participants demonstrating more complex CT than others. As such, participants’ CT articulation was generally mapped to the intermediate-level in the Sequence LT and varied levels in the other LTs. By collecting empirical data on how students expressed and articulated their CT, this study made theoretical contributions by generating initial empirical evidence to support the hypothesized learning goals and progressions in the LTs. This study also has methodological implications by employing cognitive interviews to study students’ CT and practical implications by identifying learning gaps and suggesting targeted instructional activities to support the development of elementary students’ CT.
CHAPTER 1
INTRODUCTION

Background

Computer science (CS) education is increasingly gaining attention from policymakers, educational researchers and practitioners, and the general public, such as parents and computing enthusiasts. On the national level, the “Computer Science for All” initiative is aimed at promoting CS education from kindergarten to high school and equipping students with computational thinking (CT) skills to thrive in the digital economy. The K-12 Computer Science Framework (2016) proposed five core concepts and seven core practices. The five concepts are: computing systems, networks and the Internet, data and analysis, algorithms and programming, and impacts of computing. The seven practices include fostering an inclusive computing culture, collaborating around and communicating about computing, recognizing and defining computational problems, creating, testing and refining computational artifacts, and developing and using abstraction. The Framework also proposed a CS concept and practice progression by grade band. For example, by the end of Grade 2, the students are expected to have an initial understanding of the five concepts. By the end of Grade 5, that understanding of the same concept will involve more advanced knowledge. In addition to the K-12 CS Framework, which provides overarching guidance for CS education per grade bands, the Computer Science Teachers Association (CSTA) Computer Science Standards (2017) provides detailed expectations for student performance and achievements for CS concepts and practices in the elementary and secondary level. The standards organize each core concept and practice in the K-12 CS Framework into sub concepts and practices. The detailed descriptive statements explain the suggested benchmarks of students’ CS understanding and skills by each level (grade bands). For example, Grades K-2 students can model daily processes, such as teeth-brushing and
cooking by following a step-by-step procedure, which is, essentially, an algorithm. By Grades 3-5, students know how to select from multiple algorithms the most appropriate in order to finish a task.

In elementary school settings, CS education usually takes form in promoting students’ CT in preparation for more advanced CS knowledge and skill development (Bers et al., 2014; Fessakis et al., 2013). CT is one of the Disciplinary Core Ideas for integrated K-12 science, technology, engineering, and mathematics (STEM) instruction in the Next Generation Science Standards (2013). Therefore, CT is often integrated in the context of other academic disciplines, such as mathematics and science (Rich et al., 2019; Sáez-López et al., 2016). One of the biggest advantages of CT integration is that it ensures equity by providing CS education in subject areas that are already taught to students (Weintrop et al., 2016). Compared to CS education as electives or enrichment activities available to only the few that can afford to participate, such integrated learning opportunities are accessible and available to all students.

Research and practice in CT integration at the elementary level is only in its infant stage. Researchers (Rich et al., 2017; 2018) proposed hypothesized learning trajectories (LTs) with learning goals and progressions to guide elementary CT implementation, however, these LTs have not yet been empirically tested. There is limited knowledge and standards that teachers and practitioners can resort to in terms of “when to teach what” in an integrated context: What CT content is developmentally appropriate for elementary students? What should the instructional sequence be relative to the target subject area of integration? How can teachers support CT learning through instruction? What types of understandings and misconceptions do students have? And what student learning outcomes can be expected? Answers to these questions are yet to be explored.
Statement of the Problem

The national standards such as the CSTA and K-12 CS Framework provide guidance for how to support the development of students’ CT competencies in the K-12 context. The LTs offer hypotheses of elementary students’ development of CT. However, there is little empirical evidence of how students’ CT actually develops across grade bands. To add complexity, existing research on CT has primarily focused on examining the “products” of students’ learning, such as computational artifacts that students built and programs that students wrote (Bers et al., 2014; Werner et al., 2012). While many researchers have assessed CT as a learning product, research focused on collecting in-depth, qualitative evidence to understand how students develop and apply CT is still lacking (Tang et al., 2020). Therefore, more research is needed to uncover how elementary students develop CT and the mental processes involved when applying CT in problem-solving.

Context of the Study

This dissertation study was situated within the larger grant project, “Learning Trajectories for Everyday Computing: Integrating Computational Thinking in Elementary Mathematics” (LTEC), sponsored by the National Science Foundation (Award No. 1932920). A multi-institutional effort involving University of Florida, University of Chicago, and University of Illinois-Chicago, the grant project is devoted to creating and piloting integrated mathematics and CT instruction in elementary school utilizing prototype K-8 CT LTs. The three major goals of the project involve generating evidence for 1) a better understanding of how CT can be integrated into elementary mathematics in an effective and efficient way, 2) the effect of integrated instruction on students’ mathematical understanding, and 3) refining the LTs using empirical data by piloting the integrated instruction.
The LTEC project has produced important results such as a collection of integrated mathematics + CT lessons that are based on the Common Core State Standards (2012) for Mathematics (CCSS-M). The lessons are designed leveraging synergies between Grade 3-4 mathematics concepts (namely number and operations with fractions) in the Everyday Math (Edition 4) curriculum and CT concepts (e.g., sequence, repetition, conditionals, decomposition, and variables). Another major product of the project is student learning assessment measures that are developed using the evidence-centered design (ECD), in the form of early-, mid-, and late-assessments, to be administered throughout a school year. This project also produced four prototype CT learning trajectories (LTs). These LTs with learning goals and progressions were developed by Rich and colleagues (2017, 2018) based on literature reviews to provide potential standards and expectations that help frame elementary CT instruction. However, given that the learning goals were synthesized from the literature and the progressions were hypothesized, they have not been empirically tested. Therefore, empirical evidence is yet to be established in terms of how elementary students achieve the learning goals and progress along the LTs. This dissertation study leverages the LTs, the integrated lessons, the assessments developed in the LTEC project in exploring how elementary students articulate CT and generating initial empirical evidence to understand how students’ articulated CT map to the LTs.

Currently, the integrated lessons are being piloted in schools in the midwestern U.S. throughout the school year 2019-2020. The year-long integrated lessons are divided into three stages, with each stage involving approximately one third of the lessons. The stages are indicated by the three assessments, early, mid, and late, administered at the end of each stage. Considering time restraints, this dissertation only examined students’ CT in the first two phases (i.e., phase 1 and 2 for each grade). Given that the goal of the study was to examine how students articulate
CT understanding as they progress in integrated math-CT instruction, it was important to be informed which lessons and the embedded CT concepts were introduced during those two instruction stages. Following the implementation pace of the integrated lessons, 3rd-grade students participated in eight lessons that focused on sequence, repetition, and decomposition, and 4th-grade students participated in eight lessons that focused on repetition, conditionals, and decomposition (refer to Appendix C Curriculum Mapping of CT Concepts). Hence, this dissertation study focused on the common concepts of repetition and decomposition, as well as sequence for G3 students and conditionals for G4 students.

**Purpose of the Study**

The purpose of this dissertation study was to gain a better understanding of how elementary students develop CT and apply CT in problem-solving. One way to gain such insight, as stated by Confrey et al. (2014), is to “discover the ideas, concepts, and skills that children can express, articulate, and develop when they are given the opportunity to engage with rich and interesting activities and to express and reconsider their experience and understanding” (p.245). Therefore, this dissertation study used the cognitive interview technique with a think-aloud protocol to a) qualitatively examine 3rd- and 4th-grade students’ CT as they engage in problem-solving and b) explore how students’ CT maps to the learning goals and progressions hypothesized in the CT LTs. This dissertation study aimed at gaining a better understanding of elementary students’ CT by revealing students’ articulation of CT through the cognitive interview with a think-aloud protocol.

**Research Questions**

Given the rationale discussed above, the research questions (RQs) that guided this study were:

1. How do 3rd- and 4th-grade students express and articulate CT in math-CT problem-solving scenarios? Specifically:
a. How do 3rd-grade students express and articulate CT, in the areas of sequence, repetition, and decomposition?

b. How do 4th-grade students express and articulate CT, in the areas of repetition, conditionals, and decomposition?

2. How does students’ articulated CT correspond to the learning goals and the learning progressions of the CT learning trajectories?

**Significance**

This study aimed at shedding light on understanding elementary students’ CT and generating empirical evidence to better understand students’ learning and development in CT instruction. Given that LTs are usually created by synthesizing prior literature and conducting further research, and validated based on empirical studies (Maloney et al., 2014), the next steps in computational thinking LT research is to conduct further research to validate the sequences and progressions using empirical data. This dissertation study has a few important contributions to understanding elementary students’ CT. First, the study made important methodological contributions by using the under-used CT assessment type of cognitive interview with a think-aloud protocol to reveal the thinking processes in real time as students engage in problem-solving. Second, by asking students to think-aloud while problem solving, this study generated empirical data of how elementary students express and articulate CT. Third, this study generated initial empirical evidence to support students’ CT development and progression along the LTs. Last but not the least, the study made practical contributions in identifying some learning gaps students were experiencing and proposed targeted instructional activities teachers could use to support students’ CT learning.
CHAPTER 2
LITERATURE REVIEW

Defining Computational Thinking (CT)

CT Definition Overview

Computational thinking is a fundamental skill to apply computer science concepts in problem-solving (Wing, 2006). Specifically, CT “involves solving problems, designing systems and understanding human behavior, by drawing on the concepts fundamental to computer science” (Wing, 2006, p.33). Wing (2006) explained that major computer science concepts included thinking recursively, using abstraction (i.e., representing and modeling a problem without including every detail) and decomposition (i.e., separation and modularization of a problem). CT is informed by but differs from CS in that it is “thinking like a computer scientist” (p.35) at multiple levels of abstraction rather than programming alone; It is an approach humans take to tackle problems being equipped with computing devices; It “complements and combines mathematical and engineering thinking” (p.35) and enables humans to break physical constraints and build virtual systems (Wing, 2006). While CT is not equal to CS, CT is considered to be intrinsically connected to CS and is a competency that can be effectively developed through CS education (K-12 CS Framework, 2016).

CT was later interpreted to involve “defining, understanding, and solving problems, reasoning at multiple levels of abstraction, understanding and applying automation, and analyzing the appropriateness of the abstractions made” (Lee et al., 2011, p.32). The three domains in CT (Lee et al., 2011) were abstraction (i.e., selecting certain features to model with a computing device), automation (i.e., instructing the computer to efficiently and quickly execute as demanded by human instructions), and analysis (i.e., reflecting and validating actions and decisions).
Acknowledging these aforementioned concepts, more researchers reached consensus on what CT encompasses. The Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE) converged on CT being “solving problems in a way that can be implemented with a computer” (Barr & Stephenson, 2011, p.51) and that students become “tool builders” to create physical and virtual artifacts rather than “tool users.” In addition, data collection, data analysis, data representation, algorithms, parallelization, and simulation were recognized and identified as CT concepts and capabilities. In CSTA and ISTE’s operational definition of CT (2011), data organization and analysis, abstraction, algorithmic thinking, efficient solution-seeking and transferring problem-solving processes were reiterated.

CT has also been interpreted in specific contexts (Brennan & Resnick, 2012) such as the programming interactive media, Scratch, which is a visual block-based coding platform where students can create programs from Scratch or modify pre-existing project codes to make a new project. The idea was to learn through design activities by constructing new knowledge within the constructionist framework. Three dimensions of CT (Brennan & Resnick, 2012) were proposed: computational concepts (basic computer science concepts such as sequence, loops, events, parallelism, conditionals, operators, and data), computational practices (iteration, debugging, remixing, and abstraction), and computational perspectives (expressing, connecting, and questioning in programming). This dissertation study used the definition of CT synthesized by K–12 Computer Science Framework (2016) from previous researchers (i.e., Cuny, Snyder, & Wing, 2010; Aho, 2011; Lee, 2016): “the thought processes involved in expressing solutions as computational steps or algorithms that can be carried out by a computer” (p.68) and that it
“requires understanding the capabilities of computers, formulating problems to be addressed by a computer, and designing algorithms that a computer can execute” (p.69).

CT Concepts vs. CT Skills

Acknowledging that CT involves understanding of computing concepts and the ability to apply those concepts in practice, it is important to distinguish knowledge (the “know,” in the head) from skill (the “do”, hands-on). From the assessment perspective, the understanding of computing concepts is often demonstrated as knowledge of facts and information, whereas the ability acquired from practice is the skill for actual problem-solving (Denning, 2017). Students will need to master both the knowledge and the skill to be considered competent of CT. An analogy may help clarify why that is the case: For someone to be considered an auto mechanic, they will first have to have accumulated facts and information about what can cause malfunction (the “knowledge” in the head), but more importantly, they will need to be able to identify where the malfunction is and how to fix the problem (the hands-on “skill”). Therefore, CT encompasses mastering both the “knowledge” and the “skills” (or “practices”).

CT in Elementary Settings

Challenges of Implementing CT in Elementary Schools

As researchers strive on converging on a definition of CT, implementing CT in the elementary level is a daunting task involving many challenges (Lee et al., 2011). First, allocating CT instruction time within the regular school structure is difficult. Most schools in the elementary level already follow a strict curriculum implementation (i.e., content and pacing) that is difficult to fit in a new subject to be taught. Second, most schools do not have the luxury of hiring CS teachers (Lee et al., 2011). Therefore, in the elementary level, CT is often situated in and integrated into existing subject areas taught (i.e., science, math, and so on) rather than being
taught as an independent subject (Rich et. al., 2019; Sáez-López et. al., 2016; Von Wangenheim et. al., 2017). For example, an integrated learning activity can take the form of having students program a robot to tell the life cycles of fern plants (where botany is the target subject) or create a Scratch animation where two sprites compare two fraction numbers (where math is the target subject). This integrated approach faces further challenges. For example, CS is a subject that elementary teachers may not already have expertise in and thus requires professional development (PD) before teachers can implement CT instruction. However, depending on the location, there may be a lack of PD opportunities, resources, and access to infrastructure (i.e., robotics kits, PCs, or mobile devices with learning software installed and Internet access) to support CT implementation (Lee et al., 2011).

**CT Instruction with Plugged and Unplugged Activities**

CT instruction in elementary grades often takes form as plugged and unplugged lessons and activities. Plugged activities require the use of a computer or a mobile device, often happening on visual block-based coding platforms such as Scratch and Code.org or with robotics toolkits. CT instruction with plugged lessons engage students in programming or robotics activities while reinforcing knowledge of the target subject (e.g., Bers et al., 2014; Luo et al., 2020a). Many studies explored the use of visual programming tools, such as Alice and programming-infused robotics curricula, and obtained positive results of improved programming knowledge (e.g., Bers, 2010; Bers et al., 2014; Berland & Wilensky, 2015). The following sections provide a few examples of studies that implemented CT instruction with plugged activities.

Angeli and Valanides’ (2020) study engaged 50 children between five and six years old in learning computational thinking with the Bee-bot robotic devices. The Bee-bot is a small tangible robot with directional keys that control its movements and direction. The robot can store
up to 40 commands and the commands are initiated upon pressing the “Go” button on the robot. The study examined whether there was gender difference in learning with two different scaffolding strategies. One scaffolding strategy was providing command cards that students could lay out before entering commands into the robot; the other was engaging students in collaboratively writing the commands out with help from the researcher before entering commands into the robot. The study concluded that boys benefited more from the first scaffolding strategy (using command cards) and girls benefited from the other (collaborative writing). The study also showed that the majority of the students used decomposition in problem solving (i.e., decomposing tasks into subtasks to tackle). Fessakis and Mavroudi (2013) engaged kindergarten children in two programming games, the Ladybug leaf and the Ladybug maze. In the games, children controlled the movements of the ladybug and led it to reach the leaf or go through the maze on the screen by using directional and turn commands, such as forward, backward, and turn by 90 degrees. By doing so, the children practiced path designing, applied the concepts of orientation and turning, and worked on counting and measuring skills (Fessakis & Mavroudi, 2013).

Von Wangenheim et al.’s (2017) study integrated computing education in social studies for students in 5th and 7th grade to learn CT through programming history-related games in Scratch. The study proposed an instructional unit that involved step-by-step teacher demonstration that introduced basic commands, hands-on programming activities in groups and pairs, and project development. Using the code analytical tool, Dr. Scratch (http://www.drscratch.org/), the study analyzed students’ programs in the areas of “Logic, Parallelism, User Interaction, Data Representation, Flow Control, Synchronization, and Abstraction” (Von Wangenheim et al., 2017, p5). Another study (Sáez-López et al., 2016)
presented a program that integrated CT into 5th and 6th-grade sciences and arts by having students interact and create subject area-related projects in Scratch. The program, implemented in five schools over the course of two school years, focused on improving students’ computational concepts and practices including sequence, iteration (looping), conditional statements, threads (parallel execution), event handling, user interface design, and keyboard input. By using a test with items that assess computational understanding and application and having the teachers and researchers observe and rate students’ learning and attitude, the study concluded that students had significant improvements compared to those who were not in the program. Benton et al. (2018) presented a study that integrated computer programming in mathematics for 5th and 6th-grade students using Scratch. This 2-year “ScratchMaths” curriculum was aligned to the “English National Computing and Mathematics Primary Curriculums” in the U.K. and was carefully sequenced with learning activities for upper elementary students. The study reported that the 2-year intervention had positive results in that the majority of the students engaged with programming the sprites and explored programming in the mathematics context.

Unplugged activities do not require working on a computer, but rather, they incorporate CT concepts in paper-and-pencil forms or physical acting. An example is Code.org’s My Robotic Friend lesson (https://curriculum.code.org/csf-1718/courseb/6/) that asks students to design and draw algorithms to stack cups in different patterns. There are a few benefits of using unplugged activities to teach CT. Such activities are “kinesthetic, engaging, and accessible” (Rodriguez et al., 2017, p.501) and provide opportunities for teachers to present, and students to learn, key CT concepts when access to computers or mobile devices is limited or nonexistent (Cortina, 2015); In addition, students can engage with the great ideas in CS without having to
develop any programming skills or being distracted by technical issues while working on a computer, such as installing software (Bell & Vahrenhold, 2018). Such affordances of unplugged activities make it possible for CT to be integrated into existing elementary school subjects, such as reading, writing, and art (e.g., Barr et al., 2011).

An example is Barr et al.’s study (2011) to develop students’ CT in social studies, music, and language arts classes independently of any computing devices. In this study, they explained that CT could be integrated in language arts class in that using evidence to support a thesis statement corresponds to using clearly and logically organized “data” to support computational analysis (Barr et al., 2011). Similarly, using literary elements such as establishing the plot structure and setting before filling in the detail is essentially a form of “abstraction.”

**Theoretical Perspectives in CT**

The previous sections reviewed a few studies that implemented CT in the elementary level with plugged and unplugged activities. It is important to understand the theoretical perspectives that underlie elementary CT instruction. The two most prominent and representative of such theoretical perspectives are constructionism, which emphasizes a less structured “learning by doing” and targeted, structured instruction with more explicit scaffolding and support from the teacher. The following sections review the research studies influenced by the aforementioned theoretical perspectives.

**Influences of Constructionism**

Implementation of CT in elementary schools is greatly influenced by constructionist perspectives (Papert, 1993) due to the unique affordances that allow students the agency and freedom to explore “learning by doing.” Papert’s (1993) constructionism builds on Piaget’s constructivism and entails more than constructing knowledge in the mind and extends to learning while leveraging physical tool sets in the real world. According to Papert and colleagues (1991;
1993), while constructivism allows for learning by reconstructing knowledge in the brain, constructionism is knowledge building both in the brain and in real world context while constructing a physical or virtual object. In short, it is knowledge constructions “in the head” plus constructions “in the world.” Papert (1993) compared the constructionist approach of learning to the French anthropologist and ethnologist Claude Levi-Strauss’ idea of “bricolage,” which is essentially choosing from a set of tools at hand to fix whatever that may be broken. Papert (1993) explained that it is comparable to a learner drawing from their existing source of ideas and models for learning and constructing new knowledge. He also argued that the goal of constructionism is to produce the most learning for the least teaching, as much space and opportunity should be left for learners to try putting different “tools” to use.

Based on the concept of learning-by-doing, Resnick et al. (1996) proposed two general principles that guide the design and development of computational construction kits and the “doing” experiences conducive to students’ constructing and learning. The two core ideas derived from the general principles are creating personal connections and epistemological connections between the learner and the construction kits. Personal connections speak to the organic relationship between students’ pre-existing interests, passions, intuitions, and experiences and the construction kits. In other words, students should be allowed to connect new ideas and knowledge to prior knowledge and experience by manipulating the objects and actions they are familiar with, thus achieving the goal of learning-by-doing. Epistemological connections encourage students to grow new ways of thinking and connect different domains of knowledge from the interactions with the construction kits and activities (Resnick et al., 1996). The authors emphasize the importance of “unpredictability” in the characteristics of learning environment design in that there should be a relaxed sense of control in what students should learn. As such,
construction kits should allow spaces for students to make their own personal and epistemological connections (Resnick et al., 1996). Thus, using computational construction kits is relevant in elementary CS education as it breaks down the centralized and complex CS discipline in a decentralized way and allows students to explore how to connect CS concepts to their own everyday activities and knowledge.

Influenced by constructionism, Jenson and Droumeva (2016) described a study in which they engaged 6th graders in computer-game construction for about 15 hours over a week using the Game Maker Studio which has drag-and-drop and semantic coding features. The curriculum includes 4 to 5 hours of direct instruction with the rest of the hours spent on students’ individual construction of games. In this study, CT was defined to be practicing object-oriented programming using the four CT concepts: variables, operations, functions, and conditionals. The authors acknowledged that the general game construction learning model was founded on Papert’s constructionism theory and framed the study accordingly.

This study combined structured instruction and unstructured game design time. The authors argued that since the curriculum took place during formal classroom time, in each session, they started with first providing scaffolding of coding instruction on computational vocabulary and students’ application of CT constructs, then students have the rest of each session to game construction on their own. The instruction and materials were carefully paced, and the computational concepts were introduced in an incremental fashion. For example, the concept and application of variables were introduced, then the role and syntax of operations, before moving on to functions next and conditionals in the end. A self-directed learning model was enforced with an “ask 3 before you ask me” rule, which encouraged students to first look up a question, ask a peer, before turning to a researcher for help.
In this curriculum, the instructions and explanation of computational concepts and vocabulary provides students with the “bricolage” tools that they need in order to solve computing problems. The game construction time that students had to themselves allowed space and time for them to construct new knowledge while also constructing something physical in the world. This practice taps back into Papert’s idea of “learning-by-making” and that additional level to constructivism. In addition, asking for help from researchers was used as the “last resort,” forcing students to put their knowledge to test first before getting an easy answer. It is thus appropriate to say that this study leveraged “student-centered” pedagogy given the student autonomy and sense of agency involved in this study.

TangibleK, an engaging and developmentally appropriate curriculum, is another example of implementing CT instruction based on constructionist perspectives (Bers et al., 2014). The curriculum has six lessons on engineering design, understanding robots, sequencing instructions, sensors, looping and branching, while engaging kindergartners in free-play with robotics materials. The study results suggested that kindergarten children could reach the expected level of achievement on understanding programming concepts with proper guidance and assistance, and that they exhibited more enthusiasm when working with the concepts they felt comfortable using after the lessons.

**Influences of Explicit Instruction**

Contrary to the constructionist view of “the most learning for the least teaching,” other researchers see the benefit of structured, targeted instruction. One of such structured approaches is explicit instruction (Archer & Hughes, 2010). How explicit instruction differs from constructionism is that the former emphasizes the role of the teachers rather than student agency in the learning process. Archer and Hughes (2010) defined explicit instruction to be a structured and systematic approach for teaching academic skills. The authors explained that explicit
instruction “is an unambiguous and direct approach to teaching that includes both instructional
design and delivery procedures” (p.1). Explicit instruction usually involves structured
instructional scaffolds that guide students through learning with clear learning goals,
expectations, and timely feedback and support to address learning difficulties (Archer & Hughes,
2010). There are a few distinct elements that characterize explicit instruction, for example, step-by-step demonstrations are provided in instruction to model the skill or concept being taught;
guides and supports are in place to build students’ confidence and ensure students’ performance;
students’ learning outcome is constantly and closely monitored so that teachers know when to
use instructional adjustments (Archer & Hughes, 2010). Explicit instruction is teacher-lead and
lessons usually have a strict structure. For example, the teacher opens a lesson by reviewing the
learning goals and reviewing prerequisite skills; the teacher then models the content, followed by
guided student practice with different prompts and scaffolding; students then practice without
teacher prompts; finally, the teacher closes the lesson. Explicit instruction is therefore a popular
approach to engage students with learning disabilities in special education (Archer & Hughes,
2010; Israel et. al., 2015a). With its distinct affordances on timely scaffolding and feedback,
explicit instruction is regarded as an effective approach to teaching CT to students with learning
disabilities (Israel et. al., 2015a).

Other studies in elementary CT have also leveraged structured and teacher-guided
instruction, which shares some and not all characteristics with explicit instruction, to introduce
CS concepts (e.g., Kalelioğlu, 2015). Given that CT usually involves complex programming
tasks that require sequencing long instructions (algorithmic thinking) and identifying and fixing
problems in programs (debugging), students may frequently encounter frustration during the
learning process. Therefore, structured instruction can be used to tackle these problems in
teaching and learning. Kalelioglu’s (2015) study is an example of using structured instruction. The study presented how Code.org was used as the tool to teach concepts of programming, CT and beyond without necessarily leaving time for student independent learning. The curriculum is a free Code.org course with 20 hours of instruction tutorials and lessons extended to a 5-week period. The course includes topics on computer science concepts such as binary logic, debugging, sequencing, loops, conditionals, functions, variables, decomposition, abstraction, algorithm, and more. The five-week curriculum involved an overall introduction to programming and the course itself in the first week, the first two levels of a Maze programming activity in the second week, and learning to monitor self-progress and continuing with the Maze puzzles in the third week. Then, topics of CT along with the Maze puzzles were taught in Week 4. Week 5 involved detailed instruction on algorithms and more puzzles. Based on the author’s description of the curriculum, the teacher was expected to follow strictly with the materials and activities provided on Code.org, monitor student progress and provide feedback. Much teaching and instruction was involved with no student independent coding time mentioned in this particular case. This is due to the inherent affordances within Code.org, which offers structured courses units that provide tutorials, coding activities, and unplugged activities.

**Arguments for Combining Constructionism and Components of Explicit Instruction**

Constructionism perspectives do not need to be antithetical to explicit instruction (Luo et al., 2020a). Constructionism and explicit instruction can be combined to teach CT-integrated botany to two female elementary learners (Luo et al, 2020a). In this study (Luo et al, 2020a), explicit instruction precedes “learning-by-doing” activities in each lesson. The study reported that the two participants developed CT practices in sequencing, loops, and conditionals and demonstrated the three types of engagement (behavioral, emotional, and cognitive) throughout the integrated learning experience.
In fact, many researchers have argued that a balance of structured instruction and open-ended inquiry is necessary in CT instruction (Israel et. al., 2015a; Guzdial, 2017; Ashman, Kalyuga, & Sweller, 2019). Combining explicit instruction with open-inquiry activities allows students to apply the skills learned to genuine problem solving (Israel et. al., 2015a). Guzdial (2017) made a metaphor to compare project-based learning to exercising in his blog post. He stated that while exercising is highly necessary for good human health, one should not blindly exercise without acknowledging pre-existing health issues such as heart problems and high cholesterol that may cause dire consequences. In other words, one needs to address the issues before exercising. Similarly, instead of assigning students’ complex projects and wish that somehow students struggle through the process, educators should always address learning difficulties with direct instruction first. Other researchers (Ashman et al., 2019) also corroborated that argument, stating that for better student learning outcome, explicit instruction should precede problem-solving when students have high element interactivity (i.e., more elements to process in their working memory).

Learning Trajectories (LTs) as the Theoretical Foundation

Definition of LT

Recent research has explored the use of LTs as the theoretical foundation for systematically implementing and assessing CT in the K-9 level (Rich et al, 2017; 2018). LT is originally an established construct in mathematics research and practice and it is commonly acknowledged to contain three components: learning goals, learning progressions, and the instructional activities that support the learning progressions (Simon, 1995; Clements & Sarama, 2004). Learning goals are a collection of landmarks that children are expected to meet as they engage in instruction. These goals are usually defined with broad agreement, as reflected in previous literature and/or acknowledged standards (Clements & Sarama, 2004).
Learning progressions refer to hypothetical developmental paths that students take as they progress toward increasingly sophisticated understanding during learning (Clements & Sarama, 2004; Maloney et al., 2014). Learning progressions are hypothetical in nature because such progressions are often conceptualized and created by either synthesizing the literature or working with a particular, usually small, groups of students and then applied to a different group (Maloney et al., 2014). Since students’ prior knowledge and engagement in learning cannot always be known in advance, a learning progression can only provide an *a priori* trajectory as to how learning may progress in a different group (Clements & Sarama, 2004). An example is the evaluation of the learning trajectory for young students’ understanding of length (Sarama et al., 2011). In this study, the authors presented the hypothesized developmental progression for length measurement, which was created based on a review of prior research. The LT for length starts with children’s non-ability to identify length or use of the vocabulary, to the ability to recognize length, to directly compare the length of two objects, to measure end-to-end length, and so forth. The authors collected three-year longitudinal data on five students going from pre-kindergarten to first grade and reported that the students generally reach the ability level to directly compare length by kindergarten and the level to measure end-to-end lengths by beginning of first grade, but only two progressed to the ability level of using units in length measuring. The study generated empirical evidence to support the hypothesized LT progression in young children’s understanding of length and that the progression can be replicated.

In addition to the learning goals and the learning progressions, a third key component of a hypothetical learning trajectory is the instructional activities that help children develop higher levels of thinking while moving along the progression (Simon, 1995; Clements & Sarama, 2009). Barrett and Battista (2014) emphasized that such a progression “is tied to, and must interact with,
instruction” (p.102). Without instruction, stages of development of student understanding and competencies may be obviously different. In their book, Clements and Sarama (2014) dissected early mathematics into a series of concepts, including quantity, counting, arithmetic, spatial thinking and so on, and presented learning trajectories for each concept and instructional tasks that support learning and teaching.

The construct of LTs differs from the conventional “scope and sequence” for learning (Daro et al., 2011). While a scope and sequence is a plan for what learning goals to meet and what instructional activities to be carried out, LTs afford empirical evidence that students’ understanding and skills actually develop as learning progressions have hypothesized (e.g., Clements et al., 2019; 2020; Sarama et al., 2011). Research in mathematics (e.g., Clements et al., 2019, 2020) has accrued evidence that highlights the efficacy of the LT approach. In a study that involved teaching shape composition to preschool children, Clements et al. (2019) compared an instructional approach that was based on an empirically validated LT with one that focused on one target goal (skipping the intermediate consecutive target levels). The study concluded that, after engaging the children in over eight brief sessions (9 minutes per each) over five weeks, the children in the LT-based instruction did better in terms of answering completely correctly the assessment items and using strategies than children taught to one target goal. In addition, children in the LT group expressed less counterproductive frustration than the other group (Clements et al., 2019).

Clements and colleagues (2020), in a following study to teach addition and subtraction to kindergarten children, compared the LT approach to a teach-to-target approach that was not specifically aligned to students’ current level of math skills. By engaging the preschoolers in the two different kinds of instruction, the study reported that the LT approach was more beneficial to
student learning gains even though not all students may find it necessary. The authors also concluded that the LT-based approach would facilitate greater learning in early arithmetic than the teach-to-target approach.

An important caveat of an LT is that while an LT provides a potential roadmap for how students may progress through learning, it does not and should not dictate the progression path or the time that all students take (Clements & Sarama, 2014). With students having diverse experiences and learning capability, some may traverse the LT with a shorter amount of time and may skip certain landmarks, while others may need more time and stay on certain landmarks longer. Therefore, LTs should never be imposed onto students as a mandatory progression path, but rather be used as support to understand how students learn and what instructions and activities can help students learn (Clements & Sarama, 2014).

**LTs in CT instruction**

The research in mathematics LT provides a promising outlook for CS education, especially in exploring how children progress in CT learning and what instructions can be designed to support learning. Rich and colleagues (2017, 2018) created four LTs to show K-8 students’ learning goals and progressions in four CT concepts: sequence, repetition, conditionals, and decomposition. The authors explained that these computational thinking LTs are also hypothetical in nature as they were designed by examining literature not intended to be used for the specific aim of developing LTs (Rich et. al., 2017, 2018). The learning goals and progressions in these four LTs are not tied to grade levels in that they do not specify which learning goals should be taught at which grade, but rather, they provide a general road map of CT learning in the K-8 levels. The LTs have arrows that connect multiple statements placed in boxes (Figure 2-1) that are either gray (indicating offline or unplugged goals) or white (computer-based goals). The statements are overarching learning goals (or consensus goals)
synthesized by Rich et al. (2017, 2018) from individual learning goals described in previous research studies. Some characteristics of the LTs were discussed below.

For the sequence, repetition, and conditionals LTs, Rich et al. (2017) categorized the learning goals into beginning, intermediate, and advanced levels, indicating the increasing sophistication of a computing concept and progressions in knowledge. The gray boxes reflect the unplugged goals and the white boxes refer to plugged goals. For example, one learning goal of the Repetition LT (Figure 2-2), “Some tasks involve repeating actions,” is visually displayed in the LT as a gray box because it can be taught and demonstrated without the use of a computer, whereas “Computers use repeat commands” is visually displayed in a white box because of the specific reference to computer programs. In addition, the black or gray arrow between two learning goals indicate “understanding of the source box is necessary” and “understanding of the source box is helpful, but not necessary,” respectively. For example, the conditionals learning goal “A conditional connects a condition to an outcome” requires the understanding of two source learning goals connected with black arrows, “A condition is something that can be true or false” and “Actions often result from specific causes.” The Decomposition LT does not specify any level of sophistication and the arrows show literature-supported connections instead of any prerequisite relationships between the learning goals.

Each LT details the different levels of expectations for student learning and the relationships among the learning goals. The Sequence LT (Figure 2-1), emphasizes students’ understanding and using precise and complete instructions in the beginning level, the order of instruction in the intermediate level, and the manipulation of the order in the advanced level. The Repetition LT (Figure 2-2) stresses pattern recognition, constructing instructions using repetition, and controlling the repetition (i.e., how and when to stop a repetition) using different commands
in the three levels. The Conditional LT (Figure 2-3) starts with the binary status (true or false) of conditions and evaluation of conditional statements for an intended outcome; in the intermediate level, more complexity is added with multiple conditions using different controls; and, finally, boolean variables in the advanced level. The Decomposition LT (Figure 2-4) starts with competencies to break down a system and a complex problem into smaller parts and extends into writing, reusing, and repurposing code and procedures.

**Elementary CT Assessment**

While there is some research in cultivating CT in elementary school students, it has been acknowledged that assessing elementary school students’ CT with developmentally appropriate instruments is a challenging task given students’ limited reading and understanding skills at a young age (Tang et al., 2020). A recent systematic review of CT assessments (Tang et al., 2020) identified four major assessment types commonly used in the literature, including tests, portfolio assessments, interviews, and surveys. The following is an overview of the first three types of CT assessments. Surveys are not discussed because the technicality of survey questions may not be developmentally appropriate for young children (Tang et al., 2020).

**Assessing CT Using Interviews**

Conducting interviews is identified by researchers as a useful approach for revealing students’ CT understanding and how students apply CT (Tang et al., 2020). Interviews can happen in a variety of contexts and formats (i.e., using think-aloud while problem-solving or reflecting and explaining on previously-made projects). For example, previous research reported using interviews synchronously with a think-aloud protocol (Atmatzidou & Demetriadis, 2016) to have students simultaneously articulate their problem-solving processes while they work on a programming task. Think-aloud is one of the three major techniques within the cognitive
interview, the other two being concurrent probing and retrospective probing (Pepper et al., 2018). A few decades ago, cognitive interviewing was popularly used as an investigative technique for eyewitnesses to mentally retrieve retrospective information (“speaking from memory”) for crimes such as robbery (Fisher & Geiselman, 1992; Geiselman et al., 1986). The cognitive interviewing was later adapted for survey and assessment development purposes (Carbone et al., 2002), more specifically, to provide validity evidence for assessment item development by gathering interview responses (Peterson et al., 2017; Pepper et al., 2018).

Concurrent probing means the interviewer asks questions when the interviewee is working on an item; Retrospective probing is asking the interviewee after the interviewee finishes item response to reflect and retrieve their memory on their experience during item response; Think-aloud asks the interviewee to proactively and metacognitively verbalize their thinking as they respond to an item. By using such think-aloud procedures and prompts, the verbal data collected will reveal how respondents understand, reason, and respond to an item, thus providing insights into what revisions need to be made and whether the item has clarity and relevance in what it is assessing (Pepper et al., 2018). Respondents’ answers and thinking will also reveal, if any, misalignment between what an item is conceptualized to assess (during item development) and what it actually assesses (Peterson et al., 2017). More recently, the cognitive technique with a think-aloud protocol was used by researchers to reveal students’ problem-solving thinking processes (e.g., Atmatzidou & Demetriadis, 2016; Luo et al., 2020b).

Interviews were also used to ask students to reflect on and explain their computational artifact-making considerations (i.e., how they started a project, how the project evolved during development, what were the problems encountered and solutions used, and so on). The artifact-based interview approach (Brennan & Resnick, 2012) allowed a lens to assess students’ CT and
identify potential conceptual gaps. For example, students may use certain code blocks in their project without being able to fully explain how those code blocks work. One limitation of using the interview approach was that, compared to assessments with *a priori* scoring mechanism, those with open-ended activities could be time consuming (Brennan & Resnick, 2012; Tang et al., 2020). Also, if students were interviewed about a project that they had previously made, the assessment was limited by what the interviewed student was able to remember at the moment of the interview (Brennan & Resnick, 2012). Brennan and Resnick (2012) proposed a third way for researchers to observe students’ CT in action through the design scenario approach. Unlike the artifact-based interview approach where students describe the strategies used in previous-created projects, the design scenario approach allows researchers to observe students practicing CT in real time. The design scenarios included three sets of author-developed Scratch projects with increasing difficulty. With two projects in each set, the student being assessed would select one of the projects from each set. This approach assessed students’ understanding of Scratch projects (codes), ideas to modify and extend the project, debugging skills, and the ability to remix the project. For example, students would need to explain what the selected project would do, how it could be extended, how a specific bug could be fixed, and how a feature could be added. The strengths of this approach, as explained by the authors, included a systematic examination of students’ understanding of CT concepts and practices in real-time (Brennan & Resnick, 2012).

**Assessing CT Using Portfolios**

Another common type of assessment identified by the recent systematic review on CT assessment (Tang et al., 2020) was portfolio assessment. This type of assessment evaluates students’ CT through examining student-created products against a certain rubric or criterion, e.g., assigning codes to represent the different proficiency levels for projects created by students
(Werner et al., 2012; Bers et al., 2014), or examining students’ work in various design scenarios (Brennan & Resnick, 2012).

An example of using a rubric was Bers et al.’s (2014) study that examined students’ CT ability in four domains: debugging, correspondence (i.e., choosing the correct instructions as per task requirements), sequencing, and control flow. The study assessed children’s learning outcomes after having children engage in a six-lesson (20-hour) robotics curriculum and programming activities. To successfully complete the activities, children needed to select the correct instructions and correctly sequencing the instructions in order to make the robots do a specific dance. The evaluation involved a six-point Likert scale with 0 referencing “no attempt,” 1 “did not complete the goal, task, or understanding” through 5 “Complete achievement of the goal, task, or understanding” of the CT concepts involved in an activity. The study reported that 76% of students achieved the target goals of selecting coding instructions in activities that did not involve control flows, such as, loops and conditionals; whereas that percentage of students achieving the target goals dropped when activities became more complex by involving the use of light sensors and sensor parameters and the use of conditional statements. However, there was no statistically significant difference between students’ understanding of loops and that of conditionals.

Werner et al.’s (2012) study also used a scoring scale that categorized students’ projects based on cognitive achievement levels. The study engaged middle school learners in Alice programming using the Use-Modify-Create framework over 20 hours throughout a semester. The study used the Fairy performance assessment with three programming tasks to assess students’ CT comprehension, namely algorithmic thinking, abstraction, and modeling. For example, Task 1 asked students to program the on-screen agent “HaloSilver” to turn to watch the other agent
“LeafFlame” while walking into the forest. Completing this task requires that students to correctly sequence instructions, which was essentially algorithmic thinking; the task also requires understanding of abstraction and modeling, as students needed to modify the length-of-time parameter to properly synchronize the actions (i.e., the turning should last until the other agent finishes talking). Students were evaluated to a 10-point scale, with 10 meaning that students demonstrated the desired level of cognitive achievement. The study reported that students scored the highest on Task 1, which measured CT comprehension, and the lowest on Task 2, which measured more complex cognitive abilities, such as design, problem-solving using debugging, in addition to comprehension.

The project portfolio analysis approach was also proposed by Brennan and Resnick (2012), where the authors used the user analysis tool, “Scrape,” to analyze students’ Scratch projects and to visualize what blocks were and were not used in projects. However, one limitation of this approach was that it revealed only results, rather than students’ process of developing the projects. The portfolio assessment approach examines students’ CT skills from a holistic view (Tang et al., 2020). However, such an approach is only applicable when programming platforms such as Scratch are used.

**Assessing CT with Selected- or Constructed-Response Tests**

A third type of assessment identified by Tang et al. (2020) in their systematic review is the selected- or constructed-response tests. Such tests usually constitute multiple choice questions, open-ended questions to summatively evaluate students’ CT (Tang et al., 2020). An example is Chen et al.’s (2017) study where the authors developed and implemented a six-month robotics curriculum with 125 students in fifth grade. Students used a visual programming platform to write and test their program before taking turns to test programs on a physical robot. The curriculum involved key CS concepts such as algorithms, variables, conditionals, loops, and
so on. To assess students’ CT, the authors designed and developed an instrument with 23 items (15 multiple choice questions and eight open-ended questions) grouped into six sets. The items were designed and developed using a five-component framework that aimed to assess students’ CT in terms of syntax, data, algorithms, problem representation, and efficiency. The dichotomously scored items are mainly situated in two problem contexts: everyday scenario and programming scenario. The everyday scenarios, phrased in a typical word problem, involve daily life activities that students could easily relate to. For example (Chen et al., 2017):

(Information from the stem of the item set: You are running 4 loads of laundry in a fast washing machine and each load of clothes takes 10 minutes to run) If you have 2 washing machines that can work simultaneously (i.e., at the same time), how long will it take to run all the loads?

While the use of such a traditional assessment provides quantifiable results of students’ CT, it fails to capture the processes of how students use CT (Tang et al., 2020).

Gaps in the Elementary CT Assessment Literature

To summarize, the previous sections described each of the three types of CT assessments commonly used to assess elementary students’ CT and the different contexts they were used. The strengths and weaknesses of the three types were also discussed. These different assessment types provide insights into what instruments or tools may be more appropriate given the different purpose of CT assessments: Interviews can be used to reveal students’ thinking processes in problem solving and to identify conceptual gaps but are usually time-consuming; Portfolios can be examined to holistically understand how students use CT in programming but are restricted to programming tasks; Last but not the least, tests can be used to summatively evaluate the students’ CT as a learning product but miss the CT in action. The systematic review of CT assessments (Tang et al., 2020) concluded that interviews is an under-utilized approach that affords in-depth qualitative data analysis and called for more attention in leveraging the
affordances of interviews to assess CT. Another gap identified in the CT assessment literature is that, while a small portion of studies employ a combined approach (e.g., tests with surveys and interviews) to study students’ CT, there lacks substantial research in understanding the process of how students develop and apply CT (Tang et al., 2020). Therefore, this dissertation proposed combining traditional tests and interviews using a think-aloud protocol to understand the processes of students’ CT in problem-solving by collecting in-depth, qualitative evidence. The use of cognitive interviews is methodologically similar to several previous studies that explored students’ CT (e.g., Atmatzidou & Demetriadis, 2016; Luo et al., 2020b). The method is explained in detail in Chapter 3.

Chapter 2 provided a review of previous literature, including an overview of definitions of CT proposed by researchers (e.g., Wing, 2006; 2008; Brennan & Resnick, 2012) and professional organizations (e.g., CSTA, ISTE). Chapter 2 also presented example studies of CT implementation in elementary grades in plugged and unplugged forms, the underlying theories in CT, elementary CT assessments, and how research in mathematics LTs influenced the construction of the CT LTs. Chapter 3 describes the details of the method used to study elementary students’ articulated CT using affordances of the hypothesized CT LTs.


CHAPTER 3
METHOD

Method Overview

The research questions of this dissertation study are: (1) How do 3rd- and 4th-grade students express and articulate CT in math-CT problem-solving scenarios? And (2) how does students’ articulated CT correspond to the learning goals and the learning progressions of the CT learning trajectories? Chapter 3 explains in detail the method used to answer the two research questions. First, the participants and the research setting are presented. Then, the rationale for using the qualitative methodology is explained, followed by the overall design of the study implementation using the integrated math-CT lessons and assessments. Data collection using the cognitive interview and data analysis procedures are also explained in detail.

Participants and Settings

The participants of this study were sampled from two of 3rd-grade and two 4th-grade classes at an elementary school in the midwestern United States. According to an official state database, during the 2018-2019 school year, the school had a student population that was 40.2% White, 30.9% Black, 9.8% Asian, 6.2% Hispanic, and 12.9% two or more races. The school had a diverse representation of students, i.e., 19% from low income families, 19% enrolled in special education with Individualized Education Programs, and 7% English learners. Since 2013, the school has adopted a school-wide CT and CS initiative where the school faculty and community are committed to fostering a computational thinking mindset in students in the K-5 levels. All students learn coding or programming and have access to computers. During the 2019-2020 school year, the teachers in the four classes in Grade 3 and 4 implemented the integrated math-CT instruction and assessments during math classes at school. Given the school’s history in CS
education participation, in general, students from these classes have had prior CT experience and exposure in all grade levels.

As this study examined students’ CT learning within the math+CT integrated context, the recruitment of participants had the following criteria: 1) Participants took part in at least three of the four integrated lessons for each instructional stage (i.e., phase 1 and phase 2); 2) Given the nature of the cognitive interviews, participants had the ability to verbalize their thinking; and 3) Participants provided a written consent and assent under the Institutional Review Board-approved protocols. As a result, a total of 12 Grade 3 students and 10 Grade 4 students were recruited.

Qualitative Methodology

This dissertation study used the qualitative inquiry to investigate students’ CT understanding and how that understanding corresponded to LTs synthesized from literature. Qualitative research is an inquiry approach used to obtain the meaning of problems or situations that individuals or groups may have. Creswell (2013) defined qualitative research as to begin with assumptions, possibly using a theoretical lens to obtain meaning of a specific problem or phenomenon. Creswell (2013) stated that, to study the problem, qualitative researchers collect data and evidence of a natural setting and try to make meaning by examining and analyzing the data and evidence collected (i.e., to establish patterns or themes). The data and evidence may involve people’s voices of their experiences, and the results of such inquiry usually involve the reflexivity of the researchers and a detailed description and interpretation of the problems (Creswell, 2013). Qualitative research has a few characteristics (Hatch, 2002; Creswell, 2013):

1) The natural setting: as the goal for qualitative inquiry is to make the meaning of people’s situations/problems/experiences, data are usually collected from a natural context (i.e., cultural, social, political, and so on).
2) The human factor: It’s the participants’ story to tell. Qualitative inquiry allows the participants targeted to tell their story from their unique perspectives. Researchers are the “instrument” that collect information and researchers interact directly with participants by engaging participants through talking and/or observing.

3) Data source: People’s views and voices may be obtained through interviews, observations and other forms of documentation. These different information sources can all provide data to inform the problem researchers inquire into.

4) The interpretive nature: Although qualitative research focuses on the participants’ views and voices on a problem or issue, the researchers are the ones that interpret or make the meaning from participants’ views and voices. Therefore, the interpretations are not expected to be entirely objective, as different researchers have their own background, prior biases, and so on that may influence how they make the interpretations. Rather than attempting to eliminate all possible subjectivities, qualitative inquiry encourages researchers to acknowledge how such subjectivities may have influenced data collection and analysis.

This dissertation was categorized as a “basic interpretative study” (Merriam & Tisdell, 2015), also named as “a basic qualitative study,” with a general purpose to understand people’s lives and/or experiences. In the case of this dissertation study, that experience was students’ demonstrating and articulating their CT competencies working with CT assessment items after engaging in math-CT integrated lessons.

**Design**

This dissertation study leveraged three major components of the design: the LTs as the foundation for instruction and assessment design, the integrated math-CT instruction, and the assessments (Figure 3-1). The following section explained the design in detail.
Integrated Math-CT Instruction Based on LTs

This study used the integrated math-CT lessons designed and developed under the LTEC NSF-funded grant. The lessons are aligned to the Common Core State Standards for Math (CCSS-M) and target a range of learning goals and progressions in the LTs that are developmentally appropriate for G3 and G4 students. The integrated lessons are part of a year-long implementation that is tied to Everyday Mathematics (EM, 4th edition) content progression. Specifically, the integrated lessons are designed to teach fractions and CT to 3rd and 4th grade students. There are in total 12 lessons for Grade 3 and 13 lessons for Grade 4. Computational thinking concepts and practices such as sequence, repetition, decomposition, conditionals, and variables are interwoven into the fraction lessons. There are a combination of plugged lessons (i.e., requiring the use of a computer) highlighting hands-on coding exercises where students use the Scratch programming platform to build projects while learning math and building CT skills and unplugged lessons (i.e., does not require the use of a computer) with discussion prompts and reflections. The unplugged lessons are as important because they offer opportunities for students to make connections of CT to the pre-existing intuitions in their daily life and establish new knowledge based on familiar objects and actions (Resnick et al., 1996). The integrated lessons were implemented by teachers at the school. Two researchers from the LTEC project were present during implementation and facilitated lesson implementation. These researchers were available to answer questions prior to instruction. Prior to the teaching the integrated lessons, the teachers participated in professional development related to both using the Scratch programming environment and the specifics of the lesson components. Teachers were provided with lesson plans and PowerPoint presentation slides developed by the research team.

In general, a lesson has the following structure: It begins with a cover page that starts with the “Math Connections” and the “CT Connections” that introduce the content and activities
in this specific lesson relevant to Math and CT. Then, relevant computing vocabulary is listed.
The rest of the cover page includes an at-a-glance plan of the lesson with explicit learning goals, anticipated barriers, and student options. For example, the 3rd-grade lesson “Animal Number Story” (Figure 3-2) starts with:

Math Connections: Children use Scratch to show a simple number story.

CT Connections: Children explore the Scratch workspace. Children learn that the Scratch workspace and computer programs are made up of different parts. They search for and try out different commands and learn which changes of and within commands can change the program’s output.

The at-a-glance plan of the lesson list four major components: “Login into Scratch,” “Programming Events,” “Introducing TIPP&SEE,” and “Animal Number Story TIPP&SEE” as the foci of this lesson. TIPP&SEE is a learning strategy with seven steps to help students navigate the Scratch interface while learning from existing codes in projects (Salac et al., 2020). The seven steps are: check the title of a project (T), read the instructions of the project (I), understand the purpose of the project (P), play/run the project to see what the codes do (P), manipulate the codes for Sprites (S), look at the event blocks that start the scripts (E), and explore by changing the codes (E). The explicit learning goals in the lesson, which are in the format of “I Can” statements, explain what students are expected to be able to do after the lesson. For example, “I can identify the important parts of Scratch” and “I can closely observe a Scratch program and find the scripts that caused the actions.” For anticipated barriers, one example is, “Logging in to a new program can be challenging for some children. Providing pictorial or video versions of the verbal/written directions would be helpful.” Student options list a number of ways to adapt the lesson to students’ varied preferences, such as “Include simplified directions both within the Scratch projects and on a paper.”
The G3 participants sampled for this study participated in, altogether, eight CT-integrated fraction lessons. The eight lessons cover the initial learning goals in the Decomposition LT, one learning goal in the Repetition LT, and a range of learning goals in the Sequence LT. The specific learning goals and progressions in the LTs that each lesson covered were listed in Table 3-1.

Similar to the G3 participants, the G4 participants sampled for this study participated in eight CT-integrated fraction lessons. The eight lessons cover a range of learning goals in each of the relevant LTs, as listed in Table 3-2. In addition to the three LTs listed, the G4 lessons have a heavy focus on introducing the concept of variables. However, the variable content was not discussed because it was beyond the scope of this study.

**Integrated Math-CT Assessment**

This dissertation study deliberately selected assessment items from the Integrated math-CT assessment set developed under the same NSF-funded grant. The integrated assessments were conceptualized and designed from the computational thinking LTs to assess students’ understanding of the five CT concepts: sequence, repetition, conditionals, decomposition, and variables as students progress through the integrated lessons. Given the broad context of the grant project, which integrated CT in elementary math, some assessment items were embedded in the math context (Gane et al., under review). The design of the math-CT integrated assessment items acknowledged the fact that few CT assessments in the literature followed a theoretical framework or reported validity evidence, which resulted in the lack of confidence in interpreting students’ CT performance (Tang et al., 2020). Therefore, the design of these integrated assessments was grounded in the LTs and the evidence-centered design (ECD) framework and used an argument-based approach to validation by following established design patterns and collecting empirical evidence (Gane et al., under review).
The paper-based assessment items include true-or-false, multiple-choice, open-ended, and fill-in-the-blank questions using either illustrations of the Scratch interface (e.g., code blocks), word problems embedded in the math context, or simple arithmetic problems. The construction of the assessment items followed a design pattern that specified the knowledge, skills, and abilities (KSAs) grouped from the learning goals in each LT (Gane et al., under review). For example, for sequence, one of the KSAs is: Ability to create several different sets of instructions to produce the same intended goal (2U). The number in the number-letter combination “2U” indicates that this KSA maps to the learning goal 2 in the LT, “Different sets of Instructions can produce the same outcome,” while the letter “U” indicates application of that piece of knowledge.

Each assessment item has its distinct mapping to specific KSAs (see Appendix F for KSA-Item Mapping). For example, the sequence item S.01.a (Figure 3-3) asked participants to write two different programs using the given Scratch code blocks to move the cat sprite from 0 to 5 on a number line. This item was mapped to all five of the KSA statements in the design pattern (see Appendix D for KSAs) of the sequence assessment items. The five KSAs are:

1. Ability to create several different sets of instructions to produce the same intended goal;
2. Ability to use / modify / create precise instructions to produce the intended goal;
3. Ability to use / modify / create an ordered set of instructions to produce the intended goal;
4. Ability to use a limited set of commands to produce the intended goal;
5. Knowledge that computer programs [computational artifacts] require selecting from and ordering a limited set of commands.

For each grade (G3 and G4), an early, a mid, and a late assessment were designed and planned to be implemented by teachers at three timepoints of the school year: late October 2019, February and May 2020, respectively.
Cognitive Interview with the Think-Aloud Protocol

In this dissertation study, the cognitive interview utilized a think-aloud protocol (Appendix A) that combined all three of the above-mentioned techniques (concurrent and retrospective probing and think-aloud) to elicit students’ CT understanding as students engage in solving a CT problem/item. Since the cognitive interview required one-on-one interview time, each participant was expected to be excused from class for a reasonable amount of time (approximately 20 minutes). Given experiences from the pilot study, 20 minutes was enough time for a student to finish the think-aloud for four assessment items. The think-aloud protocol started with the interviewer self-introduction and a brief introduction of what students were expected to do in the cognitive interview. Next, the interviewer modeled “thinking aloud” while working on a sample assessment item. The interviewer then told the student that they would likely be prompted to verbalize their thinking if they started to quietly work on the item instead of thinking aloud. The concurrent prompting is necessary because compared to adults, elementary students may not fully understand the purpose of the think-aloud and therefore need additional prompting in terms of providing metacognitive verbalization. The goal of the concurrent prompting questions was not to influence students’ thinking and response or help them come to the correct answer, but to remind students to articulate, as much as possible, their genuine thinking. For example, concurrent probing questions can be, “Can you tell me what you are thinking right now?” “Can you say more?” and “Can you tell me how you came up with the answer?” The goal of the retrospective questions was to understand further what have influenced students’ thinking and problem-solving strategies. Such questions can be, “Can you describe again what you just wrote?” and “What class instruction has helped you answer this item?”

The cognitive interview with a think-aloud protocol was an appropriate instrument as students verbalized their understanding of the item instruction and the reasoning processes they
went through in order to get to an answer. That reasoning reflected students’ CT understanding of the specific CT concepts.

The Pilot Study

In Spring 2019, a pilot study (Luo et al., 2020b) was conducted at a school in the southeastern U.S. The pilot study involved 13 fourth-grade students (four girls and nine boys) who had gone through four math-CT integrated lessons over the course of a month. Among the 13 students, there were eight white, two Hispanic, one African American, one Asian American, and one mixed ethnicity (as identified by their teachers). Eleven items that covered four CT concepts: sequence, repetition, conditionals, and decomposition, were sampled from the original assessment set. Each participant was given three assessment items randomly selected from the 11 items. A total of 31 student responses were collected.

The pilot study had a few conclusions in terms of students’ CT understanding: 1) items on sequence appeared to be the easiest in that students understood that the order of instructions can affect the outcome and that the order matters in programming; 2) the repetition items received responses with varied levels of understanding of the cumulative effect of repetition (repeat blocks) in programming; 3) while students understood that a condition connects to an outcome, they did not demonstrate understanding in how to evaluate a condition to be true or false; and 4) students were only beginning to understand how to decompose a system but did not understand what decomposition of a problem entailed (Luo et al., 2020b). The conclusions suggested that these four CT concepts to be further investigated in the dissertation study.

The pilot study generated important implications for the dissertation study. First, the pilot study implemented four of the math-CT lessons before students were interviewed one time during the semester (Luo et al., 2020b). To gain a deeper understanding of how students articulate CT while engaging in the integrated CT instruction, in the dissertation study, an
additional data collection cycle was added. In other words, students engaged in four integrated lessons before participating in a cognitive interview, then another four lessons before students were interviewed a second time. The two phases of data collection are explained in a later section, called “Data Collection Procedures.”

The results of the pilot study revealed an incongruity between the data and one node within the Conditionals LT. The Conditionals LT theorizes that the understanding of "a condition is something that can be true or false" precedes that of "a conditional connects a condition to an outcome." However, the data suggested the opposite. The results from the pilot study had a few implications for this dissertation study: 1) repetition and conditionals tend to be where students diverge the most in terms of CT understanding; and 2) more evidence needs to be examined to see if that incongruity stands in a different sample.

By presenting preliminary results in students’ understanding and non-understanding of the CT concepts, the pilot study also informed how assessment items should be selected in this dissertation study. In the pilot study, convenience sampling was used to select items from the assessment sets and then randomly selected for each participant (Luo et al., 2020b). The problem revealed with convenience sampling was that some items sampled did not elicit a wide range of CT, therefore limiting the CT competencies and reasoning a participant could articulate while working on such items. As such, this dissertation adopted a more rigorous method of item selection. Instead of convenience sampling, items were examined in terms of their KSAs before the ones that cover a range of CT competencies were selected. The assessment item selection criteria were explained in more detail in the following sections.
Item Selection and Grouping

Item selection criteria

Item selection was bounded by the selected concepts each research question aimed to address. Recall that the research questions involved the sequence, repetition, and decomposition LTs for Grade 3 and the repetition, conditionals, and decomposition LTs for Grade 4. Therefore, the inclusion criteria for an item selected for the cognitive interview included a) selected items needed to assess the CT in the three LTs of interest for each grade; b) selected items should reflect a wide range of CT and therefore have the capacity to elicit different levels of CT understanding, if possible; c) selected items should include ones that used illustrations of Scratch code and word problems not necessarily associated with code blocks; d) true-or-false items were excluded as such items with binary answers were less likely to generate rich thinking-aloud verbalization. Following the item selection criteria, 13 items (Table 3-3) mapped to a range of KSAs (see Appendix F for Item-KSA mapping details) were selected for the cognitive interview, including three sequence, four repetition, two conditionals, and four decomposition items (See Appendix J for the full item list). Among the 13 items, four were embedded in the Scratch context, six were presented as word problems, and three as simple arithmetic problems. Note that, given the LTs of interest in the research questions, the conditional items were not used in G3 cognitive interviews, and the sequence items were not used in G4 interviews.

Item grouping method – Phase 1

For phase 1 data collection, the nine assessment items include two sequence, two repetition, and three decomposition, and two conditionals. Items were grouped to reflect a range of KSAs (see Appendix F for KSA-Item mapping details). In other words, items were intentionally grouped to allow participants with a range of levels of CT proficiencies to articulate their problem-solving reasoning. The nine items were grouped into sets, with each set consisting
of four items for each of the G3 participants and three items for each of the G4 participants.

Specifically, the grouping followed the rules listed below:

- Grade 3 sets should have at least one item for each of the three concepts of interest. This means one sequence, one repetition, one decomposition, and one additional item from any of the three LTs.

- For grade 3, the sequence items had similar KSA mapping and so did the repetition items, therefore one item was picked from each of the two LTs. The decomposition items had varying levels in required CT proficiencies with DC.06.c being the item that was the most comprehensive in the proficiencies required. Therefore, this item was included in every set.

- A fourth item was selected from the decomposition items to provide an additional opportunity for students to articulate CT.

- Grade 4 sets should have at least one item for each of the three concepts of interest. This means one repetition, one conditional, one decomposition.

- For grade 4, since there are no repetition or decomposition items included in the original G4 Early assessment, those in the G3 Early assessment were used. Similar to what was done for grade 3 items, item DC.06.c was included in every set. One of the two repetition items was selected. In addition, the two conditionals items required similar proficiencies, therefore, one was selected from the two.

- Each set has at least one Scratch-based and one unplugged item.

**Item grouping method – Phase 2**

For phase 2 data collection, one sequence, two repetition, two decomposition items, and two conditionals were selected with some overlapping with those used in phase 1 cognitive interviews. The grouping followed the rules listed below:

- Similar to the phase 1 data collection, each grade 3 set includes one from the two sequence items, one from the two repetition items, and two of the decomposition items. The rationale for including those two decomposition items is that the items have matched the level of the KSAs assessed, yet they are very different in terms of the math content difficulty. Therefore, data may likely reveal how students express and articulate decomposition in those two different problem-solving situations.

- The two decomposition items used for G3 were included because those two items cover a wide range of decomposition KSAs.

- One from each of the LTs was selected for G4 phase 2 data collection.
Data Collection Procedures

Cognitive interviews were conducted at two different stages during the 2019-2020 school year. For each stage, each of the 12 G3 participants was presented four assessment items grouped from the sampled items and each of the 10 G4 participants was presented three. All interviews were audiotaped and then transcribed.

Phase 1

Four researchers including the author participated in a training session before conducting the cognitive interviews using the think-aloud protocol (Appendix A) at the research site. The training session included an introduction to the protocol, a think-aloud interview demonstration, and an in-depth discussion on how to adhere to protocol prompts and avoid deviations from the protocol. All four researchers had prior experience in teaching and interacting with elementary students. In mid-November 2019, phase 1 cognitive interviews took place after students had finished four lessons in the integrated math-CT curriculum. Each student participated individually in the cognitive interview wherein they were thinking aloud while solving each of the given assessment items. Each cognitive interview followed the procedures below. First, the researcher/interviewer briefly introduced themselves to the participant. Then, the researcher modeled “thinking aloud” with a sample assessment item by verbalizing the thought processes and rationale for solving the item. Then, the participant was given the paper-based items grouped according to the method described in the previous section. For each item, the interviewer reminded the participant to think aloud if there was a period (more than five seconds) of silence. After a participant finished an item, the interviewer asked retrospective questions in order to get a more complete understanding of the student’s thinking. The cognitive interview ended after a participant finished all four items and answered the retrospective questions.
Phase 2

Phase 2 interviews took place in late February 2020, after another four lessons were taught for each grade. Similar to phase 1 data collection, each student participated in a cognitive interview again. Three researchers conducted the interviews and the interview process followed the same process as in phase 1. Each interview started with a brief introduction of the researcher, followed by a demonstration of the think-aloud activity. Participants then worked on the given assessment items with the researcher making prompts for think-aloud whenever necessary. Retrospective questions were asked at the end of each interview for a more holistic understanding of student thinking. The only change made to the think-aloud protocol was that only one retrospective question (i.e., “What do you think this question is asking you to do?”) was asked instead of three, as the researchers concurred that the other two retrospective questions did not elicit meaningful data in phase 1.

Data Analysis

Qualitative data analysis is “a complex procedure that involves moving back and forth between concrete bits of data and abstract concepts, between inductive and deductive reasoning, between description and interpretation” (Merriam & Tisdell, 2015, p.202). On one hand, constructing themes is a highly inductive process, that is, beginning with small data segments, then clustering data that provide similar meaning, and forming tentative categories or themes. On the other hand, testing the tentative categories or themes against all data is a necessary process that involves deductive reasoning (Merriam & Tisdell, 2015). This dissertation study employed a primarily deductive approach by using a priori codes to categorize data. Elements of inductive reasoning (i.e., constant comparison of data excerpts) was also used to construct themes. The following sections explained the roles of the a priori codes and the interpretation criteria, and how constant comparison was done in data analysis.
**A priori Codes**

Recall that students were purposefully engaged in a math-CT integrated lesson progression specifically developed to help students meet the learning goals and move along the progressions in the LTs. Therefore, the learning goals in each of the four LTs were used verbatim as *a priori* codes to categorize students’ articulated CT. Tables 3-4 through 3-7 listed all the *a priori* codes/categories transcribed verbatim from the learning goals from the four LTs (Rich et. al., 2017, 2018).

For example, the sequence item S.04.b asks students to provide two different sets of instructions to carry eight toys from the kitchen to the bedroom, with a maximum of three toys each time. This item is mapped to multiple KSAs, indicating that successfully answering this item will require the student to create different sets of instructions to produce the same intended goal and to create complete, precise, and ordered instructions to produce the intended goal. Therefore, if a student articulated two different sets of ordered, complete, and precise instructions to achieve the same outcome, the codes S1, S3.1, and S2 were assigned to the data excerpt.

**Interpretation Criteria**

The interpretation criteria for assessment items were established and iteratively revised during a two-year validation process under the NSF-funded grant. The interpretation criteria include a detailed description for how students’ written answers to each assessment item will be categorized. The goal of the interpretation criteria was not to judge student answers to be right or wrong, but rather, to understand and document the differences of student thinking and problem-solving strategies. For example, students who strictly followed the item instructions (i.e., using only the given code blocks) and provided answer rationale with correct CT and math thinking would be coded as “3;” students who provided the intended rationale using non-given code
blocks would be coded as “2;” students who demonstrated either “only CT” or “only math”
thinking would be “1,” and answers with no evidence of CT would be “0.” The interpretation
criteria provided a framework to decide whether students’ articulation could be assigned a
specific a priori code and to which group a specific response to the assessment item belonged in
the data analysis procedures discussed in the “Analytical process” section. Given the goal of the
coding process was to capture and identify participants’ CT articulation, the interpretation
criteria was used in a way that did not penalize students for incorrect mathematical results. In
other words, if a participant demonstrated evidence of CT but reached an incorrect mathematical
answer, their CT articulation would still be recognized and coded.

**Constant Comparison**

The constant comparison analysis process used in this dissertation study was adapted
from the traditional constant comparative analysis (CCA). The traditional CCA method involved
four steps: 1) comparing each coded segment of data to the previous segments to form a
category, 2) integrating categories with similar dimensions, 3) describe and elaborate the
relationships between the categories with details, and 4) writing the general theory (Glaser, 1965;
Merriam & Tisdell, 2015). While originally intended by Glaser (1965) to develop theories in the
grounded theory approach, application of the CCA method has spread widely outside of
grounded theory and has had various adaptations. For example, Fram’s (2013) commentary
reviewed the different adaptations of the CCA used in conjunction with a theoretical framework
during data analysis and provided an example of using the CCA method in naturalistic inquiry
rather than in grounded theory. Merriam and Tisdell (2015) explain that the CCA method of data
analysis applies in all kinds of qualitative studies, even when the purpose is not to develop a
theory. Two other studies (Tokarczyk, 2012; Horn, 2011) provided a more relevant context by
using the CCA within a basic interpretative study to determine major themes derived from the
data. The CCA method is regarded as a systematic strategy for analyzing any data set in qualitative research when done in a rigorous fashion. To achieve robust data analysis, it is important to ensure the presence of a) a core conceptual element that connects all categories supported by the data, b) properties that define the categories, and c) hypotheses that explain the relationships among the categories and properties (Merriam & Tisdell, 2015).

Boeije (2002) used the constant comparative method (CCM) as a purposeful data analysis method to analyze qualitative interviews and listed five steps of the CCM application to improve the traceability and verification of qualitative analyses. Boeije (2002) interviewed for the experience of multiple sclerosis (MS) patients and the experience of patients’ spouse who were care-providers, she used a five-step interview analysis procedure: 1) Comparison within a single interview, 2) comparison between interviews within the same group, 3) comparison of interviews from different groups, 4) comparison in pairs at the level of the couple, and 5) comparing couples. Each of the five steps had an aim, important questions to ask, and results. For example, for the first step, she started by comparing fragments within one interview and did open coding, where she examined the consistency in the different fragments, (i.e., how they have in common and/or how they differ). At this point, the aim was to develop categories and assign the most appropriate codes to the fragments within one interview. The important questions in this step included, “Which codes are used to label the categories in this particular interview? What characteristics do fragments with the same code have in common?” “What is the core message of this interviewee?” and “Is the storyline consistent? Are there any expressions that are contradictory? How are all the fragments related?” (Boeije, 2002, p.397). The results of this phase of analysis include a summary of the interview, a list of initial codes, an initial conceptual profile, and memos that described the analysis process. Such a documentation of Boeije’s (2002)
adaptation of the CCA is a pragmatic approach to break down and operationalize the CCA method to analyze qualitative data outside of grounded theory (Fram, 2013)

**Analytical Process**

Preceding sections explained how *a priori* codes and the CCA were operationalized to analyze qualitative data. In this dissertation study, data analysis followed a two-step analytical process, where *a priori* codes were used to put data in categories and CCA was used for constructing themes. The following sections explain the processes in detail.

**Analysis using *a priori* codes**

The learning goals in the LTs were used as *a priori* codes to categorize students’ articulated CT at the beginning of data analysis. The interpretation criteria was used to support the identification of data that fit into different categories. At this analysis stage, the interpretation criteria provided additional reference to help decide whether an *a priori* code should be assigned. The roles of the different components are visualized in Figure 3-4.

In this step, students’ CT articulation in each response was examined against the *a priori* codes. The important questions in this step were: What understanding/proficiencies of this specific CT concept did the participant’s articulate and demonstrate? What *a priori* codes best describe the characteristics of the participant’s articulation in this particular interview? What characteristics did excerpts in this interview have in common? For example, if a participant articulated an ordered, precise, and complete program using the code blocks given in the problem instructions, then *a priori* codes pertaining to “precision and completeness in computer programs” and “order of execution” were assigned to those excerpts.

**Analysis using constant comparison**

The construction of themes using constant comparison took place after all interviews were coded using *a priori* codes. Previous research has operationalized CCA to constantly
compare data segments within a single unit (i.e., an interview), then across units (i.e., multiple interviews), to put data revealing similar meanings in the same category (Boeije, 2002).

In this dissertation study, the single unit referred to responses to one item within a CT concept. In this step, excerpts assigned the same a priori codes were grouped in the same category, while those with different a priori codes were grouped in another category. For example, item S.04.b asks participants to provide two sets of instructions for Aisha to carry eight toys from the kitchen to her room, while carrying at most three toys each trip. Participant responses that specify the number of toys in each of the multiple trips would be in one category, “using complete, precise, and ordered instructions.” On the other hand, responses without a specified number of toys for each of the multiple trips would be in a different category, “no articulation of complete, precise, or ordered instruction.”

Analyzing across units referred to constantly comparing different categories across different items within the same CT concept. For example, item S.01.a asks the participants to write two different programs using the given Scratch code blocks to move the cat sprite from 0 to 5 on a number line. Student responses may also fall in the categories, “using complete, precise, and ordered instructions” and “no articulation of complete, precise, or ordered instruction.” Response in categories for S.01.a would be compared with those for S.04.a. When comparing responses in similar categories of students’ articulated CT, there were a few important questions considered: Did participants’ articulated CT in the similar categories share similar characteristics in terms of students’ articulated CT (e.g., did responses all involve complete, precise, and ordered instructions)? What were the differences in participants’ articulation of a CT concept between different categories (e.g., the “no articulation” category did not have characteristics of complete, precise, and ordered instructions)?
Next, themes were constructed by making reflections and interpretations (Merriam & Tisdell, 2015) of the different categories of data. When drafting themes to describe the general patterns of students articulated CT, questions involved: Is the theme constructed exhaustive (i.e., speaks for all characteristics in one category)? Are different themes mutually exclusive (i.e., characteristics in one category should not appear in the other)? As reflections and interpretations were made, the themes were adjusted to be as comprehensive as possible. For example, the categories from the aforementioned items S.04.b and S.01.a would be holistically examined to devise the potential theme: Grade 3 students generally demonstrated understanding and ability in using complete, precise, and ordered instructions. Figure 3-5 visualizes the single unit and across units CCA processes.

**Mapping the correspondence**

This section explains the processes involved to address the second research question, which was: How does students’ articulated CT correspond to the learning goals and the progressions in the CT LTs. Since the *a priori* codes were verbatim translations of the learning goals in the LTs, the *a priori* codes assigned to students’ articulation provided a direct correspondence between students’ articulated CT and the learning goals in the LT. Figure 3-6 explains the relationship between the learning goals as *a priori* codes and students’ articulated CT.

**Rigor**

Multiple measures were employed in this dissertation study to ensure the credibility, dependability, transferability, and confirmability (Guba & Lincoln, 1981) of the results. First of all, to ensure credibility, multiple researchers and investigators were trained to independently perform data collection (investigator triangulation, Merriam & Tisdell, 2016). To ensure consistencies of findings, two researchers, who have had experience working on scoring CT
assessments associated with this project, coded 40% of participant interviews together to operationalize and check consistency of the coding of data using *a priori* codes (analyst triangulation, Patton, 1999). Both researchers were familiar with the assessment items, the interpretation criteria, and the LTs, and performed the data analysis for the pilot study. While coding the 40% of the participants’ interviews, the two researchers examined each data excerpt while discussing which code(s) should be assigned. Given that the two researchers coded this portion of the data collaboratively, no measure of percent agreement was calculated.

Transferability (Guba & Lincoln, 1981) was ensured by providing a rich, thick description (Merriam & Tisdell, 2016) of the research design, method, and findings. To ensure confirmability a codebook together with the Dedoose software were used to keep track of the assigning of codes with examples from the data and the codebook was refined during the process of the two researchers coding together. For example, item DC.06.c seeks evidence of students’ ability to break a polygon into two rectangles or multiple small squares, find the area for those smaller shapes, before adding them up for the final area. The two researchers concurred that, as long as a participant expresses the “breaking up” of the polygon, the code pertaining to system decomposition should be assigned. In addition, the interpretation criteria have been established and revised during a two-year validation process (Gane et al., under review). Specifically, multiple researchers (n=4) worked on refining the details of the criteria with examples and coding rationale and establishing consensus using the criteria to categorize student response to the assessments. Last but not the least, the author checked with a computer science university professor who had content expertise in K-12 CS education to cross check the interpretations made.
<table>
<thead>
<tr>
<th>Lesson</th>
<th>DECOMPOSITION</th>
<th>REPETITION</th>
<th>SEQUENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brownie Bites</td>
<td>Onramp to “1. Systems are made up of smaller parts.”</td>
<td>NA</td>
<td>Onramp to “1. Precise instructions are more likely to produce the intended outcome than general ones.”</td>
</tr>
<tr>
<td>(unplugged)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Animal Number Story</td>
<td>Onramp to “1. Systems are made up of smaller parts.”</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Sharing Equally</td>
<td>From 1 to “5. Programs can be decomposed into components.”</td>
<td>From “2. Some tasks involve repeating actions” to “1. Repeating things can have a cumulative effect.”</td>
<td>From “3.1. Precision and completeness are important when writing instructions in advance” to “4.1 Computers require precise instructions using intended commands.”</td>
</tr>
<tr>
<td>Fraction Circles 1</td>
<td>From 1 to “2. Complex problems can be broken into smaller parts.”</td>
<td>From “2. Some tasks involve repeating actions” to “1. Repeating things can have a cumulative effect.”</td>
<td>From “3.1. Precision and completeness are important when writing instructions in advance” to “4.1 Computers require precise instructions using intended commands.”</td>
</tr>
<tr>
<td>Polygon Partners</td>
<td>NA</td>
<td>NA</td>
<td>From “1. Precise instructions are more likely to produce the intended outcome than general ones” to “3.1 Precision and completeness are important when writing instructions in advance.”</td>
</tr>
<tr>
<td>(unplugged)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fraction Circles 2</td>
<td>NA</td>
<td>NA</td>
<td>Onramp to “2. Different sets of instructions can produce the same outcome.”</td>
</tr>
<tr>
<td>Storyboarding: Fraction Comic Strip (unplugged)</td>
<td>From “2. Complex problems can be broken into smaller parts” to “3. Problem decomposition is a useful early step in problem-solving.”</td>
<td>NA</td>
<td>From 3.1 to “3.2 Programs are made by assembling instructions from a limited set.”</td>
</tr>
<tr>
<td>Fraction Comic Animation</td>
<td>From “2. Complex problems can be broken into smaller parts” to “3. Problem decomposition is a useful early step in problem-solving.”</td>
<td>NA</td>
<td>From 3.1 to “3.2 Programs are made by assembling instructions from a limited set.”</td>
</tr>
</tbody>
</table>
Table 3-2. Grade 4 lesson coverage of CT concepts.

<table>
<thead>
<tr>
<th>Lesson</th>
<th>CONDITIONALS</th>
<th>DECOMPOSITION</th>
<th>REPETITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introducing Conditionals: A Rounding Shortcut (unplugged)</td>
<td>From “1. A condition is something that can be true or false” to “2. A conditional connects a condition to an outcome.”</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Scratch Creative 2: Zoo Animals Number Story</td>
<td>NA</td>
<td>From “1. Systems are made up of smaller parts” to “5. Programs can be decomposed into components.”</td>
<td>NA</td>
</tr>
<tr>
<td>Introducing Variables: Robot Boxes (unplugged)</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Variables: Math Chat</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Fraction Circles: Pizza Sharing</td>
<td>NA</td>
<td>From “2. Complex problems can be broken into smaller parts” to “3. Complex problems can be broken into smaller parts.”</td>
<td>From “2. Some tasks involve repeating actions” to “3. Instructions like “Step 3 times” do the same thing as “Step, step, step.”</td>
</tr>
<tr>
<td>Ambling Animals</td>
<td>From 2 to “5. Conditional statements are computer commands to evaluate conditions and complete connected actions.”</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Slicing Sandwiches (Comparing Same Denoms)</td>
<td>From 1 to “1.2 Sometimes multiple conditions must be considered.”</td>
<td>NA</td>
<td>From 3 to “5. Computers use repeat commands.”</td>
</tr>
<tr>
<td>Comparing Fractions: Same Numerators</td>
<td>From 1 to “1.2 Sometimes multiple conditions must be considered.”</td>
<td>From “2. Complex problems can be broken into smaller parts” to “3. Complex problems can be broken into smaller parts.”</td>
<td>NA</td>
</tr>
</tbody>
</table>

Table 3-3. Distribution of items by CT concept.

<table>
<thead>
<tr>
<th>Item ID</th>
<th>Sequence (G3)</th>
<th>Repetition (G3&amp;4)</th>
<th>Conditional (G4)</th>
<th>Decomposition (G3&amp;4)</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>S.01.a (Early)</td>
<td>R.01.a (Early)</td>
<td>C.02.e (Early, Mid)</td>
<td>DC.02.a (Early)</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>S.04.b (Early)</td>
<td>R.01.b (Mid)</td>
<td>C.03.b (Early, Mid)</td>
<td>DC.02.b (Mid)</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>S.02.a (Mid)</td>
<td>R.03.c (Mid)</td>
<td>C.06.c (Early, Mid)</td>
<td>DC.06.a (Mid)</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>R.05.a (Early)</td>
<td>DC.08.a (Early)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>13</td>
</tr>
</tbody>
</table>
### Table 3-4. *A priori* codes for Sequence.

<table>
<thead>
<tr>
<th></th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Beginning</strong></td>
<td>1. Precise instructions are more likely to produce the intended outcome than general ones.</td>
</tr>
<tr>
<td><strong>Intermediate</strong></td>
<td>2. Different sets of instructions can produce the same outcome.</td>
</tr>
<tr>
<td><strong>Advanced</strong></td>
<td>6. Some commands modify the default order of execution, altering when and which instructions are executed.</td>
</tr>
</tbody>
</table>

### Table 3-5. *A priori* codes for Repetition.

<table>
<thead>
<tr>
<th></th>
<th>Repetition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Beginning</strong></td>
<td>2. Some tasks involve repeating actions.</td>
</tr>
<tr>
<td><strong>Intermediate</strong></td>
<td>4. Repetition can go on forever or stop.</td>
</tr>
<tr>
<td><strong>Advanced</strong></td>
<td>8. Programs use conditions to end loops.</td>
</tr>
</tbody>
</table>
**Table 3-6. A priori codes for Conditionals.**

<table>
<thead>
<tr>
<th>Conditionals</th>
<th>Beginning</th>
<th>Intermediate</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. A condition is something that can be true or false.</td>
<td>0. Actions often result from specific causes.</td>
<td>2. A conditional connects a condition to an outcome.</td>
<td>3. Each of the two states of a condition may have its own action.</td>
</tr>
<tr>
<td></td>
<td>1.2 Sometimes multiple conditions must be considered.</td>
<td>4.1 Conditions can overlap, and more than one can apply.</td>
<td>4. Computers require all actions to be specified.</td>
</tr>
<tr>
<td></td>
<td>1.1 A boolean is a variable that can be true or false.</td>
<td>6. Logical operators can be used to combine conditions.</td>
<td>7. Conditional statements can create branches in the flow of execution.</td>
</tr>
<tr>
<td></td>
<td>6. Code is reusable.</td>
<td>8. Conditional statements can be combined in several ways.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11. Defining General procedures makes code more useful in the future.</td>
<td>15. Decomposition and modularization are Useful in problem-solving.</td>
<td></td>
</tr>
</tbody>
</table>

**Table 3-7. A priori codes for Decomposition.**

<table>
<thead>
<tr>
<th>Decomposition</th>
<th>1. Systems are made up of smaller parts.</th>
<th>2. Complex problems can be broken into smaller parts.</th>
<th>4. System components are often not unique within or across systems.</th>
<th>5. Programs can be decomposed into components.</th>
<th>3. Problem decomposition is a useful early step in problem-solving.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6. Code is reusable.</td>
<td>9. Code can be written in small parts (called procedures).</td>
<td>7. Often existing code from other problems can be used to solve parts of a decomposed problem.</td>
<td>10. Defining general procedures allows for easy reuse of code within programs.</td>
<td>13. Sometimes it is useful to modify and repurpose existing code.</td>
</tr>
</tbody>
</table>
Figure 3-1. The design framework.
Figure 3-2. Example math-CT integrated lesson (LTEC-2 curriculum materials).
Create 2 different scripts (sets of instructions) to move the cat so that he stops at 5 on the number line. **Use only the blocks shown above.** Write or draw your scripts in the boxes.

Figure 3-3. Item S.01.a. (LTEC-2 curriculum materials).

![Diagram](image)

Figure 3-4. Analysis using *a priori* codes.
Figure 3-5. Analysis using constant comparison for each CT concept

Figure 3-6. Direct correspondence between articulated CT and CT LTs.
CHAPTER 4
FINDINGS

Overall Data

The purpose of this study was to answer the two overarching research questions: a) how do 3rd- and 4th-grade students articulate CT as they progress in integrated math-CT instruction throughout the school year of 2019-2020 and b) how does students’ CT understanding map to the milestones (learning goals) and progression hypothesized in the CT learning trajectories. This study sought to answer the above two questions by collecting evidence of students’ articulation of CT using cognitive interviews with a think-aloud interview protocol.

Twelve third-grade (G3) students participated in phase 1 and phase 2 data cognitive interviews, with each participant completing four assessment items on sequence, repetition, and decomposition in each interview. Ten fourth-grade (G4) students participated in the data cognitive interviews, with each participant completing three assessment items on repetition, conditionals, and decomposition in each interview. Each participant interview was transcribed verbatim and analyzed in the Dedoose data analysis software. Data analysis using *a priori* codes and constant comparison allowed patterns and common themes to emerge for each of the CT concepts examined.

Data were examined relative to each of the research questions and in the context of the specific assessment item design patterns (i.e., what KSAs an item is mapped to and what evidence of CT an item seeks). Data analysis revealed interesting patterns of students’ articulated CT. In some cases, such as sequence, participants generally articulated complete, precise, and ordered instructions. Therefore, one overarching theme was devised. In other cases, such as repetition, participants generally recognized repetition in the coding item (i.e., R.03.c) but showed varied ability in creating repetition in word problems (e.g., R.01.a and R.01.b).
Therefore, two themes were devised to capture the distinct characteristics of student responses to items in different contexts. This next section described all the themes relevant to the first research question, each followed by a detailed correspondence of the students’ articulated CT to the learning goals and the learning progressions in the LTs (RQ2).

**Themes of Participants’ CT Articulation**

This dissertation study focused on the common concepts of repetition and decomposition for both G3 and G4 students, as well as sequence for G3 students and conditionals for G4 students. This section describes the themes that emerged from analyzing both G3 and G4 participants’ verbal articulation of CT in problem-solving across phase 1 and phase 2 cognitive interviews, namely: (1) G3 participants generally demonstrated understanding and using of complete, precise, and ordered instructions, (2) G3 and G4 participants generally showed evidence of understanding and recognizing repetition in the coding item, (3) G3 and G4 participants showed varied ability in recognizing and applying repetition in word problems, (4) G4 participants showed inconsistent understanding of evaluating conditions involving number comparisons, (5) G4 participants generally showed no understanding of evaluating conditions in the coding item, (6) G3 and G4 participants could identify components of a number, and (7) G3 and G4 participants used decomposition in arithmetic but demonstrated varied ability in decomposing the word problem. The following sections also presented the KSAs that guided each of the assessment items used in this study to provide more context.

**Sequence: Grade 3 (G3) Participants Generally Demonstrated Understanding and Ability in Using Complete, Precise, and Ordered Instructions**

Data analysis revealed abundant evidence in the majority of the G3 students’ understanding and using complete, precise, and ordered instructions in solving the three sequence assessment items. For example, the sequence item S.01.a (Figure 3-3) asked
participants to write two different programs using the given Scratch code blocks to move the cat sprite from 0 to 5 on a number line. This item was mapped to all five of the KSA statements in the design pattern (Appendix D) of the sequence assessment items. The five KSAs are:

1. Ability to create several different sets of instructions to produce the same intended goal;
2. Ability to use / modify / create precise instructions to produce the intended goal;
3. Ability to use / modify / create an ordered set of instructions to produce the intended goal;
4. Ability to use a limited set of commands to produce the intended goal;
5. Knowledge that computer programs [computational artifacts] require selecting from and ordering a limited set of commands.

This means that this item was designed to seek evidence of students’ abilities and knowledge to select the code blocks with the appropriate direction (move back and forward) and also the appropriate number of steps (one or two steps) for each block from a limited number of block choices. Following the four-step constant comparison analysis process, all five G3 participants who answered this question provided solutions showing evidence of using and creating complete, precise, and ordered instructions by selecting from the given code blocks to produce the intended outcome, e.g., “Move one step forward five times” (P3-4); “Move forward 2 and then move forward another 2. And then move forward 1. And then now I'm at 5” (P3-3).

In addition, all five participants provided a second set of complete, precise, and ordered instructions that would move the cat sprite from 0 to 5 on the number line, e.g., “Move two steps forward three times and then one step backward” (P3-19).

The second sequence item (S.04.b), an unplugged problem, asked the participants to provide two sets of instructions for Aisha to carry eight toys from the kitchen to her room, while carrying at most three toys each trip. Students could take as few of three trips or as many as eight trips with a specified number of toy(s) in each trip. This item was designed to seek evidence of students’ abilities and knowledge in deciding how many toys Aisha needed to carry, depending
on the number of trips she would take. This item was mapped to four of the KSA statements in the design pattern (Appendix D) of the sequence assessment items (the last KSA, “Knowledge that computer programs [computational artifacts] require selecting from and ordering a limited set of commands” was not applicable in this item as it does not require computer coding). Seven G3 participants answered this question, among which four provided solutions showing evidence of using and creating complete, precise, and ordered instructions while taking into consideration the constraints of the problem; that is, Aisha could carry only three toys at most in each trip. For example, one participant said, “Take 3 in her hands and come back and [get] 3 more and come back again and get 2 more” (P3-16) and another participant said, “You can like do 2 toys first, then 3, and 3. Or 3,3,2” (P3-10). The other three participants answering this item either did not understand the question before further scaffolding from the teacher (P3-5) or did not articulate the sequence of steps involved (e.g., saying “ask a family member to help” (P3-12)).

The third sequence item (S.02.a) asked the students to write or draw the Scratch code to make the cat go back from 5 to 1 on the number line, pick up the hat at 1, and move to 6. Here, students needed to take into consideration the order of actions to happen, the direction to go (forward or backward), and specify the number of steps to take each time. This item was mapped to all but one above mentioned KSAs (the first KSA “Ability to create several different sets of instructions to produce the same intended goal” was not applicable as the item did not ask students to provide an alternative set of instructions). All 12 Grade 3 participants answered this item and all but one provided a solution showing evidence of using and creating complete, precise, and ordered instructions by selecting from the given code blocks to produce the intended outcome. For example,

He's going to move two steps backward, that's three. So he's going to move two steps backward again, which is one, pick up the hat, move two steps forward, he's
going to be at 3, move two steps forward again, he's going to be at five, and one step forward, he's going to be at 6 (P3-14).

Repetition

Data analysis revealed abundant evidence that the G3 and G4 participants had ease with understanding and recognizing repetition in a coding context, but they had difficulty with applying this CT concept in word problem scenarios. Therefore, the two themes were: (1) the G3 and G4 participants generally showed evidence of understanding and recognizing the cumulative effect of repeating actions in the coding item and (2) the G3 and G4 participants showed varied ability in recognizing and using repeat instructions in word problems.

G3 and G4 participants generally showed evidence of understanding and recognizing repetition in the coding item

Participants showed evidence of clearly articulating their understanding of the cumulative effect of the repeat block in a coding context. Item R.03.c asked students to describe what will happen when the following Scratch code is run (Figure 4-1). This item was mapped to one of the five KSAs in the design pattern for repetition items: Knowledge that repeat commands (e.g., repeat, repeat until, forever) tell a computer to repeat specific actions/instructions. This means that the item was designed to seek evidence of students’ recognizing and understanding that, when repeat commands are used in a computer program, specific actions will be repeated for a predetermined number of times (i.e., the cumulative effect). Ten of the 11 participants who answered this item provided a solution showing evidence of understanding of the cumulative effect of the “repeat” block. For example, students responded: “When you clicked green flag, it will repeat meow three times” (P3-3); “It will play meow three times until done” (P4-15).
G3 and G4 participants showed varied ability in recognizing and using repeat instructions in word problems

Item R.05.a was an unplugged item that asked participants to decide how many apples Erika will have after following the repeated instruction: Repeat “taking one apple from DeShaun’s basket” four times. While this item was not mapped to any of the existing KSAs, one KSA was seen as relevant: Knowledge that repeat commands (e.g., repeat, repeat until, forever) tell a computer to repeat specific actions/instructions.” This means that the item was designed to seek evidence of students’ recognizing and understanding what actions were to be repeated and the result of the repeated actions. Four of the 10 participants (seven in G3 and three in G4) provided an answer showing evidence of understanding of the cumulative effect of the repeating action, e.g., “I counted with my fingers, by doing 3, cuz there are already 2 [in Erika’s basket], then plus 3, I mean, plus 1, plus 1 and that’s 4, plus another 1 and that’s 5, plus another 1 and that’s 6” (P3-1); “Cuz he [DeShaun originally] has 8 apples and she [Erika originally] has 2, then you repeat it [the instruction] 4 times, she gets 6 apples” (P4-15).

However, the majority (six) of the 10 participants who answered this item did not show evidence of understanding the cumulative effect of the repeated action. Data showed that participants either did not understand what was to be repeated, or only partially repeated the action, e.g., “It says, take 1 apple away and put it on the table and then put it on Erika’s basket. It tells me now Erika has 3 and he has 7” (P3-16); “I took four apples out of Deshawn's basket and I have to put it on her basket and now she has 3 apples” (P3-19).

Items R.01.a (in phase 1 cognitive interview) and R.01.b (Figure 4-2; in phase 2 interview) are the two versions of the same unplugged problem, which asked students to write instructions for Andre to give away 9 cookies to his 3 friends using the “repeat 3 times” instruction at least once. Both items were mapped to the KSA “Ability to use repeat commands
(forever, repeat until, repeat X times) to Use / Modify / Create instructions that create cumulative effects.”

This means that the two items were designed to seek evidence of students’ ability to identify the pattern of actions (i.e., what could be repeated) and create the cumulative effect by using “repeat” commands instead of using the same commands multiple times. The only difference between the two items was that R.01.a did not present a sample instruction, while R.01.b did. For R.01.a, 11 of the 12 students (five in G3 and seven in G4) who answered this item did not show any evidence of using repetition in their instruction, but rather, simply explained the mathematical results, e.g., “So I drew three circles and put three cookies in each so then it goes 3, 6, and 9” (P3-5); “Give each friend 3 cookies” (P4-1); “Three times 3 is 9 and 9 divided by 3 is 3” (P4-6). Similarly, for R.01.b, 7 of 11 (four in G3 and seven in G4) did not show any evidence of using repetition in their instruction, e.g., “He gave away 3 to Sally, and then 3 to Val and then 3 to Lee” (P3-12); “I see Sally's name on Andre’s instructions 3 times, and I also see Val’s 3 times and I see Lee’s 3 times here. So that means they are each gonna get 3 cookies” (P3-14); “You add 3 times 3 is 9. And 9 divided by 3 is 3” (P4-6).

The other four participants (all in G4) incorporated the “repeat” command in their solution, for example:

I just put a [repeat] block there to represent it'll be repeating something, cuz you don't want to put something in the code that says ‘give Sally one, give Val one, give Lee one just over and over again. This is just a quicker way. So, Val gets one cookie, and then Lee gets one, and then Sally gets one cookie, and then they are gonna repeat that three times (P4-14);

First, I wrote of a repeat [3 times] block. Then, then I wrote, give Sally one cookie, then give Val one cookie, and then give Lee one cookie (P4-16).
Conditionals

G4 participants showed inconsistent understanding of evaluating conditions involving number comparisons

Data analysis showed abundant evidence that the G4 participants did not articulate how to evaluate the true or false status of conditions while solving the conditional assessment items. Specifically, for item C.02.e (Figure 4-3) that asked students what sound(s) will play if the code “If 5 < 8, then play a ‘pop’ sound; If 5 > 7, then play a ‘bing’ sound” is run. This item was mapped to all three of the KSAs in the design pattern for conditional items:

(1) Knowledge that a condition is something that can be true or false.

(2) Knowledge that a conditional connects a condition to an outcome.

(3) Ability to use conditional statements (such as if-then, if-then-else, and event handlers) to evaluate a condition to determine an outcome.

This means that the item was designed to seek evidence of students’ understanding that they needed to first evaluate whether 5 is smaller than 8 or 5 is larger than 7, then decide whether “pop” or “bing” would be played. While many of the participants showed evidence of evaluating whether the number comparison expressions were correct or not (e.g., “The first one [pop sound will play] because 8 is greater than 5. And 5 is actually less than 7” (P4-1)) in phase 1 cognitive interview, most (three of four) participants who answered this item in phase 2 interview did not articulate an understanding of conditionals. Two of the three participants who responded to this item in both interviews and did articulate an understanding in phase 1 interview failed to do so in phase 2 interview.

G4 participants generally showed no understanding of evaluating conditions in the coding item

The other conditional item, C.03.b (Figure 4-4), asks students to take the user input of the number 2, evaluate the condition (2<5), and decide what the final outcome is. This item is
mapped to the same KSAs as the previous item, meaning that the item is also designed to seek the same evidence as the previous item. Only one of nine responses demonstrated evidence of evaluating the “true or false” status of conditions in this item, for example, “Yes... Because 2 is under 5 so [the pop sound will play]” (P4-5). Data analysis showed that the majority of participants (eight of nine) did not understand how to take the user input, plug the input value in the conditional statement, and evaluate whether the condition is true or false. For example, “I think yes, because if you input 2, the start sound will pop twice” (P4-12); “It’s not gonna make the sound. Because it says only for 5. It’s only gonna work for 5. And that’s 2, so that’s not gonna work” (P4-14); “I don’t know. This is tricky. Because I have no way to explain this, but I know it’ll be 1 because 1 is less than 2” (P4-6).

Decomposition

G3 and G4 participants could identify components of a number

Data analysis showed abundant evidence of participants’ identifying the components of a system in the context of mathematics (e.g., a number, an arithmetic problem, etc.) and breaking down a complex problem into smaller parts while working on decomposition assessment items. For example, item DC.02.a asked students to “write one or more addition number sentences that mean the same as the multiplication number sentence: 5 x 4 = 20.” This item was mapped to two of the KSAs in the design pattern of the decomposition assessment:

(1) Knowledge that systems are made up of smaller, distinct parts;

(2) Ability to break a complex problem into a set of simpler problems.

This means that the item was designed to seek evidence of students’ understanding and ability to identify what smaller numbers constitute the number 20. The majority of participants (four of six) who answered this item provided a solution showing evidence of identifying the
subcomponents of the number, 20. For example, “10 plus 10 equals 20, 19 plus 1 equals 20, 9 plus 11 equals 20” (P3-8); “18 plus 2 equals 20” (P3-5).

Similarly, item DC.08.a asked students to list at least three ways to add numbers to get to 10. This item was also mapped to the KSA, “Knowledge that systems are made up of smaller, distinct parts.” All five participants who answered this item provided answers showing evidence of identifying the subcomponents of the number 10. For example, “To break the number apart . . . like 5+5, what equals 10, 9 plus 1 equals 10, 4 plus 6 equals 10” (P3-2).

**G3 and G4 participants used decomposition in arithmetic but demonstrated varied ability in decomposing the word problem**

Item DC.02.b asked students to decompose the problem: (5 x 2) + (3 x 2). In addition to students’ identifying the subcomponents of a number, this item also involved students’ breaking the problem into a set of simpler problems (i.e., multiplication and addition). Therefore, the item was mapped to three of the KSAs:

1. Knowledge that systems are made up of smaller, distinct parts;
2. Ability to break a complex problem into a set of simpler problems;
3. Ability to use problem decomposition as an early step in designing a solution.

This means that the item was designed to seek evidence of students’ understanding and ability to do multiplications first before adding the two results together. Almost all participants (15 of 17) who answered this item showed evidence of breaking down the arithmetic problem into a step-by-step process, e.g., “5 times 2 is 10, 3 times 2 is 6. So 6 plus 10 equals 16” (P4-6); “Basically 5 times 2 equals 10, and 3 times 2 makes 6, and if you join them and it has a plus, if you join them, it’ll be 16” (P3-5).

Item DC.06.c asked students to list the multiple steps involved in finding the area of a polygon. This item was mapped to the same three KSAs mentioned above. This means that this
item was designed to seek evidence of students’ ability to break up the shape into two rectangles or multiple small squares, find the area for those smaller shapes, before adding them up for the final area. The majority of the responses (30 of 38) did not show evidence of using decomposition to break down the problem. Many provided a solution that was not logical or irrelevant to finding the area of the polygon, for example,

I’m thinking, do I go outside the box or if like coding, you can start here [a corner on the outline] and tell the person move down down down, go all the way around [the outline of the shape], or you can try to go inside [the shape] and get all the dots outside of the dots and inside (P3-15);

[First I] count the dots . . . then I would like, add all the sides up (P4-16);

So I think the whole in the perimeter equals, so I think the perimeter is 18 for the whole thing and then the multiplication sentence is 6X3 (P3-12).

Eight of the 38 responses to this item mentioned breaking up the polygon into smaller squares or rectangles, then finding the area of the smaller squares before adding them up for the total area. For example:

So I draw the little lines here to, like make little boxes, to see. So step 1, I would make little squares. [writing] And after I make the squares, I would count them. [writing] but I can also break it in half. . . . So if I break it [the polygon] into 2 rectangles, I could count 1, 2, 3, 4, 5, 6 [squares in the top rectangle]. Then another one, 1, 2, 3, 4, 5, 6, 7, 8 [squares in the bottom rectangles]. 6 plus 8 equals what? Oh 14. So I can make 2 rectangles. And then I could count it. After I count it [them] both, I can add [them]. Like I said, 6 on the top [writing] 8 on the bottom (P3-19).

**Corresponding Participants’ Articulated CT to LTs**

The section above described the themes of participants’ CT articulation that emerged from the data. This section described the correspondence between the patterns of the articulated CT and the theorized LTs after the mid cognitive interviews. Recall that the *a priori* codes were verbatim translations of the learning goals in the LTs. Therefore, the *a priori* codes assigned to student responses provided a direct correspondence between students’ articulated CT and the
learning goals in the LT. For each of the LTs, the range of the learning goals in the LT corresponding to students’ articulated CT was reported.

**Sequence**

The Sequence LT presents three levels of learning goals: beginning, intermediate, and advanced. Recall that the learning goals in each LT are the landmarks students are expected to meet in instruction (Clements & Sarama, 2009). The beginning level involves four learning goals about precision and completeness of instructions. Specifically, these beginning-level learning goals are:

(a) precise instructions are more likely to produce the intended outcome than general ones (S1),

(b) precision and completeness are important when writing instructions in advance (S3.1),

(c) programs are made by assembling instructions from a limited set (S3.2), and

(d) computers require precise instructions using limited commands (S4.1).

The core idea in the intermediate-level goals is producing an outcome using different sets of instructions (S2) and the order of execution (S3 and S4). The advanced-level goals are about manipulating the order of commands for an intended outcome (S6 and S7).

As reported in the previous section, data analysis revealed that G3 students showed evidence of understanding and using complete, precise, and ordered instructions. Therefore, the participants’ articulated CT was generally mapped onto most of the beginning (B) and intermediate-level (I) learning goals in the Sequence LT, with only a few participants not mapped to any. For example, Table 4-1 lists examples of participants articulating a solution to item S.02.a using Scratch code blocks (left column) and the assigned *a priori* codes, i.e., the corresponding learning goals in the Sequence LT (right column).
Repetition

The Repetition LT also presents three levels of learning goals. The beginning level involves (a) some tasks involve repeating actions (R2), (b) instructions like “Step 3 times” do the same things as “Step, step, step,” (R3) and (c) computers use repeat commands (R5). The intermediate goals are (a) repeating things can have a cumulative effect (R1), (b) repetitions can go on forever or stop (R4), (c) different kinds of tasks require different kinds of repetition (R4.1), and (d) different kinds of repetition have different commands (R5.1). The advanced learning goals are about using variables and conditions to manipulate repetition and loops (R8, R6, and R9).

There were two themes that were reported for this LT in the previous section: (1) G3 and G4 students showed evidence of understanding and recognizing repetition in coding and (2) G3 and G4 students showed varied ability in recognizing and using repeat instructions in word problems. The participants’ articulated CT was mapped to most of the beginning-level learning goals and one in the intermediate-level, with some participants mapping to more learning goals, and some not mapping to any. For example, Table 4-2 lists participant P3-3’s description of the outcome of the “Repeat 3” block (left column) and the corresponding learning goals in the Repetition LT (right column).

In another item, R.01.a, some articulated the use of repeat commands, also corresponding to two of the beginning level learning goals and the first intermediate level one. For example, Table 4-3 lists some example responses from participants describing the cookie dissemination using repetition (left column) and the corresponding learning goals in the LT (right column).

Conditionals

Just like the two LTs mentioned above, the conditionals LT also has learning goals categorized into the beginning, intermediate, and advanced levels. In this section, only the
beginning level was mentioned because the intermediate and advanced learning goals were out of the scope of this study. Specifically, the following three beginning level learning goals about evaluating the true and false status of a condition were relevant: “A condition is something that can be true or false” (C1), “Actions often result from specific causes” (C0), and “A conditional connects a condition to an outcome” (C2).

The participants’ articulated CT in conditionals showed that they had inconsistent understanding of how to evaluate the two states of a condition involving number comparisons and no understanding of evaluating conditions in coding. Therefore, the participants’ articulated CT was mapped to a range (the first three) of the beginning-level learning goals in the Conditionals LT, with some participants mapped to more learning goals, and some not mapped to any. For example, Table 4-3 lists examples of participants evaluating a mathematical expression (left column) and the corresponding learning goals in the Conditional LT (right column).

**Decomposition**

Unlike the previous three LTs, the Decomposition LT does not specify any beginning, intermediate, and advanced levels. The relevant learning goals in the Decomposition LT were: “Systems are made up of smaller parts” (D1), “Complex problems can be broken into smaller parts” (D2), and “Problem decomposition is a useful early step in problem-solving” (D3). Recall that there were two themes for students’ articulated CT in decomposition: G3 and G4 students could identify components of a number; they used decomposition in arithmetic but had varied ability in decomposing word problems. Therefore, the participants’ articulated CT was mapped to a range (three) of the learning goals in the Decomposition LT, with many participants mapped to at least the first one of the learning goals. For example, Table 4-5 lists participant P3-8’s breaking up the number 20 (left column) and the corresponding learning goal in the
Decomposition LT (right column). Table 4-6 lists examples of participants articulating how to find the area of a backward L-shaped polygon (left column) and the corresponding learning goals in the decomposition LT (right column). Table 4-7 lists participants P4-6 and P4-5’s articulation of how to break up the arithmetic problem, (5X2) + (3X2), in steps (left column) and the corresponding learning goals in the decomposition LT (right column).

**Summary of Findings**

Chapter 4 reported the findings to the two research questions of this dissertation study. First, in answering how 3rd- and 4th-grade students express and articulate CT in math-CT problem-solving scenarios (RQ1), data revealed that, in sequence, G3 students generally demonstrated understanding and using of complete, precise, and ordered instructions. In repetition, G3 and G4 students generally showed evidence of understanding and recognizing repetition in the coding item, but showed varied ability in recognizing and applying repetition in word problems. In conditionals, G4 Students articulated inconsistent understanding of evaluating conditions involving number comparisons and generally showed no understanding of evaluating conditions in the coding item. In decomposition, G3 and G4 students could identify components of a number and use decomposition in arithmetic but demonstrated varied ability in using decomposition in the word problem. Then, in answering how students’ articulated CT correspond to the CT learning trajectories (RQ2), data revealed that the participants were generally mapped to (1) most of the beginning and intermediate level learning goals in the Sequence LT; (2) different levels in the Repetition and Conditional LTs with some participants mapped to more learning goals than others; and (3) at least the first learning goal in the Decomposition LT, with some participants also mapped to two subsequent learning goals. Chapter 5 discusses these findings in detail.
Table 4-1. Participant articulation - LT correspondence (Sequence).

<table>
<thead>
<tr>
<th>Examples of participants’ response to S.02.a</th>
<th>Corresponding learning goals in the Sequence LT</th>
</tr>
</thead>
<tbody>
<tr>
<td>“So I see he’s [the cat is] at 5, so he has to move back 1,2,3,4. So there’s a ‘move 2 back [block]’, so I put 2 of those [blocks]. [writing] Move 2 steps backward, another one, move 2 steps backward, so 1, 2, 1, 2. Now move forward, wait I gotta pick up [the] hat. Then when I'm at the hat, I can move forward 1, 2, 3, 4, 5. So I can put two of these [move 2 forward blocks] and then move 1 forward. [writing ‘move 2 steps forward, move 2 steps forward’] Then 1 step forward” (P3-19).</td>
<td>1. Precise instructions are more likely to produce the intended outcome than general ones. (B)</td>
</tr>
<tr>
<td></td>
<td>3.1 Precision and completeness are important when writing instructions in advance. (B)</td>
</tr>
<tr>
<td></td>
<td>3.2 Programs are made by assembling instructions from a limited set. (B)</td>
</tr>
<tr>
<td></td>
<td>4.1 Computers require precise instructions using limited commands. (B)</td>
</tr>
<tr>
<td></td>
<td>3. The order in which instructions are carried out can affect the outcome. (I)</td>
</tr>
<tr>
<td></td>
<td>4. Computers have a default order of execution, so order matters in programming. (I)</td>
</tr>
<tr>
<td></td>
<td>5. Creating working programs requires considering both appropriate commands and their order. (I)</td>
</tr>
<tr>
<td>“I’ll just put [turn] right. Turn right and then forward, 5, and then, [he] grabs the hat, and then he turns over left, and then he goes to 9, um 4 steps” (P3-12).</td>
<td>No articulation of sequence.</td>
</tr>
</tbody>
</table>

Table 4-2. Participant articulation - LT correspondence (Item R.03.c).

<table>
<thead>
<tr>
<th>Examples of participants’ response to R.03.c</th>
<th>Corresponding learning goals in LT</th>
</tr>
</thead>
<tbody>
<tr>
<td>“When you clicked green flag it will repeat meow three times” (P3-3).</td>
<td>2. Some tasks involve repeating actions. (B)</td>
</tr>
<tr>
<td></td>
<td>3. Instructions like “Step 3 times” do the same thing as “Step, step, step.” (B)</td>
</tr>
<tr>
<td></td>
<td>5. Computers use repeat commands. (B)</td>
</tr>
<tr>
<td></td>
<td>1. Repeating things can have a cumulative effect. (I)</td>
</tr>
</tbody>
</table>
### Table 4-3. Participant articulation - LT correspondence (Item R.01.a).

<table>
<thead>
<tr>
<th>Examples of participants’ responses to R.01.a</th>
<th>Corresponding learning goals in LT</th>
</tr>
</thead>
<tbody>
<tr>
<td>“One friend has three. Another one another friend has three and the last one has three. [Writing ‘one friend has 3 cookies’] Maybe repeats three times” (P3-14).</td>
<td>2. Some tasks involve repeating actions. (B)</td>
</tr>
<tr>
<td>“First I wrote of a repeat block. Then. Then I wrote, give Sally one cookie then give Val one cookie and then give Lee one cookie” (P4-16).</td>
<td>3. Instructions like “Step 3 times” do the same thing as “Step, step, step.” (B)</td>
</tr>
<tr>
<td>“I’m drawing 3 friends and I have a circle by each friend and then I’m gonna give cookies to each friend. And then gonna see if they have an equal amount of cookies” (P3-15).</td>
<td>1. Repeating things can have a cumulative effect. (I)</td>
</tr>
<tr>
<td>“So I drew three circles and put three cookies in each so then it goes 3 6 and 9” (P3-5). “You add 3 times three is 9. And 9 divided by 3 is 3” (P4-6)</td>
<td>(No articulation of repetition)</td>
</tr>
</tbody>
</table>

### Table 4-4. Participant articulation - LT correspondence (Conditionals).

<table>
<thead>
<tr>
<th>Examples of participants’ response to C.02.e</th>
<th>Corresponding learning goals in LT</th>
</tr>
</thead>
<tbody>
<tr>
<td>“5 is less than 8 so [it will play pop]” (P4-16)</td>
<td>0. Actions often result from specific causes. (B)</td>
</tr>
<tr>
<td>“Pop and bing [will play]” (P4-1, P4-7)</td>
<td>1. A condition is something that can be true or false. (B)</td>
</tr>
<tr>
<td></td>
<td>2. A conditional connects a condition to an outcome. (B)</td>
</tr>
<tr>
<td></td>
<td>No articulation of conditionals.</td>
</tr>
</tbody>
</table>

### Table 4-5. Participant articulation - LT correspondence (Item DC.02.a).

<table>
<thead>
<tr>
<th>Examples of participants’ response to DC.02.a</th>
<th>Corresponding learning goals in LT</th>
</tr>
</thead>
<tbody>
<tr>
<td>“10 plus 10 equals 20, 19 plus 1 equals 20, 9 plus 11 equals 20” (P3-8);</td>
<td>1. Systems are made up of smaller parts.</td>
</tr>
</tbody>
</table>
Table 4-6. Participant articulation - LT correspondence (Item DC.06.c).

<table>
<thead>
<tr>
<th>Examples of participants’ response to DC.06.c</th>
<th>Corresponding learning goals in LT</th>
</tr>
</thead>
<tbody>
<tr>
<td>“So you could split this [polygon]. So there’s 1,2,3,4,6,7 right here [in the top shape]. And then 1,2,3,4,5,6,7,8,9,10,11, 11 here [in the bottom shape]. So add those 2 together, 11 plus 7 equals 18” (P4-1)</td>
<td>1. Systems are made up of smaller parts.</td>
</tr>
<tr>
<td></td>
<td>2. Complex problems can be broken into smaller parts.</td>
</tr>
<tr>
<td></td>
<td>3. Problem decomposition is a useful early step in problem-solving.</td>
</tr>
<tr>
<td>“So I think the whole in the perimeter equals, so I think the perimeter is 18 for the whole thing and then the multiplication sentence is 6X3” (P3-12)</td>
<td>No articulation of decomposition</td>
</tr>
</tbody>
</table>

Table 4-7. Participant articulation - LT correspondence (Item DC.02.b).

<table>
<thead>
<tr>
<th>Examples of participants’ response to DC.02.b</th>
<th>Corresponding learning goals in LT</th>
</tr>
</thead>
<tbody>
<tr>
<td>“5X 2 is 10, 3X2 is 6. So 6+10 equals 16” (P4-6)</td>
<td>1. Systems are made up of smaller parts.</td>
</tr>
<tr>
<td>“Five times two, and then I know that equals seven. And then three times two, three times two equals six. And have to add those together” (P4-5)</td>
<td>2. Complex problems can be broken into smaller parts.</td>
</tr>
<tr>
<td></td>
<td>3. Problem decomposition is a useful early step in problem-solving.</td>
</tr>
</tbody>
</table>
Describe what will happen when the green flag is clicked.

Figure 4-1. Item R.03.c. (LTEC-2 curriculum materials).

Andre has 9 cookies to give away to his friends Sally, Val, and Lee. He wants to give each friend an equal number of cookies. Andre wrote instructions for how to give away the cookies.

Rewrite his instructions. Use the instruction “repeat 3 times” at least once.

**Andre’s instructions:**
- Give Sally 1 cookie
- Give Val 1 cookie
- Give Lee 1 cookie
- Give Sally 1 cookie
- Give Val 1 cookie
- Give Lee 1 cookie
- Give Sally 1 cookie
- Give Val 1 cookie
- Give Lee 1 cookie

**Your instructions:**

Figure 4-2. Item R.01.b. (LTEC-2 curriculum materials).
What sound (or sounds) will play if you run this code?

If 5 < 8, then play a “pop” sound.
If 5 > 7, then play a “bing” sound.

Figure 4-3. Item C.02.e. (LTEC-2 curriculum materials).

If you run the code below, will the "pop" sound play if the user inputs 2?

Figure 4-4. Item C.03.b. (LTEC-2 curriculum materials).
CHAPTER 5
DISCUSSION

Chapter 4 described the themes of students’ CT articulation in relation to each of the four LTs and how the articulated CT correspond to the learning goals and the learning progression of the LTs. Unsurprisingly, the participants demonstrated different levels of understanding and ability with each of the four CT concepts in problem-solving. For each of the concepts, there were participants who did not articulate any understanding, whereas some others articulated more advanced problem-solving rationale and reasoning. Chapter 5 has three parts. Part one presented a detailed discussion on the findings from LT-based instruction and learning: First, students’ current CT levels after engaging in the LT-based instruction and activities; Second, the potential learning gaps of students’ learning of the four CT concepts; Third, suggestions on sample instructional activities that support students’ learning progressions on the LTs. Part two was a discussion on CT assessments embedded in different contexts, such as Scratch and word problems. Part three discussed the considerations when making interpretations of the correspondence between students’ CT and the LTs.

LT-based Targeted Instruction of CT

As described in Chapter 2, an LT consists of a collection of learning goals, learning progressions, and instructional activities that support students’ meeting the learning goals and moving along the progressions (Simon, 1995; Clements & Sarama, 2009). LT-based targeted instruction, therefore, is grounded in the LTs and complemented by explicit instructional expectations, scaffolds, and feedback that guide students through the progressions in the LTs. This study used math-CT integrated lessons and assessments that were developed by following the hypothesized learning goals and progressions in the LTs for grades K-8. Currently, given the limited research in LT-based CT instruction and activities, there is little evidence on the efficacy
of such instructional approach and how it supports students’ CT learning. However, previous research in mathematics (Clements et al., 2019; 2020; Sarama et al., 2011) has shown that LT-based instruction was indeed beneficial to some, although not all, students’ learning gains. As such, this dissertation study highlighted the benefits for LT-based targeted instruction in elementary CT. For teachers, they can purposefully identify existing learning gaps by referring to the LTs; and with targeted instruction, teachers leverage activities and scaffolds specifically designed to close the gaps. For students, with a clear learning goal and step-by-step instructional demonstrations and modeling, they build increasingly sophisticated CT knowledge, skills, and abilities. The following section was a discussion of students’ articulated CT after engaging in the LT-based CT instruction and activities, students’ articulated levels of CT as mapped to the LTs, and suggestions on the complexity of targeted instructional activities that could potentially move students further along the LTs.

**Sequence for G3**

In this study, students demonstrated understanding and skills mapped to almost all beginning and intermediate level learning goals in the Sequence LT after engaging in the integrated curriculum. The majority of participants in this study reached the intermediate level and progressed the farthest on the Sequence LT as compared to where the majority of students were in the other three LTs. This finding was consistent with the findings from the pilot study where the participants performed the best on sequence items compared to other items (Luo et al., 2020b). The fact that G3 students progressed far along the Sequence LT suggested that the students could understand and start to master the use of complete, precise, and ordered instructions in coding and daily life problems as early as in Grade 3. In fact, previous research (Kazakoff et al., 2013; Bers et al., 2014) suggested that children as young as kindergartners could successfully sequence the beginning, middle, and end of a story and select and order
instructions in simple programming tasks that did not involve control flows. This suggested that students may be ready for more advanced sequence learning at this age. Possibly, the students may already have had an advanced understanding of sequence that was not captured by the assessments due to the constraint that there was no item that was designed to assess the advanced levels of sequence understanding (i.e., manipulating the order of commands for an intended outcome). Therefore, this dissertation study suggests that future integrated instruction and assessments consider “levelling up” in introducing more advanced content in sequence earlier and assessing students’ performance using items with some level of complexity, such as, using control flow commands to alter the order of execution, etc. This suggestion is also in line with Rich et al. ’s (2017) intentions to use the advanced level learning goals as the “springboards to other learning trajectories” (p.186). In other words, given that students are progressing close to the end level in the Sequence LT, the advanced learning goals could further expand into combining repetition or conditionals with sequence. For example, explain to students that the order of execution may be changed depending on whether there is a repeat instruction involved. If yes, then the commands inside the repeat block will execute first before moving to the commands outside the repeat block. Another example where the aforementioned rule also applies is in conditional statements. If a condition is not met or is false, then the action within that specific condition is skipped. As such, the order of execution may not always be linear as the complexity of the program increases.

**Repetition for G3 and G4**

Participants in this study understood and recognized the cumulative effect of using repeat blocks in the context of coding. This finding was consistent with the pilot study in that when actions were placed within a repeat block, students knew that the actions would be repeated the specified number of times. However, when an item involved more complexity, such as, using
variables (as in one item in the pilot study), or requiring identification and creating of a repeating pattern (item R.01.a, giving out 9 cookies to 3 friends, in this study), students were less likely to articulate a clear understanding or reasoning for problem-solving. This finding had two implications highly relevant to the LT literature in terms of incrementally developing contiguous levels of understanding and skills instead of skipping levels (Clements et al., 2020). First, CT content involving a certain combination of different CT concepts (e.g., conditional + variables, or conditional + loops) should not precede students’ mastering of standalone CT concepts. In fact, a systematic review of CT learning concluded that students generally learn about standalone CT concepts in 4th grade, then move to combined CT concepts towards the end of 6th grade, for example, loops before nested loops, conditionals before conditional loops, and so on (Zhang & Nouri, 2019). As such, students engage in meaningful learning as the learning progressions are consistent with the levels of students’ knowledge development (Clements et al., 2020).

Second, the concept of pattern recognition is inherent in the very first learning goal in the Repetition LT, that “Some tasks involve repeating actions,” which implies students’ ability to “identify repeated actions within sets of instructions, or describe a task that requires repeated actions and what is repeated” (Learning Trajectories for Everyday Computing). The fact that many participants struggled to construct repeat commands based on a repeating pattern highlighted the need to include more instructional content on pattern recognition in the integrated lessons. Currently, for Grade 3, only the fourth lesson in the integrated lesson sequence (Fraction Circles I) introduced the content on the cumulative effect (i.e., repeatedly add equal shares of the whole to represent fractions). Then, the next time that the concept of repetition is reinforced is in the tenth lesson, The Frog and The Fly, where the repeat block is introduced. In this case, there was limited targeted instruction on pattern recognition, which would allow students to practice
identifying the actions and the number of times the actions need to be repeated in a specific task. Therefore, it was suggested that more targeted instructional activities that focus on pattern recognition be integrated into the lessons in a consistent and coherent manner. To introduce new content and/or skill, such as pattern recognition, teachers may apply the “I do it, We do it, You do it” (Archer & Hughes, 2011) instructional model before allowing students to engage in hands-on coding activities. The three-step process starts with teacher modeling or demonstration (i.e., “I do it”), where students are shown how to perform the skill; then the teacher provides scaffolds and guide students in practice (i.e., “We do it”); and finally, students perform the skill independent of teacher assistance (i.e., “You do it”). As such, Table 5-1 presents the targeted instruction for pattern recognition using the three processes with clear instructional goals and activities that teachers may use to help students progress in the LT.

**Decomposition for G3 and G4**

Students’ articulation of decomposition was more evident in solving arithmetic problems than in the word problem. For example, items DC.06.c (list the multiple steps involved in finding the area of a polygon) and DC.02.b (decompose the problem: (5 x 2) + (3 x 2)) were mapped to the same KSAs, which means that answering the two items require comparable decomposition competencies (i.e., breaking up systems and problems into smaller parts and use decomposition as an early step in problem-solving). Due to the participants’ familiarity with arithmetic problems involving the order of operations, students generally articulated decomposition (i.e., performing multiplication before addition) in DC.02.b.

However, participants’ solving the word problem, DC.06.c, may have been challenged by their understanding of math content knowledge (i.e., finding the area) in which the CT item was contextualized. By design, the CT assessment items utilized math content that was below the instructional grade level of the students so that math content knowledge would not prevent
participants from articulating CT (Gane et al., under review). However, data analysis revealed that the math content involved in item DC.06.c may still have presented a hurdle for the participants. Specifically, many participants presented single-step solutions that involved tracing the outline counting the dots along the outline of the polygon. Such solutions suggested that they were attempting to find the perimeter, instead of the area, even though the problem specifically asked for the area. Many participants either confused area with perimeter or lacked the appropriate math content knowledge of how to find the area. Therefore, it was hard to make inferences about students’ decomposition when participant responses were interfered with mathematical misunderstandings. Even among the participants that did decompose the problem and solved for area in multiple steps, few articulated the mathematical formula for area (i.e., area = length multiplied by width) but looked for how many smaller squares that comprise the polygon. In such cases, participants’ use of problem decomposition was recognized even when the solutions did not yield a mathematically correct result. The implication for assessment design is that item DC.06.c should be revised in order to minimize the interference of the math content knowledge.

**Conditionals for G4**

Participants in this study had difficulty evaluating conditions in items embedded in coding. This was again consistent with the conclusions from the pilot study. Previous literature (e.g., Luo et al., 2020a; Martinez et al., 2015), including the pilot study (Luo et al., 2020b), showed that students are able to propose an outcome action to a simple conditional statement. For example, students could program a robot to perform different actions depending if there is an obstacle in front. However, students faced more challenges when they had to evaluate the true or false state of a condition to reach the intended outcome, e.g.: If 5>7, sound “pop;” Otherwise, sound “bing.”
The reason that G4 students had difficulty articulating evaluating the conditional logic could be multifold. The concept of conditionals (i.e., condition evaluation and the true and false states of a condition), is content that elementary students have mostly had experience with in daily life contexts. For example, if it is snowing, then outdoor physical education classes will be canceled. Or, if there is a pandemic, school will be canceled. However, students showed difficulty in transferring their knowledge in condition evaluation from daily life experiences to coding (e.g., deciding which sound will play: “If 5 < 8, then play a ‘pop’ sound; If 5 > 7, then play a ‘bing’ sound”). Such findings again highlight the need for LT-based instruction (Clements et al., 2020) to help students connect new ideas and knowledge with their pre-existing intuitions (Resnick et al., 1996) instead of making assumptions and skipping levels.

Another potential reason for students’ inconsistent understanding of conditionals could be the limited instructional coverage and activities on conditionals in the math-CT integrated curriculum. The very first G4 lesson in the integrated lessons introduced students to evaluate conditions and their connected outcome in the unplugged format. However, that content was not repeatedly reinforced throughout the first third of the curriculum and the conditional concept was not reintroduced as the focus of the lesson until half a semester later. Such design of the lesson content may partially explain why some students could evaluate conditionals involving inequalities in phase 1 cognitive interview, but not in phase 2 interview. Therefore, instruction should allow students to incrementally develop contiguous levels of understanding and skills instead of skipping levels (Clements et al., 2020). Similar to the suggestions made in the discussion of targeted instruction on repetition, the three-process could be applied here to teach conditionals. Table 5-2 presents the clear instructional goals and step-by-step targeted activities to foster an early understanding of the “True or False” states of conditions.
In the other conditionals item (C.03.b), students faced challenges in evaluating the conditional statement involving user input. The assessment items were intended to assess the knowledge and skills in a single LT (Gane et al., under review). However, item C.03.b is potentially assessing both conditionals and variables (i.e., how to take user input) at the same time. Given that students may already have difficulties with condition evaluation, adding a second CT concept (i.e., variables, user input) would likely result in students’ not being able to clearly articulate CT competencies that were sought by the item. Therefore, it is important to reiterate Clements et al.’s (2020) LT-based instruction for incrementally developed understanding and skills and provide students with opportunities to master standalone CT concepts before combined CT concepts (Zhang & Nouri, 2019). As such, it is suggested that the revision of item C.03.b potentially focuses on retaining the conditionals component and taking out the user input component.

**Different Perspectives of CT Assessment**

Data analysis of this dissertation revealed that, in some cases, such as sequence, participants generally articulated complete, precise, and ordered instructions. In other cases, such as decomposition and repetition, even when two items elicit the same CT competencies, the students articulated different levels of CT depending on whether it was a coding item or a word problem. In such cases, the item contexts were further examined to unveil how they may have resulted in the differences in students’ CT articulation. Chapter 3 described that the assessments used in this study were developed to provide insight into student learning in the math-CT integrated curriculum. The items used either illustrations of the Scratch interface (e.g., code blocks), word problems embedded in the math context, or simple arithmetic problems. The following section discusses students’ articulated CT within the context of the different forms of assessment items.
Affordances of Scratch

Previous research (e.g., Gane et al., under review; Zhang & Nouri, 2019) has provided justifications for using Scratch in K-9 CS education. For one, Scratch is commonly used by and popular among young learners (Maloney et al., 2010). Given its block-based drag-and-drop features, Scratch allows young users to engage in coding tasks without attending to complex syntax details. Secondly, Scratch projects are designed to have “low floor, high ceiling” (i.e., easy to start and flexibilities for increased complexity), which can accommodate students with a wide range of learning goals (Maloney et al., 2010). Therefore, plenty of research studies around the world leveraged the affordances of Scratch in teaching elementary students CT and assessed students’ CT in the context of Scratch (e.g., Brennan & Resnick, 2012; Seiter, 2015; Sáez-López et al., 2016; Funke et al., 2017; Tsukamoto et al., 2017). In this study, Scratch projects that support specific learning goals were integrated into the LT-based integrated curriculum that was implemented over the course of the school year, so participants were generally familiarized with the features in the platform (i.e. the interface, code blocks). In fact, participants demonstrated the ability to use the code blocks to produce intended outcomes (e.g., in items S.01.a and S.02.a) and recognize the function of blocks (e.g., in R.03.c). Therefore, the assessment items using Scratch codes were familiar to the participants and participants could leverage previous knowledge using objects and actions that were familiar (Resnick et al., 1996).

Word Problem

Word problems may have introduced confounding variables, such as reading, comprehension ability (i.e., in-depth analysis of concepts and relationships in the problem text, Savard & Polotskaia, 2017) and content area (in this case, math) competency, thus adding variability in students’ articulated CT. For example, item DC.06.c requires students to understand that the goal was to present the steps in finding the area, and only a small number of
respondents articulated use of decomposition; whereas for item DC.02.b, which maps to the same KSAs (assessing the same level of decomposition), almost all respondents effectively used decomposition in calculating the result of the arithmetic number sentence. To make such a conclusion, it is necessary to rule out the possibility that the word problems used in this study systematically require more complex CT knowledge and skills than the non-word problems. Therefore, the KSA mapping of all items as examined. For the sequence items, the word problem (S.04.b) assesses a lower level of sequence than the coding items do; Items in the Decomposition and Conditionals LTs re mapped to identical, comparable levels; The two word problems in the Repetition LT are mapped to an additional KSA than the non-word problem is. Therefore, it was concluded that the word problems used in this study sought comparable CT knowledge and skills as the non-word problems. However, it should not be assumed that items mapped to the same KSAs have the same difficulty level. The difficulty level of assessment items in an integrated context should be holistically evaluated considering both the content knowledge of the subject area that CT is integrated into and the CT competencies items elicit.

Even though the word problems and the non-word problems address comparable CT competencies, assessment items in the word-problem format may still pose challenges to students. Decades of research in mathematics education has shed light in why word problems may be more challenging to students (Leong & Jerred, 2001; Savard & Polotskaia, 2017). Solving a word problem requires students to have declarative knowledge, that is, a student needs to have foundational knowledge and understanding of the number system and know how to use numbers to represent units. To act on the numbers, students also need to have procedural knowledge (i.e., know the rules for calculations, computation, execution, etc.). Students also need conceptual knowledge, such as, understanding the mathematical relationships in the
problem and knowing what the goal is (Leong & Jerred, 2001; Savard & Polotskaia, 2017). Therefore, word problems can be more challenging to students than simple arithmetic problems because of the semantic and mathematical structures involved (Savard & Polotskaia, 2017). Compared to arithmetic, where students can directly work on arithmetic operations, word problems require students to engage in flexible and holistic analysis of the problem. Such analysis includes translating the problem into mathematical expressions while taking into consideration the underlying concepts and relationships between quantities. While identifying the numerical units poses no obvious challenge, students are often result-driven and are inclined to go straight into calculations without first evaluating the reasons and strategies for doing them (Savard & Polotskaia, 2017). This tendency was confirmed by responses from participants who could easily articulate the calculations involved in dividing nine cookies among three friends (item R.01.a) but failed to articulate the repeating pattern. Many researchers also viewed this tendency as a lack of abstraction skill (Hazzan, 2003; Perrenet et al., 2005; Armoni, 2013).

This tendency to resort to their own familiar problem-solving strategies (i.e., using mathematical representations) without attending to the bigger problem context was theorized by researchers as “reducing the level of abstraction” (Hazzan, 2003). Specifically, the participants were not consciously and purposefully moving back up to the problem level when necessary. The following section defines abstraction in the context of CT and CS, followed by an explanation on the four levels of abstraction.

Abstraction is an indispensable skill for mathematics learning and has important implications for CS education (Grover et al., 2016). It is the cornerstone that underlies Wing’s (2006) definition of computational thinking. Abstraction, in CS, refers to the process of ignoring nonessential information and focusing and acting on only the essential aspects in a given
situation. Abstraction is not static--there is no ultimate step-by-step guidelines for thinking in abstraction, but rather, it is a fluid skill that requires thinkers to move along a continuum deciding what to ignore and what to retain as essential as problem-solving evolves (Hazzan, 2003; Perrenet et al., 2005). That continuum was described to be a hierarchy consisting of four levels of abstraction in CS (Armoni, 2013; Perrenet et al., 2005): (1) The problem level -- the highest level of abstraction-- is viewing a problem as a whole and thinking what algorithms is applicable in this specific problem-solving scenario; (2) The object level, where thinkers zoom on to the problem closer and focus on the construction of the algorithm without detailing out the algorithmic steps or the programming language to be involved just yet; (3) The program level, where thinkers delve further into the algorithmic steps in a specific programming language; and (4) The execution level -- the lowest level in the hierarchy -- where the written algorithm is put to test.

Solving a problem using abstraction is analogous to driving a car. The problem level for the driver is to know that they need to be able to operate the vehicle to reach a destination. Here, they are ignoring the mechanics and the operational process involved in accelerating and stopping a car, but rather, focusing on knowing where their destination is. Then, in the object level, with their destination in mind, they recall the steps involved in driving, without specifying the automobile brand of the vehicle. This involves knowing that they will be using the steering wheel, the throttle, the brake, and so on. Next, the driver steps into the driver seat, and focuses on the specific automobile brand that they are to operate to reach the destination (the program level). Now, the driver devises a plan to use the combination of the operation system at hand. Finally, in the execution level, the driver puts that plan to test -- operating the vehicle. Recall that thinking in abstraction requires thinkers to move in the continuum of the four levels. In the
driving analogy, thinking in abstraction means that the driver will need to be able to constantly reflect on their operation of the vehicle to make sure that they will finally reach the destination. If the vehicle has deviated the planned route, the driver will need to come back up to the program level to remind themselves what the destination is. If the vehicle is on route, the driver can “reduce the level of abstraction” by focusing on the operation more.

In the case of solving a word problem, students may delve too soon into the arithmetic calculations (the program level and the execution level), losing sight to what the problem goal is (the problem level). For example, in item R.01.a, students were asked to give out nine cookies to three friends, Sally, Val, and Lee, by using the “Repeat 3 times” instruction at least once. Here, thinking in the problem level would be understanding that they need to use the repeat instruction in their solution. However, students failed to fully understand what this entailed and proceeded to “reduce the level of abstraction” to the program level by formulating the arithmetic expression (e.g., 3 X 3 and 9 / 3) and the execution level by calculating the result (e.g., each friend gets three cookies). Students failed to attend to the broader scope of the problem, which was using the repeat instruction. Therefore, the students likely lacked the skill to move among the abstraction levels, i.e., thinking in abstraction.

To teach elementary students how to solve word problems using abstraction, Rich and Yadav (2020) proposed a framework that was adapted from the aforementioned four-level system. The four adapted levels were, from higher level to lower, the Problem Comprehension level, the Solution Visualization level, the Solution Planning level, and the Solution Enactment level. Applying this framework to item R.01.a mentioned above, students are expected to first understand what the problem is -- to use the repeat instruction in giving out the cookies instead of doing multiple actions over and over again (problem comprehension). Next, students need to
visualize what a potential solution might be by identifying the repeating pattern in the cookie-disseminating action. Then, students need to carefully plan out how many cookies to give away to each friend with the repeating pattern, before writing out the final instruction. To fully exercise thinking in abstraction, students need to re-attend to the problem whenever appropriate.

**Interpreting the Correspondence between Articulated CT and the LTs**

This section discussed how the correspondence between students’ articulated CT and the LTs should and should not be interpreted. Recall that Chapter 4 presented that students were generally mapped to all the beginning and intermediate level goals in the Sequence LT; In the Repetition LT and Conditionals LT, students were mapped to a range of learning goals, showing different levels of CT understanding; In the Decomposition LT, students were mapped a range of the beginning learning goals, with almost all mapped to at least the first one. However, many nuances exist in these general statements. The purpose of this section is to discuss how to properly interpret the correspondence and not to over-generalize the interpretations. Specifically, the context of the assessment items and the differences in proficiencies (the “know” and the “do”) are discussed below.

**Context of Assessment Items**

For repetition, conditionals, and decomposition, students’ articulated CT showed a range of responses depending on whether the item was set in a coding, arithmetic, or word problem context. Therefore, the first consideration for making interpretations of the correspondence was the context of assessment items. This study found that, given the affordances of embedding the assessment content in a familiar coding platform (i.e., the students could select from a range of given blocks), the participants generally recognized the outcome of a program and construct a working program by selecting and sequencing the appropriate blocks (e.g., items S.01.a and
S.02.a, programming the Scratch cat to move along a number line; item R.03.c, deciding the outcome of a repeating program).

When solving the CT items in arithmetic contexts, the participants generally had transferable knowledge from math that helped them solve the problems (e.g., items DC.02.a, DC.08.a, breaking up a number). Participants’ solving the arithmetic problems did not necessarily mean that they could solve a matched-level word problem as effectively (e.g., DC.06.c, breaking up the problem to find the area of a polygon; DC.02.b, breaking up the arithmetic problem to find the result). This highlighted the necessity to use assessment items set in diverse contexts in order to obtain a comprehensive understanding of students’ CT. Also, students’ CT proficiency level should be viewed as a fluidity: rather than using a single learning goal in the LT to provide a diagnostic of a students’ CT, look at a collection of learning goals that describe the range of the students’ CT competencies. In conclusion, contexts of assessment items should be considered in order to avoid overgeneralization of interpretations.

**Knowledge, Skills, and Abilities (KSAs)**

From an assessment perspective, the KSAs for the LTs describe the “proficiencies required for disciplinary competency” (Gane et al., under review). According to the Learning Trajectories for Everyday Computing (n.d.), each of the learning goals (Rich et al., 2017; 2018) in the LTs describes the expectations for what students know and can do. For example, one learning goal in the Conditional LT is, “A condition is something that can be true or false.” The “know” requires students to “Understand that a condition is a statement that can be classified as true or false” and the “do” requires students to “Assess truth-value of a condition.” While the two requirements are collapsed into one learning goal, data analysis highlighted the need to distinguish the “know” from the “do.” Participants in this study demonstrated different proficiencies of CT even under the same learning goal. Data also suggested that students may be
mapped to a range of learning goals in an LT but may in fact demonstrate only the “know” and not the “do”. Therefore, it is suggested that, when interpreting the learning goals in the LTs (Rich et al., 2017), it is necessary to consider students’ understanding and the actions that demonstrate the understanding.

For example, in repetition, the participants were mapped to a range of learning goals including “Some tasks involve repeating actions,” “Instructions like ‘Step 3 times’ do the same thing as ‘Step, step, step,’” “Computers use repeat commands,” and “Repeating things can have a cumulative effect.” However, students demonstrated more proficiencies in knowing about the cumulative effect of repetition than creating it. An example was that, when given a set of code in R.03.c, students understood that a “Repeat 3 times” block would produce the same outcome as doing the action three times (the “know”). However, when asked to create instructions using repetition in R.01.b, many students failed to create the cumulative effect using a repeat command (the “do”). Therefore, even when a student was mapped to a learning goal, it did not necessarily mean that they have demonstrated both the “know” and the “do.”

One exception, however, was in sequence, where the participants were generally mapped to most of the beginning and intermediate level learning goals, demonstrating both the “know” and the “do”. Specifically, students could decide the outcome of a given set of sequenced code, as they could construct sequenced and precise code to reach a desired outcome.

Implications

In light of the complexities involved in defining, integrating, and teaching CT, this dissertation study has a few major contributions and implications.

First, previous research has explored “what to teach” and “what should be learned” in K-9 CS education. However, a systematic match between what is offered in existing curricula and what is developmentally appropriate for students is yet to be established (Zhang & Nouri, 2019).
This study provides an example for teaching integrated CT using LT-based instruction and generates initial empirical evidence for this sample of G3 and G4 students’ current CT levels after engaging in the integrated instruction. The cognitive interviews served as a diagnostic approach to elementary students’ CT learning and investigated where this specific sample of students were having conceptual gaps by using the LTs as a theoretical reference point. The cognitive interviews offer affordances that the other assessment types (i.e., tests or project portfolios) may miss. For example, in a written multiple-choice test or a programming project, students can guess an answer to a multiple-choice question without having to provide the rationale that supports their answer or use certain blocks without knowing how they work. However, in a cognitive interview, students do need to justify their answers and students’ misconceptions and/or learning gaps, if any, can potentially be revealed. Future research can further explore the CT problem-solving processes of upper elementary grade students with different characteristics (e.g., background, learning needs, etc.) and how they develop CT in LT-based targeted instruction.

Secondly, research in mathematics education has established that an LT consists of empirically supported learning goals, learning progressions, and instructional activities that support student progressions (Simon, 1995; Clements & Sarama, 2009). This study addressed the gap in current CT research by empirically testing the LT-based instruction and activities mapped to specific learning goals in the LTs and demonstrated how they could support students’ progression in increasingly advanced CT learning. Such empirical evidence can also be used to refine the LTs by making suggestions to the LT-based targeted instructional activities for specific learning goals to support students’ progressions. The study also highlights the need to further detail targeted instructional activities for each learning goal in the LTs. With a better
understanding of learning difficulties students may have, targeted instruction can then provide targeted and better learning outcomes (Guzdial, 2017). Future research can focus on conducting longitudinal studies that examine the efficacy of LT-based targeted instruction in CT and students’ learning growth over time.

Thirdly, a lack of instructional resources and PD support has been a challenge for teachers to introduce CT in their classrooms (Lee et al., 2011). This study mapped the G3 and G4 participants to the LTs, providing confidence in the developmental appropriateness of the LT-based curriculum. With examples for targeted instructional activities for specific learning goals, teachers can better leverage the content to be introduced to upper elementary grade students and set appropriate learning goals to students. This provides teachers with resources to better support and foster elementary students’ CT learning. Future research is encouraged to understand how instruction with targeted instructional activities help teachers establish confidence in integrating CT in their classrooms.

Lastly, this study serves as an example to use the cognitive interview technique as a means to explore students’ CT. Cognitive interviews reveal the “ideas, concepts, and skills that children can express, articulate, and develop when they are given the opportunity to engage with rich and interesting activities” and allow students “to express and reconsider their experience and understanding” (Confrey et al., 2014, p.245). Such verbal data of students’ applying CT in problem-solving scenarios afford a deeper understanding of how students are developing and applying CT, which is often missed by using traditional tests that assess CT as a learning product. This study also made item revision suggestions that future research can refer to when designing elementary CT assessments.
Limitations

There are a few limitations to this dissertation study. First, given the qualitative nature of this study, only a small number of students participated in the cognitive interviews. Therefore, the findings are specific to this sample of students that participated in the math-CT integrated curriculum. The participants are from a school that has adopted a CT and CS initiative, providing students with opportunities to learn CT and CS outside of the curriculum. Therefore, the findings may not be transferable to students who do not have the same level of exposure. Secondly, while the goal of this dissertation was to better understand elementary students’ CT, due to the methodology used (i.e., cognitive interview with a think-aloud protocol), the interpretations were made based on how students articulated their CT. Thirdly, the participants sampled come from four classes, each with a different home teacher that implemented the curriculum. While all teachers covered the required lessons before each cognitive interview, the teachers may have slightly different instructional pace and teaching styles that may lead to different levels of implementation fidelity. In addition, the interpretations of the findings were limited by the number of assessment items sampled for each LT. Also, the assessment items sampled for this study cover only specific portions of the LTs. While the repetition, conditionals, and decomposition items range in difficulty and elicited different levels of CT, the sequence items received more consistent responses. Therefore, it is possible that students have more advanced understanding that is not captured by the sequence items sampled. Finally, the findings and the suggestions made in this study should be interpreted acknowledging the interrelated design and the complementary relationships among the LTs, the curriculum, and the assessments. Interpretations may not be transferable to cases where the curriculum and assessments were not aligned to the LTs.
Conclusion

This dissertation study provided a detailed qualitative account of 3rd- and 4th-grade students’ CT articulations as they progressed in integrated math-CT instruction. Using the cognitive interviews, the study collected evidence of students’ articulated CT proficiencies and examined students’ CT with respect to the learning goals and progression hypothesized in the CT learning trajectories. The findings provide initial empirical evidence to validate the LTs and highlight the need to further define instructional activities that support individual learning goals in the LTs. The study also contributes to the body of literature in elementary CT assessments using Scratch and word problems. The findings are important to researchers, educators, and teachers who are designing and implementing CT in elementary classrooms.
Table 5-1. Suggested targeted instructional activities for pattern recognition.

<table>
<thead>
<tr>
<th>Learning goals in the LT</th>
<th>Instructional goal</th>
<th>Three-process targeted instruction with sample instructional activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Some tasks involve repeating actions. (B)</td>
<td>Students identify the pattern (i.e., what is being repeated over and over again?)</td>
<td>Give away classroom supply, such as pencils or books, to two students, Sally and Lee, in the following example procedure: Give Sally a pencil, Give Sally a pencil, Give Lee a pencil, Give Lee a pencil. Explain the first repeated action in this short procedure (“I do it”). Then, ask students to join you in identifying the second repeated action (“We do it”). Level-up the activity by using different objects and/or shuffling the order. Or, try adding a third repeating action, and have students work on their own to identify what is/are being repeated (“You do it”).</td>
</tr>
<tr>
<td>3. Instructions like “Step 3 times” do the same thing as “Step, step, step.” (B)</td>
<td>Students construct instructions to show the repeating pattern.</td>
<td>Explain how to construct instructions using “Repeat X times” instead of the identical instructions twice. For example, one repeat instruction for the above procedure is: Repeat 2 times: Give Sally a pencil; (“I do it”) Then, ask students to join you in constructing the repeat instruction for the second action (“We do it”). Level-up the activity and have students work on their own on constructing the repeat instructions for a different procedure (“You do it”).</td>
</tr>
<tr>
<td>1. Repeating things can have a cumulative effect. (I)</td>
<td>Students decide the outcome of a repeating action.</td>
<td>Have students decide the outcome of a given procedure with repeat commands. For example, the outcome of the above procedure is that both Sally and Lee will get 2 pencils. (I-do-it and We-do-it may be skipped if students can independently work on the activities.)</td>
</tr>
</tbody>
</table>
Table 5-2. Suggested explicit instructional activities for condition evaluation.

<table>
<thead>
<tr>
<th>Learning goals in the LT</th>
<th>Instructional goal</th>
<th>Three-process targeted instruction with sample instructional activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>A condition is something that can be true or false.</td>
<td>Familiarize students with the “true or false” statements (e.g., whether a statement is true or false.)</td>
<td>Prepare simple statements (e.g., number sentences), such as, 3X5=20 5+5+3=13 Explain to students that both statements would be “false” because they are incorrect (“I do it”). Connect to the “Right or wrong” language.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Try the activity by using real-life examples: “I am wearing red today;” “It is raining now.” Ask students to join you in deciding whether the statements are true or false (“We do it”).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Have students write statements and have a peer decide whether the statements are true or false (“I do it”), e.g., 2&gt;5 3/1&gt;4/1</td>
</tr>
<tr>
<td>Learning goals in the LT</td>
<td>Instructional goal</td>
<td>Three-process targeted instruction with sample instructional activities</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>A conditional connects a condition to an outcome.</td>
<td>Students understand that a conditional statement always has an outcome.</td>
<td>Add an outcome action to the “True” state of the condition. For example, If “3X5=15” is true, add 1; If false, do nothing. Explain how you evaluate this conditional statement (“I do it”). Ask students to join you in deciding the actions for the following conditional statements (“We do it”): If “2&gt;5” is true, add 10; If false, do nothing. If “3/1&gt;4/1” is true, add 10; If false, do nothing. Have students work on deciding the outcomes for the following (“You do it”): If “today is Monday” is true, jump once; If false, clap once. If “3/1&lt;4/1” is true, say “correct!” If false, say “wrong!”</td>
</tr>
<tr>
<td>Each of the two states of a condition may have its own action.</td>
<td>Students tell/execute the outcome action depending on the true or false state of the condition.</td>
<td>Explain that the words “… is true” is always omitted in coding, for example, “If 3X5=20 is true, add 1”; Is the same as “If 3X5=20, add 1”; And the words “if false” is often replaced by “else,” for example, “If ‘today is Monday’ is true, jump once; If false, clap once.” Is the same as “If ‘today is Monday’ (is true), jump once; Else (If false), clap once.”</td>
</tr>
</tbody>
</table>
APPENDIX A
THINK-ALOUD PROTOCOL

Prepping the participant:
[Intro] Hello. I am *researcher name*, a student at * school name*. I am interested in learning about how you solve the following problems. This activity will take about 15 to 20 minutes. Your parents said it’s Okay for you to do it. And I just need you to sign this (assent) before we start. I am recording our conversation so I don’t have to take notes. The set ID is # - # (e.g. 3-1).

We are going to do a “think-aloud” activity. Basically, I’m asking that you tell me what you are thinking as you work through the problems. I didn’t write these questions, you can say whatever you are thinking while working on them. You will not be graded on this. Whatever you say won’t hurt my feelings.

Now I will show you an example of “thinking aloud” when solving a problem [show participant the item].

Kristen uses her fraction strips to compare $\frac{1}{3}$ and $\frac{1}{4}$.

<table>
<thead>
<tr>
<th>$\frac{1}{3}$</th>
<th>$\frac{1}{3}$</th>
<th>$\frac{1}{3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{1}{4}$</td>
<td>$\frac{1}{4}$</td>
<td>$\frac{1}{4}$</td>
</tr>
</tbody>
</table>

Kristen writes this number sentence: $\frac{1}{3} < \frac{1}{4}$

Do you agree with Kristen? _________

Use Kristen’s fraction strips to help explain your answer.

[anno: So the question reads, Kristen uses her fraction strips to compare $\frac{1}{3}$ and $\frac{1}{4}$. And she writes $\frac{1}{3}$ is smaller than $\frac{1}{4}$. Do you agree with Kristen? Use fraction strips to help explain your answer. So I don’t agree with Kristen because $\frac{1}{3}$ should be bigger than $\frac{1}{4}$. So if you divide the same area into three equal parts, each is $\frac{1}{3}$, and that’s how big each part is (referring to the top strip and marking the first $\frac{1}{3}$ ). If you divide the same area into 4 equal parts, each is $\frac{1}{4}$ and this big (referring to the bottom strip and marking the first $\frac{1}{4}$). So $\frac{1}{3}$ is bigger than $\frac{1}{4}$. ]

Now I will give you a couple of problems. Please remember to talk out loud just like I did when working on a problem. If you go silent for more than 5 seconds, I might ask you to describe what you are thinking. I will also ask you to tell me what you think of the item. Is it okay?

Prompting questions -DURING think-aloud:
- (At the very beginning) Now go ahead and start by reading aloud the question.
- Can you tell me what you are thinking right now?
Questions should be asked with the goal of probing for the following if the student did not provide enough “thinking”
  - For multiple-choice: why an answer is or is not selected?
  - For open-ended tasks: probe for the rationale of the given solution.

Wrap-up reflection question-AFTER think-aloud:
  - What do you think this question is asking you to do?
  - What subject or subjects you learn at school do you think this is?
  - What do you think that your classmates will need to know in order to answer this question correctly (probe for examples/detail)?

Remember: we are NOT trying to help participants answer the questions correctly, rather, we are uncovering whatever the participant is thinking when problem-solving.
APPENDIX B
SAMPLE INTEGRATED MATH-CT LESSON

Lesson 2-3A
Action Fractions
Math+CT

Animal Number Story
Math Connections: Children use Scratch to show a simple number story.
CS Connections: Children explore the Scratch workspace. Children learn that the Scratch
workspace and computer programs are made up of different parts. They search for and try out
different commands and learn which changes of and within commands can change the program’s
output.

Before You Begin
Create your class via your teacher account on the Scratch site and generate a list of student logins.
Create individual credential cards to distribute to children. Decide how to manage these credentials in
the event that children forget (e.g., keep a master list, have children tape their credential cards to their
desks). Decide how children will get to the Scratch website (scratch.mit.edu). Consider pre-loading a
bookmark on each child’s desktop.

Vocabulary
event • sprite • script • block • remix

Warm Up 5–10 min
Materials
Starting with Scratch
Children discuss using coding for extra math practice.
“I Can…” Statements
Children read the explicit Math and CS goals.

Focus 45–50 min
Logging in to Scratch
Children log in and finish setting up their accounts.
Programming Events
Children discuss events in a programming context.
Introducing TIPPSEE
Children learn a strategy for exploring Scratch projects.
Animal Number Story TIPPSEE
Children use TIPPSEE to explore a project.

“I Can…” statements
• I can log into my Scratch account.
• I can identify the important parts of Scratch.
• I can closely observe a Scratch program and find the scripts that caused the actions.
• I can remix, save, and share Scratch programs.
• I can make changes to a program so that it does what I want it to.

Anticipated Barriers
• Logging in to a new program can be challenging for some children.
  Providing pictorial or video versions of the verbal/written directions would be helpful.
• Children often forget the procedure for remixing, even after being taught.
• Children often forget to remix.

Student Options
Consider these options for adapting the lesson to your students’ preferences:
• include simplified directions both within the Scratch projects and on a paper.
• Offer alternate versions of the directions (e.g., video).

Computational Thinking
• COMPOSITION: Systems are made up of smaller parts.

3.NBT.2
Animal Number Story project; Animal Number Story TIPPSEE Journal page

Student Credential cards (one per child)
1 Warm Up 5-10 min

Starting with Scratch

Start with a whole-class discussion about using Scratch to get extra practice with math and build coding skills. Tell children that they will be using computers to program in Scratch many times this year, and that today they will focus on how to get started. Explain that children should pay close attention when you demonstrate steps on your computer, so that they will have an easier time when they go to their own computers to use and modify the Scratch project (program).

Briefly poll the class about their experience with Scratch. If there are any children who claim to be very experienced, consider asking if they are willing to help their classmates and adjust seating arrangements to spread them out accordingly. You may also wish to briefly talk to children about how to be respectful of and careful with the computer equipment.

Distribute login credentials to each child, and keep an extra set for you to model the steps for a first-time login for children.

I Can ...

Display the "I Can ..." statements and remind children that these statements express the goals for today’s lesson and can give them clues about what to expect. Carefully read each statement and ask them to use their thumbs to show how true they feel each statement is for them right now (thumbs up for yes, thumbs down for no, thumbs sideways for maybe). Remind them that during the wrap up you will look at the "I Can ..." statements again, and see how their opinions about the statements have changed.

2 Focus 45-50 min

Logging in to Scratch

Display your screen and model the steps to log in and arrive at the Scratch class home page. (See the Scratch Login Instructions document.)

When ready, have children go to their individual computers to attempt these steps. Each child should log in using their credentials and follow the prompts to finish setting up their accounts. Children will need to enter their own Birth Month, Birth Year, Gender, and Country. Circulate and assist children as needed.

After each child has had sufficient time to log in, have them close their devices or gather as a group away from their desks. Lead a whole-class discussion on their experience getting to the Scratch site and logging in. Be sure to address any challenges (e.g., typing in the url, finding the bookmark, etc.) that children experienced.
# Programming Events

Engage children in a conversation to introduce the idea of events as they relate to programming. A question and answer dialogue could go like this:

- Can anyone give me an example of an event? Sample answers: festival, 4th of July, wedding. Right – an event is a significant thing that occurs.
- How about this? (tap a child on the shoulder) – the child will probably look at you. What happened? I tapped X on the shoulder. What was the response? Sample answer: She looked at you. Was that an event? Answers vary.
- In programming, that is an event. That was something I did to X, and X had a specific response to that event. Typically, when you tap someone on the shoulder, they look at you. Are there other events that happen to you that have a specific response to them? Sample answer: The bell rings at the beginning of the day, and we get in line at our classroom number.
- Computers have events they can respond to. What are the things you can do to a computer to make it do something? Sample answers: Type on keys, use the mouse to click places.
- In Scratch, you will learn how to make the computer do fun things when you type keys or click specific places.

# Introducing TIP&SEE

Display your screen showing your Scratch class home page. Introduce the purpose of TIP&SEE. Explain that Scratch is very powerful, so it has a lot of buttons, tabs, and pictures, and it can be overwhelming. You are going to teach them a strategy, TIP&SEE, which will help them navigate Scratch and focus on the important parts of each project.

Narrate your steps while you scroll down to the class studio and find and open today’s first project: Scratch Basics - Zood Goes Home (https://scratch.mit.edu/projects/230348882/). Model using TIP&SEE to look at a Scratch project by using the following scripted think-aloud. Display the TIP&SEE poster next to the project, and be sure to point out each part of the Scratch project page as you refer to it.

Now we are going to use a strategy for exploring Scratch projects. This strategy is called TIP&SEE.

First, we get a TIP (point to the first four letters of the word) from the project page. A tip is a clue or a helpful hint. Now let’s look at our Scratch Project Page to see what we can find out about this project.

- T stands for Title. The title is at the top. First we read the title, which may tell us what the project might be about. Could someone help me read the title? Scratch Basics - Zood Goes Home. What do you think might happen in this project? Answers vary.
- I stands for Instructions. The instructions are in the top right bar. These tell you how to run the program. Could someone help me read the instructions?
• P stands for **Purpose**. The purpose comes after the instructions. This tells you what you, the programmer, are supposed to do with this project today. Could someone help me read the purpose?

• P stands for **Play**. It is not just enough to play — when I play, it is very important that I remember three questions — what event did I do, what sprites did something, and what action did each sprite do? Event, sprite, and action. Each time I do something, I can record what I saw happen on my TIPPSIE journal page. Scratch programs begin when you press the green flag. (Press the green flag to play the program.)

What happened? **Zood talked.** Now let’s press the space key. What happened? **Zood walked.** You can also click on sprites. If we click on Zood, what happens? **Zood walks the other way a bit.** What about if we click on the rocket? It’s blast off.

I have previewed the Scratch Project page using TIPPSIE. Now it is time to **SEE** inside so that I can learn about the code that makes it work!

We go inside the project to find code that made the actions happen. Remember that when we played the program, we noted event, sprite, and action. For example, we can find the code that made Zood talk when I pressed the green flag. Inside the project, we start with:

• S stands for **Sprites**. What does S stand for? **SPRITES!** I can see that there are two sprites in this project. I first click on the Zood sprite to find some code.

• E stands for **Events**. What does E stand for? **EVENTS!** Now I can find which script did which action. A script is a set of instructions for a computer that go together. In Scratch, a script is a set of blocks, or instructions, that are linked together.

Look at Zood’s green flag script! Look at those say blocks!

Okay, let’s find the script that makes the spaceship blast off! What did we do to make the spaceship blast off? **clicked on the spaceship.** Click on the spaceship sprite, then look at its Events for when this sprite clicked. I see that it kept moving up.

• E stands for **Explore**. What does this E stand for? **EXPLORE!** Now is the time when I get to figure out how blocks and scripts work by making changes to a script and seeing what happens. I can change a number or other information that is typed in a block. I can also add a block or delete a block.

I’m going to try changing how long the block waits. (Change the number in the wait ____ secs block from 0.1 to 0.2, press green flag, then press the space key to run the script.) Wow! Zood went slower when I changed that block.

I wonder what happens when I add another say block. I am going to try to have Zood ask us to help him get home. (Add a say ____ for ____ sec block from the Looks category to the script starting with when green flag is clicked and have it say “Can you help me get home?”) Now let’s play it with the green flag again and see what happens. Wow! I can make Zood speak!

There seems to be lots of things that I can do when I explore and I can’t wait to keep experimenting with Scratch!
Animal Number Story TIPPSEE

Tell children that they will now get a chance to try TIPPSEE on a different project, Animal Number Story (https://scratch.mit.edu/projects/277391112/). Direct them back to their computers and distribute the Animal Number Story TIPPSEE journal page. As necessary, remind children about the steps they need to open the project. Have children use the project to complete the journal page individually or in pairs.

Wrap Up

When children have had sufficient time to make changes to their projects, have them Remix, Save, Rename, and Share. Explain to children that when they start with someone else's project, and they want to change how it works, they will need to Remix, Save, and Share it.

Then call them together to discuss today's experience. You may wish to have children Pair and Share for some questions. Suggestions:

- Describe Scratch to your friend. Share can with the group. Answers vary.
- Which "I can..." statements did you do today? Think/Pair/Share. Follow up: Which "I can..." statement was most fun? Answers vary.
- What was your favorite/the most surprising/the trickiest thing about using Scratch today? Answers vary.
- Did your project do anything you did not expect it to do? Answers vary.
- What can you change to make a project do something different? Sample answer: You could change the blocks. You could change the numbers or the words in the blocks.
- What do you understand and remember about the sprites / blocks? Answers vary.
- What questions do you still have about Scratch? Answers vary.

Now "I Can..." Review today’s “I Can...” statements and ask children to use their thumbs to show their opinion of each statement.
APPENDIX C
CURRICULUM MAPPING OF CT CONCEPTS

<table>
<thead>
<tr>
<th>Lesson Name</th>
<th>Grade</th>
<th>EM Placement</th>
<th>DECOMPOSITION</th>
<th>REPETITION</th>
<th>SEQUENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stage 1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brownie Bites (unplugged)</td>
<td>3</td>
<td>1-12</td>
<td>Onramp to 1*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Animal Number Story</td>
<td>3</td>
<td>2-3</td>
<td>Onramp to 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sharing Equally</td>
<td>3</td>
<td>2-9</td>
<td>C*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fraction Circles 1</td>
<td>3</td>
<td>2-12</td>
<td>A</td>
<td>B</td>
<td>C;D</td>
</tr>
<tr>
<td><strong>Stage 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polygon Partners (unplugged)</td>
<td>3</td>
<td>4-4A</td>
<td></td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>Fraction Circles 2</td>
<td>3</td>
<td>5-1A</td>
<td>Onramp to 2; B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Storyboarding: Fraction Comic</td>
<td>3</td>
<td>5-3A</td>
<td>D</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>Strip (unplugged)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fraction Comic Animation</td>
<td>3</td>
<td>5-3B</td>
<td>D?</td>
<td>B?</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lesson Name</th>
<th>Grade</th>
<th>EM Placement</th>
<th>CONDITIONALS</th>
<th>DECOMPOSITION</th>
<th>REPETITION</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stage 1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Introducing Conditionals: A</td>
<td>4</td>
<td>1-3</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rounding Shortcut (unplugged)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scratch Creative 2: Zoo</td>
<td>4</td>
<td>1-6A</td>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Animals Number Story</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Introducing Variables: Robot</td>
<td>4</td>
<td>2-2A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boxes (unplugged)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Variables: Math Chat</td>
<td>4</td>
<td>2-2B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Stage 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fraction Circles: Pizza Sharing</td>
<td>4</td>
<td>3-1A</td>
<td>D</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>Ambling Animals</td>
<td>4</td>
<td>3-4A</td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slicing Sandwiches (Comparing</td>
<td>4</td>
<td>3-6A</td>
<td>I</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>Same Denoms)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comparing Fractions: Same</td>
<td>4</td>
<td>3-6B</td>
<td>I</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>Numerators</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* The numbers and letters correspond to the learning goals and arrows in the learning trajectories. See Appendix E LEARNING TRAJECTORIES WITH INSTRUCTION MAPPING for more details.
<table>
<thead>
<tr>
<th>LT</th>
<th>KSAs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Sequence</strong></td>
</tr>
<tr>
<td></td>
<td>• Ability to create several different sets of instructions to produce the same intended goal (2U)</td>
</tr>
<tr>
<td></td>
<td>• Ability to use / modify / create precise instructions to produce the intended goal (1U; 3.1)</td>
</tr>
<tr>
<td></td>
<td>• Ability to use / modify / create an ordered set of instructions to produce the intended goal (3; 3A)</td>
</tr>
<tr>
<td></td>
<td>• Ability to use a limited set of commands to produce the intended goal (3.2U)</td>
</tr>
<tr>
<td></td>
<td>• Knowledge that computer programs [computational artifacts] require selecting from and ordering a limited set of commands. (4; 4U; 4.1; 4.1U; 5; 5U)</td>
</tr>
<tr>
<td></td>
<td><strong>Conditionals</strong></td>
</tr>
<tr>
<td></td>
<td>• C.A: Knowledge that a condition is something that can be true or false. (1)</td>
</tr>
<tr>
<td></td>
<td>• C.B: Knowledge that a conditional connects a condition to an outcome. (2)</td>
</tr>
<tr>
<td></td>
<td>• C.C: Ability to use conditional statements (such as if-then) to evaluate a condition to determine an outcome. (5U)</td>
</tr>
<tr>
<td></td>
<td><strong>Decomposition</strong></td>
</tr>
</tbody>
</table>
APPENDIX E
LEARNING TRAJECTORIES WITH INSTRUCTION MAPPING

SEQUENCE
- Advanced
1. Precise instructions are more likely to produce the intended outcome than general ones.
2. Different sets of instructions can produce the same outcome.
3. The order in which instructions are carried out can affect the outcome.
4. Computers have a default order of execution, so order matters in programming.
5. Creating working programs requires considering both appropriate commands and their order.
6. Some commands modify the default order of execution, altering when and which instructions are executed.
7. The position of a new command can affect outcomes.

REPETITION
- Intermediate
1. Repeating things can have a cumulative effect.
2. Some tasks involve repeating actions.
3. Instructions like “Step 3 times” do the same thing as “Step, step, step.”
4. Repetitions can go on forever or stop.
5. Computers use repeat commands.
6. Variables can be used to control the number of repetitions.
7. Different kinds of repetition have different commands.
8. Programs use conditions to end loops.
9. Loops can be nested to accomplish complex tasks.
## APPENDIX F
### KSA-ITEM MAPPING

<table>
<thead>
<tr>
<th>secondary reference to item</th>
<th>R.05.a</th>
<th>R.01.a</th>
<th>R.01.b</th>
<th>R.03.c</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>G3 Early Q08</td>
<td>G4 Mid Q09</td>
<td>G4 Mid Q04</td>
<td>Q08</td>
</tr>
<tr>
<td></td>
<td>app.</td>
<td>app.</td>
<td>app.</td>
<td>app.</td>
</tr>
</tbody>
</table>

R.A: Knowledge that creating computational artifacts can involve using repeated actions/instructions to accomplish an intended goal (2; 2U)

R.B: Ability to Use / Create / Modify a set of instructions that produce the same outcome by using two different approaches: (1) using repeat commands and (2) using the same commands multiple times (3; 3U)

R.C: Knowledge that repeat commands (e.g., repeat, repeat until, forever) tell a computer to repeat specific actions/instructions (5; 5U)

R.D: Knowledge that different kinds of tasks require different kinds of repeated instructions and therefore different repeat commands (1; 4.1; 4.1U; 5.1; 5.1U)

R.E: Ability to use repeat commands (forever, repeat until, repeat X times) to Use / Modify / Create instructions that create cumulative effects (1U; 5A)

<table>
<thead>
<tr>
<th>primary reference to item</th>
<th>S.04.b</th>
<th>S.01.a</th>
<th>S.08.a</th>
<th>S.04.a</th>
<th>S.10.a</th>
<th>S.02.a</th>
<th>S.12.b</th>
<th>S.04.d</th>
</tr>
</thead>
<tbody>
<tr>
<td>G3 Early Q07</td>
<td>X</td>
<td>X</td>
<td>--</td>
<td>X</td>
<td>--</td>
<td>--</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Q10 (G4 Early Q08)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G3 Mid Q02</td>
<td>X</td>
<td>X</td>
<td>--</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>G3 Mid Q08</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q09</td>
<td>X</td>
<td>X</td>
<td>--</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>--</td>
</tr>
<tr>
<td>Q10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G4 Early Q07</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S.A: Ability to create several different sets of instructions to produce the same intended goal (2U)

S.B: Ability to use / modify / create precise instructions to produce the intended goal (1U; 3.1)

S.C: Ability to use / modify / create an ordered set of instructions to produce the intended goal (3; 3A)

S.D: Ability to use a limited set of commands to produce the intended goal (3.2U)

S.E: Knowledge that computer programs [computational artifacts] require selecting from and ordering a limited set of commands. (4; 4U; 4.1; 4.1U; 5; 5U)
<table>
<thead>
<tr>
<th></th>
<th>C.06.a</th>
<th>C.01.a</th>
<th>C.02.e</th>
<th>C.03.b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>G4 Early Q04</td>
<td>Q05 (G4 Mid Q02)</td>
<td>G4 Mid Q05</td>
<td>Q06</td>
</tr>
<tr>
<td>C.A: Knowledge that a condition is something that can be true or false. (1)</td>
<td>--</td>
<td>--</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>C.B: Knowledge that a conditional connects a condition to an outcome. (2)</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>C.C: Ability to use conditional statements (such as if-then, if-then-else, and event handlers) to evaluate a condition to determine an outcome. (5U)</td>
<td>--</td>
<td>--</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>DC.03.b</th>
<th>DC.02.a</th>
<th>DC.06.c</th>
<th>DC.08.a</th>
<th>DC.08.b</th>
<th>DC.02.b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>G3 Early Q03 (G3 Mid Q06)</td>
<td>Q04</td>
<td>G3 Early Q05 (G3 Mid Q07)</td>
<td>Q06</td>
<td>G3 Mid Q04 (G4 Mid Q03)</td>
<td>Q05</td>
</tr>
<tr>
<td>DC.A: Knowledge that systems are made up of smaller, distinct parts (1; 1U)</td>
<td>--</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DC.B: Knowledge that a <strong>program</strong> is made up of different levels and components (5; 5U)</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>DC.C: Ability to break a complex problem into a set of simpler problems. (2; 2A; 2U)</td>
<td>--</td>
<td>X</td>
<td>X</td>
<td>--</td>
<td>--</td>
<td>X</td>
</tr>
<tr>
<td>DC.D: Ability to use problem decomposition as an early step in designing a solution. (3; 3U)</td>
<td>--</td>
<td>--</td>
<td>X</td>
<td>--</td>
<td>--</td>
<td>X</td>
</tr>
</tbody>
</table>
Understanding student's problem-solving in computational thinking-integrated math (IRB201900240)

1. Background:

Since its popularization by Janette Wing in 2006, computational thinking (CT) has gained a tremendous amount of attention. It has been acknowledged in the literature that computational tools can be used to improve math and science understanding (Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013; Weintrop, Beheshti, Horn, Orton, Jona, Trouille, & Wilensky, 2016). To effectively integrate computing in math and sciences, it is important that the integrated curriculum is designed in a coherent way and supports the development of students’ learning (Sengupta et al., 2013). A few studies explored different ways to integrate computing and computational thinking in K-8 mathematics curricula with specific learning objectives and investigated how such curricula affect math learning. In Calao, Moreno-León, Correa, & Robles’s (2015) study that introduced coding in math classes, it was found that students had significant increase in math skills. In a study that used Code.org to teach problem solving, students reported to have improved math knowledge and self-confidence (Kalelioğlu, 2015). However, little detail is revealed in terms of how students reason through and conceptualize computing tasks and what kind of barriers and challenges students face in integrated CT instruction. This study will further explore students’ experience in the context of elementary math by using CT-integrated fractions lessons and report students’ reasoning and problem-solving in CT assessment.
2. Specific Aims:

The purpose of this study is to examine students’ experience in the CT+fractions lessons, developed under an NSF grant, and understand students’ reasoning and problem-solving strategies using a think-aloud protocol. Class observations and screencast data will be collected to inform the following research questions:

1. How do students conceptualize computing tasks within paper and pencil assessment?
2. What are the barriers and challenges students encounter in learning computing through integrated CT+fractions instruction?
3. Research Plan / Study Description:

Overview: There are two parts of the study. The first part involves the teaching of the six CT+fractions lessons and the second part is the cognitive interviews with students using the think-aloud protocol.

Intervention: The study will happen at [school] during regular semester period. For this study, three PhD students will teach the CT+fractions lessons to three classes of 4th graders. As the primary instructor, each PhD student will teach one hourlong class per week for 6 weeks in the month of March and April. After the lessons, students may choose to participate in a cognitive interview where they are asked to solve two to three computing tasks in the assessment using the think-aloud protocol. Participants’ responses will be audio recorded.

We are not responsible for recruiting participants, but rather, we will work with those who are already in the class. Students may participate in the lessons without participating in this research. Parents of registered students will be provided an opt-out letter a week in advance and are requested to return the signed letter within two weeks if they wish to opt out.

CT-integrated fractions (CT+fractions) lessons: This study uses the “Action Fractions” lessons, an integrated math-computational thinking curriculum aligned to the CCSS-M, and it provides fractions instruction for 4th graders. Computational thinking concepts and practices such as decomposition, sequencing, variables are interwoven into the fractions lessons. The lessons include discussion prompts, reflections, unplugged activities, and hands-on coding exercises where students use the Scratch programming platform to build projects while building computational thinking skills. Students’ Chromebooks will be recorded using screen casting software to capture their problem-solving processes.

Assessment: Assessments used in this study were developed according to specific learning trajectories that include a set of questions that measure five CT constructs: decomposition, sequencing, repetition, variables, and conditional logic. The assessments involve True or False questions, multiple-choice items, word problems, and open-ended questions that ask for possible solutions.

Think-aloud protocol: 45 participants (representative sampling) will be selected to participate in the think-aloud tryouts of the assessment. This protocol first preps students by telling them that they will not be graded and that they can say what they wish to say. The researcher then models the think-aloud process by using a simple math problem. Each participant is then expected to work on two to three items using think-aloud. They will be prompted to talk out loud while working through the problems. The think-aloud interview will take place in the meeting space next to the 4th grade classroom, which is on the second floor of the elementary building at [school].

Rationale: Participants recruited for this study involve registered 4th-grade students at [school], students are automatically considered participants unless they are opted out by their parents or legal guardian. Since participation is voluntary, participant attrition is a potential internal validity threat because there is no mechanism in place to make sure that all participants remain present throughout the study.

In addition to [school], a second location for data collection is XXX Elementary School in XXX. This school is already piloting the same CT+fractions curriculum and the assessments (in grade 3 and 4) under a different UF IRB (IRB201802875). Therefore, this IRB will focus only on the collection and analysis of the think-aloud interview data at three time points: early, mid, and late stage of the 2019-2020 school year. Participants will be recruited from registered 3rd and 4th grade students at
XXX Elementary and those who will provide the consent and the assent. The data collection and analysis will follow the same procedures done at [name]. Data collection at XXX will be assisted by two non-UF researchers, Todd Lash, and Gakyung Jeong, who are approved under the IRB for the University of Illinois at Urbana-Champaign to conduct research-related activities relevant to this study. That IRB approval letter is uploaded to the Miscellaneous section.

**Data analyses**: All qualitative data collected (observations, screen captured data, and think-aloud interviews) will be coded and analyzed by the three PhD researchers. For RQ2, the Collaborative Computing Observation Instrument (CCOI) will be used to analyze screencast data.

**Instrument**: Participants' conversations and interactions among themselves and on the Chromebooks during computing tasks are screen-recorded and then analyzed using Israel et al. (2019)'s Collaborative Computing Observation Instrument (CCOI). This instrument has different nodes that categorize behavior, including how participants capture a peer's attention, the content of the problem that they are seeking help with, how they show excitement and curiosity, how they respond to a peer's problem, and how the problem is solved, etc. The CCOI allows to capture participants' time on tasks, ways of help-seeking and providing help to peers in need, how they individually and/or collaboratively deal with challenges and solve problems.
### 4. Possible Discomforts and Risks:

Since I will be working with students in a classroom setting, there should be minimal to no risk. Participants’ identification will be kept confidential and protected.

### 5. Possible Benefits:

Participants will learn computing skill and gain understanding of basic computational thinking concepts. This will be a pilot study of a potential larger scale dissertation study that will involve more participants. More students will benefit from this study that teaches computational thinking.

### 6. Conflict of Interest:

No COI is identified.
Dear Parent/Guardian,

I am Feiya Luo, a PhD student in the College of Education at the University of Florida conducting a research study “Understanding student's problem-solving in computational thinking-integrated math”. Under the supervision of my advisor, Dr. Maya Israel, a professor in the College of Education at UF, the purpose of this study is to explore how students express computational thinking understanding while working on assessment items. The study results will help us determine if it might be useful to use these lessons for developing computational thinking skills.

This study involves a 15-20-minute-long interview where an interviewer will ask a few questions as your child solves the computing tasks. Your child will not be required to continue if she or he does not wish to. The interview will be audio-recorded. Only the researchers on this project will have access to the recording for data analysis purposes and the data will be de-identified.

The activity outcomes and recordings will only be accessible to the researchers under the approved IRB. At the end of the study, the recordings will be deleted. All student identities will always be kept confidential to the extent provided by law. I will replace your child’s name with a pseudonym when reporting data. The actual name of your child or the name of the school will never be used in any public reports of the study. Participation or non-participation in this study will not affect your child’s grades or placement in any programs.

You and your child have the right to withdraw from participation at any time without consequence. There are no known risks or immediate benefits to the participants, and no compensation is offered. Study results will be available upon request.

I am requesting that you sign and return this form if you wish to have your child participate in this study. If you wish to have your child participate, please return the signed form within a week of receipt. If you have questions about this study, please contact me at [contact information] or [contact information]. Questions or concerns about student rights as a research participant may be directed to the IRB office, University of Florida, Gainesville, FL 32611.

I have read the procedure described above. I provide consent to have my child, __________________________, participate in Feiya Luo’s study of “Understanding student's problem-solving in computational thinking-integrated math”.

______________________________________________  __________________
Parent/Guardian  Date
Modification to Previously Signed Consent

Dear Parent/Guardian,

I am Feiya Luo, a PhD student in the College of Education at the University of Florida. Earlier this school year, you consented to my study, “Understanding student’s problem-solving in computational thinking-integrated math”. This is to inform you of the modification to the original signed consent:

- Conduct two follow up interviews to understand students’ computational thinking over the course of the school year. The interview format and procedure remain the same as the original one. Each interview will last about 15 to 20 minutes and will be audio-recorded.

Please sign below if you wish to have your child participate in the two follow-up interviews. Please kindly return the signed form within three days of receipt.

The data collected from your child’s interviews will be matched across all three interviews for the purpose of understanding their learning growth. Only the researchers on this project will have access to the recording for data analysis purposes and, at the end of the study, the data will be de-identified, and the recordings will be deleted. All student identities will always be kept confidential to the extent provided by law. I will replace your child’s name with a pseudonym when reporting data. The actual name of your child or the name of the school will never be used in any public reports of the study. Participation or non-participation in this study will not affect your child’s grades or placement in any programs.

If you have questions about this study, please contact me at [blank] or [blank]. Questions or concerns about student rights as a research participant may be directed to the IRB02 office, University of Florida, Gainesville, FL 32611. [blank].

I have read the procedure described above. I provide consent to have my child, ____________________________, participate in the two follow-up interviews.

_____________________________________________  __________________
Parent/Guardian                          Date
Hello,

My name is Feiya Luo and I am a doctorate student at the University of Florida. I am interested in your thinking in the coding activities and problem tasks. I will be asking you a few questions as you solve the tasks.

There are no known risks to participation, and this interview will take about 15 to 20 minutes. If you don’t like a task, you don’t have to do it. If you want, you can quit the interview at any time. I also want you to know that whatever you decide, this will not affect your grades in class.

Your parents said it would be OK for you to participate. If you want to participate, please sign this form.

Feiya Luo

I would like to participate in this study.

____________________________________________  __________________
Student                                         Date
Modification to Letter of Student OPT-IN

Hello,

My name is Feiya Luo and I am a doctorate student at the University of Florida. Earlier in the school year, you participated in my study, solving problem tasks. I would like to do two follow-up interviews with you this semester, and in each interview, I will be asking you a few questions as you solve problem tasks.

There are no known risks to participation, and each interview will take about 15 to 20 minutes. If you don’t like a task, you don’t have to do it. If you want, you can quit the interview at any time. I also want you to know that whatever you decide, this will not affect your grades in class.

Your parents said it would be OK for you to participate. If you want to participate in BOTH interviews, please sign this form.

Feiya Luo

I would like to participate in this study.

__________________________________________  __________________
Student                                      Date
Aisha has 8 toys that she wants to carry from the kitchen to her room. She can carry 1, 2, or 3 toys at once. One way to carry the toys is listed below. Write two other ways that Aisha can carry the 8 toys to her room.

Example:

- Carry 3 toys to room.
- Carry 2 toys to room.
- Carry 3 toys to room.

One Way:

Another Way:

Item S.04.b.
Create 2 different scripts (sets of instructions) to move the cat so that he stops at 5 on the number line. Use only the blocks shown above. Write or draw your scripts in the boxes.

Item S.01.a

Create 2 different scripts (sets of instructions) to move the cat so that he stops at 5 on the number line. Use only the blocks shown above. Write or draw your scripts in the boxes.

Item S.02.a
Deshaun’s basket has 8 apples and Erika’s basket has 2 apples.

**Instructions:**
Repeat 4 times:
- Take 1 apple from Deshaun’s basket and put it on the table.
- Take 1 apple from the table and put it in Erika’s basket.

If you follow these instructions, how many apples will be in Erika’s basket? _________

Item R.05.a

---

Andre has 9 cookies that he wants to give away to 3 friends. He wants to give each friend an equal number of whole cookies. Write instructions for giving the cookies to the 3 friends. Use the instruction “repeat 3 times” at least once.

Item R.01.a
Andre has 9 cookies to give away to his friends Sally, Val, and Lee. He wants to give each friend an equal number of cookies. Andre wrote instructions for how to give away the cookies.

Rewrite his instructions. Use the instruction “repeat 3 times” at least once.

**Andre’s instructions:**
- Give Sally 1 cookie
- Give Val 1 cookie
- Give Lee 1 cookie
- Give Sally 1 cookie
- Give Val 1 cookie
- Give Lee 1 cookie
- Give Sally 1 cookie
- Give Val 1 cookie
- Give Lee 1 cookie

**Your instructions:**

Item R.01.b

Describe what will happen when the green flag is clicked.

Item R.03.c
What sound (or sounds) will play if you run this code? ____________________________

If $5 < 8$, then play a "pop" sound.
If $5 > 7$, then play a "bing" sound.

Item C.02.e

If you run the code below, will the "pop" sound play if the user inputs 2? _________________

![Image of Scratch code]

Item C.03.b

Write one or more addition number sentences that mean the same as the multiplication number sentence:

$$5 \times 4 = 20$$

**Addition Number Sentences:**

Item DC.02.a
Pretend you want to find the area of the shape outlined in white.

This problem requires multiple steps. Break down the problem into steps. Describe your steps.

Item DC.06.c

Decomposing means breaking something down into parts. Decompose the number 10. List at least three ways you can add numbers to get to 10.

Item DC.08.a

List the steps you would use to solve this multi-step problem:

\[(5 \times 2) + (3 \times 2) = ?\]

Item DC.02.b
LIST OF REFERENCES


BIOGRAPHICAL SKETCH

Feiya Luo completed her PhD in educational technology. Prior to pursuing her PhD, Feiya received her master’s degree in translation and interpretation from the Middlebury Institute of International Studies at Monterey in California. A teacher with many years of teaching experience, Feiya's research interests involve computational thinking integration and CS education in the elementary level. Feiya believes in leveraging interactive learning technologies such as robotics to promote learners' interest in various subjects, such as science and math.