

12-10-2025

Elementary Teachers' Use of Sphero Robots to Teach Computational Thinking

Marquita Jackson
Walden University

Follow this and additional works at: <https://scholarworks.waldenu.edu/dissertations>



Part of the [Educational Technology Commons](#)

This Dissertation is brought to you for free and open access by the Walden Dissertations and Doctoral Studies Collection at ScholarWorks. It has been accepted for inclusion in Walden Dissertations and Doctoral Studies by an authorized administrator of ScholarWorks. For more information, please contact ScholarWorks@waldenu.edu.

Walden University

College of Education and Human Sciences

This is to certify that the doctoral dissertation by

Marquita Jackson

has been found to be complete and satisfactory in all respects,
and that any and all revisions required by
the review committee have been made.

Review Committee

Dr. Darci Harland, Committee Chairperson, Education Faculty

Dr. Heng-Yu Ku, Committee Member, Education Faculty

Chief Academic Officer and Provost
Sue Subocz, Ph.D.

Walden University
2025

Abstract

Elementary Teachers' Use of Sphero Robots to Teach Computational Thinking

by

Marquita Jackson

MPhil, Walden University, 2024

MA, Full Sail University, 2012

BS, Winston Salem State University, 2010

Dissertation Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy

Education

Walden University

February 2026

Abstract

The power of play has often been used in K–5 classrooms, but it has not always been implemented as a way to teach computational thinking skills. The use of educational robots such as Sphero have been explored, but not with young learners. The research problem addressed through this study was the lack of understanding of K–5 teachers’ experiences in using Sphero robots to teach computational thinking. The purpose of this basic qualitative study was to explore K-5 teachers’ experiences in implementing Sphero robot activities and specifically to teach computational thinking. The conceptual framework included Magana’s T3 model and Angeli’s computational thinking framework. Eight K–5 teachers recruited across the U.S. with different levels of experience implementing Sphero robots to teach computational thinking were interviewed. Multiple rounds of coding were conducted using thematic analysis. Study findings showed that teachers used technology in transformational ways to navigate technical challenges that leveraged curiosity with student autonomy, and used robots for guided play that led to algorithmic thinking. Teachers used robots to teach computational thinking skills such as debugging in environments where failure, reflection, and interaction were integral. This study may influence social change as stakeholders make decisions on ways robots might be used as educational tools to motivate young students and develop their computational thinking skills. Results may also be used to inform the role robots might have in early grades, and better support teachers to implement them more successfully, providing purposeful early exposure to computational thinking that may benefit more children through robots and play.

Elementary Teachers' Use of Sphero Robots to Teach Computational Thinking

by

Marquita Jackson

MPhil, Walden University, 2024

MA, Full Sail University, 2012

BS, Winston Salem State University, 2010

Dissertation Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy

Education

Walden University

February 2026

Dedication

I dedicate this achievement to my children, Laila and Tristan Jackson. I began my dissertation journey when Laila was just a week old. Thank you for being my motivation for excellence. I took a break from my dissertation when Tristan was born and resumed once he reached school age. Laila and Tristan, you walked every step of this journey with me. Your patience, love, and unshakable belief carried me through. This achievement is as much yours as it is mine. And remember, with hard work, dedication, and perseverance, you can accomplish anything your heart desires.

Secondly, I dedicate this achievement to my mother, Martina Clarke. I thank you for your support. Your days of taking the kids for the summer, visiting, cooking, and cleaning so I could work on my paper did not go unnoticed. Thank you for supporting me through every aspect of my life.

I'd also like to dedicate this achievement to my late grandmother, Dorothy Mae Webster. I lost you so long ago, and I still remember keeping you close to my heart during my undergraduate graduation as you fought pancreatic cancer. I thank you for guiding me through life as my very special guardian angel.

Finally, I want to express gratitude to my husband, Dr. Tremeze Jackson, who stood by my side throughout this journey. Your patience and generous support meant the world to me.

We did it!

Acknowledgments

I would like to acknowledge Dr. Darci J. Harland as my chair, counselor, and number one motivator during this dissertation process. I still remember meeting you during my first residence in 2014. I appreciate your effort in understanding my learning style and in applying those teaching methods to support my success. Dr. Harland, you have been the best part of my 11-year journey with Walden University.

I would also like to acknowledge my second member, Dr. Heng-Yu Ku, and Professor Dr. Howe. Dr. Ku, thank you for your patience and expertise throughout this journey. Dr. Howe, please know that I completed data analysis (with the guidance of Dr. Harland), and I hope I made you proud.

My final acknowledgement goes to my amazing village of family and friends. You all have been so patient, supportive, and encouraging through this journey, and I could not have done it without each and every one of you. Tanya, Theda, Bianca, Tracy, Cesily, Kina, CJ, Chad, Shaquandra, and Charleen, thank you for the laughter, letting me cry, and seeing me through the end of this journey. To my siblings, Mario and Jr. (De' Laria), my first sources of strength. From our rough and humble beginnings to this milestone, we have risen together. I hope this achievement honors our journey and makes you proud.

As this chapter concludes, I eagerly anticipate the adventures that life will unveil.

Table of Contents

List of Tables	v
List of Figures	vi
Chapter 1: Introduction to the Study.....	1
Background.....	2
Problem Statement.....	3
Purpose of the Study	4
Research Questions	4
Conceptual Framework of the Study	5
Nature of the Study	5
Definitions.....	7
Assumptions.....	8
Scope and Delimitations	8
Limitations	9
Significance.....	10
Summary	10
Chapter 2: Literature Review	12
Literature Search Strategy.....	13
Conceptual Framework.....	14
T3 Framework.....	14
Computational Thinking Framework.....	23
Coding and Computational Thinking.....	32

K–5 Teachers and Computational Thinking	38
Teachers’ Instructional Practices for Computational Thinking	39
Training in Computational Thinking	46
Teachers’ Perceptions of Experiences of Teaching Computational Thinking	53
Summary and Conclusions	59
Chapter 3: Research Method.....	63
Research Design and Rationale	63
Role of the Researcher	64
Methodology	65
Participant Selection Logic	65
Instrumentation	66
Procedures.....	67
Data Analysis Plan.....	71
Issues of Trustworthiness.....	76
Credibility	77
Transferability.....	77
Dependability	78
Confirmability.....	78
Ethical Procedures	80
Summary	81
Chapter 4: Results	82

Setting	83
Demographics	83
Data Collection	87
Data Analysis	88
Data Analysis for RQ1	89
Data Analysis for RQ2.....	98
Evidence of Trustworthiness.....	109
Results.....	109
Theme 1: From Tinkering to Transforming.....	109
Theme 2: Student-Centered Learning to Enhance Computational Thinking	
Skills	118
Theme 3: Planning Perfect Instructional Practices for Computational	
Thinking.....	123
Theme 4: Exploring Failure	142
Summary.....	154
Chapter 5: Discussion, Conclusions, and Recommendations	157
Interpretation of the Findings.....	158
Theme 1: From Tinkering to Transforming.....	158
Theme 2: Student-Centered Learning to Enhance Computational Thinking	
Skills	159
Theme 3: Planning Perfect Instructional Practices for Computational	
Thinking.....	161

Theme 4: Exploring Failure	163
Limitations of the Study.....	164
Recommendations.....	165
Implications.....	166
Conclusion	167
References.....	170
Appendix A: Interview Protocol.....	199
Appendix B: Codebook.....	207
Appendix C: Right to Publish Figures	211

List of Tables

Table 1. Search Terms	14
Table 2. Interview Questions Aligned to RQs	67
Table 3. A Priori Codes Aligned to T3 Framework.....	74
Table 4. Participant Demographics of Experience and Current Position	87

List of Figures

Figure 1. The T3 Framework	16
Figure 2. Elements of Computational Thinking	26
Figure 3. Codemap for Theme 1: From Tinkering to Transformational.....	90
Figure 4. Codemap for Theme 2: Student-centered Learning to Enhance Computational Thinking Skills.....	95
Figure 5. Codemap for Theme 3: Planning Perfect Instructional Practices for Computational Thinking	99
Figure 6. Codemap for Theme 4: Exploring Failure.....	105

Chapter 1: Introduction to the Study

Educational practices have evolved to include more technology use as technological advances have become more accessible to youths. Digital and technological experiences have surrounded children, yet schools rarely focused on exploring the technology world, especially in the early grades (Jurado et al., 2020). Many educational sectors used 21st-century skill sets such as coding and computational thinking (Wing, 2006). However, most primary schools did not instruct in coding or computational thinking despite the desire for K–6 computing education (code.org, 2019).

Although there are various definitions of computational thinking, academics provided several interpretations. Computational thinking is a method of understanding and solving problems using computer science concepts and techniques (Wing, 2006). Wing (2006) garnered global recognition for computational thinking and elucidated its role in equipping learners with essential abilities for problem solving. Selby (2014) characterized computational thinking as a subset of problem solving techniques and solutions that could be executed using computing equipment. Expanding on Selby's research, Angeli et al. (2016) devised a unique framework to describe the computational thinking skills in Grades K–6.

Computational thinking in the K–5 educational setting has often been associated with providing experiences for students in coding, either using mobile device applications (apps) such as Scratch (Kang & Lee, 2020) or types of robots such as Lego Mindstorm (Üşengül & Bahçeci, 2020) and Kibo (Jurado et al., 2020), but Sphero robots had not been studied as extensively with elementary students (Rath, 2015). Therefore, in the

current study, I explored K–5 teachers’ experiences of implementing Sphero robots to teach computational thinking. Findings may support public school administrators in assisting in-service educators with the professional development needed to offer support with using robotics to teach computational thinking.

The background, problem statement, study purpose, research questions (RQs), conceptual framework, and nature of study are covered in Chapter 1, along with a justification for the research design. I offer the operational meanings of pertinent terminology to remove any uncertainty in interpreting crucial ideas. Additionally, the chapter includes the research assumptions, limits, scope, and delimitations. I summarize the study’s main points, importance, and implications for constructive social change at the end of Chapter 1.

Background

As technology became more accessible to youths, educational practices evolved to include more technology use. One aspect of technology that was used as a teaching tool was robotics. Robotics was more frequently utilized in Grades 6 and older to promote computational thinking, and coding was also studied empirically (Chevalier et al., 2020; Fagerlund et al., 2020; Gunbatar & Turan, 2019), but research on elementary teachers’ implementation of robots for teaching computational thinking was less prevalent (Chalmers, 2018; Ensign, 2020; Kang & Lee 2020).

Previous researchers used numerous research methods to determine the value of teachers introducing computational thinking in educational contexts. Kale et al. (2018) emphasized the need to demonstrate to teachers how to apply computational thinking to

improve their classes and help the students understand the material. In one study conducted in the early elementary grades, Angeli et al. (2019) explored using a Blue Bot. A Blue Bot was a programmable, transparent, bee-like robot that students used directional coding to maneuver. Angeli et al. found that introducing the robot to the lesson changed the lesson from a traditional lesson, added rigor, and increased the computational skills needed to complete the task. Most robots for instructional practices used directional coding to operate the robot. However, Sphero is different from most robots used in the educational setting because Sphero robots use a block-style coding program (Fofang et al., 2020). The block-style program uses a mobile application that allows students to code from a mobile device, unlike the Lego robots where students used a desktop. The block coding used with Sphero robots is scaffolded well for young learners, allowing them to advance and increase rigor in mathematics as they are ready (Weintrop, 2021). To better understand teachers' experiences utilizing robots to teach computational thinking to K–5 learners, it was essential to look at the Sphero robots' utilization.

Problem Statement

The societal issue on which the current study was based was that teachers may lack the necessary training, resources, or support to integrate technology such as Sphero robots into their classrooms effectively (Barana et al., 2020; Butler & Leahy, 2021; Jocius et al., 2021; Kong et al., 2020; Mason & Rich, 2019; Mills et al., 2025). If teachers are ill-equipped or lack self-assurance in utilizing these tools, students may not wholly reap the advantages of acquiring computational thinking skills, thereby impeding the

cultivation of crucial problem solving abilities (Angeli et al., 2019; Jaipal-Jamani & Angeli, 2017; Noh & Lee, 2020). The research problem addressed in this study was the lack of understanding of K–5 teachers’ experiences in using Sphero robots to teach computational thinking. Although statistical evidence showed that teaching students to program robots improved their computational skills (Noh & Lee, 2020), there was limited knowledge about how K–5 teachers used these robots. Research indicated that learning to program enhanced elementary students’ computational thinking skills; however, the best methods for teaching computational thinking in the lower grades had been debated (Hunsaker, 2020). No study had been conducted on how in-service teachers used Sphero to teach computational thinking. The current study aimed to fill the gap by investigating K–5 teachers’ experiences using Sphero robots to teach computational thinking.

Purpose of the Study

The purpose of this basic qualitative study was to explore K–5 teachers’ experiences of implementing Sphero robots to teach computational thinking. Though computational thinking is a field of expanding importance, academic work on computational thinking has begun to emerge philosophically and experimentally (Angeli et al., 2020). Using data from 10 individual interviews with 10 K–5 elementary teachers who used the Sphero robots with their students, I learned about teachers’ experiences in teaching computational thinking to elementary students.

Research Questions

I developed two RQs based on the research problem and purpose, which guided this study:

RQ1: What are K–5 teachers’ experiences in implementing Sphero robot activities?

RQ2: How do K–5 teachers use Sphero robots to teach computational thinking?

Conceptual Framework of the Study

The conceptual framework I used for this study included two parts. The first was Magana’s (2017) T3 model. The T3 framework explains how innovations in technology, such as robotics, are applied in education. Magana described the three stages of technology use, including (a) translational technology use, (b) transformational technology use, and (c) transcendent technology use. I used the T3 model to answer the first RQ and describe teachers’ learning activities when teaching with robots. The second part of my conceptual framework was Angeli et al.’s (2016) computational thinking framework. This framework includes five skills or elements of computational thinking for what could be observed or assessed with K–6 students. The components of the framework are (a) abstraction, (b) generalization, (c) decomposition, (d) algorithms, and (e) debugging. To establish how teachers implemented these computational thinking components and used them in robot classes, I used the computational thinking framework to answer the second research question. More details about how the conceptual framework was used are provided in Chapter 2.

Nature of the Study

In this qualitative study, I applied the basic qualitative inquiry design to explore K–5 teachers’ experiences of implementing Sphero robots to teach computational thinking. Ravitch and Carl (2016) described qualitative research as involving systematic

and contextualized methods to interpret how humans view, approach, and make meaning of their experiences, contexts, and the world. “Generic qualitative inquiry or basic qualitative inquiry investigated people’s reports of their subjective opinions, attitudes, beliefs, or reflections on their experiences” (Percy et al., 2015, p. 78). My study fit this description because I explored K–5 teachers’ experiences implementing Sphero robots to teach computational thinking. The data collection method for this study was interviewing teacher participants. When interviewing teachers, I asked participants to share their experiences with lessons taught using robots. I also explored which computational thinking skills teachers encouraged students to practice. I used a semistructured approach for the interview process. Merriam (2002) explained that with a semistructured approach to interviewing, most questions are predetermined before the interview, and other questions develop as the interview happens. Ravitch and Carl (2016) described qualitative interviews as the center of many studies because qualitative interviews provide deep, rich, individualized, and contextualized data that are central to the research.

For the current study, I aimed to include five Grade K-2 teachers and five Grade 3-5 teachers who used Sphero robots to teach computational thinking skills. The one data source for this basic qualitative study was individual interviews. The participant selection process was purposive sampling, which meant identifying and selecting groups of knowledgeable or experienced individuals with a topic or interest. Ravitch and Carl (2016) explained that purposive sampling provides context-rich and detailed accounts of specific populations. I recruited teachers nationally using my professional learning network. I used my conceptual framework to develop the interview protocol. There were

specific inclusion criteria for teachers to be in the study. The participants had to (a) be currently employed as a K–5 teacher and (b) teach computational thinking using Sphero robots.

Definitions

In the study, I employed the following operational definitions:

Coding: The last stage of the software development process because it relates to writing code in a particular programming language (Piazza & Mengual Andrés, 2020).

Computational thinking: Solving problems, designing systems, and understanding human behavior by drawing on the concepts fundamental to computer science (Wing, 2006).

Programming: A technical matter closely linked to various computer technologies. From its inception, programming has been mostly a technical matter. A formal study of algorithms and their attributes became possible with the advent (and necessity) of computer-independent (“universal,” in the parlance of the time) programming languages, which enabled the expression of algorithms in a machine-neutral manner (Lodi & Martini, 2021).

Robot: A programmable and versatile manipulator designed for transporting goods, components, tools, or specialized systems (Latombe, 2012). A robot can perform various activities and execute predetermined movements to accomplish many objectives.

Robotics: The advancement of automation by enhancing the robot as already understood and encompassing the procedures related to the movement of the machine (Latombe, 2012).

Assumptions

I identified three assumptions for this research. The initial assumption was participants would answer the interview questions truthfully. Participant confidentiality was maintained, and the participants were considered volunteers. Participants were free to leave the study at any moment without consequence to foster an environment in which they could openly and honestly relate their personal experiences. The second assumption was that participants would report similar experiences of the phenomenon and that the study's inclusion criteria were suitable. Third, because participants were teachers, I also assumed that the responses of teachers would demonstrate deliberation and concern for the issue of teachers not being adequately trained in teaching computational thinking. Additionally, teachers would have had to have a genuine interest in participating in the study.

Scope and Delimitations

In this basic qualitative study, I explored K–5 teachers' experiences implementing Sphero robots to teach computational thinking. The research topic and the questioning strategy were created for a unique demographic of teachers who employed the Sphero robot during instructional time, and these young learners defined the scope of this study. Due to the scarcity of teacher training programs that offered robotic integration and computational practices at the preservice level, this study's delimitations included the underrepresentation of individuals in the target population. Due to my career as a science, technology, engineering, and math (STEM) teacher and my experiences with the topic, anyone from my school district was excluded from the study.

Limitations

There were various possible limitations including design and methodology limitations. The small sample size hampered generalizability. Being the sole interviewer presented its own bias. I was passionate about using educational robotics and was a former teacher of STEM, so I had to limit my interjections and stick to the interview questions. The standard procedure for qualitative research is for the researcher to minimize their biases and the potential negative impacts of assumptions (Patton, 2015). Patton (2015) suggested adopting a systematic interview guide to help guarantee uniformity in how questions are posed, decreasing the probability of leading or discriminatory questions. To counteract my bias, I engaged in reflexivity, which involved actively reflecting on and documenting my biases, preconceptions, and potential impacts during the study process (see Patton, 2015) and engaging a colleague or mentor to do a thorough examination of the interview questions and data analysis to identify any biases and verify interpretations. Also, I rigorously stuck to interview questions by formulating interview questions that refrained from straying into subjective viewpoints or criticism. I deliberately restricted interjections throughout interviews, enabling participants to articulate their experiences without being swayed by my viewpoints. I also participated in training sessions with a peer and dissertation chair to enhance my interviewing competency and cultivate techniques for preserving impartiality.

The significance of triangulation in my research resided in my plan to conduct interviews with teachers who taught students at various grade levels, ranging from kindergarten to fifth grade, and to report multiple perspectives to corroborate and

authenticate the results. Engaging participants in evaluating the precision of the transcriptions and interpretations of their interviews was essential to guarantee the accurate representation of their viewpoints. I documented the research process, especially any biases and limitations, to allow readers to assess the study's rigor and trustworthiness.

Significance

This study was significant because it contributed to the gap regarding how K–5 teachers used Sphero robots to teach computational thinking to K–5 students. Results from this study may enhance the existing knowledge in educational technology. The findings of this study may be used to assist stakeholders in making informed assessments on how robots are used to provide opportunities for students to practice problem solving skills and computational thinking skills. The findings of this study may lead to societal transformation as stakeholders become better educated on strategies to enhance their support for teachers in integrating robots into their lessons of computational thinking. In addition, the acquired knowledge could aid districts in making informed decisions regarding the investment in robots for classrooms and enhancing students' educational experiences. Finally, the findings of the study may enhance educational technology by clarifying how teachers used robots to foster computational thinking in K–5 children.

Summary

In Chapter 1, I provided the contextual framework, problem statement, and objective of the research, which pertained to the imperative of understanding how K–5 teachers used Sphero robots to teach computational thinking. Furthermore, I identified

the primary RQs regarding the experiences of K–5 teachers in implementing Sphero robot activities and the use of Sphero robots by K–5 teachers to teach computational thinking. In addition, I presented the theoretical framework of computational thinking and clarified its role in supporting the research. Furthermore, the study’s nature and the definitions that served as the foundation for the literature review and RQs were explained. Lastly, I elaborated on the assumptions, scope and limitations, and significance of the study. In Chapter 2, I review the literature pertaining to how teachers use Sphero robots to teach computational thinking, and the conceptual framework pertaining to how teachers use educational robots to teach computational thinking.

Chapter 2: Literature Review

As the education system continues to change for the benefit of students, computational thinking education has gained popularity in K–12 classrooms because of its impact on children’s development of critical thinking and digital skills (Angeli & Giannakos, 2020). Integrating educational robots and block-style programming exercises into collaborative learning spaces has encouraged the growth of computational thinking, critical thinking, social skills, and problem solving skills (Silva et al., 2021). Problem-based learning could provide students with the cognitive skills of computational thinking essential for comprehending phenomena in their environment and solving issues using computational methods (Shin et al., 2021). Effective teacher development programs are vital for cultivating the necessary subject expertise, pedagogical skills, and self-assurance that teachers require to teach computational thinking in the context of programming effectively (Kale et al., 2018; Rich et al., 2020). With limited university programs and professional development opportunities focusing on computational thinking, teachers are often left to discover ways to teach computational thinking independently. Providing teachers with the support they need could ensure that students receive the best possible education and are equipped with the skills they need to succeed in the digital age.

The research problem addressed in the current study was the lack of understanding of K–5 teachers’ experiences in using Sphero robots to teach computational thinking. The purpose of this qualitative study was to explore K–5 teachers’ experiences of implementing Sphero robots to teach computational thinking. The primary RQs addressed K–5 teachers’ experiences in implementing Sphero robot

activities and K–5 teachers’ use of Sphero robots to teach computational thinking. In this chapter, I review the search strategies, conceptual framework, recent literature relevant to the topic, and framework on computational thinking along with a technology model. I conclude by describing what was known and unknown about this topic.

Literature Search Strategy

The literature review method involved utilizing articles and materials sourced from the Walden Library databases. The databases that were used the most were ERIC and Education Source. Academic Search Complete, Sage, and Taylor and Francis yielded comparable outcomes. In addition, Google Scholar was used to broaden the scope of research.

In pursuit of my research objective, I restricted my selections to peer-reviewed materials published from 2018, when I commenced my research. I primarily focused on studies conducted between 2021 and 2025 because the knowledge gained in education over the past few years yielded more significant results concerning the application of robotics in computational thinking instruction. Computational thinking, application of robotics, coding, programming, and elementary school educators were among the searches conducted. Furthermore, I searched using keywords about professional development, training, and teacher perceptions (see Table 1).

Table 1*Search Terms*

Topic idea	Search term
Computational thinking	(computational thinking or algorithmic thinking or programming or computer science) and (mathematics didactics or mathematics or math) and (“secondary school” or “primary school” or “elementary school”)
K–5	K–5 or elementary or primary early education
Elementary school	elementary school or primary school or grade school or elementary education or elementary students
Coding	coding or programming or computer science or computing
Programming	programming or coding or computer science
Robots	Robotics
Elementary teacher	elementary teachers or elementary educators or elementary instructors
Teacher perceptions	teacher perceptions or teacher attitudes or teacher views or teacher perspectives
Training	training or education or development or learning
Professional development	professional development or professional learning or professional training or professional education

Conceptual Framework

This study’s conceptual framework included Magana’s (2017) T3 framework and Angeli et al.’s (2016) computational thinking framework.

T3 Framework*History of T3 Framework*

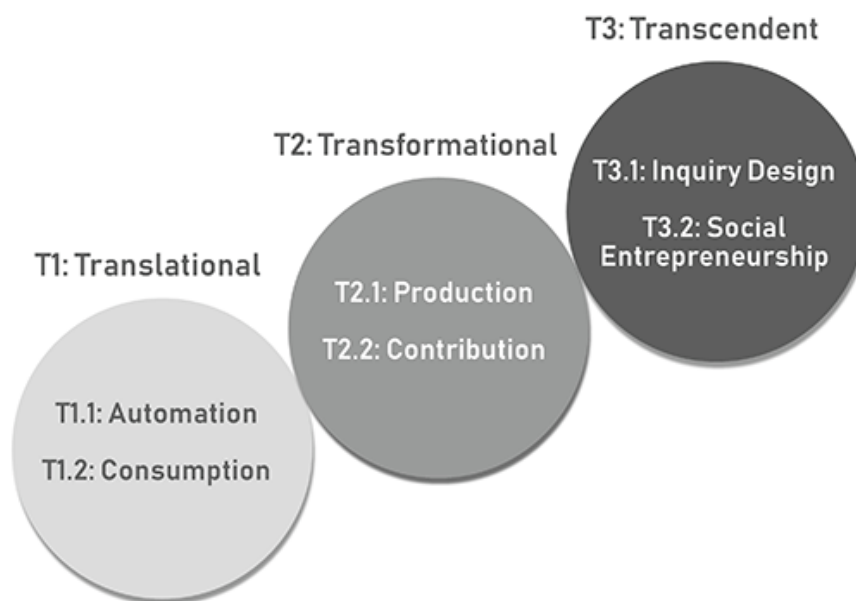
Magana (2017) developed the T3 framework model after examining two current technology integration frameworks: the technological pedagogical content knowledge (TPACK) model and the substitution, augmentation, modification, and redefinition

(SAMR) model. The TPACK model emphasized the significance of teachers' knowledge of the technology of content knowledge and pedagogical knowledge in the classroom setting. Magana found that the TPACK model lacked clarity on the steps to develop an understanding of technology. The TPACK model showed the importance of a teacher having the technical knowledge to implement technology in the classroom. However, Magana felt that the TPACK model lacked guidance or standards for self-assessment for teachers. The lack of standards made it difficult for administrators or teachers to use it as a tool related to technology usage in the classroom (Magana, 2017).

Alternatively, the SAMR model built from the TPACK Model but was more of a scale that categorized technology usage (Magana, 2017). SAMR, which stood for substitution, augmentation, modification, and redefinition, was better used to describe technology usage as far as the pedagogy but lacked the resources for technological implementation in actual instructional practices (Magana, 2017). Therefore, Magana (2017) created the T3 framework to improve these previous models and better described the process of integrating technology in teaching and learning environments. The purpose of the T3 framework was to provide a pathway grounded in research and theory and promote students' acceleration through technology (Magana, 2017).

Constructs of T3 Framework

Magana's (2017) T3 framework included three stages of educational technology usage: translational, transformational, and transcendent (see Figure 1).

Figure 1*The T3 Framework*

Note. From *Disruptive Classroom Technologies: A Framework for Innovation in Education*, by S. Magana, 2017, Corwin. Reprinted with permission (see Appendix C).

T1: Translational Technology Use. The first stage of the T3 framework was translational. Magana (2017) defined translational as “one was translating tasks or experiences from an analog modality to a digital modality” (p. 28). An example of translational technology would have been a teacher having students complete paper-pencil assignments digitally. Technology use in the translational domain aided in applying educational technologies to administrative, instructional, or learning tasks. Translating tasks by digitization added value by saving time, accumulating efficiency, and enhancing accuracy (Magana, 2017). It was important to note that translational technology did not change the learner, or the task learned (Magana, 2017). The translational domain had two steps: automation and consumption (Magana, 2017).

Automation was the first step to adding value to technology usage (Magana, 2017). Automation was when either teacher or learner used technology to automate or add value to instructional or learning tasks (Magana, 2017). Automation technology saved time, helped reduce task-related errors, increased the number of assignments completed, and may have improved the quality of a finished product (p. 29). In my exploration of teachers' use of Sphero robots, automation technology might have included having students use the essential function of driving robots related to a lesson. The use of the driving feature would have resulted in fewer task-related errors. The use of the driving component of Sphero would have increased the number of tasks completed in a certain amount of time.

Consumption was the next step of the translational stage: accessing some digital form of content-related information (Magana, 2017). Consumption described teachers and students consuming media in a digital realm (Magana, 2017). Magana (2017) depicted consumption as the ability to consume information in the digital medium. Consumption of content-related information was, to some degree, more valuable than use in the automation step. Magana (2017) explained that with the automatic use of digital tools, information could be consumed and simultaneously stimulate learners. The consumption process had been positively disrupted, while students traditionally received instruction directly from the teacher (Magana, 2017). In my study, consumption technology included using Sphero as a digital tool to consume interactive content. Using Sphero during lessons could also have saved time with fewer teacher-led lessons.

Providing the students the opportunity to learn without teacher direction increased exploration in learning.

T2: Transformational Technology. The next stage in the T3 framework was Transformational. Magana (2017) described the stage of transformational technology as when students excelled in their learning in a way that would not have occurred without using technology. A significant component of transformational technology in education was creating a student-centered learning atmosphere (Magana, 2017). With the help of technology, students could express their academic abilities and content knowledge (Magana, 2017). Transformational learning was student-centered using educational technology (Magana, 2017). Transformational learning allowed students to have a singular view of learning no longer but to create shifts in mindsets of preconceived notions before learning (Magana, 2017). The two steps of the transformational stage included production and contribution (Magana, 2017).

The first step of transformational was production and was defined as how students used their digital knowledge to produce a digital artifact to show what was learned (Magana, 2017). Students' production components included authentic evidence of growth and mastery with digital tools, the quality of work produced with technology, and the method students used to create the work. Within production, Magana (2017) provided three production strategies for students in the transformational phase of the framework, including (a). students producing mastery goals, (b). students tracking their growth and mastery, and (c). students creating and archiving knowledge and thought artifacts.

Magana (2017) provided three value indicators for whether the technology was used in production. With the first strategy, students producing mastery goals gave them a specific learning standard that may have connected with many learning dimensions (Magana, 2017). The second strategy was used to motivate students by having them track their growth and mastery, building their self-awareness and self-determination (Magana, 2017). The third strategy focused on the students generating their own formative assessments to show mastery and transition from the teacher-generated formative assessments (Magana, 2017). In examining teachers' experiences implementing Sphero robots, production technology might have included students using the Sphero robot to explain content learned in the classroom digitally. For example, students could program the robot to follow the life cycle of the food chain.

The second step in transformative technology use was the contribution, defined as the student's ability to create digital content knowledge that contributed to the learning of other students (Magana, 2017). Magana (2017) revealed that to achieve the contribution stage of transformational technology; teachers had to transform the learning environment from a competitive viewpoint to a learning community where students worked together to build tasks inclusively. By transforming the learning environment, students learned how to put themselves in another person's frame of mind in lieu of singular thinking as it pertained to learning. Within this learning method, students learned empathy and consideration of how other students learned and processed/interpreted information (Magana, 2017). In my study, contribution technology use included students using the Sphero robot to create tutorials to teach other students how to use the robot for coding.

Three strategies supported the contribution of transformational technology, which included (a) students contributing to and tracking their observance of classroom commitments, (b) students producing authentic tutorials designed to contribute to others' learning, and (c) students using digital tools to curate their authentic learning tools (Magana, 2017).

Students' contributions to and tracking of their observance of classroom commitments helped establish the development of norms and expectations of the learning environment (Magana, 2017). Students developed an ownership mentality of the learning environment by contributing and tracking their observance. Adding in the component of students producing authentic tutorials to contribute to others' learning increased the level of contribution and the sense of community (Magana, 2017). Creating tutorials also shifted students from passive learners to active knowledge producers (Magana, 2017). Students' use of digital tools to create authentic learning tools allowed them endless opportunities to create digital portfolios that showed the progression of content knowledge over time. Some examples of digital tools students could use included but were not limited to blogs or self-created websites.

T3: Transcendent Technology Use. Magana (2017) described transcendent technology use as teaching and learning that pushed the scope of what was known to reach new performance levels. Magana (2017) explained that student passion was critical to transcendent technology. There were two steps within transcendent technology: inquiry design and social entrepreneurship (Magana, 2017). The purpose of the two stages was to promote student use of technology to contribute something of value to the

benefit of society (Magana, 2017). Transcendent use of technology encouraged students to create solutions to problems they were passionate about (Magana, 2017). The transcendent use of technology held a precedent for students' success in building digital platforms for their learning capabilities but also in supporting the learning of others (Magana, 2017).

The first step of transcendent technology was inquiry design. Inquiry design was essential to how students investigated significant problems and found solutions for those problems (Magana, 2017). Students were at the center of learning and their academic development (Magana, 2017). Transcendent technology use had three supporting strategies. The three supporting strategies were (a) students using technology to investigate a wicked real-life problem that mattered to them, (b) students using technology to design original inquiries and generate resolutions, and (c) students communicating, defending, and iterating their unique knowledge contributions. Each strategy assisted students with deeper learning. Students used technology to design their inquiries and generate resolutions focused on researching and solving problems on topics they were passionate about (Magana, 2017). Students communicated, defended, and iterated their unique knowledge contributions by having students communicate their solutions to stakeholders as they responded to questions about the students' investigations (Magana, 2017). In my study, transcendent technology use included students using the coding skills they learned using Sphero robots, likely combining multiple content areas.

The second step of transcendent technology use was social entrepreneurship, when students' roles as creative thinkers allowed the learners to own their learning

(Magana, 2017). Social entrepreneurship stimulated learning experiences that were powerful and meaningful. Students imagined, designed, and created new tools or platforms to solve wicked problems that mattered, which consisted of students imagining, designing, and creating new tools or platforms (Magana, 2017). Moreover, students being able to create solutions to problems that mattered to them and use their technology tools to collaborate and reflect was the process that social entrepreneur motivated. Students' beta tested, iterated, and generated robust versions of digital solutions demonstrated by students who built digital content and tested the impact by gathering feedback; once the feedback was given, students used the information to improve their digital content (Magana, 2017). Furthermore, students could scale the implementation of their robust digital solutions and support students by having them engage in sharing solutions that gathered feedback and informed and scaled their solutions (Magana, 2017). In my study, social entrepreneurship included students using the coding skills they learned using Sphero robots to find solutions to real-life problems they identified and sharing those solutions with stakeholders within the community.

Justification for the Use of the T3 Framework

The T3 framework was a relatively new framework often cited in the literature regarding how technology was used. Some critics suggested that the stages of the T3 framework had not been proven to improve student outcomes (Brown, 2020). However, Magaña created the T3 framework to guide teachers in monitoring and improving their technology use and for administrators to evaluate technology use. Researchers could have

used the framework to produce empirical research to determine differences between the various stages of technology use and its impact on the classroom.

The T3 framework is a good choice for this study because it fits its purpose and methodological design well. The T3 framework aligns with my research by examining teachers' use of Sphero robots in relation to the activities used to teach computational thinking. The three stages of the T3 framework add learning levels and serve as a guide to understanding how teachers choose to use Sphero robotics for instructional purposes. In this research, I aim to explore K–5 teachers' experiences of implementing Sphero robots to teach computational thinking. Each stage is a building block to successfully integrating technology in the classroom. Understanding each stage adds new elements to describe a student-centered way of learning to improve the use of robotics in education. I used the T3 model to answer RQ1 and describe teachers' experiences when teaching with robots. I used the T3 model to develop a priori codes, which will be further discussed in Chapter 3.

Computational Thinking Framework

History of Computational Thinking

The history of computational thinking dated back to Papert and Harel (1991), who discussed the early workings of computational thinking. Papert and Harel challenged students' ideas of learning differently while still obtaining the skills needed to grasp the concepts. Papert and Harel discussed the difference between instructionism and constructionism, which differed in teaching strategies and how students learned. Instructionism focused on propositional knowledge, whereas constructionism focused

more on concrete knowledge (Papert & Harel, 1991). The presence of computers aided in the concept of computer literacy and the formation of computational thinking (Papert & Harel, 1991). Constructionism, however, suggested that learning was understood through being constructed (Papert & Harel, 1991).

Wing (2006) brought worldwide attention to computational thinking and explained that computational thinking added fundamental skills learners needed to solve problems. Many researchers after Wing contributed to understanding the elements that made up computational thinking. For example, Selby (2014) researched how teaching programming could increase computational thinking skills. Selby studied programming pedagogy, taxonomy, and the difficulties of learning programming. Building from Selby's work, Angeli et al. (2016) developed a distinct framework to describe further what computational thinking skills looked like in grades K–6.

Selby (2014) described computational thinking as a broader topic of cognitive processes. Problem solving reinforced computational thinking or similar aspects of computational thinking (Selby, 2014). Computational thinking took strategic planning to multiple domains, such as engineering and science (Selby, 2014). Selby (2014) researched Bloom's taxonomy and elements of computational thinking to construct a theory of computational thinking taxonomy that focused on the relationship between cognitive processes, programming pedagogy, and levels of computational thinking skills.

Selby's (2014) findings identified a connection between computational thinking and the analysis, synthesis, and evaluation levels of Bloom's taxonomy. The study participants included 143 respondents, consisting of teachers, academics, and industry

professionals (Selby, 2014). According to Selby, computational thinking was distinguished as a subset of problem solving strategies and solutions that could be implemented with computing devices. Selby (2014) used her research to highlight the connections between computational thinking and computer science as it related to education.

Another framework that focused on the use of computational thinking in education was the International Society for Technology in Education (ISTE). This organization listed core components of computational thinking and created competencies geared toward educators. The core components included decomposition, gathering and analyzing data, abstraction, algorithm design, and how computing impacted people and society. The ISTE computational thinking competencies differed in several ways from the framework of Angeli et al. (2016). For instance, although both used the term “abstraction,” the definitions were slightly different. Angeli et al. defined abstraction as a decision about what information was needed and what information was not required. In contrast, ISTE defined abstraction as the process of reducing complexity by focusing on the main idea. ISTE’s computational thinking had core components different from those of Angeli et al. ISTE’s computational thinking included gathering and analyzing data, which was defined as the process of collecting and storing data in a way that was easy for the computer to understand and served as a method to make patterns or predictions. However, Angeli et al.’s framework focused on students’ behaviors and skills. In contrast, ISTE computational thinking was geared more toward assisting educators in planning and creating computational thinking activities for students.

Constructs of Computational Thinking Framework

In Angeli et al.'s (2016) framework, there are five skills of computational thinking: (a) abstraction, (b) generalization, (c) decomposition, (d) algorithms, and (e) debugging. Angeli et al. organized these skills according to grade levels: K-2 (ages 6-8) or early elementary, 3-4 (ages 9-10) or upper elementary, and 5-6 (ages 11-12) or intermediate elementary (Angeli et al., 2016). Focusing on developing young children's computational thinking through scaffolded interactions, Angeli and Valanides (2020) used the five skills to further research the effects of integrating computational thinking into education, specifically focusing on (a) abstraction, (b) generalization, (c) decomposition, (d) algorithms, and (e) debugging (see Figure 2).

Figure 2

Elements of Computational Thinking

Element	Definition
1. Abstraction	The skill to decide what information about an entity/object to keep and what to ignore (Wing, 2011).
2. Generalization	The skill to formulate a solution in generic terms so that it can be applied to different problems (Selby, 2014).
3. Decomposition	The skill to break a complex problem into smaller parts that are easier to understand and solve (National Research Council, 2010; Wing, 2011).
4. Algorithms	The skill to devise a step-by-step set of operations/actions of how to go about solving a problem (Selby, 2014).
a. Sequencing	The skill to put actions in the correct sequence (Selby, 2014).
b. Flow of control	The order in which instructions/actions are executed (Selby, 2014).
5. Debugging	The skill to identify, remove, and fix errors (Selby, 2014).

Note. From “A K–6 Computational Thinking Curriculum Framework: Implications for Teacher Knowledge” by C. Angeli, J. Voogt, A. Fluck, M. Webb, M. Cox, J. Malyn-Smith, and J. Zagami, 2016, *Educational Technology & Society*, 19(3), 47–57. Reprinted with permission (see Appendix C).

Abstraction. The first construct of Angeli et al.'s (2016) computational thinking framework was an abstraction. Abstraction was the skill of removing characteristics or attributes from an object or an entity to reduce it to a set of fundamental traits (Wing, 2011, as cited in Angeli et al., 2016). Abstraction was when students could simplify a problem by hiding details that were not relevant (ISTE, 2016). Students in the early elementary category might have practiced abstraction when they used directional language to solve problems, created story maps using robotics, or created and wrote coding paths (Angeli et al., 2016). Upper elementary students were able to dive more into the element of creativity to solve problems by creating digital elements or digital games (Angeli et al., 2016). Intermediate elementary students used abstraction to create a new model or representation to solve problems (Angeli et al., 2016). Creating new models or representations could have been conceptual, mathematical, mechanical, textual, or graphical. With more research, Angeli and Valanides (2020) suggested that more research be done to study the developmental perspective of abstraction. Teachers encouraging students in the skill of abstraction might have probed students with reminders such as, "What information do you think you needed to solve this problem? Or "Is there any information here you might not need?" An example of abstraction related to my study would be when teachers encouraged students to decide what information to keep or ignore when using Sphero robots.

Generalization. The second computational thinking skill was a generalization. Generalization was defined as the ability to simplify a solution to apply it to a different problem (Angeli et al., 2016). Angeli et al. (2016) stated that abstraction and

generalization were often completed to provide greater utility. Generalization in early elementary included students building upon older coding paths to create new ones (Angeli et al., 2016). Upper elementary generalization merely extended or remixed what had been completed (Angeli et al., 2016). Generalization in intermediate elementary comprised the extension of the creation to increase complexity (Angeli et al., 2016). Teachers encouraging students to generalize might ask probing questions such as, “Was there anything you learned last time that might help you solve this problem?” When teachers described Sphero lessons where students practiced generalization, it included a progression in lessons where students could take more minor codes and build them up to make more significant, fully functioning codes.

Decomposition. The third computational thinking skill was decomposition. Angeli et al. (2016) defined decomposition as the skill of breaking complex problems into simpler ones. Decomposition allowed students to simplify problems into smaller components to make them easier to understand and solve (ISTE, 2018). Early elementary students learning the skill of decomposition tested students’ aptitude to break down larger tasks into smaller, more manageable tasks (Angeli et al., 2016). Applying decomposition, upper and intermediate elementary students created a solution for the task broken into small subtasks (Angeli et al., 2016). Angeli and Valanides (2020) found that with the help of scaffolded tasks, even young students could decompose complex problems.

Additionally, teachers who encouraged students to learn the skill of decomposition might have asked students, “How might you break this problem down into smaller pieces?” or “What was the first task you needed to do to complete this

challenge?” Decomposition applied to my study when teachers constructed Sphero lessons. Students were given opportunities to use coding to solve more significant problems or challenges that they needed to break down into smaller pieces.

Algorithms. The fourth computational thinking skill was algorithmic thinking (Angeli et al., 2016). Angeli et al. (2016) defined algorithms as the skill of devising a step-by-step action for solving problems. Algorithms consisted of sequencing and control flow (Angeli et al., 2016). Sequencing was defined as the talent of putting actions into the correct sequence, and the flow of control was defined as the order in which instructions or steps in an algorithm were evaluated (Angeli et al., 2016; Angeli & Valanides, 2020). Early elementary algorithmic thinking was when students understood the importance of steps in action and could place instructions in the correct order (Angeli et al., 2016).

For example, upper elementary students used algorithmic thinking to define steps and establish instructions. However, adding an iteration or repeating those steps required further learning (Angeli et al., 2016). In contrast, intermediate students employed algorithmic thinking to make decisions based on conditions, add variables, and articulate mathematical and logical expressions (Angeli et al., 2016).

Furthermore, teachers played a crucial role in encouraging debugging. The final element of computational thinking was debugging. Angeli et al. (2016) defined debugging as the skill of identifying, removing, and fixing problems. Notably, students at all grade levels had to recognize when instructions did not correlate with the desired actions and subsequently be able to act and correct errors (Angeli et al., 2016; Angeli & Valandines, 2020). In this regard, teachers who encouraged students in debugging skills

might have provided them with a code and asked the students to find and fix the problem on their own. Thus, a teacher's support in this process was essential for developing students' computational thinking abilities.

Debugging. The final element of computational thinking was debugging. Angeli et al. (2016) defined debugging as the skill of identifying, removing, and fixing problems. Students at all grade levels needed to recognize when instructions did not correlate with the desired actions and then be able to act and correct errors (Angeli et al., 2016; Angeli & Valandines, 2020). Teachers encouraging students in debugging skills might have provided students with a code and had the students find and fix the problem on their own. A teacher speaking to a student about debugging might have asked a student, "Where did the code not work in the process? or "How could you fix the problem in the code?" Debugging applied to my study when students had to work through their Sphero challenges, identify when results were not correct, and be able to edit the code to fix problems.

Justification for the Use of Computational Thinking Framework

Several computational frameworks could have supported my study. Still, the Angeli et al. (2016) computational framework had qualities that best helped my exploration of K-5 teachers' experiences implementing Sphero robots to teach computational thinking. One computational thinking model I could have used was Ling-Ling et al. (2021), who studied a revamped computational thinking framework. With five different core components from those of Angeli et al., Ling-Ling et al. (2021) took a more holistic approach to students' learning. The goal of the holistic computational

framework was to focus more on knowledge generation by teachers in lieu of knowledge acquisition of students, which did not fit my study. ISTE (2016) also had identified core computational thinking skills as part of their computer science competencies. Still, those core components were not a good fit for my study because the ISTE framework focused more on the teacher's outcomes of data and professional development versus the actual instructional practices that would be implemented for the students. Angeli et al.'s computational thinking model outlined skills for each learner in grades K–6. Previous research applied the Angeli et al. framework to early elementary robot activities, making it a natural fit for my study (see Angeli & Valanides, 2020). Angeli et al.'s (2016) computational framework included five computational thinking skills that teachers should have encouraged young learners during learning activities. Because Angeli et al. clearly explained what computational thinking skills looked like at the various stages (early, upper, and intermediate elementary), it provided a precedent for studying these skills with this population. Since my research focused on grades K–5 and robotics, the Angeli et al. (2016) model was the best choice for my study.

The computational thinking framework (Angeli et al., 2016) was particularly relevant to my study as it emphasized the integration of computer science into K–5 classrooms and provided me as the researcher with practical examples of the skills included in computational thinking activities. Understanding each element gave me the knowledge needed to interpret teachers' lessons and to answer RQ2, about how K–5 teachers used Sphero robots to teach computational thinking. The framework thoroughly dissected each element of computational thinking, offered valuable insights to explore

and better comprehend what computational thinking looked like in the classroom with students of this age. In my study, I developed interview questions aligned to the various elements of Angeli et al.'s (2016) framework to target specific computational thinking skills. By utilizing this framework as a foundational guide, I hoped to effectively explore and understand the ways in which educators incorporated computational thinking through Sphero implementation.

Coding and Computational Thinking

Although computational thinking was broadly defined as “thought processes of formulating and solving problems with the use of computers” (Angeli et al., 2016, p. 47), much of the research on computational thinking was more narrowly focused on coding. It was essential first to define coding before understanding its placement in education. Coding was defined as the languages used to enable computing (Monteiro et al., 2021). Coding allows teachers to enhance learners’ difficulty levels based on their level of understanding (Cervera et al., 2020). Teachers often taught computational thinking using robotics and had students learn coding and programming (Stewart et al., 2021; Mills et al., 2025). Teachers’ use of computational thinking and technology increased the importance of discussing different languages of coding and coding literacy. However, there was a difference between coding and programming. Oddie et al. (2010) defined programming as a solution to an understood problem that was analyzed, and then an algorithm of the solution was translated into a code. Visual programming, such as block-style programming, was often used to teach coding in primary schools (Sáez-López et al., 2019).

Scratch and ScratchJr were programs often used to teach students coding.

Research on computational thinking with elementary-aged students frequently focused on students' use of programs such as Scratch and ScratchJr (Kılıç, 2022). Scratch was an educational programming platform developed at MIT to make programming simpler for young students to learn by using a block-style coding language where students could drag and drop commands into a sequence of codes to produce an action of a character (Yukselturk & Altıok, 2017). ScratchJr was developed and defined as a simplified version of Scratch, later created, and geared toward students ages 5-7 to encourage younger students to learn programming (Strawhacker et al., 2018). Chou (2020) examined computational thinking and the use of ScratchJr in a mixed-method study of an experiment-based computer class of 12 third-grade students. Findings indicated that students significantly increased computational thinking skills with an 8-week program. ScratchJr showed effectiveness in helping students to learn sequence, events, debugging, and parallelism, but not loop concepts.

Likewise, Pérez-Marín et al. (2020) found similar connections to computational thinking when students used Scratch in their quasi-experimental design study, exploring the use of metaphors to teach 132 students ranging in age of 9-12. The unique part about using metaphors was the connection between input and output as it related to the human body to teach students computational thinking using Scratch. One significant result showed student growth from pre-test to post-test regarding content knowledge and computational thinking skills. The study demonstrated the effect of using a small amount

of time to increase students' skills in computer programming concepts like memory, programming, conditionals or loops, and computational thinking.

While the Pérez-Marín et al. study reported success in learning computational thinking via Scratch to teach science and language arts content, Rodríguez-Martínez et al. (2020) examined a similar population, but for math and found increases in students' computational thinking. Results from the quasi-experimental design indicated that 47 sixth-grade students' computational abilities were scarce in the beginning, suggesting a solution to this scarcity was teaching computational thinking in elementary education to advance students' learning (Rodríguez-Martínez et al., 2020). Rodríguez-Martínez et al. reported a significant increase in computational thinking just after five sessions of the program, which resulted in greater success at the end of the program. All three studies demonstrated growth using the Scratch program but with differing approaches and age groups.

There were other ways that students could learn computational thinking and coding that did not involve Scratch and ScratchJr. Leonard et al. (2021) taught coding by integrating choreography using a virtual platform that allowed 15 fifth-grade students to physically dance and convert the dance into a code to make an avatar mimic their movements. Scratch and other programming environments inspired the virtual platform Virtual Environment Interactions. Results from the mixed methods study showed that students successfully comprehended and utilized computational thinking concepts by developing codes to convert physical dance moves into digital dance moves. The results also showed that students were able to grow in the areas of social interactions and also

indicated growth in students' ability to learn computational thinking skills. Students shared that testing and debugging remained the most challenging concepts among the computational thinking concepts.

Unplugged coding was the term used for unique lessons to teach coding and computational thinking without requiring any technology or devices (Bustuttil et al., 2025). In a mixed methods study, Tonbuloğlu and Tonbuloğlu (2019) examined the use of unplugged lessons to teach coding and computational thinking to 114 fifth-grade students with similar computer experience but no prior experience in coding. Students were given a pretest to measure prior computational thinking skills. Data were collected over 10 weeks with scaffolding instruction to support computational thinking skills. Results showed that students positively responded to the unplugged lessons, displaying growth in computational thinking skills. However, there were no statistically significant changes to their problem solving abilities. The data also indicated that students had difficulty connecting the unplugged lessons to lessons typically associated with computer lessons.

In addition to highlighting the success of their research, Tonbuloğlu and Tonbuloğlu (2019) were also able to emphasize the areas of weakness for their study, providing an opportunity for future research. While Tonbuloğlu and Tonbuloğlu focused their research on upper elementary students, Relkin et al. (2021) focused their study on younger elementary students by conducting a quasi-experimental longitudinal design study that focused on 667 first and second-grade students' computational thinking abilities with the use of unplugged lessons. While data from Relkin et al.'s study

highlighted that young students' early exposure to computational thinking was beneficial, it was essential to note that their skills and abilities would continue to improve as they developed physically and mentally.

In addition to app-based programs to teach coding, other research focused on screen-free robots. For example, educational robotics was integrated into engineering and science instruction (Usengül & Bahçeci, 2020). Educational robotics was often used to help students understand computer science concepts (Noh & Lee, 2020). Educational robotics gave students a more relatable experience in mathematical problem solving (Cervera et al., 2020). There were several types of educational robots used in instruction. One form of robot often used in education and for research was tangible robots. Welch et al. (2022) described tangible robots as screen-free robotic coding toys that moved along a path or within a grid, including Code-a-Pillar, Botley, Cubetto, and Robot Mouse. Results showed that tangible screen-free robotic coding toys benefited students by increasing their spatial awareness, measuring skills, and sequencing (Welch et al., 2022). While tangible coding toys benefited younger students learning to code, older students benefited from web app-supported robots. In a quantitative study with six classes and about 125 fifth-grade students from a Title I school in the southeast United States, Shen et al. (2022) developed a curriculum and assessment system using a humanoid robot to assist upper elementary students with learning computational skills while solving programming and everyday reasoning problems. Shen et al. used pre and post-tests to compare the students' initial performance with their computational thinking skills. The results illuminated how to integrate and evaluate critical thinking in everyday logic and programming scenarios

among students. This study allowed students to apply and test their knowledge of the computation skills learned during the program.

On the other hand, in a qualitative study using a STEM class of 22 sixth-grade students in a New Jersey suburb, Munn (2021) investigated how students used computational thinking skills of abstraction, decomposition, pattern recognition, and algorithms when using educational robots during learning. Munn reiterated the need for research in the upper grades to transition the curriculum planners for education to include robotics and computational skills in math and science to increase students' ability to compete globally. Shen et al. (2022) and Munn connected with the focus of my study by expanding the research to upper elementary and emphasizing the use of educational robotics to teach computational thinking. However, the research of Sáez-López et al. (2019) contributed valuable quantitative and qualitative data regarding the use of educational robots that used block-style coding language and how beneficial this particular style of coding was to assisting students in gaining skills needed for not only computational thinking but also content matter like mathematics and science using the mBot robot. Sáez-López et al. (2019) proposed conducting an additional study in the field of visual block style coding. Block style coding involved using robots like Sphero to assist students in developing their mathematical and computational thinking skills. The need for future Sphero robot studies was a topic of particular significance in Sáez-López et al. (2019). The age of students was closely aligned with the student demographic I intended to examine and the kind of robots I investigated in my respective research endeavors.

Overall, research showed that elementary students learned computationally by learning to code in various ways, such as Scratch/ScratchJr, programming, and/or educational robots. Research showed that robots successfully taught computational thinking to students of multiple ages (Herro et al., 2022). However, there was a significant gap between upper and primary elementary levels regarding the amount of research conducted. Research indicated a pivotal need to expose students to computational thinking and computer science skills as early as possible (Relkin et al., 2021). The literature also showed an increased push in other countries that lacked technological advances and robotics access. This made my research even more critical as my research aimed to examine K–5 teachers’ experiences in implementing Sphero robots to teach computational thinking. Although the literature highlighted commonly used robots, little was known about the Sphero robots, even though they were widely used in schools. I sought to understand what K–5 teachers’ experiences were in implementing Sphero robot activities and how K–5 teachers used Sphero robots to teach computational thinking. The hope was that this research would act as a guide to assist teachers when it came to best practices and selecting the best educational robots to use to teach computational thinking.

K–5 Teachers and Computational Thinking

In this section of the literature review, I synthesized the data about K–5 teachers and how they taught computational thinking. I organized this section to discuss their instructional practices, the training and professional development they received about computational thinking, and teachers’ perceptions of teaching computational thinking.

Teachers' Instructional Practices for Computational Thinking

Providing teachers with pedagogical support and lesson suggestions to teach computational thinking became necessary as the demand for teaching computational thinking in the classroom increased (Kwon et al., 2021). The purpose of my study was to explore K–5 teachers' experiences of implementing Sphero robots to teach computational thinking. I encouraged teachers in my study to discuss instructional strategies and lesson ideas they had used when using robots to teach computational thinking. The literature included various teaching strategies teachers could use to build students' computational thinking skills, some using technology and some more traditional technology-free strategies. Additionally, Weintrop et al. (2016) created four sets of computational thinking practices to encourage advancing computational thinking out of the computer labs and more integration into subject areas in the classroom.

Additionally, the International Society for Technology in Education (ISTE) developed computational thinking competencies for educators as a way to deepen learning while building digital learning skills and providing a guide for teachers to integrate computational thinking practices into their instruction (ISTE, 2018). Although computational thinking standards had been created (ISTE, 2018), and programs and activities were made available to classrooms (Weintrop et al., 2016), little empirical research had been done on implementing these resources. Instead, teachers' instructional practices in the classroom to teach computational thinking were described in practitioner journals, such as journals from the National Science Teacher Association.

Some teachers used instructional strategies that taught computational thinking by exploring concepts like Earth Sciences. Massicotte et al. (2021) examined the integration of computational thinking in a middle school science lesson focused on weather. In grades 6-8, students engaged in weather challenges that incorporated computational skills through instructional strategies, including a phenomena framework, vocabulary exploration, prediction, and place-based learning. At the end of the unit, students reviewed computational thinking concepts and wrote reflections indicating how each idea was used. Students demonstrated their decomposition abilities by making weather predictions using radar and weather data. A color-coded chart supported the instructional strategy of visualization, which showcased students' ability to recognize patterns in the weather. Data abstractions were indicated by separating more extensive data into miniature representations of temperature trends.

Another Earth Science lesson to teach computational thinking focused on tide measurement (Shin et al., 2021). Shin et al. focused on modeling instructional strategies like scaffolding using technology to support the computational thinking aspect of the lesson and engaged students beyond normal abilities. Teachers concentrated on the learning goals to drive instruction on computational thinking. Other instructional strategies explained in the article included collaboration to create tangible items to support computational thinking and encouraged students to become active thinkers with their peers.

Teachers used additional instructional strategies to teach computational thinking using robotics; however, these instructional strategies were similar to those integrated

when not using robotics to achieve computational thinking (Rehmat et al., 2020). The actual proof of applying computational thinking skills was highlighted in the work students completed when their thinking was applied to robotics. For example, Newley et al. (2018) utilized computational thinking concepts in an afterschool program that used Bybee's robotics to increase student knowledge of engineering, critical thinking, and programming while creating statues of celebrities who could move. A critical instructional strategy for this lesson was collaborative learning among students and modeling from the instructor to ensure students' safety. A unique instructional practice described by Newley et al. (2018) included organizing students in workshops to bring all groups together at the end of a phase to allow students to summarize completed tasks and explain what they would do next.

The workshop model involved teachers presenting a concept to a small group, followed by student practice. Subsequently, students reconvened with the teacher to summarize their learning and identify areas requiring further attention (Newley et al., 2018). Another instructional strategy used was visualization throughout the lesson. Visualization could assist with comprehension of the computational concepts needed to progress in the robotic challenge (Newley et al., 2018). The instructors also used informal evaluations such as exit tickets, observations, or asking questions while students were working and taking notes as an instructional strategy to provide feedback to students. A more formal instructional strategy used in the classroom was applying rubrics to provide feedback. Rubrics were used to grade projects at the end of the lesson, and they often

included scale scores for each rubric category to reinforce computational thinking concepts and allowed students to learn these concepts in conjunction with coding.

While Newley et al. (2018) utilized robots to teach computational thinking, Barth-Cohen et al. (2019) took a no-tech approach to teaching computational thinking. Sixth-grade students practiced computational thinking skills by programming other classmates to walk like robots—the lesson started with the instructor explaining the instructions to collaborative groups. The instructional strategy of assigning group roles ensured that each student had a defined task within the cooperative group and the lesson. During phase one of the lesson, the students worked collaboratively to design their code before the robot (classmate) reenacted it. The students then tested their code for effectiveness and made edits where necessary. The instructor checked for understanding before students moved to the second phase of the lesson. During the second phase of the lesson, after students had tested their code and gained approval from the instructor, the students exchanged their code with other groups. The exchanging of the codes between groups allowed students to peer review the different groups' code; this was an instructional strategy that instructors used as a formal evaluation. Barth-Cohen et al. (2019) noted that specific issues arose during the lesson that other instructors should consider, including the size of the step students used, which impacted the code and the peer review results. Overall, Newley et al. (2018) and Barth-Cohen et al. (2019) presented instructional strategies that could be used with or without robotics to support computational thinking in the classroom setting.

Teachers investigated math-science integration to improve computational thinking instruction. Rich et al. (2019) suggested adding computational thinking with math and science to teach computer science in elementary school. A programmable paper circuit comprising copper tape, LED lights, and a programmable microcontroller was used to teach 6-8th graders about science history and computational thinking and math by Searle et al. (2021). Students and teachers explored energy conversion, energy storage, and polarity while learning computational thinking principles like algorithm writing and debugging. This article discussed collaborative learning, student-centered instruction, lesson differentiation for student assistance, and cross-content integration. Rich et al. (2019) conducted a qualitative study of eight elementary teachers in the Midwest to examine how they used computational thinking to improve math and science instruction. Teachers helped pupils by framing, prompting, and reflecting. Framing used the description to frame the task and forced students to practice, priming their thinking (Rich et al., 2019). Teachers utilized prompting to encourage pupils to develop computational thinking abilities (Rich et al., 2019). Teachers employed reflection to have students articulate their computational processes. These are some ways teachers integrated computational thinking into math and science.

Teachers used problem-based learning to teach computational thinking. Shin et al. (2021) noted that problem-based learning allowed students to collaborate on practical projects, encouraging inquiry through problem solving. In a quantitative study with four sixth-grade teachers and 200 sixth-graders, Kwon et al. (2021) examined how a problem-based curriculum, including computer science, affected student learning and attitudes.

The problem-based learning project gave students a realistic programming experience that tackled daily social concerns, focusing on creating a school culture of compassion. Elementary pupils learned computational thinking better via problem-based learning and authentic evaluation (Kwon et al., 2021). Kite and Park (2018) used computational thinking and NGSS standards to teach students in an Environmental Science AP class to redesign a fictional city. Weintrop et al. (2016) defined four main computational thinking practices: data, modeling, simulation, computational problem solving, and systems thinking. Kite and Park evaluated this approach. Kite and Park covered Wientrop's approach and used problem-based learning and computational thinking. The lecturer introduced technologies and gave background and context to help students understand and absorb the material. Like Shin et al. (2021), the session focused on student-centered collaboration, but instructors were encouraged to scaffold students' cognitive processes. Kite and Park assessed and provided comments using group sessions, rubrics, and the final presentation.

There were several benefits to using problem-based learning to support computational thinking in the classroom. Moreover, it offered additional examples of instructional strategies to enhance computational thinking. Stakeholders expected problem-based learning to positively affect students' education by utilizing various initiatives such as robotics, unplugged lessons, or programming (Kwon et al., 2021). In fact, Yang et al. (2020) demonstrated that teaching through problem-based learning could significantly help students develop skills via activities driven by overarching guiding questions.

Furthermore, implementing problem-based learning and student-centered activities may have greatly benefited students by enhancing their understanding of computational thinking concepts, especially those designed by Angeli et al. (2016). Although not directly identified as problem-based learning, Rehmat et al. (2020) conducted a qualitative case study that employed three common instructional strategies: questioning, modeling, and motivation and encouragement.

Specifically, teachers used questioning as an instructional strategy to engage six families with students aged 5-7 in computational thinking practices. For example, modeling illustrated to students how to code while completing both unplugged and plugged activities. Additionally, modeling allowed adults to guide students through different game levels, providing context for the robot's movement, as Newley et al. (2018) noted.

Moreover, the encouragement helped motivated students and enhanced their overall engagement in the activity. Importantly, it should be noted that the instructors of the lessons were not certified teachers; instead, they were part of a program where parents learned computational thinking strategies to foster learning among younger children. Ultimately, the expected benefit of these strategies was the students' success in increasing their computational thinking skills.

The fact that there were not many articles that referenced instructional strategies to increase computational thinking supported the need for my research. Most of the articles I found offered instructional strategies that teachers used that were more aligned with middle school learning. The lack of articles providing instructional strategies across

all levels of education highlighted the gap in the literature to support teachers in providing instructional strategies to implement computational thinking. The view of computer programming shifted from learning to coding to ensure that all students were exposed to computational thinking across subjects (Sung & Black, 2021). Finding the most effective ways to implement computational thinking across subject areas may have assisted with increasing computational thinking exposure in younger grades.

Training in Computational Thinking

Preservice Teachers

Computational thinking was a 21st-century skill that teachers were encouraged to teach for students' success; however, not many teachers knew the term or how to incorporate it into their classroom settings (Ghani et al., 2022). Moreover, one challenge was how to best train teachers in computational thinking and ensure they had the knowledge to teach their students (Mouza et al., 2018). Additionally, literature showed a gap in research between preservice teachers and tenured educators. Yadav et al. (2017) suggested that training preservice teachers was the most suitable time for teaching computational thinking and thinking computationally. Furthermore, Ketelhut et al. (2020) and Yadav et al. (2017) both argued that part of the problem with finding a successful way to teach computational thinking was the lack of a definitive definition of the term, as well as the lack of support for teachers in lesson planning and instructional practices. To address this, Ketelhut et al. (2020) conducted a qualitative study to create a model for preservice teachers to partner with mentor teachers. This included a two-part professional development program on computational thinking, supporting mentor teachers to assist

preservice teachers. The study included 13 mentor teachers with experience ranging from 5 to over 30 years and 11 preservice teachers. Overall, data from reflections, questionnaires, and focus group interviews indicated that although mentor teachers were successful during their professional development, they found it difficult to determine how to fit computational thinking into their curriculum and how to help preservice teachers do so. Additionally, teachers reported that their school environments did not support science instruction and lacked computational thinking resources. Of particular importance in Ketelhut et al.'s research was that professional development for teachers was not always enough to support the successful implementation of teaching computational thinking.

Preservice teachers were also challenged with learning coding, programming, and robotics before being in a classroom independently. In this regard, Govender and Govender (2021) stated that learning to program required complex problem solving, reasoning, and decision-making skills. Furthermore, coding and programming had been implemented in countries worldwide due to the high demand from companies that produced technology and software (Gökbulut & Bakangöz, 2021). In addition, Cam and Kiliçer (2022) indicated how the educational view of robotics and coding included virtual reality, which helped mechanical engineering while motivating students to explore connections on a multidimensional level.

Moreover, Govender and Govender (2021) conducted a case study with third-year preservice computer teachers to explore their programming experiences using Scratch for Arduino. The study aimed to transition beyond visual programming to a more integrated approach that incorporated physical computing through electronics and robotics. The

results showed that this approach significantly impacted preservice teachers' self-efficacy, perceptions, and attitudes. Consequently, the study emphasized the need for educators to learn effective pedagogical methods to successfully integrate coding and robotics into the classroom. To further support this, Govender and Govender (2021) proved that professional development was vital to teacher success in implementing computational thinking.

In a related study, Capobianco et al. (2020) utilized preservice teachers in their research to introduce engineering into 45 preservice teachers' education programs. They explained how to shift science teachers from learners to teachers in developing engineering design-based science. By employing the teacher-as-learner framework, Capobianco et al. (2020) stated that studies often lacked the acquisition of desired content by preservice teachers. They indicated that if one could examine how and when a preservice teacher generated an understanding of engineering design-based science and design-based science instruction, preservice programs could better pinpoint how to integrate instructional supports into their educational program.

Returning to the findings of Govender and Govender (2021), they explored the impact of physical computing on preservice computer teachers through Scratch for Arduino, demonstrating that integrating electronics and robotics significantly enhanced students' self-efficacy and learning. In contrast, Capobianco et al. (2020) gathered data through self-interviews and reflective narratives in a design-based methods course, successfully shifted preservice teachers' learning experiences. Similarly, Butler and Leahy (2021) conducted a qualitative study with 51 preservice teachers in a two-year

digital specialism program, emphasizing computational thinking. Notably, unlike Govender and Govender's participants, these teachers had considerable prior knowledge of computational skills.

Professional Development for In-service Teachers

Another way to improve computational thinking in classrooms was to provide professional development for in-service teachers. Several studies examined professional development focused on helping teachers teach computational thinking to students. However, teachers needed more than content knowledge for proper learning and understanding. Specifically, to fully grasp the concept, teachers had to learn pedagogies to support student learning (Kong et al., 2020). For example, using a multiple case study approach, Monteiro et al. (2021) worked with 11 preschool educators and 17 primary school teachers to develop integrated computational thinking activities, where the teachers were trained in computational thinking, robotics, and coding for 50 hours of in-service education. The main argument in this research was that student retention of information in computational thinking was based on how the educator delivered instructional practices and the educator's understanding of coding literacy.

Furthermore, the findings indicated a crucial role in engaging students aged 3 to 6, including tangible activities and assisting in the methodological approaches to computation. In particular, play-based learning, like storytelling, games, and artistic expression, helped increase computational thinking and coding literacy competencies. In light of this, the findings provided a greater connection to my study as I planned to focus on teachers' activities to teach computational skills in mathematics. Nevertheless,

although there were some successes, the researchers noted that the presence of COVID-19 negatively impacted their research and results. The scarcity of resources hindered the study. Other significant challenges included classroom management, lack of student participation, and the overall learning process. Correspondingly, Strawhacker et al. (2018) discovered results similar to those of Monteiro et al. (2021) that related to the value of professional development among primary teachers. In their study, Strawhacker et al. (2018) conducted a mixed-method study where data were collected from 222 students in kindergarten through second grade at six different schools across the U.S. Notably, six educators received specialized training using ScratchJr. to teach students to investigate the correlation between teaching styles and student outcomes. As observed in this research, Strawhacker et al. (2018) utilized the ScratchJr program and assessment tool ScratchJr Solve It to introduce coding to students. This study revealed that students succeeded in classes with more structured classroom management and student-driven activities.

In contrast, teachers who offered flexibility and prioritized student goals over their own demonstrated higher student success. Finally, expert facilitator results also positively altered student data results. Overall, the success and challenges of professional development were critical as I examined the activities teachers used to teach computational thinking. Together, these studies demonstrated how professional development in computational thinking, focusing on instructional strategies and meaningful activities, supported students' learning and the teacher's ability to teach computational thinking.

Although there was not much literature on professional development in computational thinking for secondary education, secondary education teachers faced a different problem with implementing computational thinking due to the discipline (subject area) they were teaching. In this context, the research of Kelter et al. (2021) and Jocius et al. (2021) contributed successful strategies for implementing the professional development of computational thinking; however, their approaches to professional development differed. Specifically, Kelter et al. (2021) conducted a qualitative case study with eight teachers from urban and suburban public high schools, focusing on constructionist co-design styles as a form of professional development, with none of the teachers having previous participation in co-design training. The resulting research findings highlighted the importance of teachers' learning as they created lessons that integrated computational thinking. While acknowledging that this practice of co-design professional development may not have been ideal and faced the challenge of time constraints, it allowed teachers to understand the pedagogical skills needed to teach computational thinking.

Conversely, Jocius et al. (2021) aimed to conduct mixed-method research in which 24 teachers participated in summer professional development designed to provide pedagogical support for computational thinking-infused lessons using the 3C model that focused on code, connect, and create. Importantly, Jocius et al. (2021) pointed out that the code section of the 3C aided teachers in enhancing their technological content knowledge. As a result, this connection allowed educators to merge subject matter with computational thinking, supporting teachers as they developed lessons infused with

computational thinking. When interpreting the data, one would have found success in teachers integrating computational thinking using the 3C model, despite some challenges such as lack of time, technological resources, difficulties in scaffolding, and collaboration between students and teachers, as well as teachers and students. Ultimately, Kelter et al. (2021) and Jocius et al. (2021) suggested that their professional development practices be implemented in preservice educational programs to alleviate the issue of insufficient computational thinking instruction before it began, focusing on code, connect, and create. Importantly, Jocius et al. (2021) pointed out that the code section of the 3C aided teachers in enhancing their technological content knowledge. As a result, this connection allowed educators to merge subject matter with computational thinking, supporting teachers as they developed lessons infused with computational thinking. When interpreting the data, one would have found success in teachers successfully integrating computational thinking using the 3C model, despite some challenges such as lack of time, technological resources, difficulties in scaffolding, and collaboration between students and students, as well as teachers and teachers. Ultimately, Kelter et al. (2021) and Jocius et al. (2021) suggested that their professional development practices be implemented in preservice educational programs to alleviate the problem of lack of computational thinking instruction before it began.

Overall, the literature made it clear that professional development was vital to educators' development of understanding and teaching computational thinking. It had to be said that the challenges presented by professional development also included the actual implementation of computational thinking in the classroom. The lack of time and

resources was a constant issue in education; the hope was to understand possible professional developments or educator training that demonstrated success for teachers and students. Professional development was essential to my study as we aimed to determine how teachers used Sphero robots to teach computational thinking. The next section of the literature review highlighted how teachers used robots during instruction.

Teachers' Perceptions of Experiences of Teaching Computational Thinking

Researchers argued that the most valuable aspects of coding and computational thinking were problem solving abilities, logical reasoning, and creativity, developed because they were transferable talents for actual life applications (Garcia-Penalvo et al., 2016). Applying these skills to the world benefited students only if teachers agreed. It became crucial to prepare educators as there was a greater focus on introducing computer science to elementary and secondary learners (Monjelat & Lantz-Andersson, 2020). Training and regularly involving teachers in professional development activities became increasingly necessary to help educators learn, unlearn, re-learn, and restructure their beliefs, attitudes, and ideas about integrating computational thinking into the classroom (Ogegbo & Ramnarain, 2022). The most crucial thing to know was how teachers viewed coding, the help they received, their level of achievement, and the difficulties they faced when teaching it (Wu et al., 2020).

Educators were expected to conduct lessons according to state standards and assessments, but most elementary education teachers focused more on reading and mathematics instruction (Amrein-Beardsley, A. (2022); Buck et al., 2010; Schneider, J., & Gottlieb, D. 2021). Consequently, instructors might not have emphasized instructing

computational thinking without computer science student evaluations (Rich et al., 2021).

The pressures teachers faced could have caused teachers not to value the benefits of computational thinking or even take the time to teach it. Teachers without sufficient training had emotional and cognitive obstacles to teaching computational thinking (Mason & Rich, 2019). Teachers' beliefs and perceptions regarding implementing computational thinking abilities in the classroom varied by the required teaching subject (Juškeviciene, 2020).

Hadad et al. (2021) investigated the pedagogical approaches implemented in educational robotics and coding to understand the extent of teachers' importance and the advantages of providing professional development through a miniature private online course utilizing eighty elementary school educators. This work's primary objective was to incorporate teachers' perspectives into a holistic understanding that illuminated a pedagogical approach encompassing tactics and teaching methods for educational coding and robotic learning. Hadad et al. (2021) described "teacher centrality" as how instructors engaged with their pupils, determining whether they served as facilitators by providing step-by-step instructions or permitted students to investigate and learn independently while providing guidance and support. How an instructor taught influenced their perception of how robotics and coding should be taught so that students acquired the most significant possible understanding. Hadad et al. (2021) suggested that instructors allowed students to explore various learning paths through coding and robotics.

Additionally, students should have engaged in novel coding and robotics experiments, akin to how Lou et al. (2023) advocated for teachers to integrate

computational thinking and robotics into their classroom lessons. Lou et al. demonstrated that students who collaborated better understood the skills being taught. While each teacher in the study had a different experience, the results and the students' progress were consistent. My study aligned with the findings of Hadad et al. (2021) and Lou et al. in that they offered insights into how K–5 educators could incorporate coding and robotics to teach computational thinking.

Jin and Harron (2023) examined the perceptions and progression of computational thinking skills among in-service teachers enrolled in an online graduate emerging technologies course. The age, experience, and education of the participants varied significantly. Furthermore, the results highlighted teachers' perceptions of computational thinking skills, particularly problem solving and creativity, which were enhanced by the online course. However, after completing the course, educators' perspectives regarding critical thinking and collaborative learning diminished. Jin and Harron (2023) explained that this decline in collaboration and critical thinking could be attributed to a lack of opportunities for collaborative activities among the course participants.

In contrast, in another study on computational thinking professional development, Hershkovitz et al. (2023) developed a four-week program where teachers engaged in a computational thinking professional development initiative that utilized a constructionist, hands-on, co-design methodology. Unlike the findings of Jin and Harron (2023), Hershkovitz et al. (2023) ensured collaboration among participants by having them jointly develop the lesson plans they implemented. This approach was oriented around the learner and emphasized the development of a profound comprehension. Moreover,

incorporating a co-design element in teacher professional development for computational thinking and throughout the implementation of computational thinking-enriched curricular units in the classroom proved beneficial, as the study results showed great success among educators. Thus, both Jin and Harron (2023) and Hershkovitz et al. (2023) demonstrated that professional development initiatives on computational thinking, whether complete courses or four-week programs, could successfully enhance teachers' understanding and application of computational thinking in the classroom.

Rich et al. (2019) and Fessakis and Prantsoudi (2019) conducted similar studies to gather important information for creating successful professional development programs for teachers that addressed the integration of computational thinking in the classroom. Specifically, they investigated educators' attitudes, beliefs, and perceptions regarding the content of computational thinking concepts. Notably, Rich et al. (2019) focused on elementary educators, whereas Fessakis and Prantsoudi (2019) surveyed K–12 teachers in Greece. Rich et al. (2019) demonstrated a significant disparity between the number of math and science connections. This finding appeared to reinforce the notion that instructors, on the whole, found it easier to integrate computational thinking into their mathematics curriculum compared to their science instruction. Therefore, these results highlighted the importance of using computational thinking across content areas.

Additionally, Fessakis and Prantsoudi (2019) found two main teacher groupings: one group viewed computational thinking as a creative and epistemological skill that utilized mathematics and computer science in different scientific fields, often equating computational thinking with mathematics. In contrast, the other group perceived

computational thinking as separate from computer science or associated it with specific aspects of computer science (Fessakis & Prantsoudi, 2019). Ultimately, the foundation of my research revolved around understanding the utilization of Sphero robots by teachers in the instruction of computational thinking. This exploration may provide valuable evidence for understanding instructors' perspectives on computational thinking and its implementation.

Current global interest was in incorporating programming instruction into educational settings (Monjelat & Lantz-Andersson, 2020). From an international perspective, Wu et al. (2020) compared the perceptions of K–12 educators in Finland, Mainland China, Singapore, Taiwan, and South Korea regarding the significance of 21st-century abilities, particularly in computational thinking and coding skills. A one-way ANOVA was used to analyze the survey results. The gender, age, and teaching location of each teacher were considered when analyzing the data. Results indicated that, compared to educators in Finland, those in Mainland China, Singapore, Taiwan, and South Korea held more positive opinions of their computer science proficiency. Male and female teachers had significantly different perspectives on how critical future skills would be for their students' careers; the attitudes of female teachers were also more favorable.

Compared to educators in Finland, educators' views in Mainland China, Singapore, Taiwan, and South Korea on the value of computational thinking in the classroom and schoolwide computational thinking support were more favorable (Wu et al., 2020). The computational thinking and programming abilities of male and female

teachers differed. Male teachers rated their programming and computational thinking abilities higher than female teachers. While Wu et al. (2020) compared K–12 teachers in multiple countries, Ling et al. (2017) examined a larger sample of primary teachers in Malaysia and their perceptions of integrating computational thinking. Using a sample of 159 primary school teachers in Malaysia, Ling et al. (2017) found that most participants aimed to incorporate computational thinking, exhibited a positive attitude toward the concept, and assessed its ease of use. However, there were also a few misconceptions about the idea (Ling et al., 2017). Teachers' opinions and perspectives were influenced by various elements that varied significantly between countries; therefore, special implementation of their investigation was necessary for each one (Fessakis & Prantsoudi, 2019). These global studies demonstrated the importance of studying teachers' perspectives on computational thinking in primary education.

Studies showed that teachers benefited from professional development to effectively integrate computational thinking into the classroom (Fessakis & Prantsoudi, 2019; Monjelat & Lantz-Andersson, 2020; Rich et al., 2019). Teachers' beliefs, attitudes, and perceptions regarding computational thinking and their difficulties when instructing it were critical to implementing computational thinking (Hershkovitz et al., 2023). Furthermore, exploring the global outlook on computational thinking revealed that educators from diverse nations held divergent opinions regarding how coding should be taught, the level of experience a teacher needed to teach computational thinking and coding, and how using coding impacted students and future generations (Wu et al., 2020; Ling et al., 2017). Although there was literature supporting teachers' perceptions as they

pertained to computational thinking and robotic usage, most of the literature on these topics was conducted outside of the United States. The remaining gap explained how U.S. K–5 teachers used robots to teach computational thinking and clearly depicted teachers’ perceptions of computational thinking. The lack of clarity regarding teachers’ perceptions, as provided in the literature, highlighted the need for additional research on the topic. My research aimed to explore K–5 teachers’ experiences in implementing Sphero robots to teach computational thinking. Understanding teachers’ perceptions of computational thinking would hopefully allow for more consistent opportunities for teachers to obtain the assistance they needed in implementing robotics and computational thinking in the classroom.

Summary and Conclusions

An analysis of the existing literature on computational thinking and using robots by K–5 instructors stressed the necessity of comprehending how teachers incorporated robotics and coding into their classrooms, how teachers selected activities for teaching computational thinking, their level of readiness, and their perspectives. The literature highlighted the importance of providing educators with adequate training and resources to facilitate the integration of robotics and coding in their teaching practices (Barana et al., 2020; Hadad et al., 2021; Kong et al., 2020). Additionally, the studies emphasized the need for increased collaboration among educators, researchers, and industry experts to develop effective strategies for teaching computational thinking skills through robotics and coding (Cassidy et al., 2020; Kelter et al., 2021; Yadav et al., 2017). The findings suggested that integrating robotics and coding in K–5 classrooms could be a powerful

tool for promoting STEM education and preparing students for the future workforce (Baroutsis et al., 2019; Fessakis et al., 2019; Ríos Félix et al., 2020). However, what was not yet understood was how teachers used robots to teach computational thinking and teachers' experiences implementing Sphero robots.

Although there was research on the connection between coding and computational thinking, there were still gaps to explore. The research on computational thinking with elementary students by introducing a framework that appropriately applied computational thinking skills for young learners was led by Angeli et al. (2016) and Angeli and Valanides (2020). These components could be taught together or separately to improve classroom computational thinking education (Angeli et al., 2016). According to a review of the relevant literature, the implementation of coding-integrated learning initiatives and novel tablet-based tools was identified as an effective strategy for promoting computational thinking skills and programming literacy (Strawhacker et al., 2018). In my review of the literature on computational thinking and coding, the themes that emerged were coding and programming using online tools, gamification, and educational robots (Chalmers, 2018; Chapman & Rich, 2018; Sáez-López et al., 2019). The research focused on a variety of methods to teach computational thinking, including robots (Usengül & Bahçeci, 2020) and mobile technology to code and run them (Karakasis & Xinogalos, 2020), as well as non- or low-tech options for coding (Angeli & Valanides, 2020).

Nevertheless, several gaps remained. One gap was understanding the difference between coding and programming and the relationship with computational thinking, which was crucial for students to fully grasp the concepts and apply them effectively

(Goldenberg & Carter, 2021). Much of the coding and computational thinking research had been concentrated at the middle school (Newley et al., 2018) or high school levels (Kite & Park, 2023). In addition, Scratch and ScratchJr were programming applications that were widely used in elementary education (Chou, 2020; Monteiro et al., 2021), but Sphero robots and their mobile apps were also commonly implemented, and little research had been done connecting these classroom experiences with how teachers built students' computational thinking skills. These gaps were essential because choosing which aspects of computational thinking to emphasize in the curriculum could be challenging. The skills and knowledge of the instructors were the most crucial factors in determining the success of computational thinking and coding education.

Although there was an increased discussion about teaching computational thinking skills in grades K–5 (Hudson & Baek, 2022), the current gap was whether early elementary-level teachers were confident in teaching computational thinking to young learners. Aslam et al. (2020) conducted a study that proved how teachers utilized the ISTE standards to develop students' computational thinking skills. Data from study results in the last five years showed that one component of improved computational thinking success was teachers' training and professional development (Barana et al., 2020). This could hinder the overall implementation and impact of computational thinking in education (Thompson et al., 2020). An investment in professional development for teachers was essential to ensuring their success in integrating computational thinking into their teaching practice (Kong et al., 2020; Rich et al., 2019). While some studies explored the importance of providing ongoing support and resources

for teachers to enable them to effectively incorporate computational thinking into their curriculum and maximize its benefits for students (Rich et al., 2019), and teachers' perceptions also played a significant role in the integration of computational thinking (Akgunduz & Mesutoglu, 2021; Ladachart et al., 2020; Rich et al., 2021). Understanding how teachers used robots with K–5 students to develop computational thinking could provide insights into best practices and what support teachers needed, like training, as they often struggled to incorporate computational thinking into their curriculum (Joicus et al., 2021). My study expanded on the current research by expanding the research done with K–5 teachers and Sphero robots, a less studied technology.

In Chapter 3, I will provide a detailed account of the approach employed for the planned study. The chapter is divided into four sections. These include research design and justification, role of the researcher, methodology, and trustworthiness issues.

Chapter 3: Research Method

The purpose of this basic qualitative study was to explore K–5 teachers’ experiences of implementing Sphero robots to teach computational thinking. While computational thinking has been gaining significance, a growing body of academic research explored computational thinking from philosophical and experimental perspectives (Angeli et al., 2020). To accomplish my purpose, I analyzed data from individual interviews conducted with 10 K–5 elementary teachers who incorporated Sphero robots into their teaching practices. I aimed to gain insights into the teachers’ experiences providing computational thinking skill development for early elementary students.

In Chapter 3, I detail the methodology I planned to use for this study. I include the research design and its justification, the researcher’s responsibilities, and the methodology, which contains the selection of participants, instrumentation, data collection, and plans for data analysis. In conclusion, I discuss the ethical implications of this research and the four facets of trustworthiness—credibility, transferability, dependability, and confirmability.

Research Design and Rationale

Two fundamental RQs served as the foundation for my study. The importance of equipping all students with computational thinking skills was widely recognized; however, there remained uncertainties regarding the knowledge that educators need to implement computational thinking in the classroom and the most effective scaffolding to

assist instructors in incorporating computational thinking into subject areas for Grades K–12 (Mills et al., 2025; Rich et al., 2019). These two RQs were the following:

RQ1: What are K–5 teachers’ experiences in implementing Sphero robot activities?

RQ2: How do K–5 teachers use Sphero robots to teach computational thinking?

I chose a basic qualitative design for this study. According to Patton (2015), the basic qualitative study addresses how individuals build meaning. Qualitative inquiry, sometimes known as generic qualitative inquiry, entails posing open-ended questions to individuals and observing subjects of interest in authentic environments to address issues, enhance programs, or formulate policies (Patton, 2015). I selected a basic qualitative design because my research focused on contributions from teachers to gain fundamental knowledge about the experiences of teachers who used robotics to teach computational thinking. According to Rallis and Rossman (2012), in-depth interview investigations allow researchers to elicit individual viewpoints regarding a topic. The use of open-ended questions to learn about the real-world experiences of teachers justified the use of a basic qualitative design.

Role of the Researcher

As the sole researcher of this basic qualitative study, I was committed to identifying and engaging with all aspects and responsibilities. I defined the recruiting criteria and set the necessary interview protocols. In addition, I created the participant criteria and interview methodology. Each interview was conducted utilizing a well-established protocol authorized by my committee, and a peer assessed it to ensure its

effectiveness. In addition, I analyzed the data and reported and interpreted the findings. In doing so, I used what I had learned from the conceptual framework and allowed the information to guide my study. I ensured that an ethical stance was taken to remove all bias from the research process. The development of the interview questions was a tedious process. I wanted to ensure that my line of questioning was engaging, direct, and concise. My role as the researcher did not conflict with my present position as a STEM lab teacher because I ensured that I maintained my role as the researcher and refrained from incorporating personal ideas, criticisms, or opinions during the research process. As a STEM lab teacher, I direct instruction with children, providing instruction on various subjects including agriculture and robotics. Instead of recruiting locally, I found participants who fit the inclusion requirements through my professional learning.

Methodology

This section describes the methodology I used in the study. Its components included detailed accounts of participant selection, recruiting, participation protocols, instrumentation, an interview guide, reflective diaries, data collection methods, and a comprehensive data analysis strategy. Furthermore, I describe the trustworthiness and ethical implications of this research.

Participant Selection Logic

Participants for this study included teachers who taught in a K–5 setting. I used a purposeful sampling and recruitment strategy for this study. Patton (2015) defined purposeful sampling as selecting cases rich in material that allows for a comprehensive understanding of issues significant to the study’s objective. This sampling strategy was

justified because the fundamental aim of utilizing a purposeful sample was to strategically concentrate the selection of cases on the goal of the inquiry, the primary questions, and the relevant data (Patton, 2015). Purposeful sampling aimed to obtain participants who could make valuable contributions to the study through their expertise. Ravitch and Carl (2016) explained that purposeful sampling provided context-rich and detailed accounts of specific populations. My study referred to their understanding of implementing robotics to teach computational thinking.

For this study, I aimed to recruit five Grade K-2 teachers and five Grade 3-5 teachers who used Sphero robots to teach computational thinking skills. The inclusion criteria for teachers to be in the study were that they (a) had to be currently employed as a K-5 teacher and (b) had to teach computational thinking using Sphero robots.

Instrumentation

To answer my RQs, I developed 15 interview questions aligned with the RQs (see Table 2). The first seven questions were aligned with RQ1 and helped me better understand how they used Sphero robots. I used the T3 model to interpret answers to these questions. Interview questions 8-15 aligned with RQ2. I aligned these interview questions with specific computational skills. See Appendix A. I also developed a full interview guide. The interview guide was developed following Turner (2010) and Castillo-Montoya's best practices (2016). The guide was what I used when conducting the interviews and included my opening comments, introduction, and full interview questions, along with potential prompts I used to probe participants further. See Appendix A.

Table 2**Interview Questions Aligned to RQs**

RQ1	RQ2	IQ	Interview question
X		IQ1	How do you introduce Sphero robots to students?
X		IQ2	Describe a favorite activity you've implemented using the Sphero robots.
X		IQ3	Describe a Sphero activity that did not go well when implementing it with your students.
X		IQ4	What are some of the challenges of using Sphero robots with young students?
X		IQ5	How do you assess Sphero robot activities?
X		IQ6	Have your Sphero lessons replaced lessons you used to do without technology? Or have they provided activities that were previously not possible? Explain?
	X	IQ7	How do you use Sphero robots to teach mathematical thinking and logic?
	X	IQ8	When you are teaching coding to young students, what tricks (strategies) have you used to help students learn how to follow instructions when coding? (Flow of control)
	X	IQ9	When teaching using Sphero robots, what activities do you use to teach students to put actions in the correct order when coding?
	X	IQ10	When using the Sphero with students, what sort of lessons or activities have you done where students had to identify patterns between older and newer problem solving tasks to solve a new problem?
	X	IQ11	Please provide an example of a lesson where students had to use the Sphero robot with a model or some sort of visual representation to solve a problem.
	X	IQ12	In your Sphero lessons, when faced with a large code, what strategies have been successful for you in how to teach students to break down the code into smaller sections?
	X	IQ13	In your Sphero lessons, how do you teach or introduce the algorithms to students?
	X	IQ14	When faced with challenges, how do you help students to alleviate frustrations with debugging while coding with Sphero robots?

Note. IQ = interview question; RQ = research question.

Procedures

The following section outlined the specific steps involved in the data collection process. I outlined the protocols for recruiting, participation, and data collection. Before

recruitment, I obtained authorization from the Institutional Review Board (IRB) affiliated with Walden University.

Procedures for Recruitment

I had designed an infographic that offered a concise overview of the study and the criteria that potential participants had to meet to be included in the recruitment process. The infographic also provided instructions on how potential interviewees could contact me if they were interested in participating in the study. The inclusion requirements for this study included participants who were currently employed as K–5 teachers and those who taught computational thinking using Sphero robots. I used my professional learning network to recruit individuals that might have fit my inclusion criteria. I also used the social media network X (formerly known as Twitter) to publish the infographic on my social media profile to attract educators.

As part of my recruitment strategy, I intended to maintain meticulous records of the sources through which each participant discovered the infographic. Upon receiving an email inquiry, I promptly provided potential participants with the consent form in the body of the email. If participants consented to participate, they replied by providing their contact information and a few questions related to the inclusion criteria. The data provided in the form was exclusively utilized for verification purposes to prevent imposter participants from entering the study (Roehl & Harland, 2022). I avoided sending attachments or links that seemed suspicious when sending out invitations.

Procedures for Participation

The process for participation began after receiving participants expressed interest in the study via email. I responded promptly within 5 days to arrange a suitable time for further communication and to express my gratitude. The interviews were scheduled via email, and I planned to conduct them virtually using Zoom. I notified the participants of the Zoom meeting one week in advance to confirm the date and time. I followed up and sent the link the morning of the meeting to verify that the time was still convenient for the participants, as scheduling conflicts could arise from being a teacher.

Verbal confirmation of consent was obtained at the beginning of the interview as I commenced recording, instead of receiving a signature. I continued recruiting until I had 10 participants or until I achieved saturation. Each interview lasted about 45 to 60 minutes. I also asked participants to engage in member checking, which took 10 to 15 minutes of their time. After the first interview, I requested that my committee members review the initial interview recordings and provide feedback on my execution of the interview protocol. The input occurred before I conducted any more interviews.

Procedures for Data Collection

Concerning data collection, I practiced a responsive interviewing style using an interview protocol guide. Although I planned to have set questions that would be asked of each participant, according to Rubin and Rubin (2011), the responsive interviewing style added flexibility to the interview process. It allowed for prompting or probing questions to get more out of the interview from the participant. Using the interview protocol guide allowed me to create a script. I followed the script to stay on track with the interview and

be mindful of the participant's time. I started the interview by thanking the participant for agreeing to participate. I provided the participant with a brief description of my research topic. I reminded the participant that the interview would be recorded and obtained verbal consent for recording. Once I obtained verbal consent to record, I asked the participants if they had any questions before beginning. As a way to assist the participant with feeling any nerves with the interview process, I asked the interviewee some clarifying questions about themselves, which also helped me to obtain background information about the participant and their teaching background and experience, to be sure they fit the study's inclusion criteria.

Patton (2015) suggested using prefaces, transitions, announcements, and introductory statements to help with the interview flow. I planned on using these methods to help guide the interview. I had probes and follow-up questions per question to get as much information as possible from the interviewee to help with my study. During the interview, I limited my feedback to the interviewee to ensure that I was not persuading them or making them feel their response was inaccurate. I controlled my facial expressions so as not to sway the interviewee's response or devalue their responses.

I reiterated the purpose of the interview as a basis for focus and feedback. Lemon (2017) emphasized the importance of mindfulness during the research process. During interviews, I wanted to be mindful to ensure I did not unintentionally send negative signals to the participant through sounds, facial expressions, or body language. I limited the notes I took so as not to make the interviewee nervous. When the interview was over, I thanked the interviewee for their time and asked them if there were any further

questions or anything to add to the study. After the interview, I wrote in my research journal any pertinent information or reflections I needed to add to share with my dissertation chair and methodologist. Patton (2015) stated that the post-interview period was crucial for contemplation and expansion. Ortlipp (2008) stated that actively maintaining a reflective journal ensured that experiences, opinions, thoughts, and feelings were recognized and fully integrated into the research process, including design, data collection, analysis, and interpretation.

After each interview, I took the recorded audio files of each interview and transcribed them using Kultura, a media system provided by Walden University. I listened to the audio several times before looking over the transcript. Once I understood the interview context, I reviewed the transcribed text and made the necessary corrections while listening to the audio recording. I then listened to the audio again to add any punctuation that the Kultura system or I had missed. I conducted a thorough examination of the transcription, making sure that it accurately reflected precisely what the participants said. Another part of the editing process was redacting confidential information and adding brackets to support where the participant omitted any words. I followed the formatting procedures required by Walden University or the suggestions of my committee.

Data Analysis Plan

I received IRB approval on December 10, 2024 (IRB12-10-24-0534501) and began data collection soon afterward. For this basic qualitative study, I performed my data analysis by analyzing the data through coding. As part of my research design, I used

the T3 model (Magana, 2017) for RQ1, and the computational thinking model (Angeli et al., 2016) for RQ2. Specifically, I applied a priori codes aligned with the T3 model for the first coding cycle to answer RQ1: What are K–5 teachers' experiences in implementing Sphero robot activities?

Ravitch and Carl (2016) defined coding as essential for researchers to effectively organize data into manageable units or chunks for analytical purposes. Before using a priori coding, Saldana (2016) advised that researchers should comprehensively understand the concept of identity to apply codes to the data effectively. A priori codes were described by Saldaña (2016) as provisional codes or metasummaries of connected papers revealing emerging themes. I developed a priori codes aligned with Magana's (2017) T3 framework (see Table 3). Once I had identified text segments that pertained to the framework, I applied open and pattern coding to determine how similar data within each a priori code was.

Open coding let researchers contemplate and engage with their data, giving academics analytic leads for further study (Saldaña, 2016). Finally, I determined the themes from the coding that answered the RQs (Percy et al., 2015). I coded manually rather than using electronic methods of coding. Electronic coding was an effective method for maintaining organization, but it was essential to remember that the researcher was ultimately responsible for the coding process. While electronic coding could assist, it did not replace the researcher's role in coding. Although my a priori codes were predetermined, once I began coding within each a priori group, I kept a codebook to organize new pattern codes that I used during the coding cycles. Decuir-Gunby et al.

(2011) stated that examining interview data was a complex process of making sense of and understanding it. Decuir-Gunby et al. (2011) defined coding as involving the allocation of codes, which had been previously specified or operationalized in a codebook, to raw data.

Table 3*A Priori Codes Aligned to T3 Framework*

A priori code	Aligned to literature topic or framework domain/element
Translational-Automation (T1-Auto)	Automation is when either teacher or learner uses technology to automate or add value to instructional or learning tasks (Magana, 2017). For example, the use of the driving feature would result in fewer task-related errors. The use of the driving component of Sphero would increase the number of tasks completed in an amount of time.
Translational-Consumption (T1-Consump)	Consumption, in which teachers and students access and consume digital content knowledge and information from online sources or other electronic media (Magana, 2017). For example, using Sphero as a digital tool to consume interactive digital content.
Transformational-Production (T2-Prod)	Production aims to improve students' capacity to evaluate their learning progress through self-assessment, boost their belief in their ability to succeed, and use their existing knowledge and skills to acquire new material effectively to create visible learning (Magana, 2017). For example, include students using the Sphero robot to digitally explain content learned in the classroom. For example, students can program the robot to follow the life cycle or the food chain or to produce some sort of final product.
Transformational Contribution (T2-Contrib)	Contribution, application of knowledge in distinct and practical situations, so showcasing, exemplifying, and conveying their profound comprehension of subject matter (Magana, 2017). For example, groups working together to meet goals they set, doing tasks they determined, to solve a problem using coding and robots.
Transcendent-Inquiry Design (T3-InquDes)	Inquiry Design facilitates students' utilization of digital resources to discern, explore, formulate hypotheses, and devise solutions to issues that hold significance for them (Magana, 2017). For example, students might use the coding skills they learned using Sphero robots, likely combining multiple content areas to address a wicked, real-life problem they are passionate about.
Transcendent-Social Entrepreneurship (T3-SocialEntre)	Social Entrepreneurship instructs students to purposefully and contextually develop novel and evolving software coding environments and communication platforms in order to systematically produce and expand more resilient digital solutions to issues that are significant to them (Magana, 2017). For example, students creating solutions using coding and robots, beta test, iterate and develop multiple solutions to wicked, real-life problems that are shared with stakeholders in the community.

In the interview protocol, I targeted specific elements of computational thinking in my interview questions. Therefore, to answer RQ2, how K–5 teachers used Sphero robots to teach computational thinking, I used open coding and then pattern coding to determine categories and themes. After the initial open coding, I employed pattern coding to group the emerging categories into broader themes, which allowed for a more nuanced understanding of the data and ensured a thorough analysis of how computational thinking was represented across the responses. According to Saldaña (2016), open coding was considered the initial form of coding.

Furthermore, pattern coding referred to regular or consistent occurrences in the data that appeared more than twice. This process helped identify recurring patterns and relationships between categories, providing deeper insights into the participants' experiences and how computational thinking was integrated and understood in different contexts. By synthesizing these themes, I was able to trace both explicit and implicit representations of computational thinking skills, highlighting areas of strength and potential gaps in participants' knowledge or application. This ultimately guided a more targeted interpretation of how these skills were being fostered in educational settings and informed recommendations for future curriculum development.

I included them in the member-checking process to ensure that I had correctly interpreted my participants' responses. Member checking enhanced the trustworthiness of findings derived from qualitative investigations (Erdman & Potthoff, 2023). I did this by providing them with a summary of my analyzed data. I allowed the participants to review and amend any misconceptions or omissions in the impressions of a single participant's

interview. Using the single-participant interview method of member checking allowed me to interpret the responses of those interviewed thoroughly.

Part of the data analysis plan was knowing how to treat discrepant data. Discrepant case analysis focused solely on the outliers, the peculiar cases, to gain insights into the normal or ordinary ones (Waite, 2011). Identifying discrepant data was crucial as it could not be disregarded simply because it deviated from the norm. Omitting inconsistent data was not ethical. To ensure ethical research practices, I aimed to identify and report discrepant data to maintain data integrity. Although I was using a priori coding, if patterns in the data fell outside my predetermined codes, I developed new codes and reported them in my results. If I encountered individual teachers with experiences that diverged significantly from others, I still reported that data, even if it did not contribute to the final themes. However, I was transparent in reporting how I treated all discrepant data.

Issues of Trustworthiness

Trustworthiness was a crucial element in qualitative research. By implementing specific methodologies and techniques designed to ensure precision (Ravitch & Carl, 2016), the reader gained assurance that the data effectively represented the phenomenon. These procedures were employed to reduce prejudice and enhance the precision of the results (Patton, 2015). Lincoln and Guba (1986) suggested that credibility served as an analogous concept to internal validity, transferability as an analogous concept to external validity, and dependability as an analogous concept to objectivity, which was the criterion for ensuring trustworthiness. I used several strategies to increase

trustworthiness, including testing the RQs, peer debriefing, triangulation, member checks, thick descriptions, audit trails, a code-recode strategy, and a reflexive journal. In this section, I explained the methods I employed to determine trustworthiness by assessing credibility, transferability, dependability, confirmability, and adherence to ethical principles.

Credibility

For qualitative research, Ravitch and Carl (2016) and Guba (1981) defined credibility as when a researcher demonstrated the capacity to consider and address intricate aspects that arose during an investigation and navigated patterns that defied simple explanations. Erdman and Potthoff (2023) suggested that to establish credibility in a qualitative study, a researcher had to establish rigor through structural coherence, field experience, triangulation, member verification, peer examination, interview techniques, and establishing the researcher's authority. I used the triangulation strategy for this study by interviewing individuals teaching various elementary grades. I also used the strategy of member checks by ensuring that my data was constantly reviewed to prevent researcher bias, as Ravitch (2016) recommended. In addition, I used the peer debriefing strategy by having a colleague review my data.

Transferability

Transferability referred to the ability of qualitative studies to apply and be relevant to broader contexts while retaining the original context's distinctive details and depth (Guba, 1981; Guba & Lincoln, 1985; Ravitch, 2016). The contextual aspects that influenced my study involved a specific emphasis on the works of Angeli et al. (2016)

and the development of a framework that clarified computational thinking and its fundamental components. In my study, I also utilized the Magana (2017) T3 framework for innovation, which offered an approach to improving instructional methods. Using the above frameworks, I was able to provide thick descriptions to help with the transferability of my research. To provide transferability, I described my participants' context, so that individuals could better understand if the study results might also apply to other situations. Transferability was further demonstrated by my commitment to maintaining a comprehensive audit trail of my practices. The study was also transferable as I presented a thorough account of my methodology and interview protocol.

Dependability

Dependability was defined as the ability of data to remain consistent and stable over an extended period, assuring its sustainability (Guba, 1981; Guba & Lincoln, 1985; Ravitch, 2016). In this study, I assessed the reliability of the interview questions, and member checking enabled participants to review the interpreted results for accuracy. I used triangulation and sequencing of methodologies to ensure dependability in my research. I also developed a clear justification for my selections, demonstrating that I had a suitable data collection plan. Establishing a practical research approach was crucial for ensuring the reliability of the results.

Confirmability

Confirmability referred to the recognition and examination of how biases and preconceptions influenced the understanding of data. It involved actively addressing and minimizing these biases through a disciplined process of self-reflection. There were two

methods to ensure that the study was transparent: by offering comprehensive explanations of the methodology and by meticulously documenting the study's limitations. Engaging in open reflection about the data-gathering and analysis process showcased self-awareness and mitigated bias (Ravitch & Carl, 2016). I maintained a research journal to demonstrate the thought process driving the research decisions. This journal included details of chronological research activities, kept a record of the dates and times of recruitment, the steps taken each day to find participants, and the outcomes of those efforts (Cronin, 2012).

Contextual information about data gathering was recorded in field notes to capture my thoughts and impressions before, during, and after engaging with the data (Matteson, 2021). Initial data reflection, which included quick ideas, impressions, and gut feelings about crucial concepts during the data-collecting process, was written down as soon as possible after the data was collected in a "dear diary" format. It was essential to document spontaneous decisions, behaviors, insights, and thoughts (Cutliffe & McKenna, 2004; Houghton et al., 2013; Shenton, 2004). Self-evaluation of my performance in the interviews focused on enhancing my skills in extracting information from participants, primarily through iterative questioning (Flicker, 2004).

For my self-assessment, I explored introspection and intuitive insights into the honesty and trustworthiness of the participants. The information was derived from the interview process and the application of iterative questioning techniques (Flicker, 2004; Shenton, 2004). I employed the technique of bracketing, which involved documenting my personal experiences, historical background on the subject, and the emotions elicited

during the process to recognize and distinguish them from the subject under investigation (Orange, 2016; Weatherford & Maitra, 2019). Using a reflexive journal allowed me to rigorously document my research process, ensured transparency, minimized bias, and maintained confirmability in my study.

Ethical Procedures

The trustworthiness of qualitative research depended on how researchers followed ethical procedures. In all academic pursuits, including adult and community education, ethics became so vital that it prioritized research and publication and compelled researchers to protect the dignity of the subject matter they studied against any potential harm (Olaniran & Baruwa, 2020). The researcher served as the data collection and analysis instrument in a fundamental qualitative study (Yoon & Uliassi, 2022). As part of this position, the researcher had to prioritize participants' well-being and enhance the study's accuracy and consistency (Stewart, 2010).

For this study, I followed ethical procedures by applying to the IRB at Walden University. Once this application process was completed, I received approval from the IRB to continue with the research. Subsequently, I interviewed my participants and made audio recordings to ensure accuracy. During these interviews, I conducted them privately, ensuring confidentiality by using a headset to prevent unintended listeners. Moreover, I created a folder with subfolders for each participant, ensuring they were password-protected. Every subfolder was labeled with the participant's assigned color code to maintain organization. In addition, all written notes or journals were securely stored digitally. As for data retention, I stored and retained the raw data for five years before

disposing of it. It is also important to note that demographics were not considered in the research study, and all communication with participants was conducted directly through email.

Furthermore, any information obtained was inaccessible to any supervisors or coworkers. Participants were able to contact me via email from social media posts showcasing the infographic, ensuring a low-pressure and non-coercive participation experience. After the interviews, I provided each participant with a member-checking summary to ensure their ideas were correctly interpreted. Finally, the data was exclusively utilized for the research at hand and not employed for any other purpose. Rest assured that the interview questions were designed to be safe and posed no harm.

Summary

Chapter 3 included a methodological description of this study, including a thorough explanation of the research design for this basic qualitative study, the outline of my role as the researcher, details of the participant recruitment and selection procedures, a description of the instrumentation used, and a plan for data analysis. In addition, I have provided a detailed explanation of the four dimensions of trustworthiness and the ethical factors I have considered. Chapter 4 will include a discussion of the data collection, data analysis, the study results, and evidence of trustworthiness.

Chapter 4: Results

The purpose of this basic qualitative study was to explore K–5 teachers' experiences of implementing Sphero robots to teach computational thinking. To achieve this goal, I interviewed eight elementary teachers who had incorporated robotics into their classrooms. Potential participants were solicited through Facebook groups such as Elementary STEM Specials Teachers and Sphero Educators Hub. These two groups generally consist of elementary educators who incorporate robotics in the classroom. I use these groups as part of my professional learning network. Using Magna's T3 framework and Angeli's framework for computational thinking as guides, I developed RQs and interview questions. The study's findings could contribute to social change by helping stakeholders make informed decisions on how to effectively teach students problem solving skills and computational thinking. This may lead to societal transformation as stakeholders learn strategies to better support teachers in integrating robots into lessons to promote computational thinking. Additionally, findings could advance educational technology by clarifying how teachers use robots to foster computational thinking in K–5 classrooms.

In this chapter, I present the results of this basic qualitative study. The chapter's components include detailed descriptions of participant selection, recruitment, participant protocols, instrumentation, an interview guide, reflective diaries, data collection methods, and a comprehensive data analysis plan. Additionally, I discuss the trustworthiness and ethical aspects of this research. This chapter also includes the study's findings and their relationship to the RQs.

Setting

The site for this qualitative study was virtual, with all interviews conducted remotely via Zoom. Because I recruited nationally, the participants were from varied geographical locations. Although each educational setting differed for each participant, all met the inclusion criterion, ensuring consistency in the selection. This approach allowed for diverse educational backgrounds to be represented, enabling a comprehensive understanding of the studied phenomenon.

Demographics

Participant 1 offered a unique perspective on the research, having worked as a STEM teacher with students in grades K–5. With over 26 years of teaching experience, Participant 1 taught first and second grades earlier in their career and, at the time of the interview, worked as a STEM teacher in a K–5 setting. The role of STEM teacher allowed Participant 1 flexibility to provide students with a variety of learning opportunities, including exposure to educational robots. Participant 1 had been using Sphero robots since the original Sphero Bolt, which they began using when teaching first grade. Apart from their extensive hands-on experience with technology, Participant 1 regularly participated in professional development workshops and STEM community events to stay current with the latest educational strategies and resources. Participant 1's enthusiasm for incorporating innovative technology into the classroom fostered a lively learning environment that promoted curiosity, creativity, and problem solving skills.

Participant 2, with 9 years of teaching experience, offered valuable expertise in using Sphero robots. As a K–5 technology teacher, Participant 2's first encounter with

Sphero was with the Sphero Minis. What distinguished Participant 2 from the other participants was their daily integration of Sphero into regular learning and academic lessons. Sphero robots play a central role in the learning experience, fostering student engagement and collaboration. Participant 2 discussed the importance of incorporating computational thinking to enhance learning outcomes and help students develop essential 21st-century skills. Additionally, Participant 2 shared innovative strategies with colleagues to support academic success, contributing to a vibrant, technology-integrated school community.

Participant 3 had a rich and diverse teaching background, having served as a special education educator and a K–5 STEM teacher. With over 5 years of hands-on experience in education, Participant 3 is passionate about inspiring young learners to think critically and creatively. Participant 3 challenges students to go beyond the basics of coding, fostering skills such as problem solving, innovation, and collaboration. Through engaging in inclusive teaching methods, Participant 3 aims to cultivate a love for learning and prepare students for future technological advancements. Additionally, Participant 3 often talked about seeking professional growth by integrating the latest educational technologies and methodologies into their teaching practice, ensuring continued effectiveness..

Participant 4 had over 17 years of experience in the education field and demonstrated a strong familiarity with STEM resources. Participant 4 stated “when you got into the STEM world, Sphero’s were very popular, so you couldn’t help but hear about them and be provided at least one Sphero robot.” Participant 4 first learned about

Sphero robots at a professional development event dedicated to innovative teaching methodologies. As a dedicated STEM teacher, Participant 4 actively incorporated Sphero's into their curriculum at least eight times a year, utilizing them for a variety of hands-on activities designed to enhance student engagement and understanding of robotics, coding, and problem solving skills. This frequent integration highlighted the value placed on Sphero's as a versatile tool in STEM education, supporting several key learning outcomes and fostering a stimulating environment for students to explore technology beyond traditional.

Participant 5 is no stranger to incorporating Sphero into learning practices. Participant 5 had worked as an Instructional Coach and a classroom teacher. Fiercely determined to obtain Sphero robots, Participant 5 purchased all of the Spheros either secondhand, on sale, or through donations. That demonstrated the importance of using Sphero with students. Recognizing the potential of Sphero to enhance engagement and understanding, Participant 5 consistently sought innovative ways to integrate this technology into various lessons. This dedication underscored a strong commitment to leveraging technological tools to support student learning and curiosity.

Participant 6 has worked in education for over 30 years and had also tested educational robots extensively. Their experiences included not only classroom teaching but also participation in robotics trials and evaluations. After learning about Sphero robots at the White House during Computer Science Week, this unique experience helped them see the engaging benefits of Sphero robots for all learners, from young children to adult learners. Participant 6 strongly advocated for the integration of robotics in

education as a means to inspire creativity, enhance problem solving skills, and foster collaboration among students of diverse.

Participant 7 has worked with Sphero robots since their inception. As a district STEM Teacher Leader, Participant 7 was provided with a set of Sphero robots to learn and incorporate into the classroom while also teaching other teachers. Having taught various grades before becoming a STEM teacher, Participant 7 understood what it took to engage students. While the primary focus of Participant 7 was developing the school garden, they always found time to integrate Sphero learning throughout the school.

Participant 8 had 7 years of experience as an Instructional Coach in elementary education, demonstrating a strong commitment to fostering innovative learning environments. This passion for Sphero started originally with LittleBits before they merged with Sphero, reflecting a long-standing enthusiasm for integrating technology into education. In this role, Participant 8 taught teachers how to effectively use Sphero robots in the classroom, providing professional development and practical strategies. Having had experience with all Sphero Robots, including the educational and programmable models, the new Indie robot by Sphero became a fan favorite among educators and students alike, appreciated for its versatility and engaging features.

It is worth noting that most participants have been using Sphero since the release of the original Sphero Bolt. When asked about robot usage, participants could mention any robot they had experience with, including various models such as Spark+, Sphero Bolt, Sphero Mini, and others. The new Sphero robot, the Indie, has been a favorite among participants. Participant 2 stated, “And then Indie is great as well, because it’s all

colors, so they can easily figure it out. Green means go, and red means stop, and these other colors are going to make it turn.” Additionally, many participants appreciated the intuitive design and the educational value of robots, noting that they help facilitate learning about programming, motion, and color recognition. See Table 4 for participant demographics.

Table 4

Participant Demographics of Experience and Current Position

Participant number	Teaching experience in years	Current position
1	26	STEM teacher
2	9	Technology teacher
3	5	STEM teacher
4	17	STEM teacher
5	20	5 th grade teacher
6	30+	Instructional coach
7	19	STEM teacher
8	7	Instructional coach

Data Collection

For this qualitative study, interviews were my primary source of data collection. I conducted a total of 8 virtual interviews in Zoom using the interview protocol described in Chapter 3. I audio-recorded in two ways. I used the embedded record feature within Zoom, and I also used the Voice Memos on my iPad as a backup recording. Interviews ranged between 25-80 minutes. Additionally, no unusual circumstances occurred during the data collection process, unless explicitly stated.

The interview with Participant 1 took place on February 17, 2025, and lasted 25 minutes. My next interview was with Participant 2, which took place on February 18,

2025, and lasted 52 minutes. Participant 3's interview was conducted on March 27th, 2025, and lasted 60 minutes. The interview for Participant 4 was conducted on April 1st, 2025, and lasted 31 minutes. Participant 5's interview was conducted on April 4th, 2025, and lasted 62 minutes. The interview with Participant 6 was conducted on April 15, 2025, and lasted 1 hour and 22 minutes. Participant 7's interview was conducted on April 15th, 2025, and lasted 58 minutes. Participant 8's interview was conducted on May 1st, 2025, and lasted 45 minutes.

To prepare the interview data for analysis, I transcribed the audio recordings from Zoom to create detailed written transcripts. Afterward, I copied the transcripts into Microsoft Word, carefully removed the timestamps, and made necessary edits to improve clarity, accuracy, and accurately captured and organized them for easy reference during analysis. During this process, I made sure to redact anything that would identify the participants or specific site for the sake of confidentiality.

Data Analysis

I used open coding, as recommended by Saldaña (2016) for qualitative research. I conducted coding using Excel. Within Excel, each participant had their own tab where the quotes from the transcripts were added and then coded. The final tab within the Excel document combined all the coded quotes from each participant into a single tab. To aid in the coding process, I developed a codebook following the guidelines of DeCuir-Gunby et al. (2011). I had a very challenging time creating my initial codebook. My initial codebook code names were more or less an identifier of the question that was being asked, rather than a code name. The lack of a firm code name presented delays. During a

workshop hosted by my dissertation chair, I was able to see exemplars from other doctoral candidates that highlighted that I did not entirely grasp the concept of what I was doing. Armed with more knowledge and understanding, I set out to redesign my codebook with new code names. The new code names were carefully selected to ensure clarity and wholeness of the quotes.

The final tab within the Excel document combined all the coded quotes from each participant into a single tab. This allowed me to see all the quotes associated with each code name at once. Once sorted alphabetically by code name, I used a website called Trello to organize them by category. This gave me the opportunity to organize and manage the quotes. What I like most about Trello is that from the dashboard, I can see all the code names, their categories, and their occurrences. I think it is worth mentioning that even up to this point, the code names are being updated and or merged when there are two codes with similar definitions.

Through the data analysis process, I identified a total of 31 codes, which I subsequently organized into 6 categories. A finalized list of codes and their definitions can be found in Appendix B.

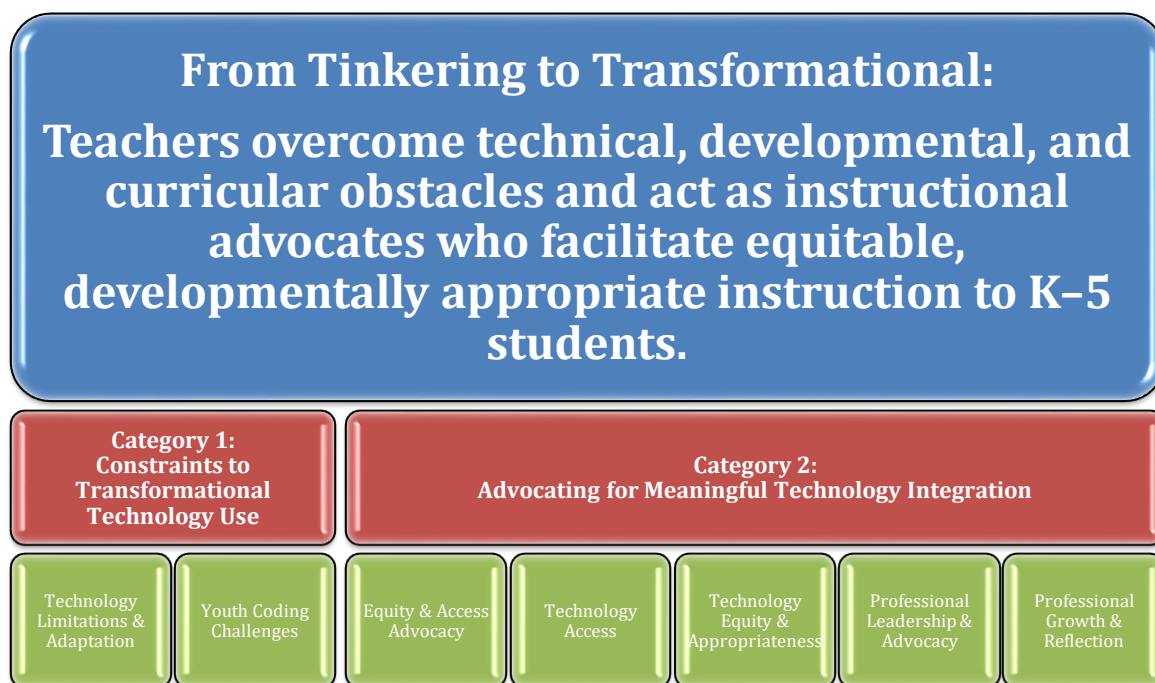
Data Analysis for RQ1

Each set of categories and codes are linked to the two RQs associated with this qualitative study. RQ 1 asks what teachers' experiences are in implementing Sphero robot activities. There are two themes for RQ1 with Theme 1 having two categories, and Theme 2 having 1 category. Theme 1 states that teachers overcome technical, developmental, and curricular obstacles and act as instructional advocates who facilitate

equitable, developmentally appropriate instruction to K–5 students. I also developed a shorter abbreviated theme, which for Theme 1 was From Tinkering to Transformational. Under Theme 1, there are two categories (a) Constraints to Transformational Technology Use and (b) Advocating for Meaningful Technology Integration (see Figure 3).

Figure 3

Codemap for Theme 1: From Tinkering to Transformational



Category 1: Constraints to Transformational Technology Use

The first category for Theme 1 is Constraints to Transformational Technology Use, and it has two codes: Technology Limitations and Adaptation and Youth Coding Challenges. Technology Limitations and Adaptations not only focus on the technological challenges encountered during the initial design phase of the Sphero robot but also address various operational issues such as speed, control precision, device connectivity,

physical handling, battery life, and environmental resilience. Participant 4 shared that “Sometimes the Sphero’s don’t connect to the iPad. You have to get another iPad, another app. For some reason, they don’t feel like connecting. So that’s a little frustrating, but that doesn’t happen as often.” By considering these factors, the design process becomes more comprehensive, ensuring that the robot performs reliably and effectively in a wide range of real-world applications, from educational settings to industrial uses. This holistic approach enables continuous improvement and innovation, ultimately leading to a more versatile and robust robotic system that can meet diverse user needs and environmental challenges.

The second code for Category 1, Youth Coding Challenges, focuses on the challenges of younger students and the adapted strategies for engagement. It captures how these limitations in emotions and basic academic skills – such as reading, vocabulary, or early math – create obstacles for students learning to use and program robotics tools like Sphero. For example, Participant 3 stated “Kindergarten. Oh, my goodness...there are reading barriers, so it’s too much for me to go around and make sure that you’re putting the blocks together on the canvas correctly because it’s only me.” These barriers often require additional scaffolding to support participation. The difficulties young learners face in coding highlight the cognitive demands of computational thinking, underscoring the need for schools to provide structured support and guidance that helps students turn struggle into deeper understanding and lasting skill development.

Category 2: Advocating for Meaningful Technology Integration

Under the same theme, Category 2 focuses on Advocating for Meaningful Technology Integration. This category emphasizes the importance of ensuring equitable access to technological resources and fostering a positive attitude toward technology use among students. There are five codes associated with this category, each highlighting a different aspect of advocacy and integration. The code names for Category 2 include Equity and Access Advocacy, Technology Access, Technology Equity and Appropriateness, Professional Leadership and Advocacy, and Professional Growth and Reflection.

The first code is Equity and Access Advocacy, which strongly supports introducing robotics and coding in K-2 to develop interest and confidence early in education. Participant 1 feels strongly about students being introduced to coding at a young age by stating, “there are a lot of kids who know a lot more than I do, even with the robots and with Sphero and coding. I teach them the basics, and then there are some kids who will jump in and say, “No, you can do this,” and you know, so I think all kids can definitely code. They’re very capable.” This early engagement aims to build foundational skills, encourage curiosity, and promote lifelong learning in STEM fields. Additionally, the category advocates for ongoing professional development for educators to effectively incorporate technology into their teaching practices, ensuring that all students benefit from innovative and inclusive learning environments.

Technology Access encompasses the various methods teachers use to obtain their Sphero robots, including personally purchasing them, using school-provided resources, or

participating in borrowing programs. It highlights the teacher's unwavering determination to provide students with innovative and engaging learning experiences through accessible technology. Participant 5 stated, "No funding, so I fully funded my whole thing." This commitment ensures that all students have the opportunity to benefit from hands-on, interactive learning tools, fostering a dynamic and inclusive classroom environment.

The code, Technology Equity and Appropriateness emphasizes the importance of selecting suitable educational tools tailored to students' readiness and accessibility needs. This approach involves teachers intentionally choosing simpler, developmentally appropriate robots—such as the Indie, Code & Go Mouse, Bee-Bots, and others—that align with students' age and cognitive development levels. Participant 2 mentioned that "Sphero bolts, we primarily use them in K[indergarten], but the coding piece, the algorithmic thinking piece usually starts between the second and third grade, when they've got enough reading skills to do that." By doing so, educators ensure that every student has equitable access to learning experiences that are both challenging and attainable, fostering engagement and developing foundational skills in a supportive environment.

The next code, Professional Leadership and Advocacy, highlights how dedicated teachers are not only capable of leading district and state-level professional development sessions but also of inspiring innovation and technological integration within their schools. Participants spoke about actively advocating alongside school leadership to expand training opportunities and encourage the widespread use of Sphero robots among fellow educators. Participant 5 shared how they encourage teachers and administrators to

play with Sphero. “[I] show them how to use the Sphero. It’s not one more thing to do. It just replaces something that you’re already doing.” Such initiatives foster a culture of continuous learning, technological advancement, and collaborative growth within educational communities.

The final code for Category 2 is Professional Growth and Reflections. Participants reflected upon their experiences of using Sphero robots and how much they learned and grew during the process. Participant 6 stated, “I think what I’ve learned the most, and learned a lot, is because I think I reflect a lot of my teaching practices.” This theme really highlights the participants’ evolving mindsets, adaptability, reflections on their teaching practices, and realizations about underutilization and the potential for deeper integration. The teachers’ comfort level with Sphero compared to other robots was also emphasized in this section.

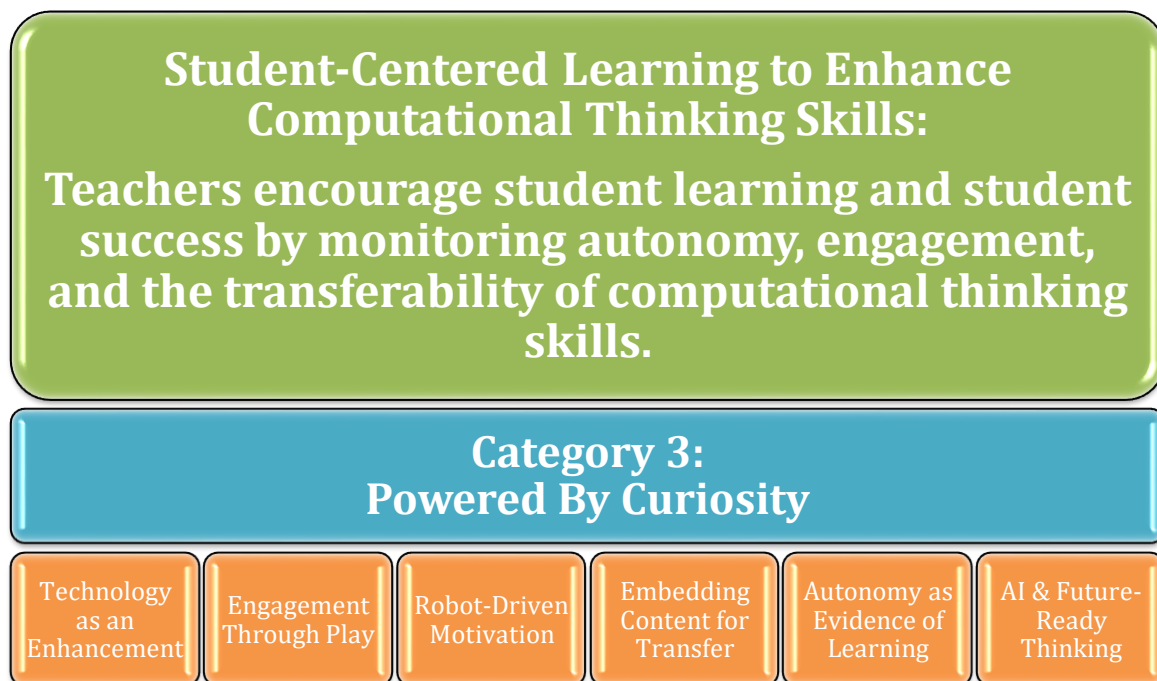
Category 3: Powered by Curiosity

The Theme 2 still relates to the RQ1 and concentrates on how teachers encourage student learning and student success by monitoring autonomy, engagement, and the transferability of computational thinking skills. Additionally, this theme emphasizes the importance of fostering a curious mindset among students to enhance their ability to transfer computational skills across different contexts. It highlights the role of teachers in encouraging inquiry and exploration to facilitate deeper understanding and skill development. The shortened Theme 2 title is Student-Centered Learning to Enhance Computational Thinking Skills. The one category for this theme is Category 3, titled Powered by Curiosity, and it has six associated codes. The codes for Category 3 include

Technology as Enhancement, Engagement through Play, Robot-Driven Motivation, Embedding Content for Transfer, Autonomy as Evidence of Learning, and AI and Future-Ready Thinking (see Figure 4).

Figure 4

Codemap for Theme 2: Student-centered Learning to Enhance Computational Thinking Skills



The first code under Theme 2, Technology as an Enhancement, emphasizes how Sphero enhances, but does not replace, core content, helping to deepen engagement and application. This approach highlights the supportive role of technology in education, aiming to complement and enrich traditional teaching methods without overshadowing them. Participant 2 truly captures the essence of Technology as an Enhancement when Participant 2 stated, “Yeah, Sphero definitely provided activities that wouldn’t be possible.” This quote highlights how Sphero contributes to meaningful student learning.

Evidence of student excitement, collaboration, and focus truly embodies Engagement through Play, which is the second code. Participant 5 expresses this well when Participant 5 said, “They see it as a game. And I think that’s the biggest part.” This code examines how students view coding as a game, which boosts their persistence and resilience. Students stay actively engaged through physical interactions with the robots, increasing motivation and learning.

The third code for Theme 2 is Robot-Driven Motivation. Working with robots (e.g., Sphero) encourages students to persist through challenges, keep trying, and overcome frustration, thereby promoting a growth mindset and problem solving attitude. When Participant 6 stated, “The reason I use Sphero is that I don’t get to pick what motivates kids, and they love Sphero, really love them.” Additionally, engaging with robotics fosters creativity, teamwork, and critical thinking skills—essential skills for success in the 21st-century digital world. It also helps students develop confidence in their abilities to learn new technologies and tackle complex problems, making the learning experience more interactive and enjoyable.

Embedding Content for Transfer is a code that acknowledges the teacher’s limited ability to observe students applying robotics learning in other subjects, while intentionally embedding other content areas into robotic lessons to encourage and facilitate cross-curricular knowledge transfer. This approach recognizes both the constraints teachers face and the potential for integrated learning, aiming to foster a more holistic educational experience. Participant 5 stated, “I think that...for me and my experiences, it’s really that the byproduct of students being able to solve complex

problems is that they go step by step. I don't know if it's a direct application as far as them knowing." Additionally, integrating diverse content areas helps students make connections between concepts, enhancing their understanding and retention. This method promotes critical thinking and problem solving skills that transcend individual subjects, preparing students for real-world challenges. Overall, embedding content for transfer not only broadens students' knowledge base but also encourages adaptable and transferable skills across various disciplines.

The fifth code, Autonomy as Evidence of Learning, the teacher uses the students' ability to independently manage robotics activities—without direct instruction—as a clear indicator that they have mastered skills and internalized prior learning. Students are able to articulate their learning and/or thinking process. When Participant 4 stated, “When I ask them questions about the assignment or about what they're doing, if they can explain it to me, then they got it,” it really shows how students are able to take charge of their own learning to grasp concepts.

Considering the future, it is essential that teachers navigate the rapidly changing landscape of technology. The AI and Future-Ready Thinking code highlights AI, machine learning, and preparing students for future careers. “I mean, even though they're 5 to 11 years old... You still need to be cognizant of what their future holds, even though we don't know what it holds,” stated Participant 7, expressing the importance of always looking ahead. Additionally, integrating innovative teaching strategies and fostering adaptability can further empower students to succeed in an uncertain world. Teachers are responsible for creating an environment that encourages curiosity, resilience, and lifelong

learning, ensuring that students are equipped to thrive in the technological advancements of tomorrow.

Data Analysis for RQ2

RQ2 has two themes and three categories. RQ2 was “How do K–5 teachers use Sphero robots to teach computational thinking?” The question is important because it explains the teaching methods teachers use to help students understand computational thinking. Theme 3 emphasizes how teachers facilitate students’ progression from guided play to algorithmic thinking using developmentally appropriate and scaffolded instructional strategies. The shortened theme name is Planning Perfect Instructional Practices for Computational Thinking. The first category under the RQ2 is Category 4, which is Strategic Instructional Design for Computational Thinking. This category focuses on creating effective teaching strategies that enhance students’ understanding and application of computational concepts, ensuring that instruction is both purposeful and targeted. The five code names for Category 4 include Flexible Planning and Standards Alignment, Frequency of Integration, Exploration-Based Introduction, STEM Integration Practices, and Assessment Practices (see Figure 5).

Figure 5

Codemap for Theme 3: Planning Perfect Instructional Practices for Computational Thinking



Category 4: Strategic Instructional Design for Computational Thinking

Every good teacher knows that proper planning is the key to great instruction. Flexible Planning and Standards Alignment serves as the foundation for effective teaching. The first code name under the initial category emphasizes the importance of adaptability in lesson design. Participant 5 provides a glimpse into planning by stating, “I’ve used Sphero for ELA, Math, and Science.” Lessons are built around science and computer science standards, but with an added focus on flexibility and creativity, encouraging students to explore and innovate within structured guidelines.

Frequency of Integration captures how often teachers incorporate Sphero into their instruction. It is essential to recognize that there is a time constraint within

elementary education. Participant 3 stated they implement Sphero's "at least once per quarter." However, it is not about the frequency of integration; it is about the quality of instruction, which is a good introduction to the next code name, Exploration-based Introduction.

The third code, Exploration-Based Introduction, focuses on student-centered learning. "I usually give them the basics and let him explore first, and just see what it does," says Participant 8. This expresses a type of Exploration-Based Instruction that students can use to enhance their learning. It intentionally uses open-ended, low-pressure, and curiosity-driven robotics activities that help students learn computational thinking concepts by engaging with Sphero in a playful, discovery-driven way.

When students iterate through the engineering design process, using physical blocks and maps to plan, build, test, and revise their robot's movements, it fits the fourth code name, STEM Integration Practices. Some of the participants are STEM teachers, and their experiences highlight the benefits of Sphero robots on multifaceted levels. Participant 7 explains how they use STEM when they mentioned, "I use STEM. And I use a lot of arts integration, language arts integration, and I combine it with the science content." The participants test the limits not only of the robot but also of the students, challenging their thinking, problem solving skills, or expanding their creativity.

Modern assessment practices are shifting from traditional testing toward more authentic, formative approaches that better capture students' skills and understanding. The Assessment Practices code refers to assessing students not only on their final coding or robotics product but also on their ability to collaborate, communicate, and work as a

team during the process. Assessments are also largely informal and observational, with an emphasis on student autonomy and growth. Participant 5 shared their view on assessing by stating, “I think part of it is anecdotally, and how the students are reacting to the assignment at hand or the task at hand.” Furthermore, these innovative assessment strategies promote a deeper engagement with learning, enabling educators to tailor their instruction to meet individual student needs.

Category 5: Scaffolding Computational Thinking Practices

In educational practice, scaffolding helps teachers break down complex tasks into manageable steps and gradually transfer responsibility to students as their skills grow. This instructional strategy is especially important in fostering independence and confidence among learners. This is the purpose of Category 5, which is Scaffolding Computational Thinking Practices. There are six code names associated with Category 5, each representing different aspects or methods of scaffolding within computational thinking education. The six code names under Category 5 include: Fundamental Driving Introduction, Fundamental Computational Thinking (CT) Skills, Algorithmic Foundations, Understanding Shapes, Visual Decomposition, and Real-World Integration. By implementing these strategies, educators can effectively support students in developing their problem solving skills and computational proficiency.

When students are introduced to Sphero, there are several learning curves students must navigate. The first code, Fundamental Driving Introduction, involves introducing young students to robotics through simple, foundational driving activities that focus on basic control skills before gradually progressing to more complex programming concepts.

Participant 6 discussed how they were able to add buy-in with students prior to getting them into coding by saying, “Well, I like to show them how to drive them [Sphero]. Because they like the driving aspect. And it’s instant gratification, obviously. And then we move into the coding pieces, which they really do like.” This approach helps build confidence and foundational knowledge, making it easier for students to understand and apply more advanced robotics techniques in future lessons.

Developing Foundational CT Skills is the second code related to equipping learners with the cognitive tools to analyze problems, recognize patterns, and design efficient, step-by-step solutions. The code captures how students are encouraged to plan, anticipate, and mentally rehearse the sequence of steps needed to accomplish a goal using Sphero. Participant 2 likes to “review all those expectations before we start, and they’re practicing a lot of that as far as computational thinking, understanding the steps I need to take if I want it to do this.” Steps like this help to reinforce foundational computational thinking principles such as sequencing, prediction, logic, and problem solving skills. It is also the recognition that block coding requires explicit instruction and scaffolding.

The third code, Algorithmic Foundations form the core of computational thinking, enabling learners to design precise, step-by-step procedures for solving complex problems. With Algorithmic Foundation, teachers use real-life analogies and hands-on activities as a code name to introduce algorithms as a sequence of step-by-step instructions. Participant 2 described how they introduce algorithmic thinking to students when they mention “we start really basic with just out like what an algorithm is. An algorithm is a step-by-step process. So, relating it to their everyday lives. That’s generally

where we start.” This method makes it clear for students to understand and connects it to a concept they can grasp easily.

Integrating mathematics into robotics helps students understand how geometry, algebra, and physics work together to control robotic movement and decision-making. The fourth code Understanding Shapes involves teaching students how to use logical reasoning to grasp and manipulate relationships between speed, duration, and distance when programming robots. It is important to understand how math plays a huge role in the classroom when using Sphero. Participant 7 provided insight into implementing math with Sphero, stating, “Well, you can use multiplication, subtraction, or addition, whatever they’re working on in that grade level.” Sphero activities can be used to teach additional mathematical concepts, including angles, distance, time, proportional reasoning, geometry, and so much more.

The fifth code, Visual Decomposition, incorporating visual aids when teaching coding in robotics, improves students’ understanding by helping them visualize logic flows, spot errors, and grasp complex structures more clearly. “So, using a lot of screen casting or putting up a deck with pictures of exactly what they need to do helps” is an example of how Participant 2 uses visual aids to support students during decomposition with Sphero. Teachers can assist students in breaking down large coding tasks into manageable parts using visual aids and modeling.

The last code name for Category 5 is Integrating real-world context into education boosts student engagement and critical thinking by connecting academic content with everyday applications. Real World Integration, as a code name, frames computational

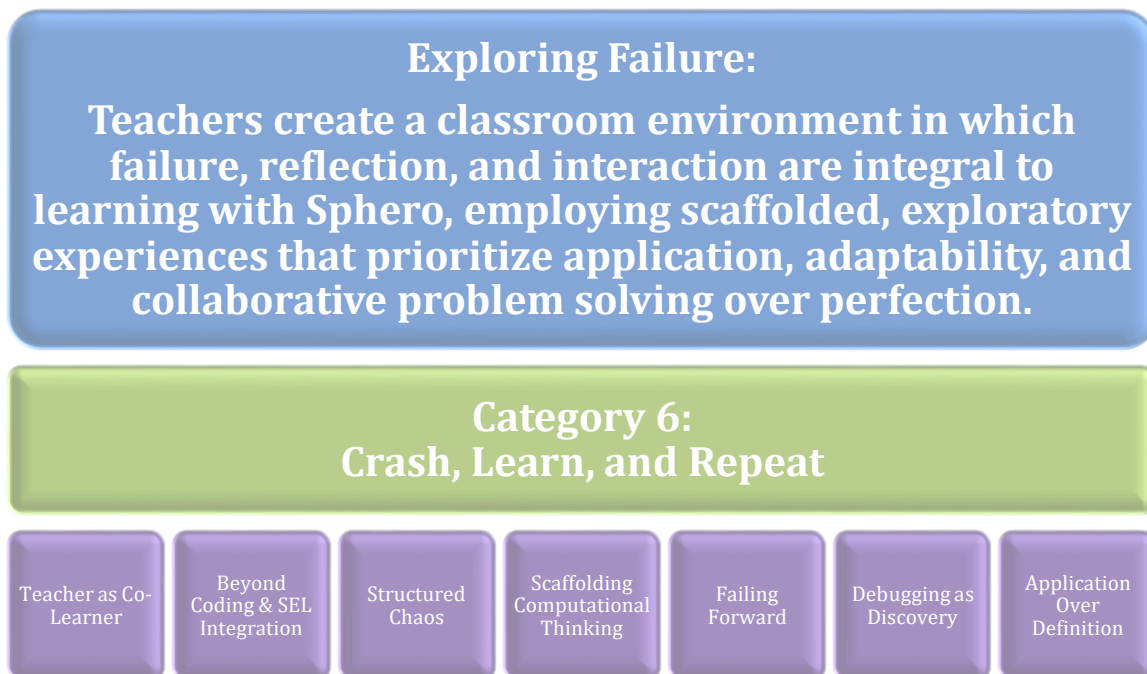
thinking as a means of teaching problem solving strategies that students can apply beyond coding, recognizing its broader value for real-life and cross-disciplinary contexts. Participant 6 had a unique activity that gives a great depiction of Real World Integration, stating, “They build this little city, and while coding, they deliver mail at certain places and stop and go and stop and go.” This approach not only makes learning more relevant and interesting but also prepares students for practical situations they will encounter outside the classroom, fostering skills that are essential in today’s dynamic world. Moreover, by integrating real-world scenarios, educators can inspire students to see the relevance of their education, encouraging lifelong learning and adaptability in an ever-changing society.

The final theme for this qualitative study is Theme 4, which highlights how teachers create a classroom environment in which failure, reflection, and interaction are integral to learning with Sphero, employing scaffolded, exploratory experiences that prioritize application, adaptability, and collaborative problem solving over perfection. Theme 4 only has one category. The shortened theme for Theme 4 is Exploring Failure. Category 6 is titled Crash, Learn, and Repeat. Participants discussed how the “crash, learn, and repeat” approach empowers students to embrace challenges with curiosity and confidence, transforming setbacks into opportunities for deeper understanding. Teachers foster resilience by creating learning environments where failure is viewed as a natural and valuable part of the problem solving process. There are seven codes for Category 6. The code names include: Teacher as a Co-Learner, Beyond Coding and Social-Emotional Learning (SEL) Integration, Structured Chaos, Scaffolding Computational Thinking,

Failing Forward, Debugging as Discovery, and Application over Definition (see Figure 6).

Figure 6

Codemap for Theme 4: Exploring Failure



Category 6: Crash, Learn, and Repeat

Acting as a co-learner allows participants to learn alongside their students—experimenting, questioning, and problem solving together—thereby fostering a more dynamic and inclusive learning environment. The first code Teacher as a Co-Learner enables the participants to model learning alongside students while embracing uncertainty and shared discovery. Participant 2 stated, “Just because your name is over the door doesn’t mean that you’re going to really know any more sometimes than the kids.” Additionally, adopting this approach encourages participants to remain curious and open-

mindset, which can inspire students and promote a growth mindset. It also helps build a classroom culture based on mutual respect and collaborative exploration, ultimately enhancing engagement and understanding for all learners.

With Beyond Coding and SEL Integration, students not only learn to solve problems computationally but also to communicate effectively, manage frustration, and support their peers through challenges. The integration of Sphero activities across various curriculum areas—such as digital citizenship, classroom routines, and social-emotional learning—extends beyond just algorithmic thinking and technical skills. It also demonstrates how participants help students manage their frustrations while coding, fostering resilience and perseverance. Participant 5 highlighted an innovative approach by using Sphero as an incentive to reduce chronic absences and explained:

This allowed [students] to start to develop a passion for learning, so they knew attendance was important. As soon as they started to show up more often, chronic absences began to decline because they understood they needed to be at school to participate in the club, which in turn increased attendance.

Additionally, participants felt these activities promoted teamwork, creativity, and real-world problem solving, making learning more engaging and meaningful for students.

In hands-on learning environments like robotics, the third code Structured Chaos is about encouraging students to collaborate, take risks, and solve problems creatively while participants keep direction and learning goals in mind. Participant 7 had a clever idea for defining space when working with Sphero in the classroom: “the kids are spaced

out and there are pool noodles in between them. And then that's kind of what defines their space." Structured Chaos examines the classroom management tips and tricks used by participants to offer other educators ideas for handling students during chaotic yet engaging times.

Integrating scaffolding strategies into robotics instruction enhances computational thinking by helping students connect abstract concepts—such as sequencing, algorithms, and debugging—to tangible, real-world applications. Scaffolding Computational Thinking is a code about instruction that is carefully scaffolded across grade levels, from basic driving to complex coding. Participant 5 described a time when they allowed the students to explore, but recognized their assistance was needed. "I love the creativity, and I want them to be able to explore. But there are times when it's like, all right. Direct modeling instruction is what you have to do." This approach ensures students develop confidence and competence in robotics through a gradual increase in challenge and support, fostering deeper understanding and engagement.

The fifth code, Failing Forward, was about encouraging students to see mistakes as vital parts of learning, turning failure into a driver for growth and improvement. Participants view failure as an integral part of the learning process, embracing it as an opportunity to iterate and improve through trial and error. Participant 6 recognizes the value of Failing Forward when students are "considering ideas, trying everything. And my favorite part is when it doesn't work — they don't just shut down and say, 'Oh, yeah, we failed.' Instead, they keep working on making it succeed." Ultimately, failing forward

empowers students to see learning as an ongoing process, where each setback serves as a steppingstone toward mastery and innovation.

The sixth code, Debugging and Discovery is a code that was used to describe how students learn to analyze their code, test hypotheses, and uncover new solutions—transforming mistakes into moments of insight and innovation. With Debugging as Discovery, students are encouraged to identify and correct errors through self-reflection, peer collaboration, and teacher guidance. They work together to solve challenges, often debugging and iterating as a team to find the best solution. Participant 1 shared insight on debugging: “I think they need to see...it’s easier for them to find the bugs and fix things if they can do it a little bit at a time.” Viewing debugging as discovery transforms frustration into curiosity, empowering students to approach challenges with persistence and a sense of possibility.

The seventh code, Prioritizing Application over Definition, emphasizes the importance of learning by doing, allowing students to develop understanding through experience rather than memorization. “We use the terminology to help them become comfortable with it because it’s used so often. It’s just repetition,” explains Participant 5. Application over Definition describes the teacher’s emphasis on ensuring students grasp and apply computation thinking concepts, even if they have not yet mastered the formal vocabulary with those concepts. In essence, valuing application over definition transforms computational thinking from theory into action—empowering students to think critically, explore confidently, and create with purpose.

Evidence of Trustworthiness

I upheld issues of trustworthiness in a number of ways. In this section, I will describe how I ensured credibility, transferability, dependability, and confirmability. First, I ensured credibility by using triangulation. I did this following the strategies suggested by Erdman and Potthoff (2023) that I described in Chapter 3. Triangulation for this study was the use of multiple teachers with varying amounts of Sphero experience. Next, regarding transferability, I heavily relied on the contextual aspects influencing my study, specifically emphasizing the works of Angeli et al. (2016) and developing a framework that clarifies computational thinking and its fundamental components. In my study, I also utilized Magana's (2017) T3 framework for innovation, which offered an approach to improving instructional methods. Thirdly, I ensured dependability by following the suggestions of (Guba, 1981; Guba & Lincoln, 1985; Ravitch, 2016). I demonstrated a suitable data collection plan and established a practical research approach. Finally, for confirmability, I maintained a research journal that acted more as a journal where I shared my thoughts and overall views of the research process.

Results

In this section, I have organized the results by theme. For each, I include paraphrased ideas and quotes, showing evidence of them.

Theme 1: From Tinkering to Transforming

Theme 1 was From Tinkering to Transforming. This theme that emerged from the interviews highlights how teachers overcome technical, developmental, and curricular obstacles, acting as instructional advocates who facilitate equitable and developmentally

appropriate instruction for K–5 students. Additionally, these participants demonstrated resilience and innovation in adapting their teaching strategies to better serve diverse student needs, ultimately fostering a more inclusive and effective learning environment. They are committed to continuous professional growth, seeking out new methodologies and incorporating feedback to improve their instruction. Their dedication not only benefits their students but also strengthens their advocacy efforts among colleagues and stakeholders, fostering a collaborative and supportive educational environment.

Category 1: Constraints to Transformational Technology Use

The journey toward transformative technology in education often encounters various obstacles, as students and teachers must navigate how to adapt to technological limitations that affect access, understanding, and implementation while still working beyond youth coding challenges. In Category 1, participants discussed challenges that can arise both technologically and academically, making it essential for educators to develop innovative solutions and supportive strategies. Overcoming these hurdles is crucial to ensuring that all students can benefit from the opportunities that modern technology offers, fostering an inclusive and progressive learning environment.

As Sphero’s technology has advanced over the years, the reliability of the Sphero robot posed a greater challenge for teachers when they first implemented robotics in the classroom. Participant 8 recalled a time a lesson failed due to the instability of the Bluetooth connectivity of Sphero, “We had too many Sphero’s in a room. They were trying to do an obstacle course, but the technology wasn’t good back then. They kept reconnecting to other people’s Sphero’s. It was bad.” Participants 2, 4, 6, and 7 also

expressed issues with connectivity, but Participant 6 confirmed that Sphero has since fixed this problem, expressing, “They’ve changed that, and they’ve troubleshooted that, and they’ve made it easier.” And although connectivity is not as much of an issue as it has been in the past, Participant 8 has learned from earlier mistakes and is offering a solution, noting, “we’ve taught the kids really well how to reconnect their Sphero’s. That’s the key, because they also have trouble finding theirs. Teaching them the appropriate space to stay away from your robot is important.” Teaching students how to reconnect their own devices is important because it empowers them to become more self-sufficient and confident in managing their technology. This skill is essential in today’s digital age, where connectivity and device management are integral to both learning and everyday life.

Youth Coding Challenges often present unique academic challenges, as early learners often struggle with reading comprehension and vocabulary essential to understanding programming instructions. Participant 2 highlighted one of the main academic struggles mentioned by participants 3, 4, and 6 when they said, “Lack of reading...a lot of times they struggle with looking for the aim button, they don’t know what aim looks like, nor can they read it.” This shows a challenge teachers experience when implementing Sphero robot activities. Participant 6 emphasized this concern, stating, “the hardest thing for me is working with little itty bitties, because when I work in kindergarten...my superpowers have never been pre-readers.” As always, teachers find a way to prevail and overcome challenges. Participants 2, 3, 4, 6, and 7 all mentioned how important it is to scaffold and support students during the implementation process by

using websites like code.org or heavily incorporating visuals. Participant 4 mentioned, “I’m not saying no kid can just get in there and play around. But the majority need to be taught certain things, there’s certain vocabulary and that they need to understand, to even get the block program to work.” And with block programming being a goal of some educators when using Sphero robots, it highlights their focus on simplifying coding concepts for students. Participant 6 supported the idea that a child can code when saying, “they are totally capable of drag and drop.” This approach aims to make programming more accessible and engaging for beginners, fostering a better understanding of computational logic through a visual and hands-on method. Participants’ insights demonstrated that with proper scaffolding, visual aids, and a supportive approach, young students can develop foundational coding skills regardless of initial literacy levels.

Category 2: Advocating for Meaningful Technology Integration

The second category for this theme was Advocacy for Meaningful Technology Integration, starting with a commitment to equity, leadership, and reflective practice; empowering participants to ensure that technology enhances rather than obstructs authentic, accessible learning. Participants strongly felt that students should have access to coding and robotics in grades as low as Kindergarten through Grade 2. Participants 2, 3, 4, and 7 supported the idea of teaching coding and robotics in early grades, as they observed that “tablets and phones have been placed in the hands of babies for years, giving them access to technology at a young age. Some toddlers are capable of using a device better than some adults.” With this advanced knowledge of technology, students can grasp computational concepts quickly. Participant 7 looked towards the future,

stating, “I have to be realistic and think about children who might be able to understand coding and get into a summer camp, maybe at Georgia Tech...they’re more than capable of that, they’re exposed to so much at a young age.” This type of thinking among participants is what takes classroom lessons from translational to transcendent.

During the interview process, participants were asked, “What would you say to someone who feels like K–5 students are too young to learn using robots?” The responses from the participants were unanimous, filled with giggles and sarcasm, with comments like “Sure. Wake up and smell the coffee. You have to change at times. And if I’m not doing what’s best for the future, I’m not doing my job, and I’m not improving my job,” expressed Participant 7. Participants agreed that the use of Sphero and robotics allows students to tap into areas of education that they may have success with. Participant 8 pointed out, “kids who are not good at sports, or they may not be your top dog academically, but they like robots, and they really want to learn robotics, and they become good at it. I’ve had strugglers become good robotic coders,” while participants 5, 6, 7, and 8 emphasized the importance of introducing students to robotics early to plant seeds for the future, ensuring they are not negatively affected. Participant 5 indicated, “Students go into middle school or high school, and they see computer science as a course...if they don’t know what that is, they might gloss over it, and it’s something that they could be really, really successful at.” Overall, providing equitable exposure to robotics at a young age proved to be the most effective approach according to the participants.

Equity for participants does not just end at advocating for students to have opportunities to learn coding at an early age; it continues with participants ensuring that students have access to technology for their learning. Unfortunately, the price of Sphero seems to be a common theme amongst participants; Participant 5 confirmed, stating, “It’s expensive. They’re costly. If you don’t have Title I funding or direct funding for your school, then you don’t have robots. Being realistic, considering you’re spending \$2,500 on 15 robots that might only last two years, is important.” But when you believe in something, you make a way. Participant 5 felt so strongly about students having access to Sphero in their district that they made it happen themselves, “No budgeting. I fully funded my robots. I was on eBay buying every robot I could to take to schools in my district because it’s a Title I district. So, there’s a high level of need.” Where Participant 1 chose to participate in a donor choose option, they recounted, “I thought they were really cool and bought one and then did some Donor’s Choose to get a few for my classroom.” The difference between Participant 5 and Participant 7 demonstrated that not all Title I schools have access to funding to provide robotics, and that this depends on how the administration allocates funds. Even though all participants have access to Sphero robots, participants also proved that exposure can’t occur without the necessary tools.

Having access to Sphero is extremely important, but what is done with the Sphero is just as important, which is why the focus on Technology Equity and Appropriateness cannot be overlooked. Sphero Bolts are widely used amongst participants 2, 3, 5, 6, 7, and 8, and Participants 1 and 4 revealed they do not like using the Sphero Bolt with

students younger than third grade. Participants 4 commented, “I don’t usually use the Sphero’s with younger kids. I typically don’t, because I have other robots that are more age-appropriate.” When asked what robots were more appropriate than the Sphero Bolt, Participants 1 and 4 explained the use of other robots. Participant 1 shared “I like Code and Go Mouse, Dash robot, Bee Bots, and Indie robots, for younger students.”

A fan favorite amongst all participants is the new Indie robot by Sphero.

Participant 4 provided an overview of the Indie robot for context: “They’re like cars ... I think they’re easier to use because they’re for younger students. Indie robots come with the color mats. So, it’s easier for them to see. Okay, this color is telling it to do this.”

Participant 8 explained the age appropriateness of the Indies in this example: “I do a lot with the Indies because of the way the Indies are, and that each square does something. So, if they have a little obstacle course, it’s very simple for K-1 students.” Even though the simplicity of the Indie makes it more desirable for the participants to use with younger students, participants 4 and 5 explained why Indie robots are used in some cases with older students. Participant 4 stated “Since I am in a low socioeconomic ESOL [English of Speakers of Other Languages] school, our kids, on average, fall below grade level. So, it’s easier to use things for younger kids that even 5th graders can use to help them with understanding.” Participants’ understanding of age-appropriate robots and students’ academic needs is essential for designing effective educational programs that leverage technology to enhance learning outcomes.

Participants take advocating seriously by not only advocating for access, but also encouraging administrators and other teachers to see the benefits of using Sphero in the

classroom with students. Participants 5, 7, and 8 all work with their administration and other teachers to demonstrate how to use Sphero with students. Participant 5 commented, “The hardest thing is, every school has a different administration. They determine what’s implemented. If your administration is looking at you as the expert, they’re going to ask you, How can we incorporate these into the gen[eral] ed[ucation] classroom?”

Participants also urged early exposure to Sphero and computational thinking with new teachers. Participant 8 explained that they take the time to work with new teachers with 1 or 2 years of experience and pointed out, “They don’t know how to manage them. It’s like giving them a new piece of anything that’s foreign. You rarely see students leave college knowing how to integrate any type of coding or technology into a classroom.” The advocating continues with Participant 7 speaking at conferences and shared, “I’ve facilitated district-wide workshops over the years, local, in-house professional learning, and I’ve presented at State Conferences.” Participants who shared their experiences using Sphero at conferences had the opportunity to introduce Sphero to other stakeholders and potential policymakers. Participants 7 and 8 expressed the need for regular standards with Sphero. Participant 8 asserted, “There needs to be regular standards in place for all students. We teach them science standards. We teach them math standards. Why shouldn’t there be a computer science slash technology standard implemented regularly?” While Participant 5 declared in their state, “We have computer science standards, but there’s no explicit instruction for them at the elementary level. So, how do we expose that to students? Also, meeting the teacher’s needs, because there are assessments and other things that they’re worried about.” The lack of guidance on how to implement Sphero

makes it increasingly difficult for participants, but instead of quitting, they continue to advocate for students' success.

Participants shared their experiences with implementing Sphero robots, emphasizing their Professional Growth and Reflective Practices. Some topics mentioned by participants included how Sphero benefits students. Participant 7 asserted, "So, it's kind of like if we don't use the technology available to us, we're not preparing our kids for their future jobs...Honestly, in 10 years, I don't know what the careers will look like." Participants reflected on their teaching practices as well when Participant 8 declared, "Oh, this is like the hardest thing [teaching students to break down code]. I struggle with this one all the time." Whether it is the struggle to teach a certain skill or a common struggle of lack of time amongst educators, participants 2, 3, 5, and 7 shared their sentiments on time constraints. Participant 7 confirmed that time constraints do not help when teaching using Sphero, as they emphasized, "Because in a 45-minute lesson, you don't have a lot of time. You're looking at a 5-minute introduction, 5 to 7 minutes roughly to set up the challenge, and then the kids have to draw or do something else." These frustrations turn into unmet desires as participants 3, 5, and 8 considered their desire to have been taught robotics as children, Participant 8 emphasized, "I wish somebody had taught me when I was little. I've always been science-minded. I like robots and coding a whole lot more, and how they have changed teaching practices for educators." Learning from the participants' experiences can provide valuable insights into their perspectives, challenges, and successes. This process helps understand diverse

viewpoints, foster empathy, and improve future programs or interventions using Sphero robots.

There are several educational robots that can teach students to code, but most participants prefer Sphero robots for their ease of use, durability, and excitement that they bring to the classroom. Some participants were not very confident in using Sphero for anything beyond the draw-and-drive feature. Participant 4 shared an ah-ha moment when they reflected on “coding is more of the block feature, and I don’t delve into that area. We live in the draw and the drive. I learned that I don’t use Sphero for computation. I should not stifle my students’ experience with Sphero’s.” Participant 1 shared similar sentiments when they revealed, “I don’t have to be a perfect coder or programmer. I can teach them the basics. They will take it further. Some of them show me things, so...I don’t have to know. I don’t have to have all the answers.” Participant 6 went further and shared, “I think what I’ve learned the most...is to just trust the process. And I love this about teaching. I know enough to get them all started...But one thing I learned about myself is that I don’t have to know everything.” Additionally, Category 2 emphasized ongoing collaboration, ongoing professional development, and the importance of staying abreast of emerging technological advancements to sustain effective integration.

Theme 2: Student-Centered Learning to Enhance Computational Thinking Skills

Theme 2 was Student-Centered Learning to Enhance Computational Thinking Skills. This theme was about how teachers encourage student learning and student success by monitoring autonomy, engagement, and the transferability of computational thinking skills. This theme emphasizes students’ engagement and motivation while

learning coding and explores how these qualities influence their overall academic journey. Participants shared their personal experiences of how students persevered through various challenges, highlighting resilience and determination. Additionally, the discussion covers how technology has played a crucial role in enhancing students' learning experiences, making education more accessible, interactive, and effective. The theme aims to showcase participants' perspectives on the importance of fostering motivation and integrating technology to support student success across different learning environments.

Category 3: Powered by Curiosity

There is only one singular category for Theme 2. Category 3 is titled Powered by Curiosity and explores participants' experiences as they navigate using technology to enhance learning in their classrooms. This category also explores the similarities between Engagement Through Play and Robot-Driven Motivation in students as they use the Sphero robots in an academic setting. Furthermore, it provided participants with a space to examine how Embedding Content for Transfer and Autonomy as Evidence of Learning can enhance their understanding and application of computational thinking. Category 3 concludes with AI and Future-Ready Thinking, as participants share the benefits of preparing students for their future.

When participants discussed Technology as an Enhancement, they focused on how Sphero use enhances lessons without becoming another daunting task. Participants 2, 3, 6, 7, and 8 expressed the benefits that Sphero brings to classroom lessons. For example, Participant 2 believed "Sphero provided activities that may not have been

possible because you can't replicate coding and execution without a robot." Participant 7 provided examples of how they use technology to enhance learning when they reported, "I try to expose the kids to everything. I mean, we do green screen technology, movie-making, and script development. We do pretty much anything technology-related."

Participant 5 shared a lesson using multiple forms of technology with their Sphero when they described:

So, we put a GoPro on top of the Rover and filmed it as it went through the obstacle course students created. We brought it into some video editing software and added like film grain. And we added a voice-over that sounded like the Apollo 13 Mission. But that one was probably one of the best activities I've ever done.

This lesson is an example of how creative teachers can be when implementing Sphero robots. Participant 2 exclaimed, "The Sphero has given students that hands-on piece where they can see their code in action. It's more interesting to see it change color or speak to you than watch the little character bounce across the maze on a screen."

Participants 5, 6, 7, and 8 discussed the potential to enhance lessons with Sphero.

Participant 5 proclaimed, "Because of the Sphero, we have added activities to our curriculum that are more aligned to the Sphero. You can only do with the Sphero."

Participants has proven that technology as an enhancement bring value and the ability to strengthen instruction, promote engagement, and make learning more accessible and relevant to all students.

Combining engagement through play with robot-driven motivation demonstrates how interactive, hands-on learning experiences can boost student focus, persistence, and enthusiasm for solving problems. Participant 6 proclaimed, “I don’t get to pick what motivates students, and Sphero does just that.” Participants commented that student motivation made them relentless. Participant 5 stated, “They wanted to do it! It wasn’t like a forced engagement or anything like that—they were truly engaged. They wanted to, and they were so excited.” Participant 2 added to the proclamation of students being motivated by Sphero when they pointed out, “they failed many times, I don’t understand, but they don’t give up, they stick with it. It’s not like nobody loses it and gives up. They just want to figure it out and make it work, which is really cool.” Engagement through play is essential for fostering creativity, learning, and social skills in children. By incorporating playful activities into daily routines, participants created a motivating, enjoyable environment that fostered critical thinking and problem solving skills. Participants 3, 6, and 8 agreed that competition is healthy. “I think it’s them wanting to get to the end. I think it’s the competitive nature. I’m going to figure this out. I think students find it engaging” (Participant 8). Ultimately, combining play and robotics shows how intentional engagement can foster deeper understanding, intrinsic motivation, and lasting interest in computational thinking.

Embedding Content for Transfer while promoting Autonomy as Evidence of Learning enables learners to apply knowledge independently, demonstrating real understanding and meaningful learning. Participants 2, 4, 5, and 8 shared how students are able to gain skills in analyzing when using Sphero. Participant 8 explained, “Sphero

increases the students' ability to analyze steps to make it transferable into the classroom...I think that attention to detail. Once you get a kid to see that attention to detail, it follows them into other things." Participants 4 and 5 discussed ways students Embedded Content for Transfer while promoting Autonomy as Evidence of Learning.

Participant 8 affirmed:

If they can explain it to me, then they understand it. Or if another student is struggling, I'll ask, did you ask another student for help? When students help each other, I know they're understanding. When the other student walks away, they can achieve what they set out to do.

Participants 6 and 8 highlighted the independence students obtain when working with Sphero. Participant 6 conceptualized the idea when they commented, "They don't want my help, and they try to be nice about it. And task completion is always a sign of success for them." Ultimately, when students can transfer and apply knowledge on their own, their independence becomes the clearest sign of deep, lasting learning, and it is evident that they are transcendental learners.

As students become more engaged in transcendental learning, participants explained how the use of Sphero robots prepares them for the future through AI and Future-Ready Thinking. Participant 3 explained, "I think the AI features of Sphero are probably underutilized." This highlights the potential to further incorporate AI capabilities into the learning process, encouraging students to explore these tools more deeply and fully leverage their benefits for enhanced education and future readiness.

Participants 7 expressed:

Early exposure to AI will help students learn to use it properly rather than for completion, showing that it can help in every area when applied correctly. That would be something that can motivate other teachers to jump on the wagon and see about incorporating it into their classroom and being willing to learn.

Through hands-on activities with Sphero robots, Participants 2, 3, and 7 stated how students actively explore how AI and robotics can enhance collaboration, critical thinking, and innovative design — essential skills for thriving in a technology-driven future. Participants expressed through these experiences not only foster technical understanding but also inspire creativity and problem solving abilities, preparing students to navigate an increasingly automated world.

Theme 3: Planning Perfect Instructional Practices for Computational Thinking

Theme 3 was Planning Perfect Instructional Practices for Computational Thinking. In this theme, teachers facilitate students' progression from guided play to algorithmic thinking using developmentally appropriate and scaffolded instructional strategies. Theme 3 is quintessential to understanding how K–5 teachers use Sphero robots to teach computational thinking, as it strategically breaks down complex concepts into manageable, engaging activities that foster problem solving skills and technological literacy among young students. There are two categories that dive deeper into teachers' practices of using Sphero to teach computational thinking. Category 4 focused on the instructional design, while Category 5 focused on scaffolding practices.

Category 4: Strategic Instructional Design for Computational Thinking

Category 4, titled Strategic Instructional Design for Computational Thinking, is a comprehensive framework that includes five supporting codes. These supporting codes serve to enhance the instructional strategies, facilitate better understanding of computational concepts, and ensure an effective learning experience for students. The integration of these components aims to promote a deeper engagement with computational thinking skills through well-structured and strategic instructional methods.

To develop well-structured, strategic instructional methods, participants began by focusing on understanding through Flexible Planning and Standards Alignment. Participants 8 and 6 shared a glimpse of their planning process, while other participants presented lesson examples to showcase how their plans had been effective. Participant 8 mentioned, “So, a lot of planning is looking at our standards first and where ... it make sense to make it fit...Or where can we integrate Sphero best.” Participant 6 shared, “When I start the planning process, I look at the standards — science and computer science. For my planning process, I always like to integrate it with science.” It is important to note that participants not only use the Sphero to meet science and computer science standards but also integrate it with other subject areas. Participant 5 shared how they use the Sphero with English Language Arts (ELA):

So, in ELA, I’ve done acrostic poems. Students come up with an acrostic poem, which is like, let’s say, they write their name on a big piece of paper. And then they go to each letter and then have them say the word that’s associated with that letter of their name. Sphero stops, speaks what

it is, moves on to the next one, speaks what it is, moves on to the next one, and so forth.

All participants have integrated Sphero into their science lessons. Participants 1 and 3 shared how they use the Sphero with a model. Participant 1 recounted, “Well, the model would be like the cell model. They had to use it to go around the cell model and identify the parts of the cell,” where Participant 3 explained:

Constructive and destructive forces. I like having the kids create many models. I noticed that when they practice angles, speed, and distance with Sphero’s, some of those same concepts are applied to the creation of their models. We need to create this kind of slope, this kind of angle to run everything away. Our obstacle needs to be this tall to keep the surge of water from overcoming it. Those are things that are a standard based on which I’ve noticed: a lot of those same concepts have been applicable to programming.

The ability to integrate does not stop there, Participants 8 and 7 shared their experiences with planning and integrating Sphero with Mathematics and STEM. Participant 8 highlighted, “Also, just some great math activities are achieved...productive struggle will happen with those bigger concepts. It may not be as easy with an addition problem, but that productive struggle is huge.” The participants demonstrated their versatility in planning and applying standards when using Sphero robots in the classroom through various activities. The use of the Sphero can be impaired at times by the participant’s ability to integrate due to time constraints.

All participants have different integration frequencies with the Sphero robots. Participants 1 and 4 shared their integration frequency, focusing on how often they engage in collaborative activities. Participant 1 shared they implement Sphero robots “I’d say at least once a month now,” and Participant 4 mentioned “I would say I integrate Sphero’s twice a school year for grades 2 through 5. So roughly, 8 times a school year.” This limited disclosure helps to understand their level of participation but may omit details about the quality or specific nature of their contributions. Participant 2 provided more context on the integration process when they explained:

It depends on the topic. So, generally, I try to integrate them monthly, since I only see students once a week. So, we’re talking about a week, one week every month that I’m doing Sphero with a grade level at some capacity. That is my goal. But it really kind of just depends on what we’re learning.

The frequency of integration reported by participants provided a realistic and insightful view of how often they incorporate Sphero robots into their teaching. This information helps to understand the practical application and potential barriers to consistent usage.

Participants may not be able to use the Sphero instructionally as much as they would like, but they shared unique ways that they introduce Sphero’s to students when they use Exploration-Based Introduction. Participants 3, 5, 6, and 7 believe that Exploration-Based instruction is the best way to get students excited about learning. Participant 5 declared, “I give the Sphero to them, and I give them the opportunity to explore and see what it does. I stand back. Mostly because kids are intuitive, and they go

okay, we can figure it out.” Participants view exploration as essential for student-centered learning prior to actual instruction. Participant 7 explained:

And then the kids just play first, if it’s at the very beginning introduction. See what the robot can do. How can you change the light colors? What do these colors mean, depending on which robot they’re using? The Indies have the color-coded mats to explore. So, they have to explore and see how those work. And then, once the kids understand the basics, I’d introduce them to their actual STEM challenge.

Participants 1, 2, 4, and 8 adopted a different approach to Exploration-Based Introduction compared to the explained method by Participant 5:

It’s that exploration piece, and it seems kind of counterintuitive, because when you give them exploration time and free code, there aren’t any directions to be followed. However, when they have that time, it takes away some of the mystery, and then, when they are asked to follow explicit directions, they don’t have that desire to try something else, because they’ve already tried it. So now, when they go through, they’re a little bit more attentive to the directions given to them and are trying to complete the task.

Regardless of the rationale for using Exploration-Based Introductions, participants often find that these approaches can effectively engage their audience, foster curiosity, and set a compelling tone for subsequent content.

Effective STEM Integration Practices go beyond isolated subject instruction, fostering interdisciplinary learning environments where students collaborate to apply knowledge and solve complex, meaningful problems. Participants 1, 5, and 7 spoke of the positives of STEM Integration Practices. STEM Integration Practices with Sphero allowed participants to take learning to new levels of discovery. Participant 1 shared:

With the Sphero mini, I've done an activity with fourth graders where they had to build a paper bridge that holds weight. I've extended it so they now have to add ramps to their bridge, and those ramps have to hold enough weight for the Sphero Mini to get up and over...They have to get the Mini up the ramp, across the bridge, and down, and the bridge they build has to be made out of paper.

Participant 7 recounted, "I mean, obviously, they're doing a lot of collaboration number one. And then the critical thinking about how to get the robot to do what I want it to do is important to building problem solving skills."

Participant 7 further explained the benefit of STEM Integration Practices when they noted, "And the robots allow them to think computationally, as well as program and work as a group. The whole purpose of STEM is to collaborate and communicate, and critical thinking skills come along as well." STEM Integration Practices encourage students to think beyond traditional classroom boundaries, fostering creativity, problem solving skills, and an interdisciplinary understanding that prepares them for real-world challenges. Participant 8 shared an experience they had with students during a lesson and recounted:

We built this particular activity using Legos and built it on a piece of poster board. The kids got to build and have ownership of the city. And then, it was funny... they kind of made it hard. They wanted to see if they could make all the turns and twists to get from Point A to Point B in their city. So, I thought that was pretty good. They even tried to combine their cities with those of other classmates. They upped the ante on their coding without realizing it, which is what I liked about it.

Participants' activity examples showcased the collaboration and critical thinking skills involved in STEM Integration Practices, highlighting how these activities assist participants in teaching computational thinking by integrating core content with computer science.

In robotics education, Assessment Practices must account for creativity, collaboration, and iterative problem solving, recognizing that learning often occurs through experimentation and revision and is observed by participants. Participants 1 through 8 unanimously agree that their assessment method is observation. Participant 7 further explained, "Because I teach STEM, I do not offer a grade. In my class, all my assessments are formative. And it's me watching." Participants 1, 3, and 4 are also STEM teachers who do not use a typical grading system. Participant 4 proclaimed, "I normally assess based on the engagement. If the kids are engaged and completing the assignments, then I can say that's a win. Since I'm STEM, I don't have hardcore grades. So, I just go off anecdotal observation." Along with observation, engagement is important to participants 6 and 7 as well. Participant 7 asserted:

The engagement piece is huge. Are they following through, and are they successful in the parts that they were able to complete? Can they get through this part? So, that's really the main focus. It's that anecdotal piece, not that you just completed the task.

There are other ways to assess the students on tasks involving Sphero robots. Participant 8 shared, "I've used rubrics before," while Participants 2 and 3 stressed the importance of hearing how students explain their thinking and understanding. Participant 3 mentioned: "Honestly, just peeking over their shoulder. Hey, before you move that, let me look at your canvas. I want you to demonstrate understanding by interpreting this for me and predicting what will happen before you start this code." Participants talked about how effective assessment practices involve more than just measuring performance, it also includes ongoing feedback, reflection, and adaptation to foster meaningful learning outcomes.

Category 5: Scaffolding Computational Thinking Practices

Continuing with Theme 3, Planning Perfect Instructional Practices for Computational Thinking, Category 5 focuses on Scaffolding Computational Thinking Practices. There are six code names for Category 5. Category 5 highlights the literature presented by Angeli et al. (2016), guiding participants to provide insight into the application of abstraction, generalization, decomposition, algorithms, including sequence, and debugging within the classroom while using Sphero robots. Additionally, this category emphasizes the importance of integrating these concepts to enhance students' problem solving skills, creativity, and understanding of robotics technology. It is about

the design of activities that encourage critical thinking and hands-on learning with Sphero robots, ultimately strengthening the connection between core content and computer science.

Participants emphasized the importance of teaching students how to drive the Sphero robot before teaching them to code it, as explained in the Fundamental Driving Instruction. Participant 2 explained, “Understanding the fundamentals of how to aim the robot and how to make it go forward, go backward, those kinds of things are important when learning to drive the robot, which leads to coding essentials.” To reduce the chaos, Participant 7 provided a demonstration that not only sparked students’ interest but also prepared them for what is ahead. Participant 7 shared:

Sometimes I’ll play a short clip—maybe 30 to 45 seconds—of something cool the robot can do. Or I might just have a pre-made program I made that goes around the carpet, and they’re like watching it go around them without touching the robot, or just something fun to show them how simple it is to do. And that’s usually through my large screen in the classroom. If it’s the Indies, I’m just using the floor because I just use the color mats for those ones.

Although driving the robots is fun for students, participants shared the challenges they encounter and why Fundamental Driving Instructions are crucial to student success. Participants 3 and 5 spoke of the challenges with teaching students to drive the Sphero. One shared, “Aiming is always hard to teach; some students will naturally figure it out, but others will need reminders. Aiming is important because if it’s not done correctly, the

robot will go in the wrong direction, which affects the code” (Participant 5). Participants 2, 4, and 7 discussed how speed negatively impacted students’ success when driving. Participant 4 expressed, “The other thing is in convincing them that it doesn’t need to go full speed, because they can’t drive it well to begin with. I make them keep the middle speed, not the fastest.” Participants 7 and 8 explained that the speed is a negative factor because the Sphero is a robotic ball and will often “spin out” when going at a high rate of speed and they further explained “The biggest challenge with the mini or the bolts is students have to get used to how it works and the speed and turning. It doesn’t turn like they think it’s going to because it’s a ball (Participant 8).” Participants 2, 5, and 8 shared ways they help the students to understand the fundamentals of driving with Sphero robots. For example, Participant 4 shared:

Students try their best to use the Sphero’s and the blocks that I have to create a maze. They set up different cylinders in the maze, but their goal is to avoid knocking any of them down. And there is a math part, but it does get them to practice with the Sphero’s. It gets them engaged and excited, and it teaches them how to control the speed and drive the Sphero.

Other activities mentioned by participants included having students guide the Sphero in a straight line with a piece of tape on the floor (Participant 8) or having the students race the Sphero against each other (Participant 5). Regardless of the method used, participants highlighted the importance of students understanding the fundamentals of driving before progressing to more advanced levels of coding.

Foundational Computational Thinking Skills is where participants began introducing elements of computational thinking to students, which aligned nicely with Angeli et al. framework. Participants take different approaches to introducing computational thinking to students. Participant 4 created a real-world connection to help students understand coding by:

I start by explaining what coding means, since some of them don't know. I give them a real-world example. When you touch a phone and it turns on or you're playing a game on a phone, you're telling it what to do. So coding is a means of telling a robot what to do.

Participant 6 discussed their initial start to students' understanding computational thinking skills when they said, "I think the best activity I do is showing them block-style programming so they're familiar with it. It's easier for them when they understand drop-and-drag. I usually do that by using Code.org because it's scripted." Part of helping students understand computational thinking is helping them realize that problem solving is a big part of the process, expressed by Participants 3 and 8. Participant 8 shared:

I teach them that problem solving is okay. So, if we wanted to do a task, then I usually have to use computational thinking. And decide which pieces and what sequence I need to do to get it to do what I need it to do. So that computational thinking: I have a problem... here's how I think we can solve it. And then that trial-and-error of trying it and seeing what needs tweaking where.

Participant 6 demonstrated how working collaboratively can build foundational computational thinking skills when they shared how they talk to students problem solving in groups:

Listen, you're given these instructions. It's too much. So, here's what I want you to do. Cross out everything you don't understand and leave the rest. And let's start to solve that problem that way. If you know how to solve that problem, that's where you start. And when you're working in collaboration with somebody, see if they know the other pieces, you don't know. And then I want you to think about what you can do with the information you both know.

Participants 3, 8, and 6 all encouraged students to practice abstraction by having them decide what to keep and what to ignore. This approach also extends to other computational thinking skills, such as sequencing, decomposition, and debugging.

Participant 5 further supported the connection between computational thinking and problem solving when they declared, "That computational thinking piece goes a long way with any of the coding pieces that happen. But even when you're talking about the engineering design process. They have to go step by step, identify the problems, and fix them." Participant 5 provided a clear depiction of how sequencing and debugging are used in coding and computational thinking. With a strong foundation of computational thinking skills, students can move into the complexities of Algorithmic Foundations.

Algorithmic Foundations are essential to computational thinking, helping students build logical, step-by-step methods for solving complex problems. The approach

participants have taken to teach Algorithmic thinking may differ, but all point out that the sequence of steps students take is important. Participant 8 recognized the importance to also making the connection to the students that the robot cannot do an action on its own and must be controlled by a person when they stated:

It's almost teaching them how to do a procedure. You have to understand the robot won't go on its own. I think that's where kids, especially young ones, often struggle to understand. You have to tell it to do everything, and they get extremely frustrated when it doesn't do what they tell it to do, so they immediately blame the robot. It's understanding that you're giving them a set of directions. You gave them that set of directions. They [Sphero] have no brain. They're following your directions. I think that's where the first foundational piece is: if the robot is not doing what you want it to do, you have to look at that set of directions. And I think that's a really hard skill for kids to look at. And that skill, of whether or not they can break it down. Okay, I'm watching it. And I told it to do this. It's doing that. It's doing that. Oh, wait! I told it something the wrong way, that analyzing my data piece is really hard for them to do with coding and computational thinking.

There are several effective ways to help students make meaningful connections to what is being taught, thereby enhancing their understanding and engagement. For example, incorporating real-world applications, encouraging student-led discussions, and utilizing hands-on activities can make lessons more relatable and memorable, fostering a deeper

appreciation and enthusiasm for learning. Participant 2 used the terminology algorithm daily with students and explained:

So, giving them those ideas of an order is really where we start with algorithmic thinking. And then, understanding that a list of steps helps them do a task. On my classroom board, I have a list of the steps they have to go through to get to a site or wherever we're going that day. And it's called today's algorithm. So that word algorithm is in their language. We try to embed it into what they're doing. So that it's not such a big concept. By the time we get to it officially, I'm like, you guys have been doing algorithms all year. It's that list of steps on the board, and if you don't go in that order, you're not going to get to whatever the thing we're doing is.

Using real-world connections, as Participant 2 did, made learning algorithms more relatable to students and allowed them to explore more complex understandings of computational thinking. Participant 7 shared an activity where students used a sensor to complete a maze and stated, "Using the positioning sensor on the robot to verify that students actually made a square, if the robot rolls the blocks to be in the correct order. If they're not, then you're not going to make a square." Participants 2, 5, and 8 commented on how loops help to build algorithmic foundations. Participant 8 mentioned:

And I think that takes time for kids to understand. But I think it's thinking them through. Okay, can I repeat this in a loop? And can I make that algorithm tell you that you're going to go five spaces and turn 2 times to get where we need to go? Well, if I'm going to do it 2 times. What is that

really saying? Oh, that's a loop that I need to do twice. And that's how we get to that algorithm.

Teaching algorithmic foundations not only helps students turn abstract ideas into practical actions but also equips them with essential problem solving skills. When participants guided students to plan, organize, and carry out coding tasks with clarity and purpose, their approach fostered confidence and encouraged innovative thinking, preparing them for future challenges in technology.

Understanding Shapes develops students' ability to observe, classify, and manipulate forms in their environment, building the foundation for critical and creative thinking in mathematics and beyond. Participant 2 recounted an activity where students used the Sphero to learn mathematics in bowling and shared, "there's a lot of math concepts built in there as well as problem solving. They're collaborating. They're understanding that they have to work together if they're going to solve this problem." In speaking of shapes generally and increasing mathematical thinking, Participants 1, 2, 3, and 4 spoke about using Sphero to create geometric shapes. Participant 2 explained, "the mathematical thinking comes in with the angles, time, and distance relationships on how far Sphero will go. There are several lessons we do on drawing shapes with Sphero. The mathematical pieces are there; it's just a step-by-step follow-through." Participants 5 and 6 felt strongly about how logic plays a big part in mathematics in relation to the use of Sphero robots. Participant 6 proclaimed:

There are all kinds of mathematical reasoning happening when we use Sphero. I can't separate it. And that's part of what you know. So, when I

create experiences for kids, they have to do all the mathematical reasoning, a lot of the logic that goes with that, and I think it's kind of a byproduct. So, outside of like basic math stuff—angles, numbers, increasing and decreasing—just having students go through and give them that task. And then they have to go through and look. Asking them questions like, where are those errors? What is happening? Why is this happening? And it has to make sense to them as they are learning.

Using Sphero as a dynamic, engaging way to teach math concepts—such as geometry, measurement, and algebra—makes learning interactive and fun for students, as demonstrated by the participants. These key concepts taught by participants also helped students to apply solutions to different problems, enhancing their understanding and problem solving skills across various contexts. Integrating Understanding Shapes into robotics and coding activities enables students to connect geometry with movement, design, and problem solving.

Decomposition happens when participants encouraged students to break down problems into smaller more manageable pieces; where Visual Decomposition is the same concept but participants added a visual aid to help students further understanding. Participants 2, 3, 5, and 7 shared the importance of teaching decomposition to students. Participant 5 provided an explanation:

Too much information at once can be extremely overwhelming and lead to overload, which can cause students to feel overwhelmed. Teaching them to chunk large problems into smaller ones can help them feel success in

little pieces, which we'll then take all those little pieces of success and combine into the end goal of success.

When a task is daunting and difficult for a student, they may feel discouraged while learning. Participant 7 emphasized, "And that's why I have the pictures on the board, or I have sheets of paper that they can access that give them a model to offer visual aid to be successful." Participant 5 uses a more scaffolded approach to visual decomposition and stated:

When it becomes a bit too complicated, I think I need to give more explicit instructions. So, I'll put the code on the screen. We have a touchscreen, but I'll start a new line. And we can drag parts of the code off, leaving the rest in place, then run it to see where it is. Okay, that works. Then, we can drag another piece back on. We can add the parts we've already written or that exist, one at a time, to identify where the error is happening.

Participants 1 and 3 took a different approach using task cards to assist students with visual decomposition. Participant 3 noted:

I'll give out task cards to the kids, go around, and see if they do any correctly. Then just go from one card to the next. I really encourage them to build it in small sections at a time, test it out, keep running it, make sure it's good, then add on, because if they build the whole program at once, there's more room for error. So, I really encourage them to do just a little bit at a time and test it out.

Within Sphero, visual decomposition allowed participants to translate visual information into logical sequences, bridging the gap between conceptual planning and practical execution for their students. The participants not only tackled decomposition but also provided examples of sequencing, abstraction, and debugging, which are common skills needed when applying Real World Integration.

Through Real-World Integration, participants designed lessons that link academic concepts to everyday problems, fostering relevance, engagement, critical thinking, and practical problem solving skills. They encouraged students to apply their knowledge to real-life situations, making learning more meaningful and applicable to everyday life. An example of such a lesson was presented by Participants 6 and 7, who described their lessons on oil spills in the ocean. Participant 7 recounted:

The real-world problems I've connected Sphero to have been more about the robot's movement. And the oil spill challenge directly connects to how to clean up an oil spill after it happens. What is the process going to look like? Obviously, I mean to make it relatable to the kids, learning about different materials and things they already know. But using the robot, you have to be creative. If I'm going to connect it to real content, I have to be creative about how I use that approach.

The creativity increased as Participants 2 and 5 creatively connected the use of Sphero to real-world holiday celebrations, making the activity more engaging and relevant.

Participant 2 used Sphero robots and created "Balloons Over Broadway which mimicked the Macy's Thanksgiving Day Parade," and Participant 5 had students use Sphero to

“design a vehicle that could move around Whoville and collect different holiday-themed items.” And what if learning and using robotics could aid in saving a life which was evaluated by Participant 6 as they explained the thinking after the community witnessed a brush fire and stated:

Two years ago, we had a fire evacuation. It burned all the way up to the school. There are people out there trying to keep the fire from the school, to prevent it from burning. When the kids walk by this whole windowed area, you can see the burn’s border, and we talked about there being hotspots around for months and months — like we would drive and you would still see hotspots, and you’d see the ground smoldering. It was crazy. And we talked about how robots might be able to manage that situation when we don’t want to use human life. So, when all kinds of things come up in our area, we talk about whether robots would be a good use of our time. Some things humans can do, or robots can complete.

Teaching students that their knowledge and use of robotics could save others’ lives or significantly improve safety in critical situations inspired a sense of responsibility and purpose when participants discussed real-world integration. Participants captured how, with a little creativity and lesson design, Real-world integration can transform education into an active, experiential process, empowering students to see learning as a tool for understanding and shaping the world around them.

Theme 4: Exploring Failure

Theme 4 was Exploring Failure. This theme describes how teachers create a classroom environment in which failure, reflection, and interaction are integral to learning with Sphero, employing scaffolded, exploratory experiences that prioritize application, adaptability, and collaborative problem solving over perfection. Learning through failure involves seeing setbacks as valuable opportunities for growth. It promotes resilience, adaptability, and a better understanding of one's skills and limits. By reflecting on failures, individuals can improve their strategies and ultimately succeed.

Category 6: Crash, Learn, and Repeat

Theme 4 only has one category. Category 6, titled Crash, Learn, and Repeat, describes not only on students' failures and setbacks but also on participants' failures and setbacks. Category 6 encourages participants to push through challenges and embrace them with vitality. The seven codes of Category 6 represent specific criteria used in a classroom environment. These codes are essential for understanding computational thinking, managing chaos, and the discovery associated with Category 6.

The concept of the Teacher as a Co-Learner highlights a shift from traditional instruction to collaborative inquiry, where participants model curiosity, adaptability, and continuous learning alongside their students. Participant 8 shared an experience with students who needed assistance and was transparent with them when they said, "There were a lot of questions, and they often looked to me, and they're like, well, why is it...I'm like, I don't know. So, it was a lot of us learning together." When participants and students learn together, it significantly transforms the classroom dynamics, fostering

a more collaborative and engaging learning environment. Participants 2 and 3 shared similar sentiments about learning alongside students. Participant 2 said:

They're going to teach you a whole lot more than you realize. And you're going to learn a whole lot more than you thought...So, rather than seeing yourself as the one who just delivers the knowledge to them, be willing to let them give it back.

Participant 2 also provided insight on being a Teacher as a Co-Learner when they mentioned, "We're learning to figure it out together. I didn't grow up coding in elementary school. I'm learning all this along with them. Sometimes they know more than I do. I'm like, tell us more. I don't pretend to know everything." Participants that spoke about students teaching them, shared the confidence it gives students to know they too can teach. Participants modeled lifelong learning, showing students that growth, reflection, and experimentation are shared parts of the educational journey, and that learning then extends beyond coding and social-emotional learning.

Beyond Coding and SEL Integration broadens education's focus from just technical skills to overall development, fostering interpersonal and emotional skills vital for lifelong learning and teamwork. Participant 2 shared how coding and SEL have become intertwined in learning algorithms when they reported, "We incorporate Sphero into all of our curriculum. We integrate it into our digital citizenship units and back-to-school lessons. Whether it's the rules or responsibilities, we make them part of our regular activities, not just outside of algorithmic thinking." Participants shared that one of the tasks of teaching coding is managing students' frustration with difficult codes, which

is a major part of Beyond Coding and SEL Integration. Participants 2, 5, 6, and 8 shared the same idea when deciding how to intervene with students struggling to complete a code. Participant 6 explained, “I don’t help unless they ask. I don’t alleviate frustrations; they have to learn to work through them. It’s important for everybody. Hell, I wish we had adults who could do it...I wish people knew how to reason effectively.” Having a strong feeling on the topic of productive struggle, Participant 8 shared insight and said:

I will let them struggle, then support them, then let them struggle some more, and I might give them more support. Now, if there are tears, I’ll jump in and help. But they have to learn productive struggles. They just have to, especially when it comes to coding. It is, it is not the most fun.

But they learn from it. Yeah, we tend to jump in too quickly as educators.

Beyond Coding and SEL integration focuses on the whole child, not just the computational task at hand. Participants reported having to understand the difference between frustration and learning. Participant 6 shared, “If you’re frustrated, it means you’re learning something. If you’re not frustrated, you’re probably not learning. You’re probably just going through some motion. Do I let kids get so frustrated that they cry? No, that’s the point.” When participants are equipped with the skills to manage these emotions, they encourage more meaningful educational experiences. Participant 2 provided strategies they use with students in those moments when they stated, “they do get frustrated, and usually they’ll sometimes take a breath and look at it again. I ask, did you ask a neighbor? Use your peers if you need to. Ask me, and we can go step by step.” Participant 6 also believed in using peer collaboration when students are struggling and

provided an example, stating, “Let’s just stop and talk through this. And then I bring a bunch of kids together to talk through it again. They do better talking to kids their age than they do to me. And that’s okay.” Building a healthy relationship with frustration during coding involves adopting a positive mindset, practicing patience, learning from mistakes, and maintaining perseverance to overcome challenges effectively. Participant 7 does this by ensuring that no student leaves frustrated. They commented, “But I try my best to make it so that they can be successful in one lesson. They don’t leave frustrated.” Since participants have demonstrated knowledge and success in integrating SEL and coding, they can prepare for the typical challenges they encounter with Structured Chaos.

Learning with Sphero robots is an exciting and engaging activity for students; participants shared effective classroom management strategies through the code name Structured Chaos, which helps participants maintain an organized yet dynamic learning environment. Participant 7 maintained structure during an exciting lesson by letting students know their expectations from the beginning and recognized “it’s exciting and trying to remind students that, yeah, we’re doing this. And it’s a lot of fun. But there’s also a task that has to be done. We need order.” Maintaining structure during lessons is a common theme among Participants 2, 3, 5, and 7, highlighting the importance of organized and clear instructional strategies to enhance learning outcomes. Participant 5 provided an example stating “Repetition, follow through, letting students know, and just reiterating like, how important it is that they do what I ask them to do. Because we have a goal, and it’s something that we’re trying to accomplish.” Participant 2 provided reassurance that with structured chaos, students are on task and proclaimed, “There will

be chaos. It's fine, I promise. They'll generally be on topic. It's going to be loud. It's going to be a lot of excitement. And that's how you know they are working, even though everybody's doing different things."

Having standard operating rules while the robots are in use is a must for participants. Participants 2, 3, and 7 are firm in their rules that robots are not to be touched when being operated. Participant 2 demanded "No robots are to be picked up once they are in operation; students have to learn how to maneuver them to return them without picking them up. No hands allowed, like with soccer." Participant 4 spoke of the issues with finding the robots and stated, "I've taught them how to find the robot with the Find My Robot feature. When I know I'm going to use Sphero's, I rearrange the room so that it's kind of impossible for one of them to get lost." Participant 2 shared a way to prevent the Sphero from getting lost. Participant 2 stated, "We set up big PVC pipes, a play area, which we call our playpen, and the Sphero's stay in the playpen." Participants have already discussed pricing, so having a designated space for Sphero use prevents them from getting lost during instruction. Participant 4 also commented that device management includes the speed and care of the robot, as well, stating, "Controlling them [the kids], making sure they don't turn the speed all the way up. Making sure they don't drop the Sphero's or kick them. It's more about taking care of the mechanical part once they get into using it."

In addition to device management, participants discussed with Structured Chaos that working with this age group can be difficult when sharing is involved; Participants 3, 5, and 7 all reported issues with students sharing. Participant 5 stated, "So sometimes

students don't want to share. So, you have, you have to find ways to get creative with that. They're all excited. They want to do the coding, and they want to miss their turn." Once students have violated the rules, participants must determine how to discipline them in the moment. Participant 7 shared an experience with managing discipline when using Sphero's during a lesson:

There have been some issues where students haven't been listening. And so, removal from the activity or from the opportunity to use them, is has been an option. I mean, it's few and far between. But it has happened, and it kind of sets an example for the others by saying, Hey, we have to take this seriously.

Participants 2 and 3 shared that, with structure, they can really see the learning taking place among the students, and that engagement in the Sphero activity decreased behavior problems. Participant 3 mentioned, "I absolutely had everybody engaged regardless of what their reputation may be as far as behavior goes". Participant 2 emphasized, "I know they're engaged because I'm not having any kind of behavior issues. Everybody is...everyone is excited to have them. They're excited to see what they've created come to life with the robot. There are very few behavioral issues." Participant 3 added to the connection between engagement and behavior when they declared, "I didn't see anyone walking around or doing what they're not supposed to do. Some of them noticed that someone was doing something incorrectly and said, "No, you need to do it this way." A lot of collaboration was happening." When behavior was

maintained and students were engaged, participants were able to focus more on what students need at the instructional level.

By incorporating Scaffolding for Computational Thinking, participants guided learners through the cognitive processes of decomposition, pattern recognition, and algorithmic design, enabling deeper conceptual understanding. Participants 6 and 8 explained the importance of scaffolding when learning computational thinking and robots. Participant 8 explained, “I model because they want to do the whole code in one shot. And the best thing I do is model —modeling how I code to help them understand. I am just there to scaffold what they know.” As participants scaffold instruction, they create lessons in which they gradually release students to support success. Participant 2 shared a Simon Says lesson that provided scaffolding instruction, with the intention of releasing the students to practice what they learned. Participant 2 stated:

The Simon Says lesson gives students a huge code. It builds a huge piece of the code for them. And then to start, they just kind of alter that code.

We go in and look at the code together, and I generally either project it on the board or give them screenshots of just those pieces of code. We kind of talk about the colors. And what do they look like? And what do they notice? And what do you think this is going to do? We kind of break down each little chunk of code so they can see what it might do, and then we play it out so they can see how this piece of code makes this thing happen.

And then, we work so it’s kind of like the “I do, You do” model. We do like a gradual release. The first chunk is me kind of modeling the basics

for them, and then giving them time to play, fail, and try. And then, once they've understood how it works, I give them the task. Letting them understand how all those tools work before I expect them to do something with them.

When working with younger students, participants stated they need to provide more scaffolding to support them. Participant 6 explained, "With kindergartners and first graders, I scaffold a lot more. They do all of this on their iPads. I provide additional scaffolding, including code examples and guidance on how to work with it." Sometimes the participants needed a little scaffolding from the students to understand their thinking. Participant 3 said, "It's really a reactivation of prior knowledge. Showing me that they do remember it. Asking them to break it down and scaffold it for me because they may not necessarily be able to explain where they need help." Participants have shown that supporting students throughout the coding process is essential to their success; however, students can succeed at any time, which is why it is important to teach them to learn from their failures.

Failing Forward is a shared journey for participants and students because it encompasses growth, learning from mistakes, resilience, and continuous improvement. It emphasizes that failure is not the end but a steppingstone toward success, encouraging a supportive environment where everyone collaborates to develop these essential skills. There are various reasons why failures occur during coding. At times, there are complications on the participant's part. Participant 4 shared a lesson failure that was due to a lack of knowledge on their part with block-style coding and recalled "the degrees

getting the Sphero to turn. It was over their head. I needed more training. That feature cannot be something they explore, like the drive and draw features. I feel like the block feature needs to be explicitly taught.” Participant 2 recounted a failure that occurred during the Balloons Over Broadway activity, stating:

When we did balloons over Broadway, there was a lot of falling forward.

It’s the first time they’ve coded it. They’re like, this was easy, and then, as soon as they add their float to the Sphero, the weight and the drag of the float throws off their entire code.

How students respond to failure is extremely important. Participant 7 shared an experience when students encountered failure and recalled, “I think the beauty of it is that they don’t care if they fail. They’re watching a robot and having fun with it.” Participant 8 shared, “I think it’s teaching them that failure is okay.” Participants 3 and 6 also recounted the thinking process of students while coding. Participant 3 revealed, “Strategic thinking, honestly. I feel like every kid at that point was putting their chest and checker’s brainwork into figuring out that’s not the correct order and how to do it.” Participant 6 went on to expound on an event that happened when some students were presented with a difficult challenge:

I can tell you they’re critically thinking, they’re problem solving. They’re collaborating. They are thinking creatively. They’re trying to solve this problem. And here’s my favorite. Everybody’s in. And they listen to everybody, like everybody’s involved. They’re considering ideas, trying everything. And my favorite part is when it doesn’t work — they don’t

just shut down and say, ‘Oh, yeah, we failed.’ Instead, they keep working on making it succeed.

It’s in the moments depicted by participants that the true value of learning from failures becomes apparent, highlighting how these experiences foster growth, resilience, and innovative thinking. The connection between computational thinking and failing forward highlights a mindset where setbacks are integrated into the learning process. By applying computational thinking to failures, individuals can analyze what went wrong, adjust their approach, and ultimately advance their skills more effectively.

Debugging as discovery encourages students to experiment, test, and analyze their code, cultivating persistence and curiosity through hands-on investigation. Participant 5 expressed, “I want them to have that discovery. I don’t want to tell them what to do or how things work, or how to solve problems. And that’s just from my experiences with the success I’ve seen in students while coding.” Participant 6 allowed students to use discovery as debugging because “Problem solving is problem solving. You try this. And it didn’t work. Try that. No. Then you start trying to find the best solution. It’s debugging. It doesn’t matter what you call it; it’s problem solving. We problem-solve daily.” Part of Angeli’s Computational Thinking Framework is the element of decomposition, which breaks down larger problems into more manageable ones. All participants encourage this skill when helping students understand coding. To help students become more successful coders, all participants model using small codes, which is recommended that students start with a smaller code when they start the discovery process, stating, “I code in small bits to make sure that it is working. That is not a skill

students get easily in elementary school.” Participants 4 and 5 expressed concerns about decomposition and debugging as a form of discovery. Participant 5 recalled, “But it just came to a point where I just saw that they weren’t successful, there was no production. There was a struggle. Once I broke it down into smaller chunks, they could see the problem.” Participant 4 stated:

Stop and see where you are, because if you start from beginning to end, thinking you’re about to code this perfectly, you’re going to mess up along the way, and then you’re going to be more frustrated. So, I encourage them not to start with a huge code. Start with one code press, start. See where you are.

Participant 8 had a unique perspective of comparing debugging to how students solve problems in math when they noted:

Consider this: most kids don’t check their math problems either. They assume they got the answer right, but when they don’t...Or they get a question like, Did this? You know, we see them on standardized tests all the time. Did this student take the right steps? They don’t realize that analyzing my steps is so important and challenging to teach. Discovery helps with analyzing. Analyzing is part of debugging.

Participants 2, 5, 6, and 8 explained how Sphero itself is able to help students recognize a problem in a code. Participant 2 proclaimed, “Sphero informs them. If the codes are not in the correct order, it will not behave as expected. Someone hacked my robot is a common comment. I doubt that...it will do exactly what you told it to do.” Participants

have demonstrated that decomposition is the most effective strategy for students to use when debugging, and discovery is the best approach to help them identify problems and create solutions, whether in life or academically.

When participants reflected on their instructional practices and activities, there was a notable lack of strong focus on the language being used, which led to the development of Application over Definition as a code. Participants have shown that they may not know the general terms presented by Angeli (2016), but examples of how those terms are applied have been found throughout this study. Participants emphasized the importance of understanding the application of the computational thinking element versus just recognizing the actual word. Participant 6 declared, “Oh, easy, peasy...algorithms are simply just what instructions you have to follow. Because it’s just a word. And you either know the definition, or you don’t. And this is not a word test.” Participant 5 had a differing perspective and encouraged the use of the word ‘interchangeable’ in an interchangeable manner to ensure students understood the definition through making connections. Participant 5 declared:

There’s no need for a direct explanation like, ‘Okay, this is what debugging is,’ or ‘This is what you’re doing.’ It simply happens naturally. I’ll sometimes swap the words ‘debugging’ with ‘fixing code’ or ‘solving the problem,’ and I’ll flip between those terms. This helps them start making the connection.

Overall, participants 2, 3, 5, 6, and 8 stated that they are not necessarily worried about whether students know the technical term of the skill they are learning; the application of what students are working on is most important and Participant 2 proclaimed:

They can explain to you what an algorithm is; they just might not have connected it to that vocabulary work. Yet we use the vocab. I'm more interested in understanding the concept behind it than matching vocab words. So yes, we'll say debugging. However, if they can implement a strategy to solve a problem. I'm happy, whether you call it the right thing or not. That's fine.

Participants acknowledged the importance of understanding the underlying concept being taught, rather than merely memorizing the definition. This emphasized the value of deep comprehension over superficial knowledge. Participants recognized that true learning involves connecting ideas and applying knowledge in practical situations, which ultimately leads to better retention and mastery of the subject matter. No discrepant data were found during this study.

Summary

Based on data analysis, themes emerged that were used to answer the study's RQs. There were two themes or key findings for RQ 1. The shortened version of Theme 1 was From Tinkering to Transformational. This key finding was that teachers overcome technical, developmental, and curricular obstacles and act as instructional advocates who facilitate equitable, developmentally appropriate instruction to K–5 students. Theme 2 was Student-Centered Learning to Enhance Computational Thinking Skills. The finding

was that teachers encourage student learning and student success by monitoring autonomy, engagement, and the transferability of computational thinking skills. There were also two themes for RQ 2. Theme 3 was Planning Perfect Instructional Practices for Computational Thinking. This key finding was that teachers facilitate students' progression from guided play to algorithmic thinking using developmentally appropriate and scaffolded instructional strategies. Theme 4 was Exploring Failure. This key finding was related to how teachers create a classroom environment in which failure, reflection, and interaction are integral to learning with Sphero, employing scaffolded, exploratory experiences that prioritize application, adaptability, and collaborative problem solving over perfection.

In relation to the conceptual framework, the results showed that the use of Magana's T3 Framework and Angeli's Framework of Computational Thinking served to enhance educational strategies across various disciplines. The lesson examples provided by participants demonstrated the different levels of learning using technology, as expressed by Magana's T3 Framework. Although participants did not necessarily use the same language Angeli et al. presented, they described what they taught and why, and it was clear that they were teaching computational thinking elements and discussed them intelligently. This approach also promoted innovative problem solving skills, encouraged critical thinking, and fostered a deeper understanding of complex concepts. By integrating these frameworks, participants created more engaging and effective learning environments, ultimately preparing students to succeed in an increasingly interconnected

and technology-driven world. Chapter 5 will include the interpretation of the findings, limitations of study, recommendations, and implications.

Chapter 5: Discussion, Conclusions, and Recommendations

The purpose of this basic qualitative study was to explore K–5 teachers' experiences of implementing Sphero robots to teach computational thinking. In this qualitative study, I applied the basic qualitative inquiry design to explore how K–5 teachers used robotics to teach computational thinking. I selected eight participants who worked in the K–5 education setting. I used a purposeful sampling and recruitment strategy. Interviews were my primary source of data collection for this qualitative study. I conducted all interviews virtually using Zoom and asked questions that I created, aligning with the two RQs. The findings may support public school administrators in assisting in-service educators with the professional development needed to offer support in using robotics to teach computational thinking.

There were four key findings associated with this qualitative study. The first key finding was that teachers overcome technical, developmental, and curricular obstacles and act as instructional advocates who facilitate equitable, developmentally appropriate instruction to K–5 students. The second key finding was that teachers encourage student learning and student success by monitoring autonomy, engagement, and the transferability of computational thinking skills. The third findings was teachers facilitate students' progression from guided play to algorithmic thinking using developmentally appropriate and scaffolded instructional strategies. The final finding of this qualitative study was that teachers create a classroom environment in which failure, reflection, and interaction are integral to learning with Sphero, and teachers employ scaffolded,

exploratory experiences that prioritize application, adaptability, and collaborative problem solving over perfection.

Interpretation of the Findings

The exploration of K–5 teachers' experiences with implementing Sphero robots was viewed through two frameworks. The first was Magana's T3 framework for innovation, and the second was Angeli's K–6 computational thinking framework. In Chapter 5, I include detailed descriptions, as outlined in Chapter 4, along with the interpretations of the findings. Although there was limited research addressing how K–5 teachers use Sphero robots to teach computational thinking, substantial studies existed on coding and computational thinking (Ghani et al., 2022; Herro et al., 2022; Munn, 2021; Rich et al., 2019; Shin et al., 2021) and on the use of other education robots to teach computational thinking (Fessakis & Prantsoudi, 2019; Hershkovitz et al., 2023; Ling et al., 2017; Wu et al., 2020). Some of the findings from the current study confirm, disconfirm, or extend the findings from the literature. I interpret these results in relation to themes organized by RQ and the key findings for each.

Theme 1: From Tinkering to Transforming

The study's findings for tinkering to transforming revealed that teachers encountered several issues when implementing Sphero robots and computational thinking in the K–5 classroom setting. In a review of the literature aligned to this theme, I found the use of technology to enhance learning could focus on decreasing resource disparities, particularly related to socioeconomic status, and ensure that teachers receive adequate support through professional development and upgraded resource infrastructure with a

focus on pedagogical support (A. L. L. Tang et al., 2020; Bhakti et al., 2025; Kale et al., 2018; Kwon et al., 2021; Mouza et al., 2018; Rich et al., 2020; Tyler et al., 2022). Recent literature attributed the blame to teachers' perception of educational robots, which is considered the reason for the lack of integration (Ortega-Ruipérez & Lázaro, 2023; A. L. L. Tang et al., 2020).

The findings of the current study extend the literature by identifying some of the academic barriers that educators face when implementing educational robotics in the K–5 classroom. Although some literature explained the use of robotics to decrease academic barriers for students (Castro et al., 2023; da Silva et al., 2025; Noh & Lee, 2020), the current study extends the understanding of these barriers to include academic barriers students face with reading and mathematics. In this study, participants spoke about the use of educational robots with students from lower socioeconomic areas and those that have English for speakers of other languages challenges. Participants also spoke about the use of educational robots in Grades 3-5 because of students' ability to comprehend. Some articles focused on the use of educational robots with young students to demonstrate that they can code and learn computational thinking skills (Berson et al., 2025; Hall et al., 2022; Pelizzari et al., 2023; Relkin et al., 2021; Rus et al., 2024). My study confirms the literature by showcasing creative ways K–5 teachers use Sphero robots to teach computational thinking skills.

Theme 2: Student-Centered Learning to Enhance Computational Thinking Skills

A review of the literature on student-centered learning, aimed at improving computational thinking skills, reveals that current research emphasizes the development

of problem solving skills as a key, transferable life skill (Chevalier et al., 2020; Garcia-Mills et al., 2025; Peñalvo et al., 2016). Currently, it is known that the use of robotics encourages and motivates students during their learning of computational thinking (Cam & Kiliçer, 2022; Chen et al., 2023; Haoran et al., 2025). This is particularly important to my research because it emphasizes how incorporating robotics into educational methods can enhance student engagement and comprehension of computational concepts, while also introducing elements of gamification (Santiago et al., 2025).

A review of the literature on teachers' instructional practices of educational robots to increase students' motivation has shown that integrating robotics into the classroom not only enhances engagement but also fosters a more gamified perception of instructional strategies, leading to improved learning outcomes and autonomy of learning (Chen et al., 2023; Choi et al., 2024; Quintero et al., 2022; Yang et al., 2020). My study confirms this literature, as my participants mentioned the motivation of students using Sphero and how their desire to continue learning was driven by Sphero's game-like feel. The review of the literature on instructional practices of students engaged in independent learning with educational robots encompasses an in-depth exploration of various methodologies, outcomes, and best practices, with a particular emphasis on diverse instructional strategies and techniques (Al Hakim et al., 2023). My study disconfirms the literature as the participants of my study spoke solely on collaboration as an instructional practice amongst students as they interacted with Sphero in teams or in pairs.

Existing studies suggest that students tend to perform better when educational robots are integrated into classroom activities, supported by evidence of improved

learning outcomes (Wang et al., 2023). My study confirms this literature as my participants leveraged the power of curiosity and play using Sphero to improve engagement and motivation, as well as enhanced peer collaboration. These results further underline the positive impact of robotics on educational experiences. When looking toward the future, current research emphasizes the importance of implementing AI for future-ready thinking (Junruang & Kanjug, 2025; Mohammed & Mohamed, 2025). My study confirms the literature by demonstrating that teachers believe AI integration can significantly enhance problem solving skills, adapt to rapid technological changes, and prepare individuals for upcoming challenges. This alignment between existing research and my findings stresses the growing recognition of AI's crucial role in shaping the future of education and professional landscapes.

Theme 3: Planning Perfect Instructional Practices for Computational Thinking

A review of the literature aligned to my theme of Planning Perfect Instructional Practices for Computational Thinking, shows that the current literature suggests teachers lack knowledge on how to incorporate computational thinking into their classroom settings (Ghani et al., 2022). A review of the literature on retaining information in computational thinking reveals that the way educators deliver instruction can hinder students' success in learning computational thinking (Buck et al., 2010; Monteiro et al., 2021; Schneider & Gottlieb, 2021). The existing literature stated the application of a variety of instructional strategies like scaffolding, guided play, and the use of visuals (Critten et al, 2025; Jocius et al. 2021; Newley et al., 2018; Rehmat et al., 2020). Additionally, many studies have highlighted improved problem solving skills and greater

interest in STEM subjects as significant advantages (Herro et al., 2022; Hu et al., 2024; Luo et al., 2025). My study's findings highlight the success stories of K–5 teachers who have learned how to incorporate educational robots into their classrooms, focusing on strategic pedagogical approaches and purposeful scaffolding to teach computational thinking. My findings on the connection between STEM and computational thinking are confirmed by the current literature, which highlights the growing importance of integrating these areas for future educational and technological advancements.

The current literature emphasized the use of mathematics to teach computational thinking and logic (Cervera et al., 2020; Popa & Biclea, 2023; Rodríguez-Martínez et al., 2020; Zuod & Namukasa, 2023). The findings of this study confirmed the current literature through the constant connection to students solving problems while using Sphero, particularly emphasizing the role of math in enhancing problem solving skills and understanding. Current research investigates the use of educational robots to teach components of computational thinking, including the fundamentals of algorithms and coding (Munn, 2021; Sáez-López et al., 2019; Shen et al., 2022; Vieira & Velasquez, 2023). My study confirms the literature with the use of Sphero, demonstrating its effectiveness in engaging students and enhancing their understanding of core concepts. Additionally, my study findings show that integrating Sphero into the curriculum provides hands-on learning experiences that support collaborative problem solving and critical thinking skills, which are essential elements in developing computational thinking proficiency in learners.

Theme 4: Exploring Failure

A review of the literature on exploring failure emphasized the benefit for students to learn from their mistakes while learning computational thinking with Sphero. The existing literature emphasizes the advantages of productive struggle in helping students learn computational thinking (Lee & Lee, 2024; Ritter & Schlomske-Bodenstein, 2025). The findings of this study confirm the current literature, highlighting that teachers valued experiences that taught students to embrace failures as a natural part of the learning process. Additionally, the study suggests that incorporating hands-on coding activities with tools like Sphero can effectively foster resilience and a growth mindset among students, further supporting the integration of failure-informed strategies in educational practices. This study emphasizes that teachers do not know everything and often learn alongside students, making the learning process more genuine. The current literature confirms the idea that teachers can also be learners, highlighting the perspective that continuous learning not only enhances teachers' skills but also positively impacts student outcomes, fostering an adaptable and innovative learning environment (Capobianco et al., 2020).

The existing literature emphasizes the importance of scaffolding in enabling young learners to utilize educational robots and develop computational thinking skills (Chevalier et al., 2020; Hall & McCormick, 2022; Georgiou & Angeli, 2019; Wang et al., 2021). My study extends the current literature by providing a deeper understanding of how scaffolding functions as an effective instructional practice. It explores the various techniques used to implement scaffolding and examines its impact on educational

practices, thereby offering valuable insights for educators and researchers alike. The existing literature on debugging in educational robots is limited, but there is significant research supporting debugging as a problem solving component of computational thinking (Chen et al., 2025; Sun et al., 2024; Wong et al., 2024). My study extends the current literature by providing new insights into how debugging occurs in an educational setting, specifically through the use of Sphero robots. It explores the processes involved, the challenges faced by students, and the teaching strategies that facilitate effective debugging, thereby contributing to both educational research and practical applications in STEM education.

Limitations of the Study

Limitations can significantly alter the effectiveness of the study, potentially impacting the validity of the results and conclusions drawn. Recognizing and addressing these limitations is crucial for ensuring the reliability and applicability of research findings. The participant criterion can be a limitation for the research. Since my study focused on the specific use of Sphero robots in K–5 education, this was a limitation because not many teachers use Sphero robots in elementary schools. The interview duration is a constraint; I was limited to 60 minutes with each participant for my study. The limited previous research on my topic presents a limitation because it hinders comparison of results, the development of a strong theoretical basis, and the expansion of existing knowledge. This gap highlights the importance of further studies to deepen the understanding of the subject.

Recommendations

Recommendations for further research are based on study results and limitations of the study. The first recommendation is related to the need for more comprehensive research on the use of Sphero robots in the earlier grades in elementary education. Additional research is necessary to build on the challenges students face when coding and better understand how the use of Sphero robots can enhance students' success in computational thinking.

The second recommendation is related to the study findings on the instructional practices of teachers using Sphero robots. Participants spoke freely about scaffolding instruction, ways that they introduce Sphero to students, and ways that they integrate Sphero into other content areas, however, there is no formal trainings for teachers. Most of the participants trained themselves. Therefore, more research is needed on ways teachers can effectively integrate Sphero robots into classroom learning, in order to gain a deeper understanding of why Sphero is not widely used outside of STEM subjects.

The last recommendation is related to the limitations of this study. This study was conducted with eight participants in elementary education, who participated in virtual interviews. Therefore, additional studies should be done in school settings, where the researcher can observe firsthand how the teachers are implementing Sphero robots in lessons to examine the evidence supporting the conceptual framework. In addition, extending this research into a quantitative study, students could use the Sphero robot in a classroom setting for a full school year, compared to a class with similar students academically who do not have access to Sphero, to measure their academic success.

Implications

This study may contribute to positive social change in several ways. First, at the individual level, teachers could learn instructional strategies for using Sphero robots to make learning more engaging for young learners and developing their computational thinking skills that can be applied in other areas of life. There is also potential for change at the organizational level. This study has the potential to shift how administrators and stakeholders perceive educational robots, seeing them more as learning tools than just toys. This study may also advance knowledge in the field of educational technology by encouraging more exploration of educational robotics and computational thinking at the college level, helping future teachers become more comfortable before they enter the classroom.

The study may have significant conceptual implications, as Managa's T3 Framework and Angeli et al.'s computational thinking framework can serve as valuable guides for teachers in their implementation practices. These frameworks can provide detailed directions and benchmarks to support teachers in effectively integrating new technologies into their classrooms. Additionally, although observations are commonly used in education to support teachers, these frameworks can also serve as comprehensive rubrics for administrators to validate the effectiveness of technology integration, ensuring that educational technology is used to enhance learning outcomes while maintaining high standards of implementation.

Another contribution of this study to positive social change lies in its implications for improved professional practice, especially regarding professional development. The

research highlights a substantial gap in the existing literature and current educational practices related to the effective integration of educational robots and the cultivation of computational thinking skills. Administrators could use the results of this study to inform professional development, and resource allocation for using robots as educational resources. Addressing this gap through targeted professional development programs is essential for educators to effectively incorporate these technologies and pedagogies into their teaching, ultimately fostering a more technologically savvy and innovative learning environment. This advancement not only benefits educational institutions but also contributes to preparing students for the demands of the digital age, thereby promoting broader positive social change.

Conclusion

Educational robots and computational thinking have been overlooked in education for too long. It is time to raise awareness about the benefits of using educational robots, such as Sphero, beyond STEM classrooms, clubs, and after-school programs. Sphero offers a range of educational tools designed to improve learning across all age groups. The latest robot, the Indie, have teachers excited as it demonstrated effectiveness for learners of various ages and skill levels. Sphero can assist students in learning coding, from basic directional commands to more advanced levels, such as Python. These skills are increasingly important as the world grows more reliant on technology.

The study aimed to explore K–5 teachers' experiences of implementing Sphero robots to teach computational thinking. It highlighted how participants navigated challenges such as teaching young students, overcoming technology limitations, and

addressing gaps in their professional knowledge. Additionally, the study provided an in-depth look into the various ways teachers advocate for the educational opportunities of all students, including strategies to foster equity and engagement. It also explored how these experiences contribute to teachers' professional growth, confidence, and their evolving perspectives on integrating technology into the classroom. Overall, the research offers valuable insights into the practicalities and benefits of incorporating robotics into educational curricula as well as challenges.

Although critics say that using robots with young students are not worth the hassle (Papadakis, 2020), this study provided valuable insights into how this technology can be integrated effectively into K–5 education. The research provided an opportunity to share experiences from participants and lesson examples of how teachers use technology to enhance classroom lessons. Whether by engaging students through play or motivating them with robotics challenges, it is evident that teachers appreciate how Sphero activities fostered active learning. Through a variety of diverse activities, teachers felt that their students were able to transfer content knowledge and demonstrate their understanding, showcasing practical applications of their skills. Moreover, this approach not only supports immediate learning outcomes but also indicates students' preparedness for future technological advancements. Overall, the findings highlight the argument for robotics to enrich K–5 and encourage innovative teaching practices.

In this study, participants shared how they used Sphero to teach computational thinking by ensuring their planning was flexible and aligned with the standards. The frequency of integration did not matter because, between the exploration of Sphero and

STEM integration, participants felt that through assessment practices and real-world connections learning was taken place. The key findings highlight the strong scaffolding of computational thinking, starting with learning the fundamentals of driving and building on foundational computational skills. The process continues with algorithm foundations to ensure that students grasp the logic and mathematics involved in computational thinking, utilizing skills such as visual decomposition.

This study's findings provide compelling examples of embracing failure as a vital part of the learning process. In these scenarios, a teacher acts as a co-learner as they venture beyond basic coding elements in a classroom environment specifically optimized for thriving amid structured chaos. The study demonstrates that when lessons are carefully scaffolded, students are empowered to 'fail forward' during debugging processes, which enables them to focus more on applying concepts meaningfully rather than being overly concerned with perfect definitions or skills. Teachers felt that these approaches foster a growth mindset, promote deeper understanding, and prepare students for real-world problem solving challenges, creating a dynamic and engaging learning atmosphere.

References

- Akgunduz, D., & Mesutoglu, C. (2021). Science, technology, engineering, and mathematics education for industry 4.0 in technical and vocational high schools: Investigation of teacher professional development. *Science Education International*, 32(2), 172–181. <https://doi.org/10.33828/sei.v32.i2.11>
- Al Hakim, V. G., Yang, S.-H., Wang, J.-H., Chang, Y.-C., Lin, H.-H., & Chen, G.-D. (2023). A pet-like model for educational robots: Using interdependence theory to enhance learning and sustain long-term relationships. *2023 IEEE International Conference on Advanced Learning Technologies (ICALT)*, 100–104. <https://doi.org/10.1109/ICALT58122.2023.00035>
- Altaie, M. A., & Jawawi, D. N. A. (2021). Adaptive gamification framework to promote computational thinking in 8–13 year olds. *Journal of E-Learning & Knowledge Society*, 17(3), 89–100. <https://doi.org/10.20368/1971-8829/1135552>
- Amrein-Beardsley, A. (2022). Using standardized tests for educational accountability: The bad idea that keeps on giving nothing in return. *Journal of Policy Analysis and Management*, 41(4), 1226–1232. <https://doi.org/10.1002/pam.22539>
- Angeli, C., & Giannakos, M. (2020). Computational thinking education: Issues and challenges. *Computers in Human Behavior*, 105, Article 106185. <https://doi.org/10.1016/j.chb.2019.106185>
- Angeli, C., & Valanides, N. (2020). Developing young children’s computational thinking with educational robotics: An interaction effect between gender and scaffolding

strategy. *Computers in Human Behavior*, 105, Article 106217.

<https://doi.org/10.1016/j.chb.2019.03.018>

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K–6 computational thinking curriculum framework: Implications for teacher knowledge. *Educational Technology & Society*, 19(3), 47–57.

<https://hdl.handle.net/11245/1.547418>

Angeli, C., Xerou, E., & Nicolaou, M. (2019). Investigating K-2 students' computational thinking skills during a problem solving activity about the water cycle using educational robotics. In *Proceedings of the IADIS International Conference on Cognition & Exploratory Learning in Digital Age* (pp. 85–91).

https://doi.org/10.33965/celda2019_201911L011

Anney, V. N. (2014) Ensuring the quality of the findings of qualitative research: Looking at trustworthiness criteria. *Journal of Emerging Trends in Educational Research and Policy Studies (JETERAPS)*, 5, 272–281.

<https://doaj.org/article/2a784a36b5684f12ad93e198ae21238d>

Aslam, R., Khan, N., & Ahmed, U. (2020). Technology integration and teachers' professional knowledge with reference to international society for technology in education (ISTE) standards: A causal study. *Journal of Education & Educational Development*, 7(2), 307–327. <https://doi.org/10.22555/joed.v7i2.31>

Bano, S., Atif, K., & Mehdi, S. A. (2024). Systematic review: Potential effectiveness of educational robotics for 21st century skills development in young learners.

Education and Information Technologies, 29(9), 11135–11153.

<https://doi.org/10.1007/s10639-023-12233-2>

Barana, A., Fissore, C., Marchisio, M., & Pulvirenti, M. (2020). Teacher training for the development of computational thinking and problem posing and solving skills with technologies. *ELearning & Software for Education*, 2, 136–144.

<https://doi.org/10.12753/2066-026X-20-103>

Baroutsis, A., White, S. L. J., Ferdinands, E., Goldsmith, W., & Lambert, E. (2019).

Computational thinking as a foundation for coding: Developing student engagement and learning. *Australian Primary Mathematics Classroom*, 24(2), 10–15.

Barth-Cohen, L., Montoya, B., & Shen, J. I. (2019). Walk like a robot: A no-tech coding activity to teach computational thinking. *Science Scope*, 42(9), 12–16.

Bento Miguens, A. L., Nunes Piedade, J. M., dos Santos, R. J. B., & Oliva, T. L. (2024).

Meaningful learning in mathematics: A study on motivation for learning and development of computational thinking using educational robotics. *Educational Media International*, 61(1–2), 4–15.

<https://doi.org/10.1080/09523987.2024.2357472>

Berson, I. R., Berson, M. J., McKinnon, C., Aradhya, D., Alyaesh, M., Luo, W., &

Shapiro, B. R. (2023). An exploration of robot programming as a foundation for spatial reasoning and computational thinking in preschoolers' guided play. *Early Childhood Research Quarterly*, 65, 57–67.

<https://doi.org/10.1016/j.ecresq.2023.05.015>

- Bhakti Yoga Budi, D. A. I., Okyranida Indica Yona, P. R., Maryani, I., & Nizaar, M. (2025). Twenty-first century learning technology innovation: Teachers' perceptions of gamification in science education in elementary schools. *Open Education Studies*, 7(1), 21–27. <https://doi.org/10.1515/edu-2025-0100>
- Brown, M. (2020). Book review. *Power and Education*, 12(1), 137–138. <https://doi.org/10.1177/1757743819873258>
- Buck, S., Ritter, G. W., Jensen, N. C., & Rose, C. P. (2010). Teachers say the most interesting things — An alternative view of testing. *Phi Delta Kappan*, 91(6), 50–54. <https://doi.org/10.1177/003172171009100613>
- Busuttil, L. Vassallo, D., & Schembri, P. (2025). Computational thinking: Exploring approach in early childhood education. In S. B. Kert Editor, *Effective computer science education in K–12 Classrooms*. (pp. 27–54). <https://doi.org/10.4018/979-8-3693-4542-9.ch002>
- Butler, D., & Leahy, M. (2021). Developing preservice teachers' understanding of computational thinking: A constructionist approach. *British Journal of Educational Technology*, 52(3), 1060–1077. <https://doi.org/10.1111/bjet.13065>.
- Castro, A., Medina, J., Aguilera, C. A., Ramirez, M., & Aguilera, C. (2023). Robotics education in STEM units: Breaking down barriers in rural multigrade schools. *Sensors*, 23(1), 387. <https://doi.org/10.3390/s23010387>
- Capobianco, B. M., Radloff, J., & Lehman, J. D. (2020). Elementary science teachers' sense-making with learning to implement engineering design and its impact on

- students' science achievement. *Journal of Science Teacher Education*, 32(1), 39–61. <https://doi.org/10.1080/1046560X.2020.1789267>
- Cassidy, M., Tucker-Raymond, E., & Puttick, G. (2020, March 1). Distributing expertise to integrate computational thinking practices. *Science Scope*, 43(7), 18. <https://doi.org/10.1080/08872376.2020.12291327>
- Castillo-Montoya, M. (2016). Preparing for interview research: The interview protocol refinement framework. *The Qualitative Report*, 21(5), 811–831. <https://doi.org/10.46743/2160-3715/2016.2337>
- Cervera, N., Diago, P. D., Orcos, P., & Yáñez, D. F. (2020). The acquisition of computational thinking through mentoring: An exploratory study. *Education Sciences*, 10(202), 202. <https://doi.org/10.3390/educsci10080202>
- Chalmers, C. (2018). Robotics and computational thinking in primary school. *International Journal of Child-Computer Interaction*, 17, 93–100. <https://doi.org/10.1016/j.ijcci.2018.06.005>
- Chapman, J. R., & Rich, P. J. (2018). Does educational gamification improve students' motivation? If so, which game elements work best? *Journal of Education for Business*, 93(7), 315–322. <https://doi.org/10.1080/08832323.2018.1490687>
- Chen, T.-I., Lin, S.-K., & Chung, H.-C. (2023). Gamified educational robots lead an increase in motivation and creativity in STEM education. *Journal of Baltic Science Education*, 22(3), 427–438. <https://doi.org/10.33225/jbse/23.22.427>
- Chen, S.-J., Shan, X., Liu, Z.-M., & Chen, C.-Q. (2025). Employing large language models to enhance K–12 students' programming debugging skills, computational

thinking, and self-efficacy. *Educational Technology & Society*, 28(2), 259–278.

[https://doi.org/10.30191/ETS.202504_28\(2\).TP01](https://doi.org/10.30191/ETS.202504_28(2).TP01)

Chevalier, M., Giang, C., Piatti, A., & Mondada, F. (2020). Fostering computational thinking through educational robotics: A model for creative computational problem solving. *International Journal of STEM Education*, 7, 1–18.

<https://doi.org/10.1186/s40594-020-00210-8>

Choi, W.-C., Choi, I.-C., Lam, C.-T., & Mendes, A. J. (2024). Learning sequencing with Bee-Bot: A study on improving computational thinking and motivation for young learners in programming education. 2024 IEEE Frontiers in Education Conference (FIE), Frontiers in Education Conference (FIE), 2024 IEEE, 1–8.

<https://doi.org/10.1109/FIE61694.2024.10893369>

Chou, P. N. (2020). Using ScratchJr to foster young children's computational thinking competence: A case study in a third-grade computer class. *Journal of Educational Computing Research*, 58(3), 570–595.

<https://doi.org/10.1177/0735633119872908>

Creswell, J. W., & Creswell, J. D. (2018). *Research design: Qualitative, quantitative, and mixed methods approaches* (5th ed.). Sage Publications, Inc.

<https://doaj.org/article/91bebdf43e3d4ff982a45436915a7ea9>

Critten, V., Hagon, H., Critten, S., & Messer, D. (2025). Developing computational thinking abilities in the early years using guided play activities. *Education Sciences*, 15(10), 1298. <https://doi.org/10.3390/educsci15101298>

- Cronin, B. (2012). Language matters. *Journal of the American Society for Information Science and Technology*, 63(2), 217. <https://doi.org/10.1002/asi.21698>
- Cutcliffe, J. R., & McKenna, H. P. (2004). Expert qualitative researchers and the use of audit trails. *Journal of Advanced Nursing*, 45(2), 126–133.
<https://doi.org/10.1046/j.1365-2648.2003.02883.x>
- da Silva, M. G. T., Maia, C. J. N., Curvelo, C. C. F., Garcia, L. T. S., & Gonçalves, L. M. G. (2025). Educational robotics as a pedagogical resource for K–12 students with learning difficulties. *Scientific Reports*, 15(1), 1–18.
<https://doi.org/10.1038/s41598-025-19821-x>
- DeCuir-Gunby, J. T., Marshall, P. L., & McCulloch, A. W. (2011). Developing and using a codebook for the analysis of interview data: An example from a professional development research project. *Field Methods*, 23(2), 136–155.
<https://doi.org/10.1177/1525822X10388468>
- Deterding, S. (2011). Situated motivational affordances of game elements: A conceptual model. In *Gamification: Using game design elements in non-gaming contexts* (Workshop at CHI 2011). <https://doi.org/10.1145/1979572.1979575>
- Erdmann, A., & Potthoff, S. (2023). Decision criteria for the ethically reflected choice of a member check method in qualitative research: A proposal for discussion. *International Journal of Qualitative Methods*, 22.
<https://doi.org/10.1177/16094069231177664>
- Ensign, T. I. (2020). Elementary educators' attitudes about the utility of educational robotics and their ability and intent to use it with students [Doctoral dissertation,

- ProQuest Information & Learning]. Dissertation Abstracts International Section A: Humanities and Social Sciences, 81(7-A). <https://doi.org/10.33915/etd.5546>
- Fagerlund, J., Häkkinen, P., Vesisenaho, M., & Viiri, J. (2020). Assessing 4th grade students' computational thinking through Scratch programming projects. *Informatics in Education, 19*(4). <https://doi.org/10.15388/infedu.2020.27>
- Fakiri, M., & Khaldi, M. (2025). The integration of educational robotics and artificial intelligence in learning AI concepts. *Journal of Information Sciences, 24*(1). <https://doi.org/10.34874/IMIST.PRSM/jis-v24i1.54791>
- Fessakis, G., Komis, V., Dimitracopoulou, A., & Prantsoudi, S. (2019). Overview of the computer programming learning environments for primary education. *Review of Science, Mathematics and ICT Education, 13*(1), 7–33. <https://doi.org/10.26220/rev.3140>
- Flicker, S. (2004). Ask me no secrets, I'll tell you no lies: What happens when a respondent's story makes no sense. *The Qualitative Report, 9*(3), 528–537. <https://doi.org/10.46743/2160-3715/2004.1922>
- Fofang, J., Weintrop, D., Walton, M., Elby, A., & Walkoe, J. (2020). Mutually supportive mathematics and computational thinking in a fourth-grade classroom. In M. Gresalfi & I. Horn (Eds.), *The interdisciplinarity of the learning sciences* (14th International Conference of the Learning Sciences) (Vol. 3, pp. 1389–1396). Nashville, TN: International Society of the Learning Sciences. <https://doi.dx.org/10.22318/icls2020.1389>

- García-Peñalvo, F. J., Reimann, D., Tuul, M., Rees, A., & Jormanainen, I. (2016). *An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers*. TACCLE3 Consortium. <https://doi.org/10.5281/zenodo.165123>
- Georgiou, K., & Angeli, C. (2019). Developing preschool children's computational thinking with educational robotics: The role of cognitive differences and scaffolding. *International Association for Development of the Information Society*. http://doi.org/10.33965/celda2019_201911L013
- Ghani, A.D., Griffiths, D., Salha, S., Affounch, S., Khalili, F., Khlaif, Z. N., & Burgos, D. (2022). Developing teaching practice in computational thinking in Palestine. *Frontiers in Psychology*, 13. <https://doi.org/10.3389/fpsyg.2022.870090>
- Gökbulut, B., & Bakangöz, M. M. (2021). Instructor views on technology use and coding training. *International Journal of Progressive Education*, 17(3), 299–315. <https://doi.org/10.29329/ijpe.2021.346.19>
- Goldenberg, E. P., & Carter, C. J. (2021). Programming as a language for young children to express and explore mathematics in school. *British Journal of Educational Technology*, 52(3), 969–985. <https://doi.org/10.1111/bjet.13080>
- Guba, E. G., & Lincoln, Y. S. (1981). *Effective evaluation: Improving the usefulness of evaluation results through responsive and naturalistic approaches*. San Francisco: Jossey-Bass

- Gunbatar, M. S., & Turan, B. (2019). The effect of block-based programming on the computational thinking skills of middle school students. *Turkish Online Journal of Educational Technology*, 2, 335–339.
- Hadad, S., Shamir-Inbal, T., Blau, I., & Leykin, E. (2021). Professional development of code and robotics teachers through small private online courses (SPOC): Teacher centrality and pedagogical strategies for developing computational thinking of students. *Journal of Educational Computing Research*, 59(4), 763–791.
<https://doi.org/10.1177/0735633120973432>
- Hall, J. A., & McCormick, K. I. (2022). “My cars don’t drive themselves”: Preschoolers’ guided play experiences with button-operated robots. *TechTrends: Linking Research & Practice to Improve Learning*, 66(3), 510–526.
<https://doi.org/10.1007/s11528-022-00727-8>
- Herro, D., Quigley, C., Plank, H., Abimbade, O., & Owens, A. (2022). Instructional practices promoting computational thinking in STEAM elementary classrooms. *Journal of Digital Learning in Teacher Education*, 38(4), 158–172.
<https://doi.org/10.1080/21532974.2022.2087125>
- Hershkovitz, H., Bain, C., Kelter, J., Peel, A., Wu, S., Horn, M. S., & Wilensky, U. (2023). Contribution of computational thinking to STEM education: High school teachers’ perceptions after a professional development program. *Journal of Computers in Mathematics & Science Teaching*, 42(1), 35–65.
<https://doi.org/10.70725/863082janbwy>
- Houghton, C., Casey, D., Shaw, D., & Murphy, K. (2013). Rigor in qualitative case-study

research. *Nurse Researcher*, 20(4), 12–17.

<https://doi.org/10.7748/nr2013.03.20.4.12.e326>

Hu, C.-C., Yang, Y.-F., Cheng, Y.-W., & Chen, N.-S. (2024). Integrating educational robots and low-cost self-made toys to enhance STEM learning performance for primary school students. *Behaviour & Information Technology*, 43(8), 1614–1635. <https://doi.org/10.1080/0144929X.2023.2222308>

Hudson, M. A., & Baek, Y. (2022). Increasing elementary students' computational thinking skills using a multifaceted robotics-based intervention. *Computers in the Schools*, 39(1), 16–40. <https://doi.org/10.1080/07380569.2022.2037295>

Hunsaker, E., & West, R. E. (2020). Designing computational thinking and coding badges for early childhood educators. *TechTrends: Linking Research and Practice to Improve Learning*, 64(1), 7–16. <https://doi.org/10.1007/s11528-019-00420-3>

International Society of Technology in Education. (2016). ISTE computational thinking competencies. <https://www.iste.org/standards/iste-standards-for-computational-thinking>

International Society for Technology in Education. (2018). *ISTE standards for educators*. <https://www.iste.org/standards/for-educators>

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology*, 26, 175–192.

- Jin, Y., & Harron, J. R. (2023). An investigation of in-service teachers' perceptions and development of computational thinking skills in a graduate emerging technologies course. *International Journal of Computer Science Education in Schools*, 6(2), n2.
- Jocius, R., O'Byrne, W. I., Albert, J., Joshi, D., Robinson, R., & Andrews, A. (2021). Infusing computational thinking into STEM teaching: From professional development to classroom practice. *Educational Technology & Society*, 24(4), 166–179.
- Junruang, C., & Kanjug, I. (2025). The relationship between computational and creative thinking in preschool children: An application through gamification and artificial intelligence-supported constructivist personalized learning environment. *Educational Process: International Journal*, 17.
<https://doi.org/10.22521/edupij.2025.17.337>
- Jurado, E., Fonseca, D., Coderch, J., & Canaleta, X. (2020). Social STEAM learning at an early age with robotic platforms: A case study in four schools in Spain. *Sensors*, 20(13), 3698. <https://doi.org/10.3390/s20133698>
- Juškevičienė, A. (2020). STEAM teacher for a day: A case study of teachers' perspectives on computational thinking. *Informatics in Education*, 19(1), 33–50.
<https://doaj.org/article/b0f7cf0095934b7caa9a26e6329e278d>
- Kale, U., Akcaoglu, M., Cullen, T., Goh, D., Devine, L., Calvert, N., & Grise, K. (2018). Computational what? Relating computational thinking to teaching. *TechTrends: Linking Research & Practice to Improve Learning*, 62(6), 574–584.
<https://doi.org/10.1007/s11528-018-0290-9>

- Kang, Y. & Lee, K. (2020). Designing technology entrepreneurship education using computational thinking. *Education and Information Technologies* 25, 5357–5377. <https://doi.org/10.1007/s10639-020-10231-2>
- Karakasis, C., & Xinogalos, S. (2020). BlocklyScript: Design and pilot evaluation of an RPG platform game for cultivating computational thinking skills in young students. *Informatics in Education*, 19(4), 641–668. <https://doi.org/10.15388/infedu.2020.28>
- Kelter, J., Peel, A., Bain, C., Anton, G., Dabholkar, S., Horn, M. S., & Wilensky, U. (2021). Constructionist co-design: A dual approach to curriculum and professional development. *British Journal of Educational Technology*, 52(3), 1043–1059. <https://doi.org/10.1111/bjet.13084>
- Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2020). Teacher change following a professional development experience in integrating computational thinking into elementary science. *Journal of Science Education and Technology*, 29, 174–188. <https://doi.org/10.1007/s10956-019-09798-4>
- Kılıç, S. (2022). Tendencies towards computational thinking: A content analysis study. *Participatory Educational Research*, 9(5), 288–304. <https://doi.org/10.17275/per.22.115.9.5>
- Kite, V., & Park, S. (2018). Boom bust build. *Science Teacher*, 85(3), 22–28. https://doi.org/10.2505/4/tst18_085_03_22
- Kite, V., & Park, S. (2023). What’s computational thinking?: Secondary science teachers’ conceptualizations of computational thinking (CT) and perceived barriers to CT

integration. *Journal of Science Teacher Education*, 34(4), 391–414.

<https://doi.org/10.1080/1046560X.2022.2110068>

Kong, S.-C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and learning outcomes of programming concepts, practices, and pedagogy. *Computers & Education*, 151, Article 103872.

<https://doi.org/10.1016/j.compedu.2020.103872>

Kwon, K., Ottenbreit-Leftwich, A. T., Brush, T. A., Jeon, M., & Yan, G. (2021).

Integration of problem-based learning in elementary computer science education: Effects on computational thinking and attitudes. *Educational Technology Research & Development*, 69(5), 2761–2787.

<https://doi.org/10.1007/s11423-021-10034-3>

Ladachart, L., Phothong, W., Suaklay, N., & Ladachart, L. (2020). Thai elementary science teachers' images of “engineer(s)” at work. *Journal of Science Teacher Education*, 31(6), 631–653. <https://doi.org/10.1080/1046560X.2020.1743563>

Latombe, J. C. (2012). *Robot motion planning* (Vol. 124). Springer Science & Business Media.

Lee, D., & Lee, Y. (2024). Productive failure-based programming course to develop computational thinking and creative problem solving skills in a Korean elementary school. *Informatics in Education*, 23(2), 385–408.

<https://doi.org/10.15388/infedu.2024.14>

Lee, T. Y., Mauriello, M. L., Ahn, J., & Bederson, B. B. (2014). CTArcade:

Computational thinking with games in school-age children. *International Journal*

of Child-Computer Interaction, 2(1), 26–33.

<https://doi.org/10.1016/j.ijcci.2014.06.003>

Lemon, L. (2017). Applying a mindfulness practice to qualitative data collection. *The Qualitative Report*, 22(12), 3305–3313. <https://doi.org/10.46743/2160-3715/2017.3161>

Leonard, A., Daily, S., Jörg, S., & Babu, S. (2021). Coding moves: Design and research of teaching computational thinking through dance choreography and virtual interactions. *Journal of Research on Technology in Education*, 53(2), 159–177. <https://doi.org/10.1080/15391523.2020.1760754>

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. Sage.

Lincoln, Y. S., & Guba, E. G. (1986). But is it rigorous? Trustworthiness and authenticity in naturalistic evaluation. *New Directions for Program Evaluation*, 1986(30), 73–84.

Ling-Ling, U., Labadin, J., & Suraya Mohamad, F. (2021). Information system framework for training teachers on computational thinking. *SAR Journal - Science and Research*, 119–127. <https://doi.org/10.18421/SAR43-04>

Lodi, M., & Martini, S. (2021). Computational thinking, between Papert and Wing. *Science & Education*, 30(4), 883–908. <https://doi.org/10.1007/s11191-021-00202-5>

Luo, F., Ijeluola, S. A., Westerlund, J., Walker, A., Denham, A., Walker, J., & Young, C. (2023). Supporting elementary teachers' technological, pedagogical, and content knowledge in computational thinking integration. *Journal of Science Education &*

- Technology*, 32(4), 583–596. <https://doi.org/10.1007/s10956-023-10045-0>
- Luo, Q., Hu, Y., & Wen, J. (2025). Exploring the relationship between STEM attitudes and computational thinking in young children. 2025 10th International STEM Education Conference (iSTEM-Ed), 1–5. <https://doi.org/10.1109/iSTEM-Ed65612.2025.11129259>
- Magana, S. (2017). *Disruptive classroom technologies: A framework for innovation in education* (1st ed.). Corwin, A SAGE Company.
- Mason, S. L., & Rich, P. J. (2019). Preparing elementary school teachers to teach computing, coding, and computational thinking. *Contemporary Issues in Technology & Teacher Education*, 19(4), 1 <https://citejournal.org/volume-19/issue-4-19/general/preparing-elementary-school-teachers-to-teach-computing-coding-and-computational-thinking>
- Massicotte, J., Staudt, C. J., & McIntyre, C. (2021). Weathering the virtual storm: Using computational thinking to make a forecast. *Science Scope*, 44(5), 18–27. <https://doi.org/10.1080/08872376.2021.12291411>
- Matteson, S. M. (2021). Chex Mix™ data analysis activity. *College Teaching*, 69(3), 121–125. <https://doi.org/10.1080/87567555.2020.1843389>
- Matthews, S., Nicholas, M., Paatsch, L., Kervin, L., & Wyeth, P. (2025). Social and curious: Lessons in designing digital manipulatives for young children. *International Journal of Child-Computer Interaction*, 44, Article 100725. <https://doi.org/10.1016/j.ijcci.2025.100725>

- Merriam, S. B. (2002). Introduction to qualitative research. In *Qualitative research in practice: Examples for discussion and analysis (Vol. 1, No. 1, pp. 1–17)*.
- Mills, K. A., Cope, J., Scholes, L., & Rowe, L. (2025). Coding and computational thinking across the curriculum: A review of educational outcomes. *Review of Educational Research, 95*(3), 581–618.
<https://doi.org/10.3102/00346543241241327>
- Monjelat, N., & Lantz-Andersson, A. (2020). Teachers' narrative of learning to program in a professional development effort and the relation to the rhetoric of computational thinking. *Education and Information Technologies, 25*(3), 2175–2200. <https://doi.org/10.1007/s10639-019-10048-8>
- Monteiro, A. F., Miranda-Pinto, M., & Osório, A. J. (2021). Coding as literacy in preschool: A case study. *Education Sciences, 11*(5), 198
<https://doi.org/10.3390/educsci11050198>
- Mouza, C., Yadav, A., & Ottenbreit-Leftwich, A. (2018). Developing computationally literate teachers: Current perspectives and future directions for teacher preparation in computing education. *Journal of Technology & Teacher Education, 26*(3), 333–352. <https://www.learntechlib.org/primary/p/184602/>
- Mukasheva, M. & Omirzakova, A. (2021). Computational thinking assessment at primary school in the context of learning programming. *World Journal on Educational Technology: Current Issues, 13*(3), 336–353.
<https://doi.org/10.18844/wjet.v13i3.5918>
- Munn, C. (2021). A qualitative study exploring robots as a potential classroom tool for

teaching computational thinking within a sixth-grade class. *Journal of Computers in Mathematics and Science Teaching*, 40(3), 229–264

<https://doi.org/10.70725/479467wuwye>

- Namli, N. A., & Aybek, B. (2022). An investigation of the effect of block-based programming and unplugged coding activities on fifth graders' computational thinking skills, self-efficacy, and academic performance. *Contemporary Educational Technology*, 14(1), 1–16. <https://doi.org/10.30935/cedtech/11477>
- Newley, A., Kaya, E., Deniz, H., & Yesilyurt, E. (2018). Celebrity statues: Learning computational thinking by designing biomimetic robots. *Science Scope*, 42(1), 74.
- Noh, J., Lee, J. (2020). Effects of robotics programming on the computational thinking and creativity of elementary school students. *Education Technology Research Development*, 68, 463–484. <https://doi.org/10.1007/s11423-019-09708-w>
- Oddie, A., Hazlewood, P., Blakeway, S., & Whitfield, A. (2010). Introductory problem solving and programming: Robotics versus traditional approaches. *Innovations in Teaching & Learning in Information & Computer Sciences*, 9(2), 86–91. <https://doi.org/10.11120/ital.2010.09020011>
- Ogegbo, A. A., & Ramnarain, U. (2022). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, 58(2), 203–230. <https://doi.org/10.1080/03057267.2021.1963580>
- Olaniran, S. O., & Baruwa, I. B. (2020). Ethical considerations in adult and community education research in Nigeria: Issues and perspectives. *International Journal for Educational Integrity*, 16(1), 1–10. <https://doi.org/10.1007/s40979-020-00057-3>

- Ortega-Ruipérez, B., & Lázaro Alcalde, M. (2022). Teachers' perception about the difficulty and use of programming and robotics in the classroom. *Interactive Learning Environments*, 31(10), 7074–7085.
<https://doi.org/10.1080/10494820.2022.2061007>
- Ortlipp, M. (2008). Keeping and using reflective journals in the qualitative research process. *The Qualitative Report*, 13(4), 695–705. <https://doi.org/10.46743/2160-3715/2008.1579>
- Papadakis, S. (2020). Robots and robotics kits for early childhood and first school age. *International Journal of Interactive Mobile Technologies*, 14(18), 34–56.
<https://doi.org/10.3991/ijim.v14i18.16631>
- Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism*.
https://web.media.mit.edu/~calla/web_comunidad/Reading-En/situating_constructionism.pdf
- Patton, M. Q. (2015). *Qualitative research and evaluation methods* (4th ed.). Sage Publications.
- Pelizzari, F., Marangi, M., Rivoltella, P. C., Peretti, G., Massaro, D., & Villani, D. (2023). Coding and childhood between play and learning: Research on the impact of coding in the learning of 4-year-olds. *Research on Education and Media*, 15(1), 9–19. <https://doi.org/10.2478/rem-2023-0003>
- Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., & Pizarro, C. (2020). Can computational thinking be improved by using a methodology based on metaphors and Scratch to teach computer programming to children? *Computers in Human Behavior*, 105,

Article 106196. <https://doi.org/10.1016/j.chb.2018.12.027>

Percy, W. H., Kostere, K., & Kostere, S. (2015). Generic qualitative research in psychology. *The Qualitative Report*, 20(2), 76–85.

<https://www.proquest.com/scholarly-journals/generic-qualitative-research-psychology/docview/1677664021/se-2>

Piazza, A., & Mengual Andrés, S. (2020). Computational thinking and coding in primary education: Scientific productivity on SCOPUS. Pixel-Bit.

<https://doi.org/10.12795/pixelbit.79769>

Popa, M. C., & Biclea, D. (2023). Promoting effective math learning with educational robots. *Educatia* 21, 25, 39–47. <https://doi.org/10.24193/ed21.2023.25.04>

Quintero, J., Baldiris, S., Cerón, J., Garzón, J., Burgos, D., & Vélez, G. (2022).

Gamification as support for educational inclusion: The case of AR-mBot. 2022 *International Conference on Advanced Learning Technologies (ICALT)*, 269–273.

<https://doi.org/10.1109/ICALT55010.2022.00088>

Rallis, S. F., & Rossman, G. B. (2012). *The research journey: Introduction to inquiry*. Guilford Press.

Rath, M. K. (2015). *Motion control of automated mobile robots in dynamic environment* [Unpublished doctoral dissertation]. *National Institute of Technology*.

Ravitch, S. M., & Carl, N. M. (2016). *Qualitative research: Bridging the conceptual, theoretical, and methodological*. Sage Publications.

Rehmat, A. P., Ehsan, H., & Cardella, M. E. (2020). Instructional strategies to promote computational thinking for young learners. *Journal of Digital Learning in*

Teacher Education, 36(1), 46–62.

<https://doi.org/10.1080/21532974.2019.1693942>

Relkin, E., de Ruiter, L. E., & Bers, M. U. (2021). Learning to code and the acquisition of computational thinking by young children. *Computers & Education*, 169, Article 104222. <https://doi.org/10.1016/j.compedu.2021.104222>

Rich, K. M., Yadav, A., & Schwarz, C. V. (2019). Computational thinking, mathematics, and science: Elementary teachers' perspectives on integration. *Journal of Technology and Teacher Education*, 27(2), 165–205.

<https://doi.org/10.70725/303733vqleui>

Rich, K. M., Yadav, A., & Larimore, R. A. (2020). Teacher implementation profiles for integrating computational thinking into elementary mathematics and science instruction. *Education and Information Technologies*, 25, 3161–3188.

<https://doi.org/10.1007/s10639-020-10115-5>

Rich, P. J., Larsen, R. A., & Mason, S. L. (2021). Measuring teacher beliefs about coding and computational thinking. *Journal of Research on Technology in Education*, 53(3), 296–316. <https://doi.org/10.1080/15391523.2020.1771232>

Rich, P. J., Mason, S. L., & O'Leary, J. (2021). Measuring the effect of continuous professional development on elementary teachers' self-efficacy to teach coding and computational thinking. *Computers & Education*, 168, Article 104196.

<https://doi.org/10.1016/j.compedu.2021.104196>

Ríos Félix, J. M., Zatarain Cabada, R., & Barrón Estrada, M. L. (2020). Teaching computational thinking in Mexico: A case study in a public elementary school.

Education and Information Technologies, 25(6), 5087–5101.

<https://doi.org/10.1007/s10639-020-10213-4>

Ritter, F., & Schlomske-Bodenstein, N. (2025). Fostering algorithmic thinking within a productive-failure-based workshop utilizing AI. *2025 IEEE Global Engineering Education Conference (EDUCON)*, 1–10.

<https://doi.org/10.1109/EDUCON62633.2025.11016641>

Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020).

Computational thinking and mathematics using Scratch: An experiment with sixth-grade students. *Interactive Learning Environments*, 28(3), 316–327.

<https://doi.org/10.1080/10494820.2019.1612448>

Roehl, J. M., & Harland, D. J. (2022). Imposter participants: Methodological challenges related to online recruitment and virtual data collection and how to overcome them. *The Qualitative Report*, 27(11), 2469–2485. <https://doi.org/10.46743/2160-3715/2022.5475>

Rubin, H & Rubin, I (2011). Qualitative interviewing: The art of hearing data (3rd ed.).

Thousand Oaks, CA: Sage, 360 pages. *Canadian Journal of Program Evaluation*, 27(2).

Rus, V. M., & Almășan, B. (2024). An exploration of the use of educational robotics in preschool education. *Educatia 21*, 28 (Suppl. Special Issue), 99–103.

<https://doi.org/10.24193/ed21.2024.28.10>

Sáez-López, J.-M., Sevillano-García, M.-L., & Vázquez-Cano, E. (2019). The effect of programming on primary school students' mathematical and scientific

- understanding: Educational use of mBot. *Educational Technology Research & Development*, 67(6), 1405–1425. <https://doi.org/10.1007/s11423-019-09648-5>
- Saldaña, J. (2016). *The coding manual for qualitative researchers*. Sage Publications.
- Santiago, C. P., Silva, A. R. S., Menezes, J. W. M., & Aquino, F. J. A. (2025). Gamification and self-determination theory in teaching computational thinking: An experience with Quizizz software. *Informatics in Education*, 24(3), 587–615. <https://doi.org/10.15388/infedu.2025.17>
- Schneider, J., & Gottlieb, D. (2021). In praise of ordinary measures: The present limits and future possibilities of educational accountability. *Educational Theory*, 71(4), 455–473. <https://doi.org/10.1111/edth.12488>
- Selby, C. C. (2014). How can the teaching of programming be used to enhance computational thinking skills? (Unpublished doctoral dissertation). University of Southampton, Southampton, UK. <http://eprints.soton.ac.uk/id/eprint/366256>
- Searle, K., Tofel-Grehl, C., & Macdonald, B. L. (2021). Lighting up history: Integrating mathematics and computational thinking in the science classroom. *Science Scope*, 44(5), 64–71. <https://doi.org/10.1080/08872376.2021.12291417>
- Shen, J., Chen, G., Barth-Cohen, L., Jiang, S., & Eltoukhy, M. (2022). Connecting computational thinking in everyday reasoning and programming for elementary school students. *Journal of Research on Technology in Education*, 54(2), 205–225. <https://doi.org/10.1080/15391523.2020.1834474>
- Shenton, A. K. (2004). Strategies for ensuring trustworthiness in qualitative research projects. *Education for Information*, 22(2), 63–75.

<http://iospress.metapress.com/content/3ccttm2g59cklapx/?p=1dc0853516d0424baa3147a00c250f29&pi=0>

- Shin, N., Bowers, J., Krajcik, J., & Damelin, D. (2021). Promoting computational thinking through project-based learning. *Disciplinary and Interdisciplinary Science Education Research*, 3, 1–15. <https://doaj.org/toc/2662-2300>
- Silva, R., Fonseca, B., Costa, C., & Martins, F. F. (2021). Fostering computational thinking skills: A didactic proposal for elementary school grades. *Education Sciences*, 11, 518. <https://doi.org/10.3390/educsci11090518>
- Soboleva, E. V., Sabirova, E. G., Babieva, N. S., Sergeeva, M. G., & Torkunova, J. V. (2021). Formation of computational thinking skills using computer games in teaching mathematics. *Eurasia Journal of Mathematics, Science & Technology Education*, 17(10). <https://doi.org/10.29333/ejmste/11177>
- Stewart, D. L. (2010). Researcher as instrument: Understanding “shifting” findings in constructivist research. *Journal of Student Affairs Research and Practice*, 47(3), 291–306. <https://doi.org/10.2202/1949-6605.6130>
- Sun, C., Yang, S., & Becker, B. (2024). Debugging in computational thinking: A meta-analysis on the effects of interventions on debugging skills. *Journal of Educational Computing Research*, 62(4), 1087–1121. <https://doi.org/10.1177/07356331241227793>
- Sun, Y., Okojie, M. C. P. O., Yu, W.-C. W., & Boulder, T. C. (2020). Elementary students as digital makers: Improving STEM+C teaching and learning with digital making. In S. P. Huffman, S. Loyless, S. Albritton, & C. Green (Eds.),

Leveraging technology to improve school safety and student wellbeing (pp. 262–280). Information Science Reference/IGI Global. <https://doi.org/10.4018/978-1-7998-1766-6.ch015>

Sung, W., & Black, J. B. (2021). Factors to consider when designing effective learning: Infusing computational thinking in mathematics to support thinking-doing. *Journal of Research on Technology in Education*, 53(4), 404–426. <https://doi.org/10.1080/15391523.2020.1784066>

Stewart, W. H., Baek, Y., Kwid, G., & Taylor, K. (2021). Exploring factors that influence computational thinking skills in elementary students' collaborative robotics. *Journal of Educational Computing Research*, 59(6), 1208–1239. <https://doi.org/10.1177/0735633121992479>

Strawhacker, A., Lee, M., & Bers, M. U. (2018). Teaching tools, teachers' rules: Exploring the impact of teaching styles on young children's programming knowledge in ScratchJr. *International Journal of Technology and Design Education*, 28(2), 347–376. <https://doi.org/10.1007/s10798-017-9400-9>

Tang, A. L. L., Tung, V. W. S., & Cheng, T. O. (2020). Teachers' perceptions of the potential use of educational robotics in management education. *Interactive Learning Environments*, 31(1), 313–324. <https://doi.org/10.1080/10494820.2020.1780269>

Tang, H., Xu, W., Feng, Y., & Cao, W. (2025). Global effects of robot-based education on academic achievements, computation, motivation, and performance. *Humanities & Social Sciences Communications*, 12(1), 1–12.

<https://doi.org/10.1057/s41599-025-05546-9>

Thompson, A. D., Lindstrom, D. L., & Schmidt-Crawford, D. A. (2020, January).

Computational thinking: What went wrong? *Journal of Digital Learning in Teacher Education*, 36(1), 4–5. <https://doi.org/10.1080/21532974.2019.1696641>

Tonbuloğlu, B., & Tonbuloğlu, I. (2019). The effect of unplugged coding activities on computational thinking skills of middle school students. *Informatics in Education*, 18(2), 403–426. <https://doi.org/10.15388/ifedu.2019.19>

Turner III, D. W. (2010). Qualitative interview design: A practical guide for novice investigators. *The Qualitative Report*, 15(3), 754.

<https://www.proquest.com/scholarly-journals/qualitative-interview-design-practical-guide/docview/578480397/se-2>

Tyler S. Love, Scott R. Bartholomew, & Jessica Yauney. (2022). Examining changes in teachers' beliefs toward integrating computational thinking to teach literacy and math concepts in grades K–2. *Journal for STEM Education Research*, 5(3), 380–401. <https://doi.org/10.1007/s41979-022-00077-3>

Usengül, L., & Bahçeci, F. (2020). The effect of LEGO WeDo 2.0 education on academic achievement and attitudes and computational thinking skills of learners toward science. *World Journal of Education*, 10(4), 83–93.

<https://doi.org/10.5430/wje.v10n4p83>

Vieira, C., Chiu, J., & Velasquez, B. (2023). Towards a learning progression of sequencing and algorithm design for five- and six-year-old children engaging with

an educational robot. *Computer Science Education*, 34(4), 596–616.

<https://doi.org/10.1080/08993408.2023.2255058>

Waite, D. (2011). A simple card trick: Teaching qualitative data analysis using a deck of playing cards. *Qualitative Inquiry*, 17(10), 982–985.

<https://doi.org/10.1177/1077800411425154>

Wagner, A. (2021). Evaluating science resources kindercoding unplugged: Screen-free activities for beginners. *Science and Children*, 58(5), 16.

<https://doi.org/10.1080/19434812.2021.12291668>

Weatherford, J., & Maitra, D. (2019). How online students approach bracketing: A survey research study. *Educational Research: Theory and Practice*, 30(2), 91–102.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1), 127–147.

<https://doi.org/10.1007/s10956-015-9581-5>

Weintrop, D. (2021). The role of block-based programming in computer science education [Seminar Session]. Raspberry Pi Foundation Research Seminar Series, Online.

https://www.terpconnect.umd.edu/~weintrop/papers/Weintrop_RaspPi_2021.pdf

Welch, L. E., Shumway, J. F., Clarke-Midura, J., & Lee, V. R. (2022). Exploring measurement through coding: Children’s conceptions of a dynamic linear unit

with robot coding toys. *Education Sciences*, 12(2), Article 143.

<https://doi.org/10.3390/educsci12020143>

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

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

Link Magazine, 6, 20–23. <https://people.cs.vt.edu/~kafura/CS6604/Papers/CT->

[What-And-Why.pdf](https://people.cs.vt.edu/~kafura/CS6604/Papers/CT-What-And-Why.pdf)

Wong, G. K.-W., & Cheung, H.-Y. (2020). Exploring children’s perceptions of developing twenty-first century skills through computational thinking and programming. *Interactive Learning Environments*, 28(4), 438–450.

<https://doi.org/10.1080/10494820.2018.1534245>

Wu, L., Looi, C.-K., Multisilta, J., How, M.-L., Choi, H., Hsu, T.-C., & Tuomi, P.

(2020). Teachers’ perceptions and readiness to teach coding skills: A comparative study between Finland, mainland China, Singapore, Taiwan, and South Korea.

Asia-Pacific Education Researcher, 29(1), 21–34. <https://doi.org/10.1007/s40299-019-00485-x>

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. *Communications of the ACM*, 60(4), 55–62.

<https://doi.org/10.1145/2994591>

Yang, D., Baek, Y., & Swanson, S. (2020). Developing computational thinking through project-based airplane design activities. *2020 IEEE Frontiers in Education*

Conference (FIE), 1, 1–4. <https://doi.org/10.1109/FIE44824.2020.9273950>

- Yang, Y., Long, Y., Sun, D., Van Aalst, J., & Cheng, S. (2020). Fostering students' creativity via educational robotics: An investigation of teachers' pedagogical practices based on teacher interviews. *British Journal of Educational Technology*, 51(5), 1826–1842. <https://doi.org/10.1111/bjet.12985>
- Yoon, B., & Uliassi, C. (2022). Researcher-as-instrument in qualitative research: The complexities of the educational researcher's identities. *The Qualitative Report*, 27(4), 1088–1102. <https://doi.org/10.46743/2160-3715/2022.5074>
- Yukselturk, E., & Altioek, S. (2017). An investigation of the effects of programming with Scratch on the preservice IT teachers' self-efficacy perceptions and attitudes towards computer programming. *British Journal of Educational Technology*, 48(3), 789–801. <https://doi.org/10.1111/bjet.12453>

Appendix A: Interview Protocol

Introductory script:

Greetings, and thank you so much for taking the time to meet with me. I am Marquita Jackson, and I will ask you a few questions for my research.

My research focus is on “How K–5 Teachers use Sphero Robots to Teach Computational Thinking?” This research is part of my dissertation for my Ph.D. program at Walden University. I am currently working as a STEM and Robotics teacher in a K–5 setting.

I want to inform you that this interview will be recorded only to ensure that I do not miss any pertinent information that you will be providing. If you would like a copy of the recording or transcript, it can be provided upon request. Transcripts with identifiers redacted will be shared with my university faculty along with my analysis. The interview recording and transcript will be destroyed as soon as I have completed my course.

[START RECORDING]

Background, Screening, and Introductory Questions

Today’s date is [DATE] and it is [TIME, including time zone]. Thank you for your willingness to volunteer for my doctoral research study.

[If required by IRB]

Before we get started with the official interview, I wanted to be sure you read the letter of consent, do you have any questions about participating in this study?

Do you agree to be interviewed for this study? [Wait for response of yes/no]

I’d like to learn a bit more about you and your experiences about [STUDY TOPIC].

- *How did you hear about my study? (Good idea to ask if you recruited using social media)*

Before we get started with the official interview, I'd like to learn a bit more about you and your experiences with using Sphero robots.

- *Tell me a bit about your experience and a bit about your current position.*
 - How did you hear about my study?
 - Share with me a bit about your current teaching position.
 - How long have you been teaching?
 - What grades have you taught?
 - How did you learn about Sphero robots?
 - What made you want to use Sphero robots with your students?
 - How often do you integrate Sphero robots into your instruction?

Option A: *Thank you, Let's go ahead and move into the interview questions.*

Option B: *Thank you so much for your willingness to participate. But after talking with you, I'm not sure you have the depth of experience in Sphero robotics and computational thinking that I need to answer my research questions. Thank you for your time. [They do not receive any compensation (gift cards), if applicable.]*

Table of Interview Questions: Add/remove rows as necessary.

Transition Statement: *My first group of questions relates to the implementation of Sphero robots.*

RQ 1	Interview Questions (IQs)	My Notes & Alignment to T3 & literature
What are K–5 teachers' experiences in implementing Sphero robot activities?	IQ 1: How do you introduce Sphero robots to students?	I will apply a priori codes of Translational Transformative Transcendent to the data to help answer RQ1.
	Probes:	

	<ul style="list-style-type: none"> ○ Describe some activities you do to prepare students to work with Sphero robots to your class ○ What does exploration of robots look like before you teach the lesson? ○ What is your planning process to integrate Sphero into lessons? 	
	<p>IQ 2: Describe a favorite activity you've implemented using the Sphero robots.</p>	
	<p>Probes:</p> <ul style="list-style-type: none"> ○ Why is this your favorite? ○ What else can you tell me about that activity? ○ What evidence did you have of student engagement during the activity? What types of thinking do you think they were doing? ○ Was using the robots for this activity similar to an activity you'd done in another way? How so? ○ What would you do to further improve the activity? 	
	<p>IQ 3:</p> <ul style="list-style-type: none"> ○ Describe a Sphero activity that did not go well when implementing it with your students. 	
	<p>Probes:</p> <ul style="list-style-type: none"> ○ Why do you think the activity was a challenge? ○ How did students respond? ○ What would you do differently next time, and why? 	
	<p>IQ 4:</p> <ul style="list-style-type: none"> ○ What are some of the challenges of using Sphero robots with young students? 	
	<p>Probes:</p>	

	<ul style="list-style-type: none"> ○ In what ways have you overcome those challenges? 	
	<p>IQ 5:</p> <ul style="list-style-type: none"> ○ How do you assess Sphero robot activities? 	
	<p>Probes:</p> <ul style="list-style-type: none"> ○ How do you determine if they are learning what you want them to? 	
	<p>IQ 6:</p> <ul style="list-style-type: none"> ○ Have your Sphero lessons replaced lessons you used to do without technology? Or have they provided activities that were previously not possible? Explain? 	
	<p>Probes:</p> <ul style="list-style-type: none"> ○ If the Sphero robots aren't available for an activity, are you able to teach the lesson another way? How so? 	

Transition Statement: *Now that you've shared about how you use Sphero robots, my second group of questions are related more to computational thinking.*

RQ 2	Interview Questions	My Notes & Alignment to Angeli/literature
How do K–5 teachers use Sphero robots to teach computational thinking?	IQ 8: <ul style="list-style-type: none"> ○ How do you use Sphero robots to teach mathematical thinking and logic? 	
	Probes: <ul style="list-style-type: none"> ○ What types of mathematical thinking are good to teach with Sphero robots, and why? 	
	IQ 9: <ul style="list-style-type: none"> ○ When you are teaching coding to young students, what tricks (strategies) have you used to help students learn how to follow instructions when coding? (Flow of control) 	Algorithms
	Probes: <ul style="list-style-type: none"> ● How effective have those strategies been for you? ● How does the age of student influence their ability to follow directions? How do you compensate for that? ● Some people would say K–5 students are too young to learn using robots. What would you say to them? 	
	IQ 10: <ul style="list-style-type: none"> ○ When teaching using Sphero robots, what activities do you use to teach students to put actions in the correct order when coding? 	(Sequence)
	Probes: <ul style="list-style-type: none"> ● Do your activities include only digital activities or some activities that do not require technology? ● When thinking of the activities you have used, 	

	do they always include math, or do you use other content areas?	
	<p>IQ 11:</p> <ul style="list-style-type: none"> ○ When using the Sphero with students, what sort of lessons or activities have you done where students had to identify patterns between older and newer problem solving tasks to solve a new problem? 	Generalization
	<p>Probes:</p> <ul style="list-style-type: none"> ● How effective have those strategies been for you? ● Describe any special challenges in teaching patterns because you're working with K-5 students. 	
	<p>IQ 12:</p> <ul style="list-style-type: none"> ○ Please provide an example of a lesson where students had to use the Sphero robot with a model or some sort of visual representation to solve a problem. 	(Abstractions)
	<p>Probes:</p> <ul style="list-style-type: none"> ○ When thinking of the activities you have used where students use the Sphero with a model or visual representation, what subjects do those activities link to? 	
	<p>IQ 13:</p> <ul style="list-style-type: none"> ○ In your Sphero lessons, when faced with a large code, what strategies have been successful for you in how to teach students to break down the code into smaller sections? 	(Decomposition)
	<p>Probes:</p> <ul style="list-style-type: none"> ○ Why do you think this is an important skill for young coders? ○ Or why haven't you been successful in helping students break down code? 	

	<ul style="list-style-type: none"> ○ How have Sphero lessons motivated students to struggle with large codes? If at all? 	
	<p>IQ 14:</p> <ul style="list-style-type: none"> ○ In your Sphero lessons, how do you teach or introduce the algorithms to students? 	(Algorithms)
	<p>Probes:</p> <ul style="list-style-type: none"> ● How do you connect the vocabulary to the work to help the students understand the concept of algorithm? 	
	<p>IQ 15:</p> <ul style="list-style-type: none"> ○ When faced with challenges, how do you help students to alleviate frustrations with debugging while coding with Sphero robots? 	(Debugging)
	<p>Probes:</p> <ul style="list-style-type: none"> ● If students are working to fix a problem in their code, how do you help them to realize that they are debugging? ● How long do you allow students to work on the problem before you help them to solve a problem when they are struggling with a code? 	

Final IQ.

Is there anything else about teaching with Sphero robots or using Sphero robots to teach computational thinking that we have not yet had a chance to discuss?

Additional questions you might end with:

- *What have you learned about your teaching practice through all of this?*
- *What have you learned about yourself through all of this?*
- *Do you have anything else to add?*

Closing Script: *Thank you so much for your time today. I really do appreciate you sharing your thoughts with me.*

[If snowball sampling] *I am still actively recruiting participants for my study. Do you have contact information for someone who you think might fit [INCLUSION CRITERIA], or would you be willing to forward my study invitation to others you think might be interested? Thanks!*

Thank you for your time and valuable information. When I finish transcribing your interview, I will share it with you via email. If you decide that you do not want to participate, please e-mail me.

Thank you again.

Appendix B: Codebook

Code Name	Code Definition
AI & Future-Ready Thinking	Emphasis on AI, machine learning, and preparing students for future careers.
Algorithmic Foundations	Teachers use real-life analogies and hands-on activities to introduce algorithms as a sequence of step-by-step instructions.
Application Over Definition	Describes the teacher’s emphasis on ensuring students grasp and apply computational thinking concepts, even if they have not yet mastered the formal vocabulary associated with those concepts.
Assessment Practices	Assessment is largely informal and observational, with emphasis on student autonomy and growth. Refers to assessing students not only on their final coding or robotics product but also on their ability to collaborate, communicate, and work as a team during the process.
Autonomy as Evidence of Learning	The teacher uses students’ ability to independently manage robotics activities — without direct instruction — as a clear indicator that they have mastered skills and internalized prior learning. Students are able to articulate their learning and/or thinking process.
Beyond Coding & SEL Integration	Refers to integrating Sphero activities across multiple curriculum areas — including digital citizenship, classroom routines, and social-emotional learning — rather than limiting them to algorithmic thinking or technical skills alone.
Debugging as Discovery	Students are encouraged to identify and correct errors through self-reflection, peer collaboration, and teacher guidance. They work together to solve challenges, often debugging and iterating as a team to find the best solutions.
Embedding Content for Transfer	Acknowledges the teacher’s limited ability to observe students applying robotics learning in other subjects, while intentionally embedding other content areas into robotics lessons to encourage and facilitate cross-curricular knowledge transfer.
Engagement Through Play	Evidence of student excitement, collaboration, and focus. Students view coding as a game, which increases their persistence and resilience. Students are actively engaged through physical interaction with robots, enhancing motivation and learning.
Equity & Access Advocacy	A strong belief in introducing robotics and coding in K–2 to develop interest and confidence early in education. Strong

	belief in starting robotics and coding in K–2 to build interest and confidence early.
Exploration-Based Introduction	Planning lessons and adapting them based on student needs and grade level. The intentional use of open-ended, low-pressure, and curiosity-driven robotics activities enables students to learn computational thinking concepts through engaging with Sphero in a playful and discovery-driven manner.
Failing Forward	Students and teachers view failure as an integral part of the learning process, embracing it as an opportunity to iterate and improve through trial and error.
Flexible Planning & Standards Alignment	Lessons are designed around science and CS standards, but with flexibility and creativity.
Foundational CT Skills	Captures how students are encouraged to plan, anticipate, and mentally rehearse the sequence of steps needed to accomplish a goal using Sphero — reinforcing foundational computational thinking (CT) principles such as sequencing, prediction, logic, and problem solving skills. Recognition that block coding requires explicit instruction and scaffolding.
Frequency of Integration	Captures how often teachers incorporate Sphero robots into their instruction (e.g., weekly, monthly, once per unit).
Fundamental Driving Introduction	Refers to introducing young students (e.g., kindergarteners) to robotics through simple, foundational driving activities focused on basic control skills before introducing programming concepts.
Joy in Coding/Robotics	Captures moments when students express or demonstrate clear enjoyment, excitement, or emotional connection to robotics and coding activities — indicating strong engagement and positive attitudes toward STEM learning.
Professional Growth & Reflection	Teacher’s evolving mindset, adaptability, reflections on teaching practices, and realizations about underutilization and potential for deeper integration. Teacher’s comfort level with Sphero versus other robots (e.g., Indi, Dash).
Professional Leadership & Advocacy	Leads district and state-level PD; adapts instruction based on community needs. Supports other educators in integrating Sphero; advocates for CS methods in teacher prep programs.
Real-World Integration	Sphero activities are embedded in real-world or cross-curricular contexts (e.g., art, science, SEL). Frames computational thinking as a means of teaching problem solving strategies that students can apply beyond coding, recognizing its broader value for real-life and cross-disciplinary contexts.

Robot-Driven Motivation	Working with robots (e.g., Sphero) motivates students to persist through challenges, keep trying, and overcome frustration, thereby fostering a growth mindset and problem solving attitude.
Scaffolding Computational Thinking	Instruction is carefully scaffolded across grade levels, from basic driving to complex coding concepts.
STEM Integration Practices	Leveraging Sphero robotics to apply core STEM concepts like area, perimeter, and geometry through hands-on maze navigation and design-based challenges. Students iterate through the engineering design process, using physical building blocks and maps to plan, build, test, and revise their robot's movements.
Structured Chaos	Teachers manage classroom dynamics by establishing clear rules and boundaries, which support hands-on learning with Sphero.
Teacher as Co-Learner	The teacher models learning alongside students, embracing uncertainty and shared discovery.
Technology Access	Refers to teachers owning their own set of Sphero robots, enabling full autonomy over access, scheduling, and instructional planning.
Technology as Enhancement	Sphero enhances but does not replace core content; used to deepen engagement and application.
Technology Equity & Appropriateness	Choosing tools based on student readiness and accessibility. Refers to deliberately choosing simpler or developmentally appropriate robots (like Indie, Code & Go Mouse, Bee-Bots) to match students' age and cognitive readiness.
Technology Limitations & Adaptation	Early challenges with Bluetooth and device pairing; adapted with labeling and reconnection strategies. Issues with speed control, device connection, and physical handling.
Understanding Shapes	Refers to teaching students how to use logical reasoning to understand and manipulate relationships between speed, duration, and distance when programming robots. Sphero activities are used to teach mathematical concepts, including angles, distance, time, proportional reasoning, and geometry.
Visual Decomposition	Teachers help students break down large coding tasks into manageable parts using visual aids and modeling.
Youth Coding Challenges	Teachers encounter challenges with younger students, requiring adapted strategies for engagement. Captures how limitations in emotions and basic academic skills—such as reading, vocabulary, or early math—create obstacles for students learning to use and program robotics tools like

	Sphero. These barriers often require additional scaffolding to support successful participation.
--	--

Appendix C: Right to Publish Figures

Greetings Marquita,

Congratulations on getting to the dissertation phase of your doctoral journey—and thank you for using the T3 Framework as your guiding model for innovation in education!

I am delighted to provide you with full permission to use the image you requested, as well as any other images or tables from *Disruptive Classroom Technologies*. Your citation can follow the APA formatting rules as you best see fit.

You might also find my research submission in the Oxford University Research Encyclopedia of Education helpful as another source to cite for your literature review (T3 is the only educational technology model to be published in this prestigious publication). Here is the link: <https://maganaeducation.com/wp-content/uploads/2020/11/Magana-Disruptive-Classroom-Technologies.pdf> You may also use this link in your citation/references if you wish.

Please let me know if you have any questions, or would like to chat, as I so enjoy speaking with scholars, like yourself, who are building upon my work. Please feel free to call me on my cell [REDACTED] any time you wish.

Thank you for reaching out and for honoring my work with your own.

All the best,

Sonny

Dr. Anthony J. Magana III, EdD

Sure, go ahead

Sent from [Outlook for Android](#)

...

From: Marquita Jackson [REDACTED]
Sent: Tuesday, November 4, 2025 11:02:12 PM
To: Charoula Angeli [REDACTED]
Subject: Permission to Use Figure from Computational Thinking Framework

Dear Dr. Angeli,

I hope you are doing well. I am writing to request permission to include one of your figures from your framework on computational thinking in my dissertation. Specifically, I would like to include the chart that outlines the elements of computational thinking (e.g., Abstraction, Generalization, Decomposition, Algorithms, Debugging), as developed in your published work.

The figure would be used for academic purposes only and would be fully cited in accordance with your preferred reference format. Please let me know if there are any specific conditions or wording you would like me to include when including it. I have attached the figure I want to use for reference.

Thank you very much for your time and for your influential work in this field.

Warm regards,
 Marquita Jackson
 Walden University