

**Young Children's Processes in Creating Coding Projects across  
Coding as Another Language Curriculum**

A Dissertation Submitted by

Apittha Aim Unahalekhaka

In partial fulfillment of the requirements for the degree of Doctor of Philosophy

in *Child Study and Human Development*

TUFTS UNIVERSITY

May 2023

**Committee Members:**

Marina U. Bers, Ph.D. (chair)

Lynch School of Education & Human Development, Boston College

Sara K. Johnson, Ph.D.

Eliot-Pearson Department of Child Study & Human Development, Tufts University

Remco Chang, Ph.D.

Department of Computer Science, Tufts University

Karen Brennan, Ph.D.

Harvard Graduate School of Education, Harvard University

## Abstract

Early childhood computer science education has rising popularity, as children can acquire 21st-century competencies that are essential for them such as computational skills. Children ages 5-7 worldwide created millions of programming projects on ScratchJr, which is a freely downloadable tablet-based coding application. However, there is a lack of understanding of *how* young children create these projects. The objective of this study was to understand children's processes in creating ScratchJr projects and how the processes were related to block usage outcomes. The first phase of the study was an exploratory pilot study to qualitatively understand key actions and processes when children created ScratchJr projects ( $n = 13$ ). The second and main phase of the study used Google Analytics log data to investigate the processes of a larger group of students ( $n = 153$ ) who were participating in the 24-lesson Coding as Another Language curriculum. The implications of this study are essential to inform how teachers and parents may support young children's creative and exploratory learning processes through education technologies such as tablet-based coding applications.

**Keywords:** project-based learning, process, constructionism, computer science, learning analytics

### **Acknowledgments**

"It takes a village to raise a child." I understood this quote much better from doing a doctorate in child development. The effort parents, teachers, and researchers put into improving children's development is astonishing. I have always known that the love and sacrifice my parents devoted to me was tremendous, but now I realize their hard effort is unquantifiable. Furthermore, I would not be who I am today without my two older brothers, my role models, and my life consultants.

I want to acknowledge all the children, parents, teachers, and coordinators who were part of this research during the COVID-19 pandemic. This dissertation is only possible with their committed participation.

Reflecting on my journey to where I am now, I still cannot believe how far I have come and how long I have been away from home. I will forever be grateful to my primary advisor, Dr. Marina Bers, for taking me on this journey. Thank you for making me into a much stronger person, personally and professionally. Your faith in me kept me pushing through the most challenging moments in this doctoral journey. You deeply inspired me with your passion and vision the first day I walked into your office. You have contributed so much to the early childhood CS field that I will always look up to you.

I want to acknowledge my dissertation committee for their thoughtfulness and support throughout the process. Dr. Sara Johnson, thank you for making multilevel modeling so fascinating. I appreciate all the office hours that were filled with care and made me much more confident with the technical aspects of my work. Dr. Remco Chang, you generously agreed to be on the committee when we never met and were from different departments. Thank you for reminding me to think about the practicality of my work. I am so thankful for the encouragement and great ideas when I came up with vague ideas that seemed impossible to accomplish. To my external reader, Dr. Karen Brennan, I am honored and grateful that you were a part of the committee. Your feedback added a new perspective to the conclusion of my work.

Next, I would like to thank the DevTech Research Group members. All your talent, perseverance, and great personality never stop amazing me. I am so proud to be a part of this lab, and I will never forget

all the laughter and tears we have shared. I am grateful to Dr. Parastu Dubash and Dr. Zhanxia Yang for being my CAL project family. Thank you so much, Karen Bryer, for making the seemingly impossible data collection across multiple classrooms possible. I am also thankful for the fantastic work of my inter-rater reliability team, including Kianie Ramirez and Avery Cohen. It was a blessing to have a doctoral crew, including Dr. Amanda Strawhacker, Dr. Madhu Govind, Dr. Emily Relkin, Tess Levinson, and Alexa Hasse, that shared this journey and provided mentorship and companionship through thick and thin. Thank you, Fei Tan, for being the most supportive friend ever, even before we started this program. To Dr. Kristina Aikens, thank you for hosting writing workshops and helping me to become a much better writer. I was extremely fortunate to receive your ongoing support in revising my dissertation proposal to the final version.

Allium and Miramelinda Montessori Schools, the teaching experience from both schools immensely helped me to formulate the implications of my research work. I had a lot of joy explaining to the children why I was still attending school and how a Ph.D. differs from a lower elementary level. The opportunity to share the rollercoaster world with the teachers and youngsters while writing my dissertation was more than I could ever ask for. On the darkest days, I am truly blessed to have a loving significant other and life-long friends by my side, including Dr. Oat Paritmongkol, Usa Kerdnunvong, Ta Liukasemsarn, and Dr. Ann Jaroenlapnopparat. I can't complete this journey without all of you. I am incredibly privileged to receive such excellent opportunities and support in life. In the future, I would like to provide an outstanding education to more children and help change their lives.

Lastly, the US Department of Education generously supports this research through grant P.R./Award Number: U411C190006 awarded to the DevTech Research Group.

## Table of Contents

<b>Abstract</b> .....	2
<b>Acknowledgments</b> .....	3
<b>List of Tables</b> .....	6
<b>List of Figures</b> .....	7
<b>Introduction</b> .....	8
The need to understand children's creation processes	
Current Study	
<b>Research Questions</b> .....	10
<b>Literature Review</b> .....	10
Computer Science for Early Childhood	
Developmentally appropriate pedagogy, tool, and curriculum	
Constructionism	
ScratchJr	
Coding as Another Language	
Behaviors when creating coding projects: Create, Explore, Revise, Customize	
Understanding Process of Creation	
Tool: Learning Analytics/Google Analytics	
Programming Process	
Coding Blocks Usage In Relation to Learning Experiences	
<b>Current Study</b> .....	27
Hypotheses	
Mixed-method study	
<b>Pilot Phase</b> .....	33
Qualitative Method	
Data Sources	
Procedure	
Data Analysis	

Results	
<b>Main Phase</b> .....	46
Quantitative Method	
Data Sources	
Procedure	
Data Analysis	
Results	
<b>Discussion</b> .....	64
Limitations and Future Work	
<b>References</b> .....	72
<b>Appendices</b> .....	79
Appendix A. Data Screening: Descriptive Statistics	
Appendix B. Data Screening: Pearson’s Correlations	
Appendix C. Results: Longitudinal Regression Models	
Appendix D. Assumption Testing Plots	

### **List of Tables**

Table 1. Specified Lesson Number at each Timepoint for each Grade Level
Table 2. The Lesson Number that each Complex Block gets Introduced
Table 3. Summary of Exploratory Sequential Design Phases
Table 4. Alignment of Exploratory Sequential Design Process and the current study design
Table 5. Description of the Pilot Study Sample
Table 6. Aligning the Current Study Procedure to the Adapted and Integrated Qualitative Approaches
Table 7. Project Creation Actions Coding Scheme
Table 8. Summary of the Main Phase Study Data Source
Table 9. Action Identification Log Rule
Table 10. Summary of the Variables, Data Screening and Post-Assumption Testing for Questions 3 to 5

### **List of Figures**

Figure 1. Coding with ScratchJr

Figure 2. Ada Lovelace Project

Figure 3. A ScratchJr Command

Figure 4. First Grade Classroom with CAL Curriculum Lesson 19

Figure 5. ScratchJr Block Complexity Levels

Figure 6. ScratchJr coding blocks by the end of each time point

Figure 7. Average Proportions of Actions by Experience Levels

Figure 8. Processes of a Child with high Experience

Figure 9. Sequence Plots of all 23 videos

Figure 10. Example of a Project Iteration Process

Figure 11. Example of a Coding Exploration Process that includes Explore and Create

Figure 12. Example of a Long Customization Process

Figure 13. Example of Coding Sections after Long Customization Process

Figure 14. Sequence Plots of Children with Low, Medium, and High Experience

Figure 15. Short ScratchJr Activity Snippet with Four Sections

Figure 16. Proportions of Time spent following each Creation Process across Timepoints

Figure 17. Number of Different Blocks used across Timepoints

Figure 18. Coefficients of Predictors on Variety from the Longitudinal Model

Figure 19. Ratio of Block Adherence to Number of Taught Blocks across Timepoints

Figure 20. Total Unique Blocks by Block Taught at each Timepoint

Figure 21. Coefficients of Predictors on Ratio of Adherence from the Longitudinal Model

Figure 22. Coefficients of Predictors on Block Complexity from the Linear Regression Model

## Introduction

There is a worldwide growth of computer science (CS) education in early grade levels (K-2) as computer programming is extensively crucial across disciplines (Bers, 2018a, 2019). Through coding, children can acquire skills such as computational thinking that can help them solve problems and navigate everyday life (Bers, 2020b; Relkin et al., 2021). One popular and effective CS learning pedagogy is for young children to create coding projects on a block-based programming platform such as ScratchJr (Govind et al., 2020). ScratchJr is a popular and free tablet app that allows young children to create open-ended animated stories and games through coding and designing. ScratchJr has been widely used globally with or without curriculum at home and school, with more than a million active users monthly (Unahalekhaka & Bers, 2021).

However, there is still a limited understanding of children's creation processes or patterns when creating creative projects with ScratchJr. Understanding children's processes when creating their projects is essential. Brennan and Resnick (2012) emphasized that we should assess beyond *what* students learn to *how* they learn. They elaborated that framing computational thinking as concepts is insufficient; we should articulate the framework as "processes of construction" (Brennan & Resnick, 2012, p. 6). A tool that supports processes of construction is the ScratchJr platform that offers different coding concept complexity levels; the more complex blocks have advanced computational concepts for young children, such as conditional statements. Studies found that students' coding block usage patterns can reveal their coding processes, mastery levels, and styles (Emerson et al., 2020; Grover et al., 2017). An example of a pattern is when children explore or self-discover coding blocks. This exploring pattern might be necessary for their coding creations as studies have shown that exploring patterns support young children's scientific reasoning (Cook et al., 2011; Legare, 2014; Lobo & Galloway, 2008). Furthermore, exploration broadens children's experience, leading to more revision and creativity in their solutions (Vygotsky, 2004, p. 15). Therefore, it is crucial to understand children's processes to improve teaching and learning experiences in early CS learning.



This study is part of a larger intervention that implemented the Coding as Another Language (CAL) curriculum with ScratchJr across school districts in the US (*ScratchJr Studies*, 2021). The first goal of this dissertation was to understand children's processes when creating ScratchJr projects and the changes in these processes over the course of participating in the CAL-ScratchJr curriculum. This study captured *the process* through the ScratchJr app usage *actions* from children's screen recordings and Google Analytics log data when they created ScratchJr projects. There were two phases to this study. In the pilot phase, I conducted a qualitative observation of screen recordings to understand how a small group of children with low to high ScratchJr experiences created projects. Subsequently, I identified children's key actions and processes, which were crucial for the study's main phase. The main phase used Google Analytics log data to identify the processes of 153 children when they created ScratchJr projects across time points.

The main phase also covered the study's second goal: to narrow the focus on exploring action related to the coding blocks. Specifically, exploring actions may occur before or after children learn the blocks' functionality from their lessons. Gopnik (2012) reported that children might draw different conclusions when exploring independently compared to exploring after adults teach them. Therefore, this study's second goal was to understand the relationship between children's exploring actions and coding block complexity in their final projects. Findings from this study can be applied to other early childhood coding learning platforms that use open-ended programming platforms not limited to the CAL curriculum.

The dissertation first presents the research questions for both phases of the study. The dissertation then explains a literature review of early childhood computer science and a less common but essential assessment method, process-based assessment. This section also highlights the key behaviors for child development that young children tend to exhibit when creating open-ended coding projects. The paper then goes into methods and results of the pilot and main phases in detail. The following sections are

discussion and implications for both study phases. Finally, this paper closes by describing study limitations and future work.

### **Research Questions**

#### ***Pilot Phase: Identifying Creation Processes (n = 13)***

**Research Question 1:** What processes do children with varying ScratchJr experiences follow to create their ScratchJr projects?

#### ***Main Phase: Creation Processes for Block Variety and Adherence (n = 153)***

**Research Question 2:** What are the proportions of different creation processes at the three time points?

**Research Question 3a:** What is the difference in *block variety* as the CAL program progresses?

**3b:** Which creation process is associated with more *block variety* across time points?

**Research Question 4a:** How much are children using the blocks they learned as the CAL program progresses?

**4b:** Which creation process is associated with more *block adherence* across time points?

#### ***Main Phase: Exploring Action and Block Complexity (n = 153)***

**Research Question 5a:** Do children explore complex coding blocks more frequently before they are taught?

**5b:** What is the relationship between *when* children explore complex blocks (explore on their own or explore after they were taught) and *how* complex their block usage is in their final lessons?

### **Literature Review**

#### **Computer Science for Early Childhood**

Programming skills are in high demand as there is a lack of trained workers in an increasingly technologically dependent workforce (Bers, 2018b, 2019). Many integral objects to our lives use programs, such as personal computers, smartphones, microwaves, and washing machines. Consequently, countries worldwide have been pushing for computer science lessons to start since kindergarten (Webb et al., 2017). However, having these lessons should not focus on filling the future workforce but on the

"future citizenry" (Bers, 2019). Bers presented coding as a new literacy that supports new ways of thinking so children can have a civic voice in the digital economy. Instead of coding as a STEM discipline to fill the future workforce, coding should be considered literacy to provide another platform for children to form and express their ideas (Bers, 2019). In addition, children can significantly benefit from the skills they can acquire from learning to code.

Coding should start early to make a long-lasting impact when brain plasticity is the greatest (Bers, 2018a). Through coding, children can acquire skills such as Computational Thinking (CT), which can be defined as “a set of heuristic reasoning skills that can be categorized into discrete sub-domains applicable to problem-solving in computer science and other disciplines” (Relkin et al., 2021, p. 2). Bers introduced the seven powerful ideas related to CT concepts for early childhood, including algorithms, modularity, control structures, representation, hardware/software, design process, and debugging (Bers, 2020b). CT is highly applicable to everyday life and valuable across disciplines, helping young children problem-solve, make plans, and generate solutions.

There is evidence of the benefits of coding for early childhood development (Çiftci & Bildiren, 2020; Nouri et al., 2020). Nouri et al. (2020) interviewed nineteen K-9 computer science teachers on their perspectives on their students' learning. They summarized the learning benefits in four themes: cognitive skills and attitudes, language skills, creative problem-solving, and collaborative skills and attitudes. Teachers reported that students understood the problems by breaking complex problems down into smaller steps and using diverse solutions. Furthermore, students also showed communicative and collaborative skills when working with peers. Another study on the effectiveness of preschool coding lessons showed children's significant learning gains in non-verbal cognitive abilities such as logical thinking to differentiate figures and ordering (Çiftci & Bildiren, 2020).

### **Developmentally Appropriate Pedagogy, Learning Tool, and Curriculum**

Children's developmental levels can constrain the coding concepts they can acquire. Therefore, the learning tools and teaching pedagogy must be developmentally appropriate for early childhood

classrooms (Relkin et al., 2021; Strawhacker & Bers, 2019; Unahalekhaka & Bers, 2022). For example, studies found that “control flow” (loop and if-then statements) can be challenging for young children to understand and use. Similarly, another study found that first and second graders used conditional statements in their coding projects less often than the other more straightforward concepts (Unahalekhaka & Bers, 2022). These studies suggested that using control flow statements requires children to understand multiple coding concepts, as such statements must be used with the action commands. For instance, if a child wants a character to move forward repeatedly, the command must include a move forward block (action) and a repeat loop block (control flow). Consequently, curriculums should be mindful of when and how teachers introduce complex concepts. Additionally, coding platforms themselves must also be appropriate for young children, especially when they have just started to develop their ability to read texts.

### ***The Benefits of Coding Project Creation***

One developmentally appropriate approach for young children to learn to program is by creating open-ended coding projects (Bers, 2019), as shown in Figure 1 left. Creating computer science projects allows learners to synthesize and deepen their understanding through iterating and exploring alternative solutions (Brennan et al., 2022). Resnick and Siegel (2015) compared this coding project creation approach to learning grammar and punctuation as a step to writing. A study with over 700 children (ages 5-9) reported that children in the experimental group, who created open-ended robotic projects in a seven-week curriculum, significantly improved their computational thinking scores in the post-curriculum (Relkin et al., 2021). In contrast, they found no difference in the computational thinking scores of students in the control group, which did not receive the intervention. In addition to the cognitive skills, studies reported that young children might also develop social skills when they create open-ended projects (Refer to Figure 1 right). Children learn to collaborate with others, such as peers or parents, by engaging in positive communication, including sharing their ideas and final products (Bers, 2020a, 2020b; Govind et al., 2020). Govind and others suggested that children “get to express their knowledge through teaching,

coaching, and/or playing” when they create animated open-ended coding projects with others (Govind et al., 2020, p. 61).

### Figure 1

*Coding with ScratchJr*



**The early ideas of constructionism.** The approach to creating coding projects is supported by the framework of Constructionism. Seymour Papert, the originator of this term, believed that children can construct powerful ideas when they *create* personally meaningful products (Papert, 1980). Children develop ideas rather than passively receive ideas from adults (Kafai & Resnick, 1996). Papert believed that children have a mechanism to help refine their thinking, which may contain misconceptions or false theories (Papert, 1980, p. 132). Building on this idea, Papert brought in technology as a learning tool.

Papert foresaw the potential of the computer for education before personal computers were invented (Holbert et al., 2020, p. 3). In Papert’s *Mindstorms* book, he explained that computers allow children to transform their abstract thoughts into more complex tangible products than what they can typically do in the physical world. This process is valuable as children will not limit their skills to complete drilling exercises but also apply their skills to something more consequential and concrete. Learning also happens when children reflect and share their technological product ideas with others. Constructionism recognizes that children can learn with diverse styles and knowledge representation

(Kafai & Resnick, 1996, p. 3). Moreover, what they learn through creating can also be highly complex and interdisciplinary (Papert, 1980). Papert gave an example of the Logo Turtle, an open-ended programming learning environment where children can give the turtle commands to move around and create computer graphics. Instead of doing geometric exercises on paper, children can create their geometric shapes through Logo programming.

The Logo learning environment provides a microworld, which Papert introduced as an incubator for powerful ideas to occur. A microworld is a space where children can explore, build, and modify their reality and come up with multiple alternatives (Papert, 1980, p. 126). A microworld does not necessarily need computers; another example is a playground that has a structure such as slides and swings that supports children in exploring ideas related to physics (Bers, 2020a, p. 34). Unlike the microworld, where children can test false theories to refine their thinking, schools tend to discourage the crucial process of exploring false theories or thinking errors (Papert, 1980, p. 132). Therefore, schools should incorporate more microworld learning environments for students to initiate their learning exploration.

**Constructionism in more recent contexts.** Constructionism concepts have evolved over the years from the influence of learning sciences, cognitive theories, and technology (Holbert et al., 2020). While the early constructionist ideas focused on learning at an individual level, the focus now expanded to learning at a community level, such as classrooms and virtual platforms (Bers, 2020b; Kafai & Resnick, 1996, p. 6). In their recent book on Constructionism, Hobert et al. (2020, p. 8) explained that new theories propose that cognition is at the intersection of "tools, social interaction, the environment, and the body." Specifically, Sociocultural Theory highlights how learning does not stem from a single student's mind, but learning requires students to interact with their surroundings (Vygotsky, 1978). Therefore, researchers should capture the bidirectional process in how learners interact with the environments, such as peers, instructors, and learning materials (Holbert et al., 2020). Despite these changes, constructionists' core goal is to give children powerful tools to create meaning through constructing products (Holbert et al., 2020).

***Positive Technological Development Framework.*** Bers expanded on Papert's idea of microworlds, which only focused on the child's cognitive development. Bers emphasized the whole child approach as children also encounter social, emotional, and moral domains in their microworlds (Bers, 2020a). Grounded in Constructionism, Bers developed a Positive Technological Development framework that explains how children can develop positive social behaviors when using technology: "technologies can engage children not only in thinking in new ways but also in behaving in new ways." (Bers, 2020a, p. 35) This framework presents six behaviors associated with developmental characteristics that support youth to thrive in life (Lerner et al., 2005). These behaviors are called the 6Cs, including collaboration, creativity, communication, content creation, community building, and choice of conduct. A study showed that children as young as four years old exhibited these 6Cs behaviors when coding with a screen-less programming robot for young children (Bers et al., 2019). The Positive Technological Development framework can be incorporated into the early childhood CS curriculum, which is addressed later in the dissertation.

#### ***Developmentally Appropriate Creation Tool: ScratchJr***

Papert's early thoughts of Constructionism happened when technology was limited compared to the present. Advancements in technology have made it possible for children at the kindergarten level to program their animated projects. However, the programming tool must provide a developmentally appropriate environment like a playground, in which young children can solve challenging cognitive and social problems, explore emotions, and imagine creatively (Bers, 2020b). An example of a learning tool that creates computational playgrounds is ScratchJr, a popular and free tablet app that teaches coding to children ages 5-7 (Flannery et al., 2013). ScratchJr can create a microworld where children can freely explore profound ideas and develop novel approaches to make animated stories and games (Bers, 2020a). ScratchJr's block-based programming language makes it possible for young children to code without having to read text, as children can easily recognize the symbols and color of each coding block (Refer to Figure 2). With ScratchJr, children drag each block from its palette to the coding area and snap together blocks

into a sequence. Then, the characters can act according to the coding command. Like a playground, ScratchJr also has vast project design functions in the paint editor, voice recorder, and character and background libraries. Children not only code but can also paint, clone, or even take a picture from outside to decorate their projects. The ScratchJr environment is comparable to the Logo environment that allows children to share their final products and share the making processes (Flannery et al., 2013; Papert, 1980, p. 180). For instance, children can tap on each character to see the programs that they created. Therefore, ScratchJr can easily be used across home and school settings with and without a formal curriculum.

**Figure 2**

*Ada Lovelace Project*



*Note.* This ScratchJr project created by a first grader is based on the “Ada Byron Lovelace & the Thinking Machine” by Laurie Wallmark. The command on the first page will move the character, Ada, forward seven times and go to the next scene.

***Research-based Curriculum: Coding as Another Language***

Coding is literacy for the 21<sup>st</sup> century. Bers introduced the Coding as Literacy framework, where she compares programming languages to natural languages as expression tools (Bers, 2018a). Both alphabetical literacy and coding literacy are tools that allow people to create meaningful and sharable

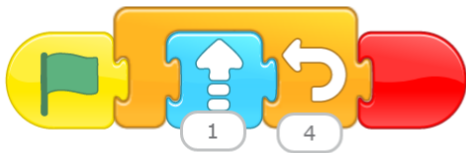


products. Therefore, teachers can model computer science learning from literacy instructions (Bers, 2019). As Bers has noted, “learning to program involves learning how to use a new language for communicative and expressive functions”(Bers, 2019, p. 499).

Using ScratchJr as an example, this programming language has an artificial symbolic system with commands (also called sequences) that run from left to right (Refer to Figure 2), unlike the more advanced programming languages that run from top to bottom. There are rules for ScratchJr sequences just like grammar; for example, each sequence must start with a yellow trigger block. Furthermore, orange control blocks like "repeat" must be used with other block types such as "motion." Consequently, a character will repeat its motion according to the repeat command (Refer to Figure 3). ScratchJr helps children understand the relationship between the execution of each coding block and the character's action by highlighting each block that gets run. Furthermore, children can create stories up to four pages per project on ScratchJr, which allows them to learn about story components (beginning, middle, and end).

**Figure 3**

*A ScratchJr Command*



*Note.* A command with trigger, repeat, motion, and end blocks that tells a character to move up four times.

Although CS has become a mandatory subject for K-12 classrooms in many US states, there is still a lack of research-based CS curriculum and professional development models in early grade levels. To fill in these needs, the Coding as Another Language (CAL) pedagogical approach was created with influences from Constructionism and the Coding as Literacy framework. The field-tested CAL curriculum integrates computer science and literacy instructions for K-2 students, aligning with the Common Core literacy standard and K-12 CS frameworks (Bers, 2019). This curriculum also aims to develop children’s

socioemotional and moral engagement by using activities guided by the Positive Technological Development framework (6 C's). There are twenty-four 45-minute lessons with various activities, which can be off-screen or on the tablet with ScratchJr. Some examples include reading stories, writing plans (Refer to Figure 4 left), unplugged coding (screen-free), and coding with ScratchJr (*Coding as Another Language*, 2022). Some lessons provide free exploration time for children to pursue their interests with ScratchJr. For example, children may revise their codes or try out a different function in the painting tool. According to the CAL curriculum, children must re-create three projects from a song, a make-up story, or a storybook at various time points in the curriculum. Each grade level has two ScratchJr projects that they must create from two storybooks to enhance their literacy skills. For instance, first graders have to read "Ada Byron Lovelace & the Thinking Machine" (Refer to Figure 2 project) and "Where the Wild Things Are." (Refer to Figure 4 right).

#### **Figure 4**

*First Grade Classroom with CAL Curriculum Lesson 19*



*Note.* (Left) A child sharing a project design plan (Right) A teacher telling Where the While Things Are story at the beginning of the lesson.

#### **Key Behaviors in Open-ended Science, Technology, Engineering, Arts, and Mathematics Learning**

This section expands on the essential actions that children exhibit through the process of creating coding projects. As Constructionism explains, when children create customized projects, they learn through exploring and revising their ideas. These key behaviors are supported across literature on

children's STEAM (Science, Technology, Engineering, Arts, and Mathematics) activities and are essential for child development.

### ***Creation***

According to Vygotsky, creative activity drives humans forward at every stage of life. Vygotsky defined creative activity as "any human act that give rise to something new" (Vygotsky, 2004, p. 7). Vygotsky categorized two activities, reproductive and creative (also called combinatorial). The *reproductive activity* is when a person repeats the previously developed patterns with no new creation. For instance, a child repetitively creates the same coding sequence with no novelty or adaptation. This process is comparable to how Papert explained the first step of the cognitive process, which is to recall an existing concept. However, if humans are only limited to recalling the previous experience with no changes, we would fail to adapt to unexpected situations (Vygotsky, 2004, p. 8). In contrast, the *creative activity* is to create something new based on previous experience. Vygotsky also talks about creativity interchangeably with imagination and fantasy (Ayman-Nolley, 2009). Creativity is crucial for "all aspects of cultural life, enabling artistic, scientific, and technical creation" (Vygotsky, 2004, p. 9).

Creativity, which stems from play, is essential for children's development. Vygotsky noted that a child's play is not a reproductive activity but a "creative working of the impressions he has acquired" (Vygotsky, 2004, p. 11). Therefore, the richness of children's creativity depends on their accumulated experience based on reality. A popular misconception is that only famous inventors are creative. However, creativity happens when a person, such as a child, imagines, revises, or creates something personally novel, regardless of the scale of the impact (Vygotsky, 2004, p. 10). Similarly, Resnick (Resnick, 2017, p. 19) described the difference between the "Big-C Creativity" inventions that are new to the world, and the "little-c creativity" inventions that can be small but personally useful in everyday life. He suggested that educators focus more on the little C to help students reach their full creative potential. Moreover, the best way to cultivate creativity is through playful project creation according to children's passion while collaborating with others (Resnick, 2017).

## ***Exploration***

Exploration starts early. Exploratory behavior is vital for children's learning from birth as it helps young children gather information from the environment, which leads to knowledge development and scientific reasoning (Cook et al., 2011; Legare, 2014; Lobo & Galloway, 2008). Newborn babies explore by gazing into the immediate environment (Gibson, 1988). Once babies can move around, they shift their exploration to the features and functionality of objects and then eventually manipulate the objects. Gibson (1988) hypothesized that understanding object affordances are a foundation for causal knowledge, evident in a more recent study with preschoolers (Schulz & Bonawitz, 2007). Similarly, a different study found that preschoolers have “intuitive theories of the physical, biological, psychological, and social world” (Gopnik, 2012, p. 1623). A study compared children (ages 4-7) and adults, and found that children explored more than adults even when they knew that the reward from their actions might not be worth the effort (Liquin & Gopnik, 2022). Young children’s thinking mechanisms are similar to the processes of science, where they explore and test their hypotheses about the world (Gopnik, 2012).

Exploration can be spontaneous or guided by instructions (Evangelou et al., 2010). However, children's exploration may differ when exploring independently than after receiving instructions from adults (Gopnik, 2012, p. 1626). Bonawitz et al. (2011) claimed that instruction promotes efficient learning; however, it comes at the cost of children being more likely to stop exploring novel information. They found that children tend to believe what adults showed them to be effective and infer that there is no more information to discover. Consequently, they explore the other information or possibilities less (Bonawitz et al., 2011). Bonawitz and others (2011) showed that preschoolers who were taught directly or indirectly about the toy functionality played with the toy less frequently, resulting in less discovery of the other toy functions. The study implied that delaying instructions for children to explore on their own may promote innovation. However, the benefits of doing so may vary as it depends on children’s unscripted action and whether the teacher knows what undiscovered information is available for children to explore independently. For instance, a study with third and fourth-grade students reported that more children learn

experimental science planning from direct instruction than discovery learning (Klahr & Nigam, 2004). Another study also found that guided instruction is more beneficial when children have no background knowledge of the learning topic, as foundational content is essential for open-ended tasks (Kirschner et al., 2006). Therefore, self-discovery and instruction-led exploration actions might both be necessary for children's learning.

Exploratory behavior is evident across numerous early childhood STEM (Science, Technology, Engineering, and Mathematics) studies (Evangelou et al., 2010; Lavigne et al., 2020; Strawhacker et al., 2020). A study with preschoolers found that exploratory behavior is essential to encourage early engineering thinking (Evangelou et al., 2010). Moreover, artifacts or tangible objects may promote exploration better than a book or sketch version of objects. A different study used CRISPEE, a tangible robotic tool that allows children ages 5-7 to learn about biodesign through programming. They found that children in individual sessions spent a great amount of their time exploring the tool's interface while spending less time building or testing programs (Strawhacker et al., 2020). Similarly, another study on computational concepts reported that preschool children who were asked to build animals from various shapes might be more motivated to explore than to build efficiently (Lavigne et al., 2020). The study's key takeaway is that young children tend to playfully use multiple ways to create rather than thinking about the most efficient steps.

### ***Revising***

After exploring new information, children may use the acquired knowledge to modify their work. Children regularly refine their coding projects by changing the ideas to use a different approach (Bers, 2020b; Papert, 1980). The essence of Constructionism is learning as creating, where the product keeps evolving through iterations and might not always stick to the pre-established plan (Brennan & Resnick, 2012; Harel & Papert, 1991, p. 6).

The ability to revise existing knowledge or theory with new information is essential for learning (Feldman & Fowler, 1997; Papert, 1980). This claim aligns with the Theory of Conceptual Change in

science learning and explains how encountering and processing new information to revise current understanding may lead to learning (Posner et al., 1982). When students face a new phenomenon, they must use existing concepts to explain it (assimilation). However, if they cannot use the current concepts to explain a new phenomenon, they will have to reorganize or revise their central concepts (accommodation). Posner et al. (1982) further elaborated on the revision phase. They explained that conceptual change would occur when the learner is unsatisfied with the existing concept, and the new concept can potentially solve the problem. In a more recent study on scientific thinking, Kuhn (2010) explained that children acquire knowledge through conceptual change by revising their theories, especially when the early theories are likely to be incorrect and incomplete. In addition to children revising their beliefs, revision actions on tangible objects can be seen across studies (Bairaktarova et al., 2011; Brennan & Resnick, 2012).

An early engineering study observed children (aged 3-5) free play with different tangible materials such as blocks, sandboxes, and water tables (Bairaktarova et al., 2011). The study's objective was to find similarities between children's play behavior and indicators of engineering behavior. One of the key behaviors that they found was problem-solving, which they explained as the willingness to "rework, redo, or redesign something to improve function of object or process" (Bairaktarova et al., 2011, p. 228). They compared this behavior to when engineers make a product that is an extension of their past creations. They observed children's reworking behavior with the more structured artifacts, including puzzles and circuits (Bairaktarova et al., 2011, p. 228).

Similarly, a coding environment like ScratchJr also has structure that may allow reworking. Another study with Scratch, a programming environment similar to ScratchJr but for children ages 8-16, observed and interviewed students on their projects (Brennan & Resnick, 2012). One of the primary practices they observed was "being incremental and iterative," which is how a project design is an adaptive process requiring small iterative steps to arrive at a solution. Children iterated their ideas and creations, which they may switch between testing and developing their Scratch projects. Creative learning

is a process that "involves moments of getting stuck, not knowing, being wrong, and failing" (Brennan, 2015, p. 280)—aligning with an idea in Papert's *Mindstorms* book that the act of refining misconceptions is crucial for learning (Papert, 1980, p. 132).

### *Customizing*

Papert encouraged the idea of not separating imagination from children's deeply personal work. He gave an example of a child who wanted to creatively incorporate math and art together when programming a graphical figure (Harel & Papert, 1991, p. 6). Constructionism highlights the importance of giving students time to visualize their ideas in meaningful ways, and their representation may change through reflection and connection with others (Sheridan, 2020). Thus, Sheridan (2020, p. 323) compared constructionist learning to studio art education, which is "about exploring form and meaning of representations." Students in art classes often have to work over multiple sessions to create a product from open-ended prompts, similar to how young children make their ScratchJr projects.

Project customization is essential for children to communicate and express ideas. A study assessed ScratchJr projects created by first and second graders at the end of their 24-lesson curriculum (Unahalekhaka & Bers, 2022). They found that the children highly customized their coding projects, especially when their projects were animated stories with multiple pages of storylines. Visual representation, such as characters and background, might help children connect ideas to coding. A literacy study presented young children's (aged 4-6) drawing as a form of visual text construction, which supported their writing. They reported that children might use drawing and writing text "together in order to create messages that are more complex than those that they could create with either drawing or text alone" (Mackenzie & Veresov, 2013, p. 27). This study elaborated that visual text creation skills like drawing can help reduce the interruption of self-expression when children are still learning to write. Furthermore, the highly customized projects might become more personally meaningful to children. For example, children can personalize their ScratchJr projects more extensively by taking photos and recording their voices in their stories.

These examples highlight a pedagogical shift from STEM to STEAM by including art disciplines (Liao, 2016). One reason that supports this shift is to engage students with more real-world applications of STEM. Furthermore, research shows that STEAM can enhance children's skills in creativity, innovation, problem-solving, and engagement in the STEM fields (Perignat & Katz-Buonincontro, 2019). STEAM education should focus on the making process, such as "exploration, creative thinking, designing, technique, creative-expression, critique, evaluation, and redesign" rather than the end product (Perignat & Katz-Buonincontro, 2019, p. 35). Therefore, it is crucial for practitioners to think about how they can effectively assess these processes.

### **Why is understanding process important?**

Studies have emphasized the importance of research examining learning processes or patterns from exploratory activities about a particular concept (Blikstein et al., 2014; Brennan et al., 2021; Holbert et al., 2020; Tissenbaum, 2020). The process can be defined as *how* a task is completed, which can be reflected in patterns of actions (Gomez et al., 2021; Iseli et al., 2021). When children design sharable artifacts, learning is more importantly reflected through the process of creating than from the final product. Kafai (1996) elaborated that learning takes place over time as children get to iteratively plan, problem-solve, and design, despite the outcome of the product. Accordingly, it is valuable for educators to understand children's learning processes to support their needs and motivation (Ryan & Deci, 2020, p. 6). However, there is a limitation to how much educators can accurately capture the learning process of each student during lessons. Technology not only supports children's learning but can also assist teachers in evaluating their students.

### ***Learning Analytics***

Although computers started to become prominent instructional tools in the 1980s (Reiser, 2001), data collection in educational settings was done manually until less than 30 years ago, before the mass scaling of technology (Krumm et al., 2018). At present, computers can collect thousands of data points when students interact with learning tools. This phenomenon led to learning analytics, which uses static



and dynamic information about learners and learning environments for educational purposes (Ifenthaler & Yau, 2020).

The field of learning analytics has flourished due to its potential to capture children's continual and real-time interactions with learning platforms (Admiraal et al., 2020; Berland et al., 2014). For example, the log data of a game application may show all the students' actions on the application across timestamps. While some traditional assessments (e.g., tests) evaluate learning outcomes, they cannot capture learning processes at a large scale without close human observation (Blikstein, 2011). Studies have found that this kind of data about students' education technology usage patterns can provide details about their mastery levels and learning styles (Emerson et al., 2020; Grover et al., 2017).

**Google Analytics** is a web analytics service offered by Google to give small to medium-sized companies insights into their customers' behavior around their products (*Google Analytics*, 2020). ScratchJr has been using Google Analytics to collect children's usage data since 2016 (Leidl et al., 2017). The purpose of collecting this data is to understand how and where children use ScratchJr. Therefore, one of the data types that the ScratchJr team has been collecting is the number of users at a particular geographic location (country, region, and city). Furthermore, the data are not identifiable as all the users' personal information, including IP addresses, gets erased. These log data have the potential to help researchers understand children's ScratchJr usage patterns, from which they can derive creation processes.

### ***Programming Process***

Computer science is more than writing codes, as it also involves thinking processes. A study described that using block-based analysis alone limits researchers understanding of children's learning; instead, "focusing on process presents an opportunity to explore the computational thinking that is incompletely represented by blocks" (Brennan & Resnick, 2012, p. 23). A different study suggested that teachers should evaluate coding projects multiple times to illustrate student learning processes (Brennan et al., 2021) and use these insights to give feedback to guide their students' next steps.

Learning analytics is a tool that can capture students' learning processes. Berland (2013) used learning analytics to categorize pathways of the programming process that high school students exhibited, including exploration, tinkering, and refining. However, it is unknown whether there are similar processes in young children's programming. A study analyzed first graders' coding process data from solving puzzles on codeSpark, an app that exposes children to computational concepts (Iseli et al., 2021). Although the coding processes were not validated, they found multiple patterns that can be linked to computational thinking concepts. For example, when children try to achieve a part of the goal before adding more commands, that is comparable to the computational thinking concept of decomposition, which is to break a problem into smaller approachable goals.

Another study implied that children who created open-ended projects with ScratchJr at home might use different processes than those at school (Unahalekhaka & Bers, 2021). The study utilized Google Analytics to collect children's ScratchJr usage data from over four million sessions in the United States. On average, they found that children at school used lower coding block variety than children at home. Consequently, children at school also used fewer intermediate and advanced coding blocks. Children at home seemed to have a more exploratory play with ScratchJr, as they also spent more time using the painting tools to customize their projects. This study implied that project creation processes that children used are related to their block usage levels and learning.

### **Coding Blocks Usage in Relation to Learning Experiences**




Not only are block usage levels and creation processes related, but usage levels can also indicate children's CS learning experiences (Emerson et al., 2020; Unahalekhaka & Bers, 2021; Wang et al., 2017). Some of the most highlighted block usage types across studies include complexity and variety (Franklin et al., 2017; Portelance et al., 2016; Strawhacker & Bers, 2019; Unahalekhaka & Bers, 2022). Various studies used block complexity to differentiate coding competency levels between K-2 students (Portelance et al., 2016; Strawhacker & Bers, 2019). A different study with upper elementary students reported how sixth graders used the highest block category variety in their Scratch projects compared to

younger students learning the same concepts (Franklin et al., 2017). This means that CS concepts and instructions should be developmentally appropriate for each grade level. Another block usage type necessary for CS learning is adherence, or how closely children can use all the blocks taught from the curriculum. Although young children vary in interests and learning paces (Harvey & Horton, 1977; Ryan & Deci, 2020; Tomlinson, 2005; Vygotsky, 2011), it is still crucial that they at least meet the minimum objective of the curriculum. Research has shown better student outcomes when teachers delivered more content from the curriculum (Lillehoj et al., 2004; Pettigrew et al., 2015). This may be why adherence is often used in program evaluation to determine how well educational programs could reach their objectives (fidelity of implementation). Therefore, the current study used block variety, adherence, and complexity as the outcomes of children's participation in the 24-week Coding as Another Language (CAL) program with ScratchJr.

### **Current Study**

The overarching objective of the current study was to understand children's ScratchJr project creation *processes*, which this study defines as the sequencing and switching of children's actions to create ScratchJr projects. Furthermore, the study linked children's creation processes to their ScratchJr coding block usage, which may be related to their coding mastery levels and learning experiences (Emerson et al., 2020; Price & Price-Mohr, 2018; Unahalekhaka & Bers, 2021). This study focused on three usage types, including block variety, block adherence, and block complexity. *Block adherence* to the curriculum was the number of recently taught blocks used by the child in the project. *Block variety* was the number of unique blocks children used. *Block complexity* was the number of intermediate and advanced blocks children used (Refer to Figure 5). This study measured block usage variety and complexity as creation is more meaningful for learning when children newly create something that has not been repeated (Ayman-Nolley, 2009). Furthermore, block adherence can still be important for learners according to the curriculum's objectives.

**Figure 5**  
*ScratchJr Block Complexity Levels*

Beginner	Intermediate	Advanced
<p>Green start, motion blocks, single character, say block, looks blocks (grow, shrink, hide, show), pop sound, record sound, end block</p> 	<p>Start on tap, control speed, wait time, return to start, go to page, repeat forever</p> 	<p>Start on bump, start on messages, send message, repeat, stop block</p> 

There were two phases in the current study to understand how different processes in creating ScratchJr coding projects related to how children used various coding blocks. The pilot phase was exploratory and observed processes in how children ( $n = 13$ ) with varying ScratchJr experience levels create ScratchJr projects. The main phase investigated the occurrences and relations of processes (from the pilot study) across more students ( $n = 153$ ) at three time points in the CAL curriculum using Google Analytics log data. The three time points as shown in Table 1 were chosen as the lessons children created major projects in the CAL curriculum. The two study phases were connected as the main phase quantified the different types of creation processes from the pilot phase.

**Table 1**

*Specified Lesson Number at each Timepoint for each Grade Level*

Grade	Timepoint 1	Timepoint 2	Timepoint 3
First	Lessons 5-6	Lessons 9-10	Lessons 19-20
Second	Lessons 6-7	Lessons 10-11	Lessons 19-20

### **Pilot Phase**

#### ***Pilot Phase: Identifying Creation Processes***

**RQ1:** What processes do children with varying ScratchJr experiences follow to create their ScratchJr projects?

#### ***Main Phase: Creation Processes for Block Variety and Adherence***

The main phase connected creation processes from research question one to how children use coding blocks (variety, adherence, and complexity). Therefore, the second research question identified the occurrences of processes across three timepoints in the CAL curriculum. I picked three-time points according to the lessons in which children created major ScratchJr projects. The third question investigated which process type was related to the greatest block variety. The CAL curriculum covers different blocks across the lessons; therefore, children should learn an increasing number of blocks as time passes (Refer to Figure 6). Thus, the fourth research question investigated whether children use the blocks they just learned and how this usage relates to the processes they used.

**RQ2:** What are the proportions of different creation processes at the three-time points?

I hypothesized that the exploration process would occur more frequently in earlier times, whereas customization would happen more prominently at the second and third timepoints. The reason is because children might explore blocks when they are still new to ScratchJr as they are curious about different block functionalities. Additionally, I hypothesized that customization would increase after timepoint one as children will learn how to use painting functions. I also expected that there would be other processes

not covered by the three identified process types (from RQ1). However, I mainly focused on the three types.





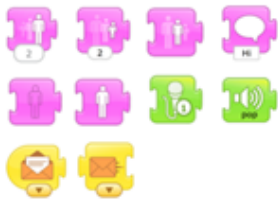

**RQ3:** a) What is the difference in *block variety* as the CAL program progresses?

b) Which creation process is associated with more *block variety* across time points?

I hypothesized that there should be an increasing number of block varieties across time points as children learn more block types. Children who explored more coding blocks would have the greatest block variety as children with this process type tend to try out unknown blocks. The next question aimed to understand whether children use blocks that they were recently taught in class as part of the CAL curriculum to create open-ended projects.

**Figure 6**

*ScratchJr coding blocks that should be covered by the end of each time point*

Grade Levels	CAL Curriculum Timepoints		
	1 Beginning	2 Middle	3 End
First			
Second			

**RQ4:** a) How much are children using the blocks they learned as the CAL program progresses?

b) Which creation process is associated with more *block adherence* across time points?

I hypothesized that children would increasingly use blocks they recently learned as the lessons progressed. Children would have more experience incorporating more blocks that they learned into their

projects. Furthermore, a process with a lot of customization would be associated with lower adherence as children would have less time to code.

### ***Main Phase: Exploring Action and Block Complexity***

The last research question examined the relationship between exploring *action* and block complexity in their final lessons. This study focused on "Exploring action" instead of "Exploration process" that might also include creating and revising actions. I defined complexity as the total number of intermediate or advanced block levels used (Refer to Figure 5). Further, exploring is split into two types, exploring blocks *before* versus *after* the lesson introduces the blocks (Refer to Table 2). In another words, exploring before the lesson means children try out complex blocks that have not been taught, vice versa. It is important to note that children will have fewer blocks to explore ahead of the curriculum.

**Table 2**

*The Lesson Number that each Complex Block gets Introduced*

Complex Block Levels	Block Type	First Grade Lesson	Second Grade Lesson
Intermediate	Tap	14	15
	Speed	15	16
	Wait	16	16
	Go to start	15	2
	Forever	6	16
	Go To Page	10	6
Advanced	Bump	14	15
	Send message	18	9
	Start on message	18	9
	Repeat	17	13
	Stop	14	19

**RQ5:** a) Do children explore the complex coding blocks more frequently before they were taught?

b) What is the relationship between when children explore complex blocks (explore on their own or explore after they were taught) and how complex their block usage is in their final lessons?

According to prior studies, children stopped exploring for novel information after they were given the information. Therefore, I hypothesized that children would explore the coding blocks less after the lessons covered them and exploring durations (before and after) will positively relate to block complexity. However, I also hypothesized that children who explored ahead of the curriculum would use more complex blocks at the end of the curriculum because they would be exposed to the complex blocks earlier and have more time to try out the new information they gained from exploring.

### Method

The overall study used a sequential mixed-method approach with a fixed design, meaning that the use of quantitative and qualitative studies was predetermined. The study took a typology-based approach, particularly, Exploratory Sequential Design (Creswell et al., 2018). The two phases in the current study aligned with the three Exploratory Sequential Design Phases (Creswell et al., 2018, p. 135), which is shown in Table 3. According to Creswell et al. (2018), this design is useful to explore important concepts that are unknown in quantitative studies. The unknown concept in this current study was young children's creation processes when creating coding projects, as there was still a lack of literature in this area.

**Table 3**

*Summary of Exploratory Sequential Design Phases (Creswell et al., 2018, p. 135)*

Phases	Description
Qualitative	<ul style="list-style-type: none"> <li>● Defining qualitative research questions</li> <li>● Collecting and analyzing qualitative data to explore phenomenon</li> </ul>
Bridging Qual-Quant	<ul style="list-style-type: none"> <li>● Designing or building qualitative feature based on the qualitative results</li> </ul>
Quantitative	<ul style="list-style-type: none"> <li>● Defining quantitative research questions</li> <li>● Collecting and analyzing quantitative data</li> </ul>



Rather than having three phases as Creswell et. al. suggested, the current study had two, “Qualitative” study as the pilot phase and the “Bridging Qualitative and Quantitative” study and “Quantitative” study as the main phase (Refer Table 4). I conducted a qualitative study with a small group of children to identify types of creation processes when creating open-ended ScratchJr projects from their tablet screen recordings. Subsequently, I quantified these processes to turn them into core variables for the main study. The following sections explain the method of the pilot and main phases in more detail.

**Table 4**

*Alignment of Exploratory Sequential Design Process (Creswell et al., 2018, p. 135) and the current study design*

Exploratory Sequential Design Phases	Aligned Study Phases	Aligned Research questions	Method
Qualitative	Pilot	RQ1	Task-analysis and deductive coding approach
Bridging Qual-Quant	Main	RQ2	Rule-based identification
Quantitative	Main	RQ3	Multilevel Regression
		RQ4	Multilevel Regression
		RQ5	T-test and Linear Regression

### Pilot Phase

#### Data Sources

The total of 23 ScratchJr screen recordings came from three sources, an online study and two in-person classrooms. In the online study, I recruited six children from K-2 levels by sending a recruitment email to the ScratchJr e-list, which has emails of parents, teachers, and school administrators. Six children participated resulting in three 30-40 minutes recordings per child. However, this study only included screen recordings from five children due to a technical issue with one child's screen recordings (Refer to Table 5). I determined children’s level of experience in using ScratchJr from a pre-survey

question that asked parents or teachers how much experience each child has, ranging from little, moderate, and high.

For the in-person classes, both classes were learning with the CAL curriculum. One classroom was from a preschool in Medford, MA, and the other was from an elementary school in RI. This study took screen recordings from different points in the curriculum when teachers asked children to create open-ended projects on ScratchJr. I labeled screen recordings from Lessons 4, 10, and 20 as children with little, moderate, and high ScratchJr experience levels, respectively. In total, there were seven videos for children with little ScratchJr experience, and eight videos each for children with medium and high experience.

**Table 5**

*Description of the Pilot Study Sample*

IDs ( $n = 13$ )	Sex	Race	Grade	ScratchJr Experience	Study Format	#Video
Child A	Male	Black	1	High	Online	3
Child B	Male	White	1	High	Online	3
Child C	Male	White	1	Medium	Online	3
Child D	Female	White	2	Medium	Online	3
Child E	Male	White	2	High	Online	3
Child F	Female	White	1	A little	In-person	1
Child G	Female	White	1	A little	In-person	1
Child H	Female	White	1	A little	In-person	1
Child I	Male	Hispanic	1	A little	In-person	1
Child J	Female	White	1	High	In-person	1
Child K	Female	White	K	High	In-person	1
Child L	Male	White	K	Medium	In-person	1

Child M	Male	White	1	Medium	In-person	1
---------	------	-------	---	--------	-----------	---

---

### **Procedure**

In the online study, I met each child and their parent or teacher over three 45 minutes sessions, where I asked the child to create open-ended projects according to the prompts. Parents and teachers were permitted to answer the child's questions and give hints; however, they were encouraged not to provide major guidance or initiate new ideas to the child. The format of the prompts was inspired by the CAL curriculum, which has free play in the early lesson and storybook recreation in the middle and the end lessons. Therefore, the first session was free play, so they could create anything they wanted. In the second session, I played an e-book called "How to Code a Sandcastle" by Josh Funk. I then asked the children to re-create this story, which is about a girl that programs a robot to help her build a sandcastle. In the third session, the children got more time to work on their sandcastle project; otherwise, they got to work on any projects they wanted if they were done early.

Children from in-person classrooms had 24-lesson CAL-ScratchJr intervention. I could not enter the schools in person due to their COVID-19 regulations; consequently, I asked teachers to screen record when children create projects and send them to the DevTech Research Group. The teachers generously supported my request but could only screen record when they were available from teaching. Therefore, the screen recordings vary from 10-30 minutes and did not always start from the beginning of the session. In lesson 4, the children had a free play session. Then children re-created two stories in lessons 10 and 19, "Ada Byron Lovelace & the Thinking Machine" by Laurie Wallmark and "Where the Wild Things Are" by Maurice Sendak.

### **Data Analysis**

The qualitative phase of this study (Refer to Table 4) adapted task-based analysis by integrating the deductive coding analysis approach to observe and analyze children's actions and processes from their ScratchJr screen recordings (Refer to Table 6). This study had to adapt the task-based approach to make the steps more suitable in an open-ended creation process with no preferred outcome. In contrast,

task-based analysis was originally created to observe the “most necessary and efficient steps toward the intended goals” in psychotherapy (Pascual-Leone et al., 2009, p. 529). Task-based analysis has been successfully applied to the other aspects of educational contexts. For instance, a study used task analysis to discover children’s mathematical disposition when creating computer games on Scratch (Ke, 2014).

Pascual-Leone et al. (2009) stated the objective of task-based analysis is to discover necessary steps in reaching a task’s goal. They split task analysis into two phases- discovery and validation. The Discovery phase “blends observation and plausible theory” (Pascual-Leone et al., 2009, p. 529); therefore, the first step of this approach is to use existing theory and frameworks to guide observation. The following steps are to come up with categories and quantify them. Subsequently, the Validation phase tests the newly developed coding scheme across new cases and establishes interrater reliability.

In addition to task analysis, I used Erickson’s deductive video coding approach, which is derived from a classroom interaction observation method (Erickson, 2006). This approach locates every occurrence of the action of interest without sampling every timeframe (e.g., 30 seconds) of the entire video. The last steps are to find the frequency of occurrences and then describe some of the occurrences in detail. The alignment of the adapted task-based and deductive analysis approaches and the current study procedure is shown in Table 6.

**Table 6**

*Aligning the Current Study Procedure to the Adapted and Integrated Qualitative Approaches*

	Task-based Analysis and Deductive Data Analysis (Erickson, 2006; Pascual-Leone et al., 2009, p. 531)	Current Study Procedure
Discovery Phase	1) Rational Model, using existing theory to guide qualitative observation	Literature reviewed young children’s actions when creating open-ended STEM projects

2a) Empirical Modeling: synthesize and organize phenomena into categories,	Designed and iterated a coding scheme with four key actions that were exhibited from the screen recordings.
2b) Deductive Video Coding:	The main researcher watched 23 video recordings and only marked the time span when the phenomenon (action) occurs.
3) Synthesis and Measurement: finding model of change, quantifying actions. Then derived creation processes from categories, or measurement of participants' progresses across time	Quantified and visualized prevalence of analyzing the action plots.
Validation Phase	1) Establish interrater reliability The second researcher coded all 23 videos by using the finalized coding scheme.

---

I started the pilot study with a literature review, which showed that children commonly display key actions when creating tangible STEAM projects, such as tinkering, refining, exploring, and customizing. I then used these findings to guide my observation on children's actions from three screen recordings, while forming a coding scheme (Refer to Table 7). The coding scheme was stable and finalized after coding three more videos. Following the finalized coding scheme, I coded the actions on NVivo as they appeared in the rest of the 23 videos. To establish interrater reliability, a second rater used the finalized coding scheme to code 23 videos. The codes were aligned approximately 96% of all time points across two raters. We discussed and resolved any coding misalignment.

**Table 7***Project Creation Actions Coding Scheme*

Categories	Actions	Definition
Coding	Create New	Start a new code from blank (excluding voice recording) with no switching to non-coding actions
	Revise	Rework on the same code after playing program or doing non-coding actions
	Explore	Try out intermediate or advanced blocks repeatedly
Project Design	Customize	Decorate project or add character/background

In the synthesize and measurement steps, I compared occurrences and sequences of actions across student groups. Therefore, I found the duration of each action by converting the timestamps and visualizing the sequences of actions in R. Before plotting, I turned raw action frequencies into proportions as the video durations differed across participants. The next step was to derive different processes that children used to create their ScratchJr projects to address the main goal of the pilot study. I achieved that by analyzing the action sequence plots and grouping actions into processes.

### Results

**Research Question 1:** What processes do children with varying ScratchJr experiences follow to create their ScratchJr projects?

The first research question examined the creation processes children with varying ScratchJr experiences follow to create their coding projects. As a foundation to understanding the *processes*, it is important to uncover the *actions* that children commonly exhibited.

#### *Frequencies of Four Key Actions*

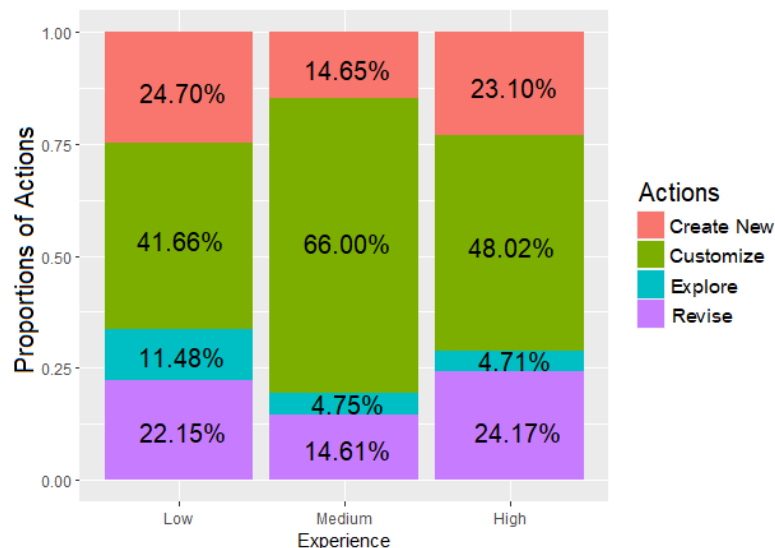
From the deductive video coding analysis, the four actions that stood out when children created their ScratchJr projects were create, revise, explore, and customize. Overall, children spent the most time (55.19% on average) aesthetically *customizing* their coding projects. As part of customization, this study

also included the action of adding a character or background to expand the story. Although adding new characters is linked to creating programs, many added characters did not have programs. The other study reported how children often add new characters to decorate the story background without the intention to program them (e.g., a sun in the sky) (Unahalekhaka & Bers, 2022). Children on average spent a similar amount of time *creating new* (19.46%) coding sequences and *revising* (19.31%) the created sequences. Children explored coding blocks that they had not learned yet, most commonly with complex (intermediate and advanced levels) blocks (Refer to Figure 5) but *exploring* on average appeared the least frequently (6.04%).

There were similarities in how children across experience levels used their time, as shown in Figure 7. Particularly, all groups of children spent, on average, the most time on customizing, similar average amounts of time on creating new and revising, and the least time on exploring complex blocks. Even though these proportions were similar across experience levels, however, there was still a difference in the overall amount of time they spent on each action. For example, children with a medium level of ScratchJr experience spent an average of 66% of their time customizing, which was more than the two other experience level groups.

### **Figure 7**

### Average Proportions of Actions by Experience Levels

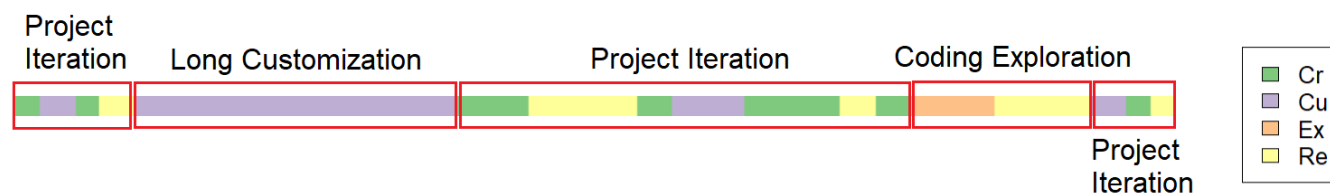


### Understanding the Three Creation Processes

I next examined the sequence plots of the four key actions (for example, see Figure 8). Across these plots, three types of processes became apparent (shown in red boxes in Figure 8). These processes could occur in any order, as young children seemed to be following these processes diversely (Refer to Figure 9 for an illustration of processes across all 23 videos).

**Figure 8**

*Processes that a child with high experience used to create ScratchJr project in one session*

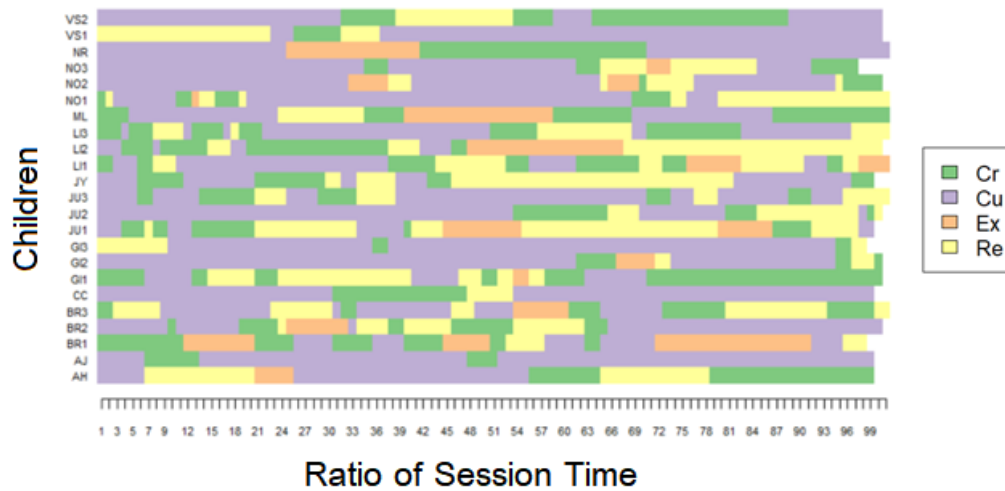


*Note.* Cr= Create New, Cu= Customize, Ex= Explore, Re= Revise

**Figure 9**

*Sequence Plots of all 23 videos with Y-axis as Children and X-axis as Ratio of Session Time*





*Note.* Cr= Create New, Cu= Customize, Ex= Explore, Re= Revise. Some sessions started with revising as children edited their existing projects.

### ***Project Iteration***

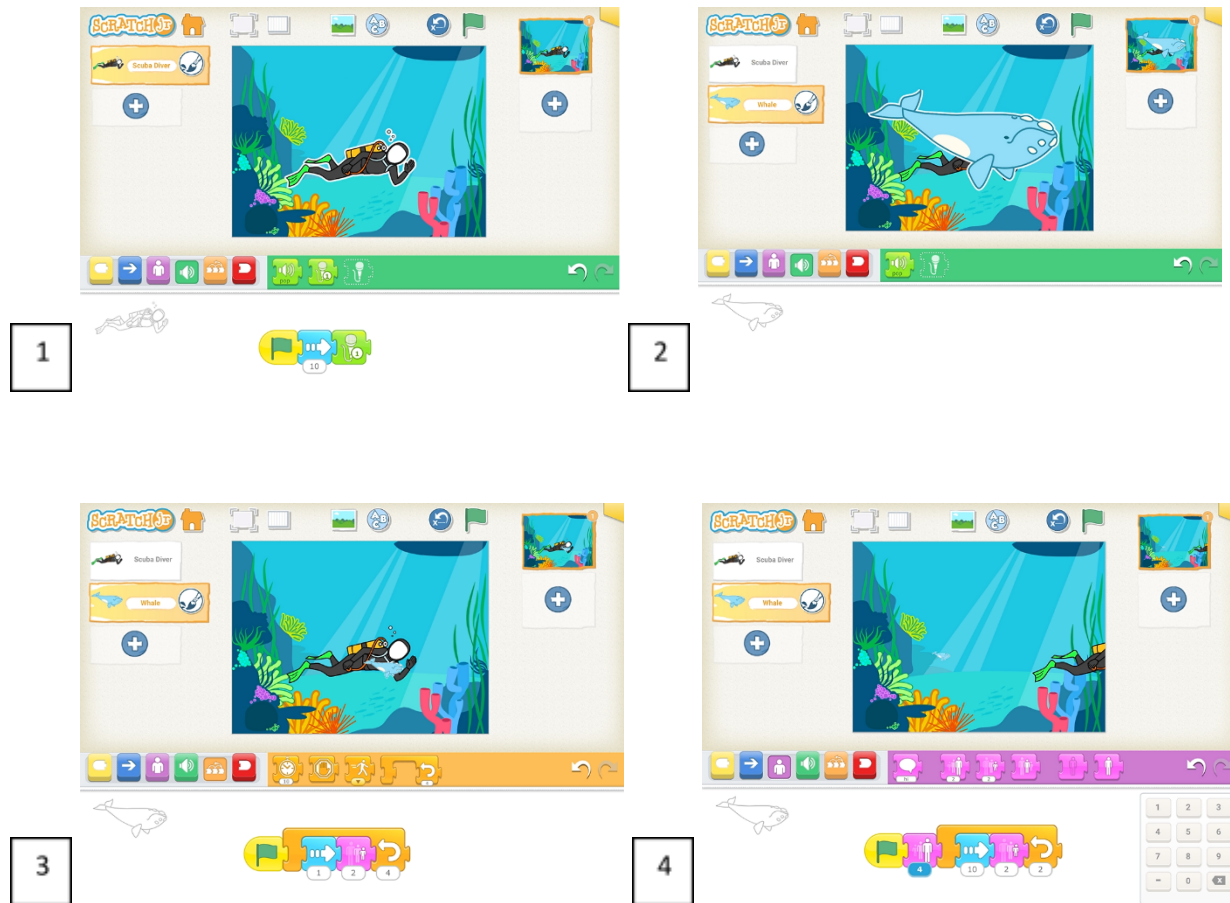
The first type of creation process was “Project Iteration”, when children switched their actions between creating, revising, and customizing. This process resulted in the expansion of their project details. Children engaged in these three actions with no particular action sequence as they are highly intertwined. The Project Iteration process aligns with literature on constructionism and making where children continually iterate their projects (Kafai & Resnick, 1996). Furthermore, iteration is also described as “cyclical processes of revisiting previous design decisions” and is integral for the design process (Adams, 2002, p. 2).

***Case Study 1.*** A first grader followed the Project Iteration process at the beginning of their session. They created a program, added a new character, created another program, then revised it. They first created a short program to make the scuba diver move forward and recorded a sound effect (Figure 10, top left). They then added a new character (Figure 10, top right), a whale, and created a repeating program for the whale to move forward and shrink (Figure 10, bottom left). After they played this program, they realized that the whale became too tiny after shrinking it four times with the repeat loop.

Therefore, they revised the whale's code by adding a grow block before the repeat loop (Figure 10, bottom right).

**Figure 10**

*Example of a Project Iteration Process that includes Create, Customize, Create, and Revise*



### **Coding Exploration**

Coding exploration was when children explored complex blocks *before* creating and/or revising their programs. Exploring is essential for child development as it is how children acquire information to understand new contexts (Cook et al., 2011; Legare, 2014; Lobo & Galloway, 2008). Furthermore, exploring behavior supports engineering thinking (Evangelou et al., 2010) as it is a foundational step in the design process. According to the Engineering Design Framework (Dorie et al., 2014, p. 5; English & King, 2015, p. 4), the act of exploring material is part of the “Problem Scoping” phase, which appears before ideating, constructing, evaluating, and revising. However, we can observe that there were instances

where children also explored complex blocks *after* they created or revised their programs (Refer to Figure 9), which was expected, as creating, or revising the program may also lead to exploring. Although exploring may come at different order, this study only focused on the process of exploration that leads to programs creation and revision, following the engineering design framework (Dorie et al., 2014, p. 5; English & King, 2015, p. 4). This study also assumed that exploring before creating or revising impacted their projects more directly.

**Case study 2.** A kindergartener demonstrated Coding Exploration process where they attempted to make a command with a repeat block with message blocks (Figure 11, top left). Their intention was to make Tac (a character) run after the girl runs to the school bus. They experimented with message blocks (Figure 11, top left) but did not get the command to work on the first try as they put “receive” and “send” orange messages in the same command (Figure 11, top right). Sending message and receive message blocks must be on different commands where the first command with a “send message” block sends a message to the second command with a “receive message” block to start playing. After they played the unexpected command, they realized the mistake and created a new command that starts with a “green flag” (Figure 11, bottom left). This command made the girl run to the bus then sent a message to Tac. Tac then ran to the bus (Figure 11, bottom right).

### **Figure 11**

*Example of a Coding Exploration Process that includes Explore and Create*



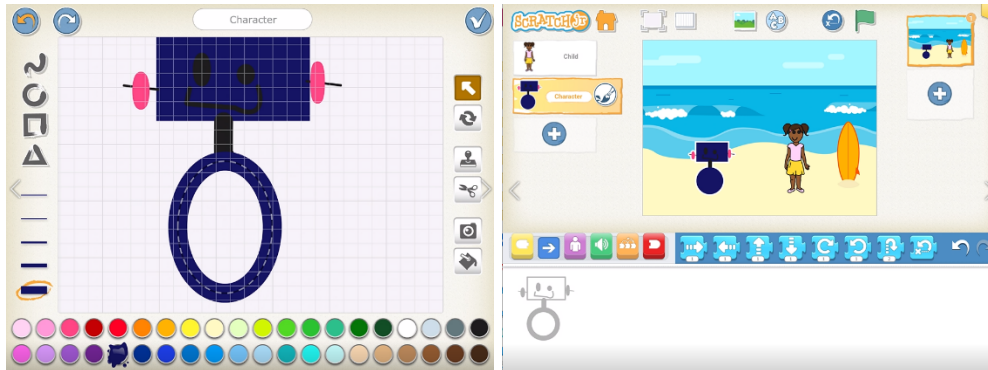
### *Long Customization*

Long customization was a process when children spent longer than 10% of their session duration customizing projects. Even if customization takes away from the coding time, the aesthetic aspect might support children to visualize their ideas in a more meaningful way. Through this pilot study, customization process seemed to motivate and engage students more with their overall ScratchJr projects. A nicely decorated project created a story framework for children to work on their code afterwards, aligned with a finding that drawing is a form of visual text construction that helps young children to compose more complex messages (Mackenzie & Veresov, 2013, p. 27).

**Case study 3.** A second grader created a story about a girl that programmed the robot to help build a sandcastle. They spent almost the entire session (Long Customization process) drawing the robot and the girl (Figure 12).

**Figure 12**

*Example of a Long Customization Process*



After they completed the customization, they created a program for the girl to say, “Build a sandcastle” then sends a message to the Robot, which then moves towards her and says, “Ok” (Figures 13, left and right). Their programs were advanced compared to the programs in the prior sessions that mostly consisted of basic motion blocks.

**Figure 13**

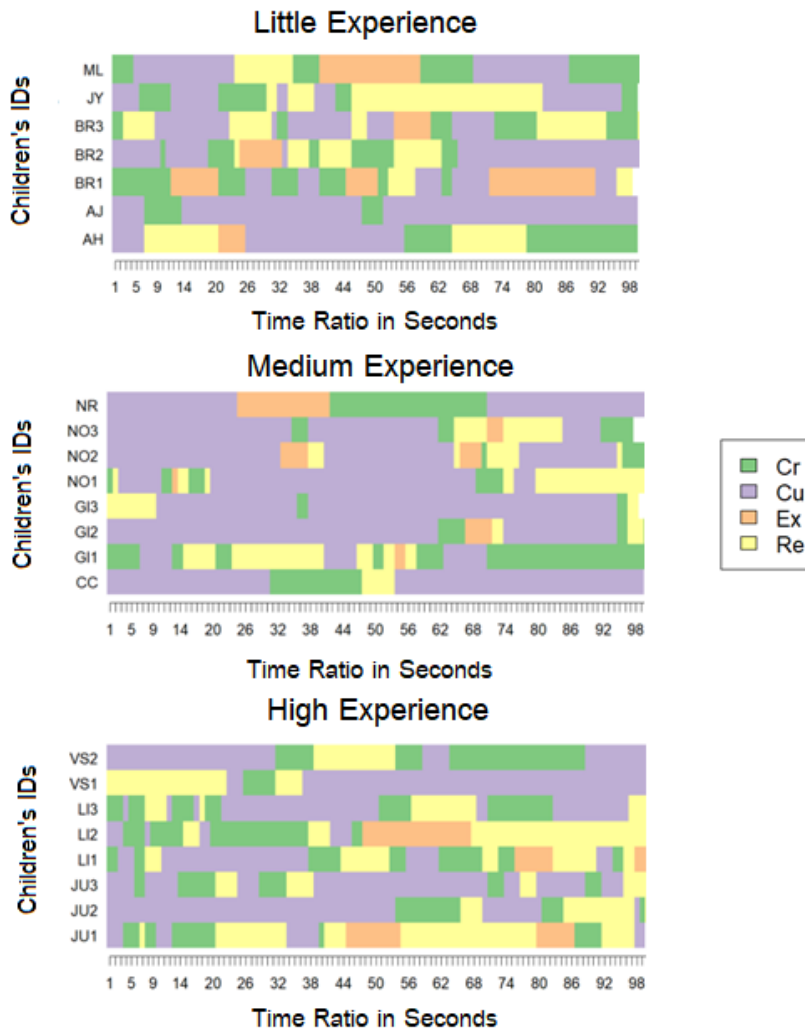
*Example of Coding Sections after Long Customization Process*



**Differences in Processes by ScratchJr Experience Levels**

**Figure 14**

*Sequence Plots Showing Processes of Children with Low, Medium, and High Experience*



The *little experience* group had the highest process variety as shown in Figure 14. Furthermore, this group spent the longest average amount of time on Coding Exploration given that they had one fewer sample size than the two other groups. The *medium experience* group spent the most time on average on Long Customization, which might be because they were at the mastery level that had recently become comfortable with painting tools. Once children got introduced to painting tools, they were hooked into customizing their projects, and drawing characters and backgrounds could take a long time. The *high*

*experience* group had more Project Iteration with some Coding Exploration because this group already mastered most ScratchJr blocks they might have spent less time figuring out the complex blocks.

### **Main Phase**

#### **Data Source**

The main phase participants were 153 students, aged from 5 to 8 years old, from two public schools in Rhode Island, USA. The children were in first ( $n = 97$ ) or second grade ( $n = 56$ ), learning from the CAL-ScratchJr curriculum (Refer to Table 8) as part of a larger study (*ScratchJr Studies*, 2021).

#### **Table 8**

*Summary of the Main Phase Study Data Source*

US State	Number 1 <sup>st</sup> Grade Classes	Number of 2 <sup>nd</sup> Grade Classes
Rhode Island	5	3

#### **Procedure**

For this phase, the Scratch Foundation developed a new ScratchJr app version, called CAL-ScratchJr. While the app users remain anonymous at an individual level, this new app version could collect identifying information at a classroom level. To do so, I assigned a unique logo for each classroom, where teachers assisted their students in tapping their classroom logos before playing with ScratchJr. Google Analytics automatically tracked each anonymous child's log data and labeled it with their classroom information.

#### **Measures**

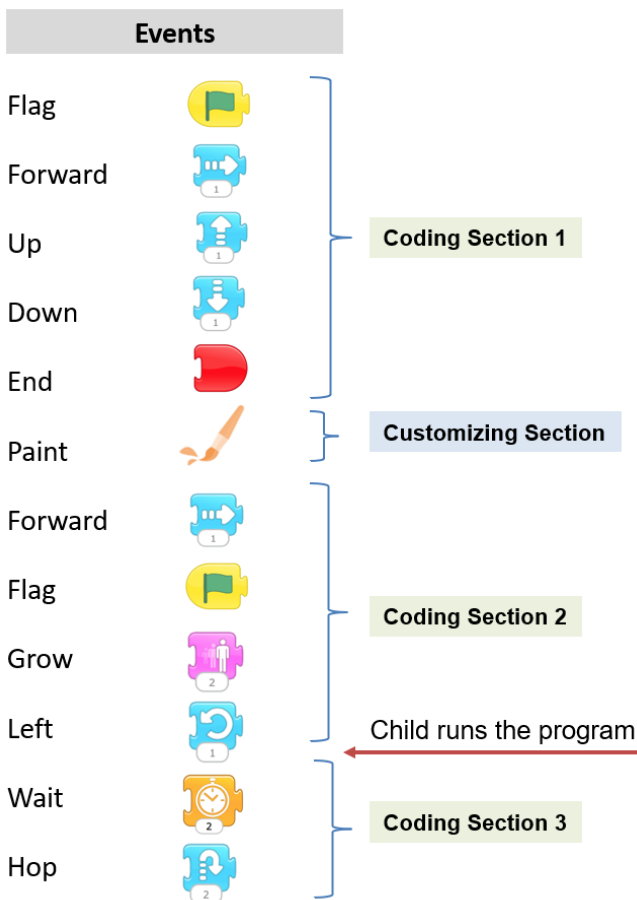
##### ***Actions on the Log Data***

Inspired by a study that used log data to extract action patterns (Guo et al., 2016), I created rules for determining children's actions on ScratchJr. I used the coding scheme from the pilot phase in Table 7 to guide the comparison between the screen recordings and the log data. Subsequently, I extracted four identified actions (create new, explore, revise, and customize) from the log data. The first step was breaking each session into multiple coding or customizing sections, which were sets of either blocks of

dragging actions or customizing actions. A coding section starts when children either create a new command or revise an existing command and ends when children run their programs or switch to customizing their projects. Figure 15 provides a visual example of a child starting the session with a coding section, which happens when this child creates a program starting with a green flag and ending with a red end block. The customizing section happens as the child paints the cat, follows by two more coding sections as the child creates new and revises programs.

**Figure 15**

*Short ScratchJr Activity Snippet with Four Sections*



*Note.* This figure shows an example of a short activity labeled into four sections (Coding-Customizing-Coding-Coding). The activity starts from the top.



**Coding Sections.** Log data from Google Analytics showed events that children did by timestamps. However, there was no identification of when children start or end creating each program, which made it challenging to differentiate coding sections. Therefore, my solution in determining “Coding Sections” was to use two types of gap times (inactive) between coding blocks. The first type of gap time was the time that children took to think and drag each coding block to create a program. This was generally faster than the second type of gap time, which was the time children tested or ran their programs, which on average took longer than 20 seconds. Therefore, this study assumed that children moved on to a new coding section if the gap time between coding blocks was longer than 20 seconds.

**Customizing Sections.** These were much simpler to identify compared to a coding section. This section included the time that children spent using painting tools and choosing characters or background.

**Table 9**

*Action Identification Log Rule*

Sections	Actions	Definition	Log Definition/ Rule
Coding	Create New	Start a new code from blank with no switching to non-coding actions	Sections with at least one trigger block
	Revise	Rework on the same code after playing the program or doing non-coding actions	Sections with no trigger blocks
	Explore	Try out intermediate or advanced blocks	Sections that have repeatedly used intermediate or advanced block
Customizing	Customize	Decorate the project or add character/background	Sections that have painting or choosing character/background

After categorizing events into sections, I labelled the four actions within each section according to the rules shown in Table 9. *Create new* referred to when children made an entirely new program.

Regardless of the blocks sequencing in each program, the coding section that the study labelled as “create new” action had to have at least one type of trigger block (e.g., start on the green flag, start on tap, start on bump), which is required for the program to run when pressing play. To elaborate, trigger blocks did not have to be dragged first in a coding section, as dragging trigger blocks last was also common. *Revise* was when children added at least one additional coding block to an existing program. Therefore, this section usually only had coding blocks that were non-trigger blocks. Furthermore, this action often happened after children tested their programs and engaged in customizing actions. *Explore* was when children tried to use intermediate or advanced coding blocks in a repeating manner. This can be reflected when children continually used the same intermediate or advanced block within a coding section. *Customize* had the most straightforward rule in the customizing section, when children used a painting tool, added a character, or added a background on ScratchJr.

### ***Three Processes from the Log Data***

From the labelled actions, I extracted three processes, “Project Iteration”, “Coding Exploration”, and “Long Customization”, with rules that closely aligned with the process identification steps in the pilot phase. There could be multiple processes in one lesson for each child. *Project Iteration* was when children switched their actions between create new, revise, and customize, with a requirement that each customize duration must be less than 10% of the session duration. *Coding Exploration* was when an exploring action led to coding actions (creating new or revising) and ended when children stopped coding and started customizing. Lastly, *Long Customization* was when customizing was at least 10% of the session duration.

### ***Pre and Post Lesson Block Exploration***

Research Question 5 investigated exploring actions before and after the curriculum taught complex (intermediate and advanced) coding blocks (Refer to Table 2). I calculated *pre-lesson* exploration by counting the number of *untaught* complex blocks that children used during exploring. In contrast, *post-lesson* exploration was the number of *taught* complex blocks usage during exploring.

### ***Variety, Adherence, and Complexity***

Block variety, adherence, and complexity were the three block usage outcomes for RQ3, RQ4, and RQ5, respectively. *Block variety* was the number of block *types* children used at three timepoints. The *Block adherence* ratio was calculated by counting the usage of unique blocks that were recently taught over the total number of recently taught blocks at each timepoint. For instance, if a first grader uses three out of six taught block types at timepoint 2, the adherence ratio was .50. Lessons included at each timepoint are shown in Table 1, and the blocks taught up to each timepoint are shown in Figure 6. Examples of recently taught blocks for first graders at timepoints 1, 2, and 3, were blocks taught up to lesson 6, blocks taught from lessons 7-10, and blocks taught from lesson 11-20, respectively. Lastly, *block complexity* was the number of non-unique intermediate and advanced coding blocks used at Time 3, which was their final project in the curriculum.

### **Data Analysis**

The objective of the main phase was to investigate the relationship between creation processes and the three block usage outcomes (variety, adherence, and complexity). Question 2 analyzed occurrences of each process at three timepoints by plotting stacked bar plots. Question 3 examined whether the duration of each process type was related to how varied children use coding blocks across the curriculum. Similarly, Question 4 investigated whether the processes were related to children's usage of recently taught coding blocks. Instead of focusing on the three process types, Question 5 examined how the timing of exploring *action* was related to block complexity in children's final lessons.

### **Research Question 3**

This study used two-level piecewise multilevel regression or longitudinal models to answer both parts of research question 3 (Refer to Equations 1.1-1.8). The timepoints were split into two phrases, Timepoint 2 compared to Timepoint 1 and Timepoint 3 compared to Timepoint 1. I created two-level models (Refer to Appendix C, Table C1), where binary variables of time phases (Timepoint 2 and Timepoint 3) were clustered within students. These models were used to predict the continuous outcome variable, *block variety*. The first-level (time variant) predictors were timepoints, duration of each process,

and session duration (mean centered at each timepoint). All durations were in minutes. The second-level (time invariant) predictor was grade level, which I only added to the intercept in the final model to understand the interaction between grade level and Timepoint 1 to predict for block variety. Furthermore, this study only allowed the intercept, Timepoint 2 vs 1, and Timepoint 3 vs 1 to vary across students.

I conducted model estimation in steps. The first model was an empty model with only an intercept and fixed and random effects. The second model was an unconditional growth model with Timepoint 2 vs 1 and Timepoint 3 vs 1 variables. I then added one variable at a time (as fixed effects), including session duration (min.), Long Customization (min.), Project Iteration (min.), Coding Exploration (min.), and Grade Level, to Models 3-7 (Refer to Appendix C, Table C1).

### Equations 1.1-1.8

*Multilevel model to predict block variety*

$$\text{Level 1} \quad Y_{ti} = \pi_{0i} + \pi_{1i}(\text{Timepoint 2 vs 1}_{ti}) + \pi_{2i}(\text{Timepoint 3 vs 1}_{ti}) + \pi_{3i,1}$$

$$\text{Level-2 Intercept} \quad \pi_{0j} = \beta_{00} + \beta_{01}(\text{GradeLevels}) + u_{0j} \quad 1.2$$

$$\text{Level-2 Slope: Timepoint 2 vs 1.} \quad \pi_{1j} = \beta_{10} + u_{1j} \quad 1.3$$

$$\text{Level-2 Slope: Timepoint 3 vs. 1} \quad \pi_{2j} = \beta_{20} + u_{2j} \quad 1.4$$

$$\text{Level-2 Slope: Lesson Duration} \quad \pi_{3j} = \beta_{30} \quad 1.5$$

$$\text{Level-2 Slope: Long Customization} \quad \pi_{4j} = \beta_{40} \quad 1.6$$

$$\text{Level-2 Slope: Project Iteration} \quad \pi_{5j} = \beta_{50} \quad 1.7$$

$$\text{Level-2 Slope: Coding Exploration} \quad \pi_{6j} = \beta_{60} \quad 1.8$$

### Research Question 4

Research questions 4a and 4b also used two-level piecewise longitudinal regression to predict the ratio of adherence (Refer to Equations 2.1-2.9). I created models (Refer to Appendix C, Table C2) with two levels of predictors. Similar to the previous question, the two binary time phases (Timepoint 2 vs 1 and Timepoint 3 vs 1) were nested under students. The continuous outcome variable was *block adherence*, or the ratio of taught block usage over the total block usage. The first-level (time variant) predictors were timepoint, number of blocks taught (mean centered), lessons duration (mean centered), and duration of each process type. All durations were in minutes. The second level (time invariant) predictor added to the intercept was grade level. This question also only allowed the intercept, Timepoint 2 vs 1, and Timepoint 3 vs 1 to vary across students.

I conducted model estimation in steps. The first model was an empty model with only an intercept in fixed and random effects. The second model was an unconditional growth model with additional Timepoint 2 vs 1 and Timepoint 3 vs 1 variables. The third model also has session duration (min.) and Number of Taught Block in the fixed effect. I then added one variable at a time (as fixed effects), including, Long Customization (min.), Project Iteration (min.), Coding Exploration (min.), and Grade Level, to Models 4-7.

### Equations 2.1-2.9

*Multilevel model to predict block adherence*

$$\text{Level 1} \quad Y_{ti} = \pi_{0i} + \pi_{1i}(\text{Timepoint 2 vs 1}_{ti}) + \pi_{2i}(\text{Timepoint 3 vs 1}_{ti}) + \pi_{3i} \quad 2.1$$

$$\text{Level-2 Intercept} \quad \pi_{0j} = \beta_{00} + \beta_{01}(\text{GradeLevels}) + u_{0j} \quad 2.2$$

$$\text{Level-2 Slope: Timepoint 2 vs. 1} \quad \pi_{1j} = \beta_{10} + u_{1j} \quad 2.3$$

$$\text{Level-2 Slope: Timepoint 3 vs. 1} \quad \pi_{2j} = \beta_{20} + u_{2j} \quad 2.4$$

$$\text{Level-2 Slope: Lessons Duration} \quad \pi_{3j} = \beta_{30} \quad 2.5$$

Level-2 Slope: No. Block Taught	$\pi_{4j} = \beta_{40}$	2.6
Level-2 Slope: Long Customization	$\pi_{5j} = \beta_{50}$	2.7
Level-2 Slope: Project Iteration	$\pi_{6j} = \beta_{60}$	2.8
Level-2 Slope: Coding Exploration	$\pi_{7j} = \beta_{70}$	2.9

### ***Research Question 5***

Research question 5a used t-test to compare pre-and post-lesson exploration. Research question 5b used a simple linear regression model to predict *block complexity* at the last timepoint. The main predictors were pre-and-post lesson explorations.

## **Results**

### **Data Screening**

I conducted data screening for research questions 3-5 according to Table 10. Some students were absent mostly due to the COVID-19 pandemic, resulting in missing values at some timepoints (40 out of 459 occasions). Out of 153 students, 121 students had data for all three timepoints (no missing values), 24 students had data for two timepoints, and eight students had data for one timepoint. Consequently, this study's regression analyses excluded missing data at different occasions from the models.

The descriptive statistics for Questions 3 to 5 are shown in Appendix A, Tables A1 to A3. I identified outliers as having z-scores of above absolute three. Using this criteria, there were two to eight outliers out of 153 children across variables at each timepoint. After removing the outliers, all three outcome variables (variety, adherence, and complexity) were normally distributed with skewness and kurtosis values approximately lesser than absolute one across all timepoints. However, predictor variables, particularly duration of process types in minutes, were non-normal at some timepoints.

Therefore, I used robust maximum likelihood estimation (ML) in the multilevel models for RQ3 and RQ4 to adjust the standard error based on the level of skewness.

Furthermore, the correlation matrices showed that there were significantly weak to mild correlation across variables (Appendix B, Tables B1 and B2), except for Long Customization and Lesson Duration that were highly correlated at .72. This high correlation was expected as Long Customization captured customizing sections that were longer than 10% of each session. Grade level was the only categorical variable, consisting of around 63% first graders and 36% second graders across timepoints. Although a third of the sample were second graders, their sample size at each time point was still large enough (greater than 30 children) to make generalization to a wider population. This data screening process showed that this study's dataset is well suited for the proposed regression analyses.

**Table 10**

*Summary of the Variables, Data Screening and Post-Assumption Testing for Questions 3 to 5*

RQs	Methods	Outcome Variables	Predictor Variables	Pre-Screening	Post-Assumption Testing
RQ3	Multilevel Regression	Variety	Timepoint 2 vs 1 (Cat.) Timepoint 3 vs 1 (Cat.) Duration of Long Customization (Con.) Duration of Project Iteration (Con.) Duration of Coding Exploration (Con.) Session Duration (Con.) Grade Level (Cat.)	Distribution Outliers: Boxplot and Z-scores Correlation	Distribution of Residuals Homoscedasticity of Residuals Linearity of independent and outcome variables

RQ4	Multilevel Regression	Adherence	Timepoint 2 vs 1 (Cat.) Timepoint 3 vs 1 (Cat.) Duration of Long Customization (Con.) Duration of Project Iteration (Con.) Duration of Coding Exploration (Con.) Number of blocks taught (Con.) Session Duration (Con.) Grade (Cat.)	Distribution Outliers: Boxplot and Z-scores Correlation	Distribution of Residuals Homoscedasticity of Residuals Linearity of independent and outcome variables
RQ5	T-test and Linear Regression	Complexity	Pre-Explore (Con.) Post-Explore (Con.) Session Duration (Con.) Grade (Cat.)	Distribution Outliers: Boxplot and Z-scores Correlation	Distribution of Residuals Homoscedasticity of Residuals Linearity of independent and outcome variables

*Note.* Con.= Continuous variable, Cat. = Categorical variable

**Research Questions 2:** What are the proportions of different creation processes at the three-time points?

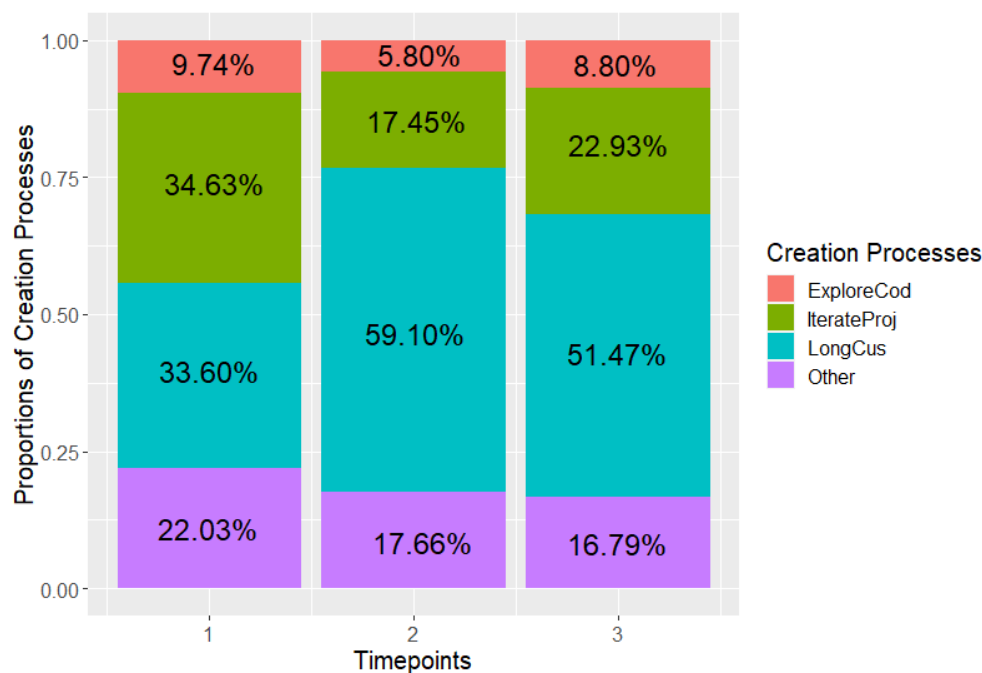
This study classified children into experience levels based on timepoint in the CAL program, with timepoints 1, 2, and 3 corresponding to little, medium, and high ScratchJr experience levels, respectively.



Like children with little experience in the pilot study, children at Time 1 in the main study (Refer to Figure 16) followed all processes more diversely and used slightly more Coding Exploration on average (9.74%) compared to the other timepoints ( $M_{t2} = 5.80\%$  and  $M_{t3} = 8.80\%$ ). Moreover, children at Time 1 used Long Customization at a similar duration to Project Iteration, both were less than 50% of all the sessions. In contrast, children at Time 2 spent around 60% of the sessions, on average, customizing their projects, which aligned with the results from the intermediate experience group in the pilot study. Children at Time 3 spent lower proportions of time on average (22.93%) Project Iteration than children at Time 1 ( $M = 34.63\%$ ), which differed from the trends in the pilot study. The “other” category was the coding process that this study did not focus on, which included the actions grouping of create-revise or separate usage of create, revise, and explore. In summary, children in this study followed creation processes differently across timepoints in the CAL curriculum. On average, children used all processes diversely in early lessons and focused mainly on Long Customization around mid-lessons. By the end of the curriculum, they split their time equally between processes that focused on customization and coding.

**Figure 16**

*Proportions of Time spent following each Creation Process across Timepoints*



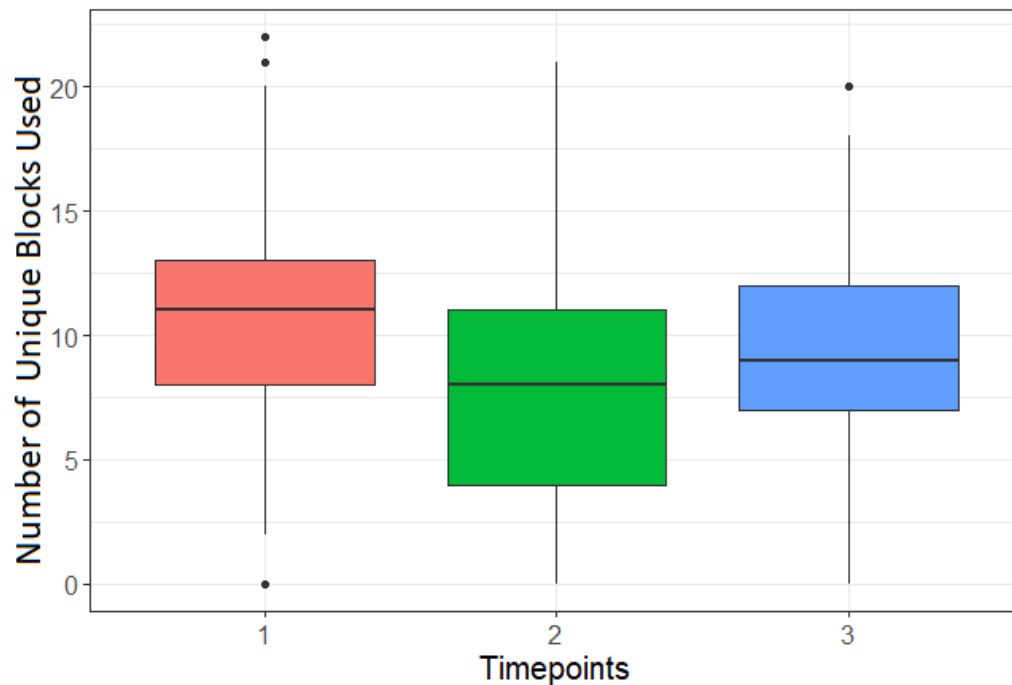
**Research Question 3a:** What is the difference in *block variety* as the CAL program progresses?

**3b:** Which creation process is associated with more *block variety* across time points?

The intra-class correlation (ICC) from the empty model (Appendix C, Table C1) showed that between-person variation accounted for only 6.79% of the total difference in *block variety*. This amount was relatively small, meaning that the variation was indifferent between timepoints within children. The third model showed that that block variety significantly dropped from Timepoint 1 to Timepoint 2 (Figure 17), when holding the total session duration constant. Compared to the average 10 block types at Timepoint 1, children used two block types less at Timepoint 2 ( $p < .001$ ) and one block type less at Timepoint 3 ( $p < .01$ ). However, this drop no longer held in model 4, which included Long Customization. In other words, the decrease in block types after Timepoint 1 was related to how long children spent customizing their projects. This study did not include Timepoint 3 in the random effect as the variance between students was very small.

### Figure 17

*Number of Different Blocks used across Timepoints*

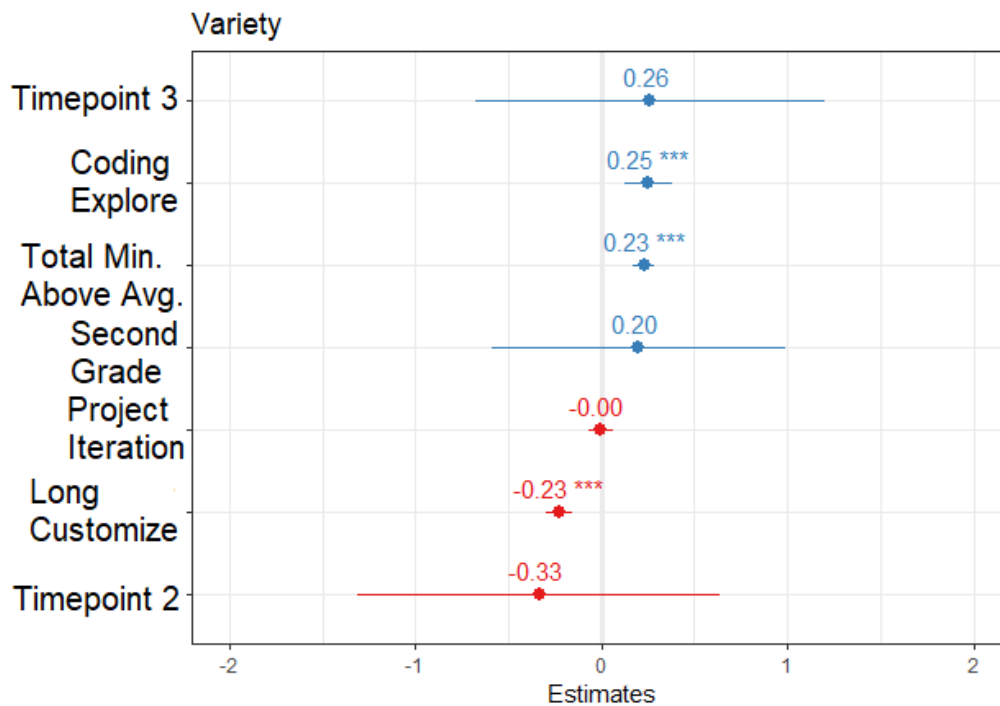


*Note.* These box plots show the distribution of the number of unique used blocks per child at each time point in the curriculum.

Coding Exploration and Long Customization process durations were significantly related to block variety at  $p < .001$  (Figure 18). With four more minutes of children following the Coding Exploration process, it is expected that children on average will use one more block type when holding the two other process types, session duration, and timepoints constant. In contrast, the model predicted that children would use one fewer block type with around five more minutes of Long Customization. The r-squared of the final model was .44. Although this study showed that block variety decreased across time, the size of this decline was only around 10-20% of the average total block each child used. Furthermore, block types may increase when children follow more Coding Exploration and less Long Customization processes.

### Figure 18

*Coefficients of Predictors on Variety from the Longitudinal Model*



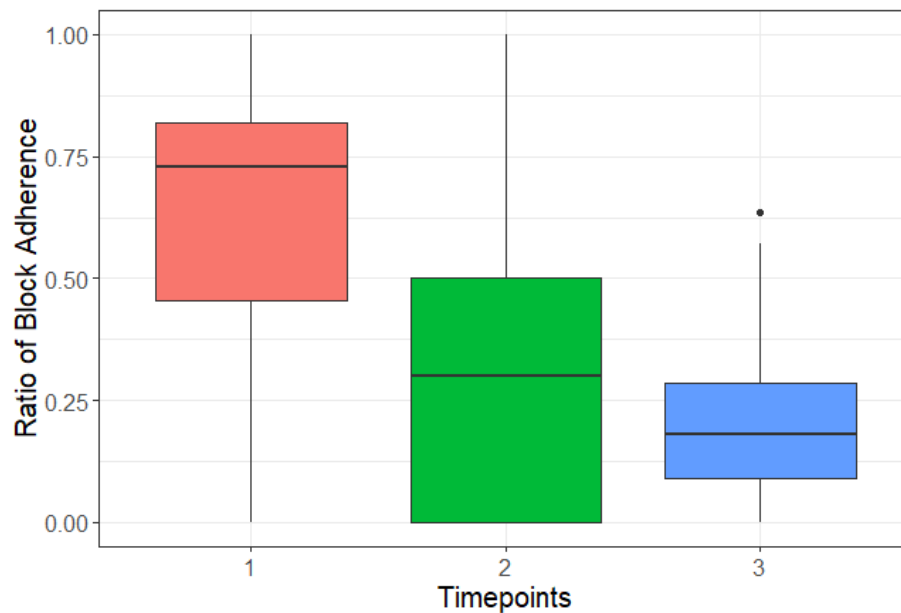
*Note.* The numbers are unstandardized coefficients. The horizontal bars are confidence intervals. \* =  $p < .05$ , \*\* =  $p < .01$ , \*\*\* =  $p < .001$

**Research Question 4a:** How much are children using the blocks they learned as the CAL program progresses? **4b:** Which creation process is associated with more *block adherence* across time points?

The intra-class correlation (ICC) from the empty model (Appendix C, Table C2) showed that between-person variation was close to zero, meaning that *block adherence* only varied across timepoints, but almost did not vary at all between children. Similar to the models from the previous question, Model 2 onwards allowed the intercept and Timepoint 2 to vary across students. This question did not include Timepoint 3 in the random effect as the variance was very small.

**Figure 19**

*Ratio of Block Adherence to Number of Taught Blocks across Timepoints*

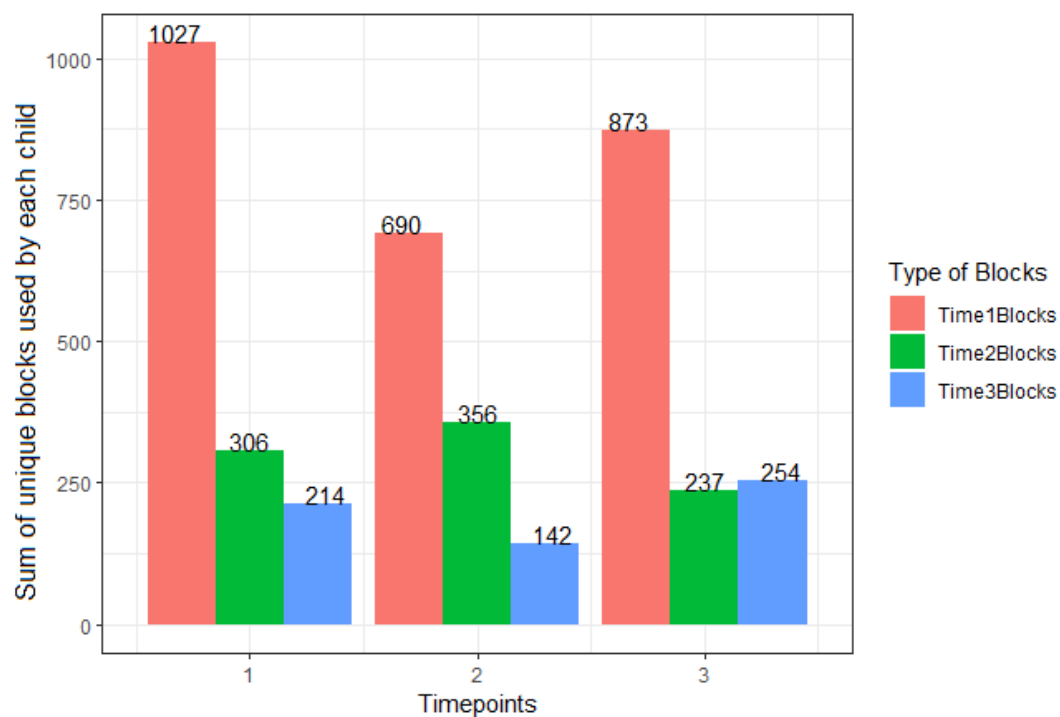


*Note.* These box plots show the distribution of the ratio of block adherence at each time point in the curriculum.

When holding the session duration and number of taught block constant, the third model showed that ratio of adherence dropped an average of 36.00% between Timepoint 1 and 2 and an average of 48.60% between Timepoint 1 and Timepoint 3. The final model predicted that first grader on average used around 77% of taught block at Time 1 ( $p < .001$ ) under three conditions: 1) their ScratchJr session had an average duration of around 30 minutes; 2) they were taught 11 coding blocks; 3) they did not use any of the three processes identified in this study. Although ratio of adherence dropped across timepoints (Refer to Figure 19), blocks that were taught at each timepoint were still most frequently used at that timepoint (e.g., Blocks taught at Time 2 were used most frequently at Time 2) (Refer to Figure 20).

## Figure 20

*Total Unique Blocks by Block Taught at each Timepoint*

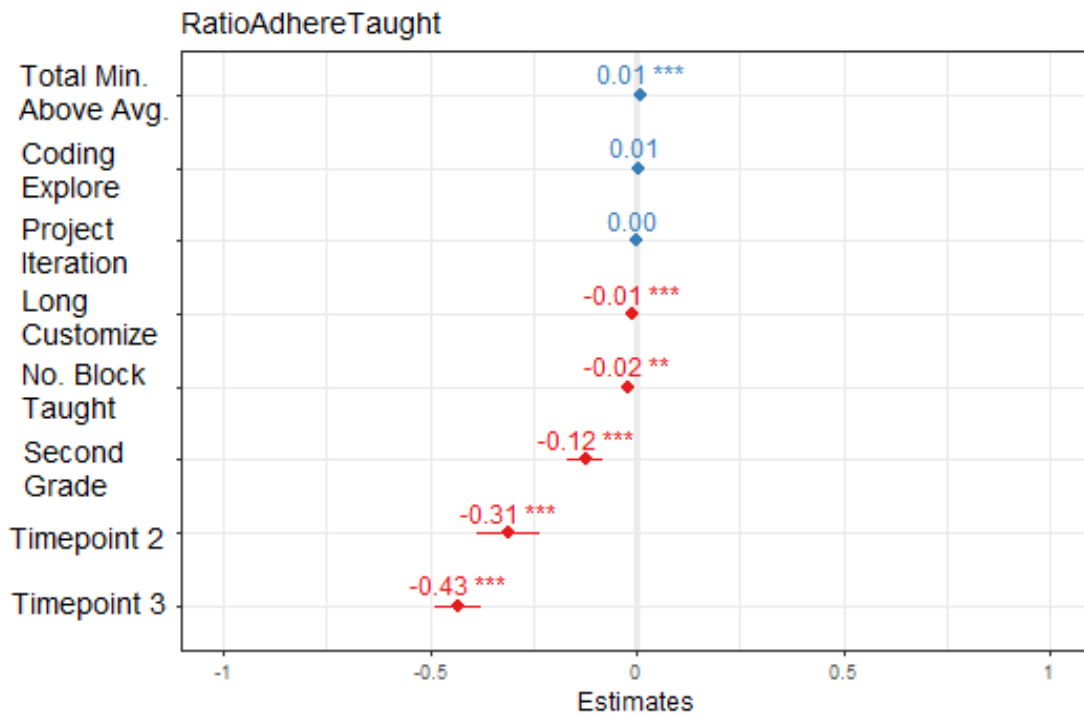


*Note.* Time1Blocks= Blocks taught at Time 1, Time2Blocks= Blocks taught at Time 2, and Time3Blocks= Blocks taught at Time 3.

Long Customization was the only process that had a significant association to adherence of block taught (Figure 21). With one more minute of Long Customization, adherence ratio was expected to decrease by 1.00%, when holding the other variables such as grade level constant. The r-squared for the final model was .56. As key takeaways, children used less recently taught block over time and Long Customization was associated with this decline. However, children still used blocks most frequently when they were recently taught.

### Figure 21

*Coefficients of Predictors on Ratio of Adherence from the Longitudinal Model*



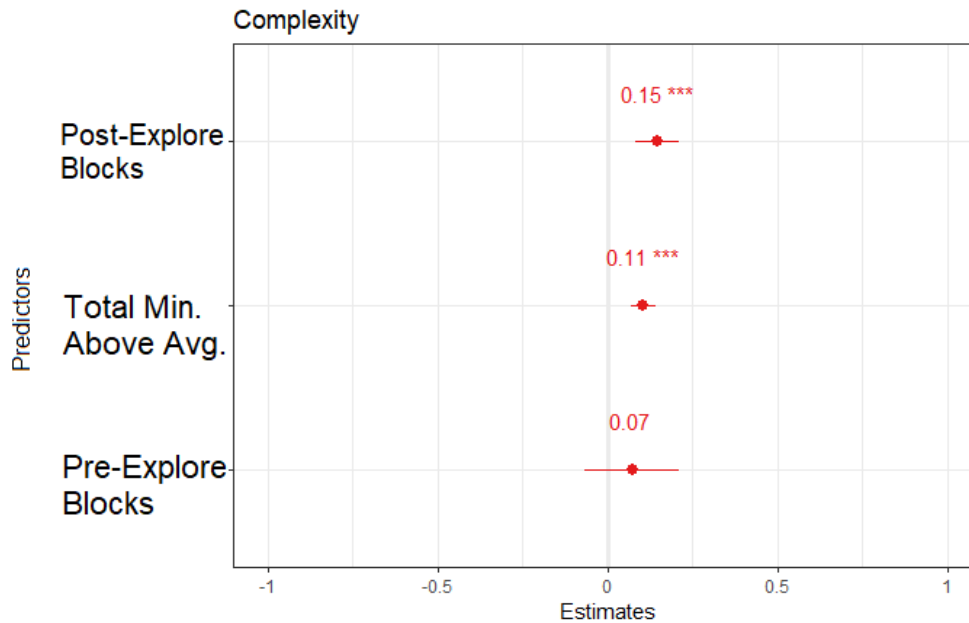
*Note.* The numbers are unstandardized coefficients. The horizontal bars are confidence intervals. \* =  $p < .05$ , \*\* =  $p < .01$ , \*\*\* =  $p < .001$

**Research Question 5a:** Do children explore the complex coding blocks more frequently before they were taught? **5b:** What is the relationship between *when* children explore complex blocks (explore on their own or explore after they were taught) and *how* complex their block usage is in their final lessons?

The number of complex blocks during post-exploration ( $\mu = 18.98$ ) was significantly higher than during pre-exploration ( $\mu = 5.42$ ) at  $p < .001$ . When children did not explore complex coding blocks at all, the average number of complex blocks used at Time 3 was estimated to be 3.17. Figure 22 shows the coefficients of all variables. If they post-explore, the number of complex blocks would increase by .14 with one increasing complex block used during post-exploration ( $p < .001$ ). Pre-exploration did not significantly impact the complex block usage. The r-squared for this linear regression model was .34. These results showed that post-exploration was more prominent in children's coding experience with the CAL curriculum.

**Figure 22**

*Coefficients of Predictors on Block Complexity from the Linear Regression Model*



### Post-Screening

The final models for research questions 3 and 4 (Appendix C, Tables C1 and C2) included all necessary random effects; any additional random effect to the model gave a convergence error. I conducted post-screening analysis on the final models for research questions 3, 4, and 5. The residual graphs are shown in Appendix D (Tables D1-D3). The residuals for all models were normally distributed, although there were some deviations from the expected normal line on each side of the tail. The Levene's tests showed that the residuals were homoscedastic as the variances did not significantly differ across children. Lastly, there was linearity in the relationship between each of the independent variables and the outcome variables.



## Discussion

Computational skills are essential for children to acquire in the 21<sup>st</sup> century (Bers, 2018a, 2019, 2020b). To gain such skills, young children can learn to program by creating coding projects (Govind et al., 2020). Millions of coding projects have been created by children monthly, and many of these projects were evaluated using rubrics (Basu, 2019; Papadakis et al., 2017; Unahalekhaka & Bers, 2021, 2022). However, there is still a lack of understanding of young children's processes when creating open-ended coding projects, even though recognizing processes is essential for understanding learning experiences (Taranu et al., 2022). Therefore, this study focused on understanding children's ScratchJr creation processes over the course of participating in the 24-week-lesson CAL curriculum.

This dissertation utilized a mixed-method approach to conduct both pilot and main phases of the study. During the pilot phase, I conducted an observation of screen recordings to understand processes employed by children when creating projects. Subsequently, in the main phase, I used Google Analytics data to better understand the varying ways in which children created projects in relation to how they used coding blocks. This approach addressed eight research questions, each of which yielded understandings into the three processes used by children when creating projects, and how each process was related with their coding block usage.

This study found that children used diverse processes to create open-ended ScratchJr coding projects. Some processes were related to coding block usage outcomes, including the variety in block types, adherence to the lesson, and complexity of the blocks. Knowing these processes is helpful as results from this study can guide how to improve the CS learning experience in early childhood. The main key takeaway is that there should be a mixture of autonomy and guidance in early childhood CS. Educators and parents may give young children the freedom to discover coding blocks while providing more guidance on the complex block functions. According to Constructionism, children develop ideas rather than passively receive ideas from adults (Kafai & Resnick, 1996). However, guidance is still essential for

young children, especially for computer science concepts that require multiple consequential steps (Klahr & Carver, 1988).

### **Teaching strategies across the curriculum may need to be different**

The current study identified three processes children followed when creating ScratchJr projects: Project Iteration, Exploring Coding, and Long Customization. These three processes are essential for young children's development and differ across student groups with varying ScratchJr experiences. In the early CAL lessons, teachers primarily introduced basic motion blocks to make characters move as children were new to ScratchJr. At this stage, children tried various processes of iterating, exploring, and customizing (selecting characters and backgrounds). In other words, they used the three process types more interchangeably. This pattern was expected as children were still new to coding and customizing functions of ScratchJr. Discovering is a way for them to gather information to develop scientific and causal reasoning (Cook et al., 2011; Legare, 2014). Therefore, children new to ScratchJr should be allowed more free-play time to discover different functions.

At mid-curriculum, children had intermediate ScratchJr experience levels and shifted from coding to spending the most time customizing their stories. At this curriculum time point, teachers taught painting functions and more blocks related to changing the characters' appearances (e.g., grow and disappear) in ScratchJr. This is expected as young children tend to invest deeply in customizing their projects. A study that assessed ScratchJr projects showed that, on average, children got higher scores on their design than on coding aspects (Unahalekhaka & Bers, 2022). The painting tools on ScratchJr are beneficial as many children program according to their decorated stories. Visual representation, like drawing, enhances scientific thinking, such as observation, problem-solving, explanation, and communication (Fan, 2015). However, too much customization means less coding time, as shown in the current study. Therefore, teachers may set expectations with children that the painting functions can enhance their coding projects, but the main focus of this work is coding. At this time point, teachers can also set goals and plans with children on how they will complete their projects. Some questions to ask

children may include how long they think they will need for customizing versus coding. Teachers can then guide children on the plans to make sure that children will focus on coding. This approach is based on the Self-Determined Learning Model of Instruction, which supports students in solving problems by creating, modifying, and applying a series of questions and solutions to reach self-selected goals (Palmer & Wehmeyer, 2003). Palmer and Wehmeyer (2003) reported that children as young as five could set goals and follow up with their plans with the teacher's assistance. Teachers may also consider allowing children to focus on customizing the visual aspects of the story before they transition into coding.

Towards the end of the 12-week program, children were highly experienced with ScratchJr and had to create final projects as stories with at least three pages. The study found that young children could complete multiple page projects by splitting their focus between coding and painting. Children in the highly experienced group from the pilot study iterated their projects the most. Project Iteration might need a certain mastery level as it requires children to combine multiple actions of programming, tweaking, and customizing to iterate their ScratchJr projects. A study with adult engineers found experts iterate their designs more frequently than novices (Adams, 2002). Iterating is integral to the design process (Adams et al., 2003) and leads to more originality in children's creative products (Taranu et al., 2022). To promote iteration, teachers and parents may encourage children to keep improving and adding details to the same project across multiple sessions, such as adding more pages and characters or making their programs more advanced. Children must have enough time to "experiment and think over" (Looijenga et al., 2015, p. 4).

### **Exploration promotes coding block variety**

Children in this study had a slight drop in average block variety when creating projects across time points, starting from ten block types in the early lessons. This decline was related to an extensive period of project customization. However, around ten block types seem to be enough to create meaningful projects. A study with adults also found that undergraduate students constantly used around eight unique block types in open-ended Scratch projects. However, the total number of blocks increased across ten weeks (Karakaya Cirit, 2022). Nevertheless, children will still benefit from using a wider variety of block types so they can apply different commands and coding concepts, given that there are more than 20 block types on ScratchJr. Results from the current study suggested that teachers should encourage children to try discovering unfamiliar or complex blocks if they want children to use a variety of blocks. Exploration or self-discovery is a significant action that enhances children's learning as it is a way for them to gather information to have a better understanding about the world, “all of us learn by constructing, exploring, and theory building.” (Papert, 1980, p. 132)

### **Young children followed preference and instruction**

Young children might not use all the blocks that were taught; however, this study still showed that they most frequently used the blocks that were recently taught. This means that young children in the study could follow instructions to a certain level, while the rest of the time was their free choice. There can be many explanations for this result. First, children might not know how to use all the advanced blocks that teachers taught later in the curriculum. According to previous studies, young children found control blocks complex and were able to use these blocks less (Strawhacker & Bers, 2019; Unahalekhaka & Bers, 2022). Second, children may want the independence to use any blocks they want in their projects. Children's ability to freely choose is essential for their learning motivation and is related to the feeling of competency over time (Ryan & Deci, 2020). Consequently, they may strongly prefer which block to include rather than following instructions. Third, they needed to keep using the foundational blocks taught early in the curriculum for all their projects. This study showed that the foundational blocks taught at

Timepoint 1, the motion blocks, were most used across all time points. This finding is expected as the motion blocks are the basic blocks that directly change the characters. Different studies also found that preschool and elementary students most commonly used motion blocks in their ScratchJr projects (Kyza et al., 2022; Papadakis et al., 2016; Unahalekhaka & Bers, 2021). For these reasons, teachers should keep in mind that children might not completely follow all the instructions in using taught coding blocks. Likely, young children will still freely use the blocks they want when creating open-ended projects. However, it can be helpful for teachers to review the complex blocks' functions before children create the projects, especially those introduced a while back. An effective teaching approach might be to strike a balance between encouraging children to include all concepts and giving them autonomy in their projects.

#### **Combination of guidance and self-discovery may lead to complex block use**

Children in this study played around with complex blocks more frequently after teachers taught the block functions. This result differed from the studies that reported how children explored more when they were not told the functions of the tangible object (Evangelou et al., 2010). This difference might be because ScratchJr's coding concepts require more thinking steps than physical toys. Therefore, ScratchJr complex blocks might not be intuitive enough for young children to initiate exploration. They may lose interest after trying complex blocks that do not work as expected, especially when they can already create animated projects with the basic blocks.

Furthermore, this study showed that the more children explored complex blocks after receiving lessons, the more complex blocks they used in their final projects. In other words, children's self-discovery after receiving some guidance may result in more advanced block usage later. Direct instruction is highly effective in concepts with multi-steps, such as computer science, geometry, and algebra (Klahr & Carver, 1988). However, free exploration time is still needed in early childhood learning experiences as children need a space to explore, build, and modify their reality and come up with multiple alternatives (Papert, 1980, p. 126). Therefore, one way to promote powerful ideas might be to combine independence and guidance, encourage guided exploration, when young children create open-ended

coding projects. Teachers and parents may want to provide free exploration time before and after teaching children the block functions.

### **Conclusion**

In conclusion, a developmentally appropriate approach for early childhood CS might be to balance free exploration and direct guidance. The early lessons can start by showing children what coding functions are available for them by making a simple program to make the character move, then allow children to explore the rest of the motion blocks. The idea of exploring and testing out early will be essential to lead exploration in more complex blocks later. Adults can continue to encourage children to explore new coding blocks as children become more experienced and confident coders. After children understand how to create functional programs, adults can introduce painting tools that will make their projects more personally meaningful. As more time to focus on coding will promote more block variety and complexity, adults should agree with the children on how much time they will spend customizing. After adults introduce the complex blocks' functions, it is essential that children still get free time to try out or discover the blocks on their own. The strength of engaging in open-ended creation is that children can put abstract and complex concepts into use (Papert, 1980). This study contributes to the early CS literature by highlighting the common processes that young children may use when creating open-ended coding projects. These processes were shown to be related to children's coding block usage, and knowing the processes can improve the early CS learning experiences.

### **Limitations and Further Research**

There are factors that impact children's creation processes, such as their coding competency and the format of when they create projects (online versus in-person). Children with little ScratchJr experience might have prior experience with coding from the other platforms. Although the research team has the baseline coding competency scores for all participants in the main study, we were not able to connect the scores to anonymous children's log data because of privacy protection.

Due to the limitation of what Google Analytics can capture, this study had to develop action and process rules. However, the rules did not perfectly capture the actions, as there were unexpected cases. For example, children new to ScratchJr might not use any trigger blocks in their commands. Their commands will not run when they press the play button on the top of the screen; however, there is a way around it by tapping directly on the block to run the program. This limitation informs what essential variables learning analytics tools should measure. In a coding app like ScratchJr, we should be able to know when children play their commands, whether they switch the ordering of the blocks in the coding area, and whether they enter how many times each block should play (the white number bubble shown in Figure 3). These measures will be able to tell us about children's testing or tinkering actions, which I often saw from their screen recordings. This problem can be solved by having a more advanced analytics tool that can simulate and re-create what children have done. For example, Shadowspect, a game that teaches geometry to middle school students by having them create shapes, can track the steps that children take to assess whether they create the shapes correctly (Gomez et al., 2021).

Furthermore, this study was not able to control influential factors on children's decision when using a certain coding block. Although the study tried to capture exploring actions before and after the curriculum covers block functionalities, I still cannot be sure if the exploration was due to peers' or teachers' influence. Going forward, we might need to ask teachers how much block guidance they usually provide when students create projects.

For future research, it will be meaningful if we can connect children's creation processes to their learning outcomes, such as coding competency and project quality. This connection is essential for instructional design, as it remains unclear from this study whether or under what circumstances a particular process is better for learning outcomes, not limited to block usage patterns. One potential question is whether children's painting time impacts their coding mastery levels. This study showed that the longer children painted, the fewer coding block varieties they used. However, children might still learn coding to a sufficient level even if they spend a large chunk of their time painting. Additionally, it

will be interesting to interview children on how or what processes they used to create their projects and why they used certain blocks over others. Young children's perception of their own processes and block choices might change as they become more competent creators across the curriculum. There may be reasons why they decided not to use the more complex blocks even if they knew how the blocks work. Importantly, children's understanding of these processes and block choices might be related to the purposefulness and quality of their projects. To test this hypothesis, studies can compare children's pre-sketch or design journal entries of their projects, children's interviews, and project scores. Lastly, to further understand why children explore complex coding blocks more after they are taught the blocks, studies may qualitatively observe children's complex block exploration behavior at different times. To do so, it is important to make sure that the sample size is sufficient, as exploring behavior does not occur frequently. Studies may find that guided exploration, or exploration after they learned the coding block functions, is more effective and intentional.



## References

- Adams, R. S. (2002). Understanding design iteration: Representations from an empirical study. *Understanding Design Iteration: Representations from an Empirical Study*, 16.
- Adams, R. S., Turns, J., & Atman, C. J. (2003). Educating effective engineering designers: The role of reflective practice. *Design Studies*, 24(3), 275–294. [https://doi.org/10.1016/S0142-694X\(02\)00056-X](https://doi.org/10.1016/S0142-694X(02)00056-X)
- Admiraal, W., Vermeulen, J., & Bulterman-Bos, J. (2020). Teaching with learning analytics: how to connect computer-based assessment data with classroom instruction? *Technology, Pedagogy and Education*, 29(5), 577–591. <https://doi.org/10.1080/1475939X.2020.1825992>
- Ayman-Nolley, S. (2009). Vygotsky's perspective on the development of imagination and creativity. *Creativity Research Journal*, 10. <https://doi.org/10400419209534424>
- Bairaktarova, D., Evangelou, D., Bagiati, A., & Brophy, S. (2011). Early Engineering in Young Children's Exploratory Play with Tangible Materials. *Children, Youth and Environments*, 21(2), 212–235.
- Basu, S. (2019). Using Rubrics Integrating Design and Coding to Assess Middle School Students' Open-ended Block-based Programming Projects. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 7.
- Berland, M., Baker, R. S., & Blikstein, P. (2014). Educational Data Mining and Learning Analytics: Applications to Constructionist Research. *Technology, Knowledge and Learning*, 19(1–2), 205–220. <https://doi.org/10.1007/s10758-014-9223-7>
- Berland, M., Martin, T., Benton, T., Petrick Smith, C., & Davis, D. (2013). Using Learning Analytics to Understand the Learning Pathways of Novice Programmers. *Journal of the Learning Sciences*, 22(4), 564–599. <https://doi.org/10.1080/10508406.2013.836655>
- Bers, M. U. (2018a). Coding, playgrounds and literacy in early childhood education: The development of KIBO robotics and ScratchJr. *2018 IEEE Global Engineering Education Conference (EDUCON)*, 2094–2102. <https://doi.org/10.1109/EDUCON.2018.8363498>
- Bers, M. U. (2018b). Coding and Computational Thinking in Early Childhood: The Impact of ScratchJr in Europe. *European Journal of STEM Education*, 3(3). <https://doi.org/10.20897/ejsteme/3868>
- Bers, M. U. (2019). Coding as another language: A pedagogical approach for teaching computer science in early childhood. *Journal of Computers in Education*, 6(4), 499–528. <https://doi.org/10.1007/s40692-019-00147-3>
- Bers, M. U. (2020a). Playgrounds and Microworlds: Learning to Code in Early Childhood. In N. Holbert, M. Berland, & Y. B. Kafai (Eds.), *Designing Constructionist Futures: The Art, Theory, and Practice of Learning Designs*. The MIT Press. <https://doi.org/10.7551/mitpress/12091.001.0001>
- Bers, M. U. (2020b). *Coding as a Playground: Programming and Computational Thinking in the Early Childhood Classroom*. Routledge.
- Bers, M. U., González-González, C., & Armas-Torres, M. B. (2019). Coding as a playground: Promoting positive learning experiences in childhood classrooms. *Computers & Education*, 138, 130–145. <https://doi.org/10.1016/j.compedu.2019.04.013>

- Blikstein, P. (2011). Using learning analytics to assess students' behavior in open-ended programming tasks. *Proceedings of the 1st International Conference on Learning Analytics and Knowledge - LAK '11*, 110. <https://doi.org/10.1145/2090116.2090132>
- Blikstein, P., Worsley, M., Piech, C., Sahami, M., Cooper, S., & Koller, D. (2014). Programming Pluralism: Using Learning Analytics to Detect Patterns in the Learning of Computer Programming. *Journal of the Learning Sciences*, 23(4), 561–599. <https://doi.org/10.1080/10508406.2014.954750>
- Bonawitz, E., Shafto, P., Gweon, H., Goodman, N. D., Spelke, E., & Schulz, L. (2011). The double-edged sword of pedagogy: Instruction limits spontaneous exploration and discovery. *Cognition*, 120(3), 322–330. <https://doi.org/10.1016/j.cognition.2010.10.001>
- Brennan, K. (2015). Beyond right or wrong: Challenges of including creative design activities in the classroom. *Journal of Technology and Teacher Education*, 23(3), 279–299.
- Brennan, K., Blum-Smith, S., & Haduong, P. (2021). Four principles for assessing student-directed projects. *Phi Delta Kappan*, 103(4), 44–48. <https://doi.org/10.1177/003172172111065826>
- Brennan, K., Blum-Smith, S., Peters, L., & Kang, J. (2022). Designing for Student-Directedness: How K–12 Teachers Utilize Peers to Support Projects. *ACM Transactions on Computing Education*, 22(2), 1–18. <https://doi.org/10.1145/3476515>
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, 1, 25.
- Çiftci, S., & Bildiren, A. (2020). The effect of coding courses on the cognitive abilities and problem-solving skills of preschool children. *Computer Science Education*, 30(1), 3–21. <https://doi.org/10.1080/08993408.2019.1696169>
- Coding as Another Language*. (n.d.). Retrieved April 21, 2021, from <https://sites.tufts.edu/codingasanotherlanguage/curricula/scratchjr/>
- Cook, C., Goodman, N. D., & Schulz, L. E. (2011). Where science starts: Spontaneous experiments in preschoolers' exploratory play. *Cognition*, 120(3), 341–349. <https://doi.org/10.1016/j.cognition.2011.03.003>
- Dorie, B., Cardella, M., & Svarovsky, G. (2014). Capturing the Design Thinking of Young Children Interacting with a Parent. *2014 ASEE Annual Conference & Exposition Proceedings*, 24.256.1-24.256.7. <https://doi.org/10.18260/1-2--20147>
- Emerson, A., Smith, A., Rodriguez, F. J., Wiebe, E. N., Mott, B. W., Boyer, K. E., & Lester, J. C. (2020). Cluster-Based Analysis of Novice Coding Misconceptions in Block-Based Programming. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 825–831. <https://doi.org/10.1145/3328778.3366924>
- English, L. D., & King, D. T. (2015). STEM learning through engineering design: Fourth-grade students' investigations in aerospace. *International Journal of STEM Education*, 2(1), 14. <https://doi.org/10.1186/s40594-015-0027-7>

- Erickson, F. (2006). Definition and Analysis of Data from Videotape: Some Research Procedures and Their Rationales. In *Handbook of complementary methods in education research* (pp. 177–188). Lawrence Erlbaum Associates ; Published for the American Educational Research Association.
- Evangelou, D., Dobbs-Oates, J., Bagiati, A., Liang, S., & Choi, J. Y. (2010). Talking about Artifacts: Preschool Children’s Explorations with Sketches, Stories, and Tangible Objects. *Early Childhood Research & Practice*, 16.
- Fan, J. E. (2015). Drawing to learn: How producing graphical representations enhances scientific thinking. *Translational Issues in Psychological Science*, 1(2), 170–181. <https://doi.org/10.1037/tps0000037>
- Feldman, D. H., & Fowler, R. C. (1997). The nature(s) of developmental change: Piaget, Vygotsky, and the transition process. *New Ideas in Psychology*, 15(3), 195–210. [https://doi.org/10.1016/S0732-118X\(97\)10001-0](https://doi.org/10.1016/S0732-118X(97)10001-0)
- Flannery, L. P., Silverman, B., Kazakoff, E. R., Bers, M. U., Bontá, P., & Resnick, M. (2013). Designing ScratchJr: Support for early childhood learning through computer programming. *Proceedings of the 12th International Conference on Interaction Design and Children - IDC '13*, 1–10. <https://doi.org/10.1145/2485760.2485785>
- Franklin, D., Skifstad, G., Rolock, R., Mehrotra, I., Ding, V., Hansen, A., Weintrop, D., & Harlow, D. (2017). Using Upper-Elementary Student Performance to Understand Conceptual Sequencing in a Blocks-based Curriculum. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, 231–236. <https://doi.org/10.1145/3017680.3017760>
- Gibson, E. J. (1988). Exploratory Behavior in the Development of Perceiving, Acting, and the Acquiring of Knowledge. *Annual Review of Psychology*, 39(1), 1–42.
- Gomez, M. J., RUIPÉREZ-VALIENTE, J. A., MARTÍNEZ, P. A., & KIM, Y. J. (2021). Applying Learning Analytics to Detect Sequences of Actions and Common Errors in a Geometry Game. *Sensors*, 21(4), 1025. <https://doi.org/10.3390/s21041025>
- Gopnik, A. (2012). Scientific Thinking in Young Children: Theoretical Advances, Empirical Research, and Policy Implications. *Science*, 337(6102), 1623–1627. <https://doi.org/10.1126/science.1223416>
- Govind, M., Relkin, E., & Bers, M. U. (2020). Engaging Children and Parents to Code Together Using the ScratchJr App. *Visitor Studies*, 23(1), 46–65. <https://doi.org/10.1080/10645578.2020.1732184>
- Grover, S., Basu, S., Bienkowski, M., Eagle, M., Diana, N., & Stamper, J. (2017). A Framework for Using Hypothesis-Driven Approaches to Support Data-Driven Learning Analytics in Measuring Computational Thinking in Block-Based Programming Environments. *ACM Transactions on Computing Education*, 17(3), 1–25. <https://doi.org/10.1145/3105910>
- Guo, H., Gomez, S. R., Ziemkiewicz, C., & Laidlaw, D. H. (2016). A Case Study Using Visualization Interaction Logs and Insight Metrics to Understand How Analysts Arrive at Insights. *IEEE Transactions on Visualization and Computer Graphics*, 22(1), 51–60. <https://doi.org/10.1109/TVCG.2015.2467613>
- Harel, I., & Papert, S. (Eds.). (1991). *Constructionism* (pp. xi, 518). Ablex Publishing.
- Harvey, K., & Horton, L. (1977). Bloom’s Human Characteristics and School Learning. *The Phi Delta Kappan*, 59(3), 189–193.

- Holbert, N., Berland, M., & Kafai, Y. B. (2020). Introduction: Fifty Years of Constructionism. In *Designing Constructionist Futures: The Art, Theory, and Practice of Learning Designs* (p. 16).
- Hout, M., & Elliott, S. W. (Eds.). (2011). *Incentives and Test-Based Accountability in Education*. National Academies Press.
- Ifenthaler, D., & Yau, J. Y.-K. (2020). Utilising learning analytics to support study success in higher education: A systematic review. *Educational Technology Research and Development*, 68(4), 1961–1990. <https://doi.org/10.1007/s11423-020-09788-z>
- Iseli, M., Feng, T., Chung, G., Ruan, Z., Shochet, J., & Strachman, A. (2021). *Using Visualizations of Students' Coding Processes to Detect Patterns Related to Computational Thinking*. 13.
- Kafai, Y. B., & Resnick, M. (Eds.). (1996). *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Routledge.
- Karakaya Cirit, D. (2022). Coding in Preschool Science and Mathematics Teaching: Analysis of Scratch Projects of Undergraduate Students. *International Journal of Contemporary Educational Research*. <https://doi.org/10.33200/ijcer.1031848>
- Ke, F. (2014). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. *Computers & Education*, 73, 26–39. <https://doi.org/10.1016/j.compedu.2013.12.010>
- Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why Minimal Guidance During Instruction Does Not Work: An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching. *Educational Psychologist*, 41(2), 75–86. [https://doi.org/10.1207/s15326985ep4102\\_1](https://doi.org/10.1207/s15326985ep4102_1)
- Klahr, D., & Carver, S. M. (1988). Cognitive objectives in a LOGO debugging curriculum: Instruction, learning, and transfer. *Cognitive Psychology*, 20(3), 362–404. [https://doi.org/10.1016/0010-0285\(88\)90004-7](https://doi.org/10.1016/0010-0285(88)90004-7)
- Klahr, D., & Nigam, M. (2004). The Equivalence of Learning Paths in Early Science Instruction: Effects of Direct Instruction and Discovery Learning. *Psychological Science*, 15(10), 661–667. <https://doi.org/10.1111/j.0956-7976.2004.00737.x>
- Krumm, A., Means, B., & Bienkowski, M. (2018). *Learning Analytics Goes to School: A Collaborative Approach to Improving Education* (1st ed.). Routledge. <https://doi.org/10.4324/9781315650722>
- Kuhn, D. (2010). What is Scientific Thinking and How Does it Develop? In *Handbook of Childhood Cognitive Development* (2nd ed.).
- Kyza, E. A., Georgiou, Y., Agesilaou, A., & Souropetsis, M. (2022). A Cross-Sectional Study Investigating Primary School Children's Coding Practices and Computational Thinking Using ScratchJr. *Journal of Educational Computing Research*, 60(1), 220–257. <https://doi.org/10.1177/07356331211027387>
- Lavigne, H. J., Lewis-Presser, A., & Rosenfeld, D. (2020). An exploratory approach for investigating the integration of computational thinking and mathematics for preschool children. *Journal of Digital Learning in Teacher Education*, 36(1), 63–77. <https://doi.org/10.1080/21532974.2019.1693940>

- Legare, C. H. (2014). The Contributions of Explanation and Exploration to Children's Scientific Reasoning. *Child Development Perspectives*, 8(2), 101–106. <https://doi.org/10.1111/cdep.12070>
- Leidl, K., Bers, M. U., & Mihm, C. (2017). *Programming with ScratchJr: A review of the first year of user analytics*. 6.
- Lerner, R. M., Lerner, J. V., Almerigi, J. B., Theokas, C., Phelps, E., Gestsdottir, S., Naudeau, S., Jelicic, H., Alberts, A., Ma, L., Smith, L. M., Bobek, D. L., Richman-Raphael, D., Simpson, I., Christiansen, E. D., & von Eye, A. (2005). Positive Youth Development, Participation in Community Youth Development Programs, and Community Contributions of Fifth-Grade Adolescents: Findings From the First Wave Of the 4-H Study of Positive Youth Development. *The Journal of Early Adolescence*, 25(1), 17–71. <https://doi.org/10.1177/0272431604272461>
- Liao, C. (2016). From Interdisciplinary to Transdisciplinary: An Arts-Integrated Approach to STEAM Education. *Art Education*, 69(6), 44–49. <https://doi.org/10.1080/00043125.2016.1224873>
- Lillehoj, C. J. (Goldberg), Griffin, K. W., & Spoth, R. (2004). Program Provider and Observer Ratings of School-Based Preventive Intervention Implementation: Agreement and Relation to Youth Outcomes. *Health Education & Behavior*, 31(2), 242–257. <https://doi.org/10.1177/1090198103260514>
- Liquin, E. G., & Gopnik, A. (2022). Children are more exploratory and learn more than adults in an approach-avoid task. *Cognition*, 218, 104940. <https://doi.org/10.1016/j.cognition.2021.104940>
- Lobo, M. A., & Galloway, J. C. (2008). Postural and Object-Oriented Experiences Advance Early Reaching, Object Exploration, and Means–End Behavior. *Child Development*, 79(6), 1869–1890. <https://doi.org/10.1111/j.1467-8624.2008.01231.x>
- Looijenga, A., Klapwijk, R., & de Vries, M. J. (2015). The effect of iteration on the design performance of primary school children. *International Journal of Technology and Design Education*, 25(1), 1–23. <https://doi.org/10.1007/s10798-014-9271-2>
- Mackenzie, N., & Veresov, N. (2013). How Drawing can Support Writing Acquisition: Text Construction in Early Writing from a Vygotskian Perspective. *Australasian Journal of Early Childhood*, 38(4), 22–29. <https://doi.org/10.1177/183693911303800404>
- Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21<sup>st</sup> century skills when learning programming in K-9. *Education Inquiry*, 11(1), 1–17. <https://doi.org/10.1080/20004508.2019.1627844>
- Palmer, S. B., & Wehmeyer, M. L. (2003). Promoting Self-Determination in Early Elementary School: Teaching Self-Regulated Problem-Solving and Goal-Setting Skills. *Remedial and Special Education*, 24(2), 115–126. <https://doi.org/10.1177/07419325030240020601>
- Papadakis, S., Kalogiannakis, M., & Zaranis, N. (2016). Developing fundamental programming concepts and computational thinking with ScratchJr in preschool education: A case study. *International Journal of Mobile Learning and Organisation*, 10(3), 187. <https://doi.org/10.1504/IJMLO.2016.077867>
- Papadakis, S., Kalogiannakis, M., & Zaranis, N. (2017). Designing and creating an educational app rubric for preschool teachers. *Education and Information Technologies*, 22(6), 3147–3165. <https://doi.org/10.1007/s10639-017-9579-0>
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books.

- Pascual-Leone, A., Greenberg, L. S., & Pascual-Leone, J. (2009). Developments in task analysis: New methods to study change. *Psychotherapy Research, 19*(4–5), 527–542. <https://doi.org/10.1080/10503300902897797>
- Perignat, E., & Katz-Buonincontro, J. (2019). STEAM in practice and research: An integrative literature review. *Thinking Skills and Creativity, 31*, 31–43. <https://doi.org/10.1016/j.tsc.2018.10.002>
- Pettigrew, J., Graham, J. W., Miller-Day, M., Hecht, M. L., Krieger, J. L., & Shin, Y. J. (2015). Adherence and Delivery: Implementation Quality and Program Outcomes for the Seventh-Grade keepin' it REAL Program. *Prevention Science, 16*(1), 90–99. <https://doi.org/10.1007/s11121-014-0459-1>
- Portelance, D. J., Strawhacker, A. L., & Bers, M. U. (2016). Constructing the ScratchJr programming language in the early childhood classroom. *International Journal of Technology and Design Education, 26*(4), 489–504. <https://doi.org/10.1007/s10798-015-9325-0>
- Posner, G. J., Strike, K. A., Hewson, P. W., & Gertzog, W. A. (1982). Accommodation of a scientific conception: Toward a theory of conceptual change. *Science Education, 66*(2), 211–227. <https://doi.org/10.1002/sce.3730660207>
- Price, C. B., & Price-Mohr, R. M. (2018). An Evaluation of Primary School Children Coding Using a Text-Based Language (Java). *Computers in the Schools, 35*(4), 284–301. <https://doi.org/10.1080/07380569.2018.1531613>
- Reiser, R. A. (2001). A history of instructional design and technology: Part I: A history of instructional media. *Educational Technology Research and Development, 49*(1), 53–64. <https://doi.org/10.1007/BF02504506>
- 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*, 104222. <https://doi.org/10.1016/j.compedu.2021.104222>
- Resnick, M. (2017). *Lifelong kindergarten: Cultivating creativity through projects, passion, peers, and play*. MIT Press.
- Resnick, M., & Siegel, D. (2015). *A Different Approach to Coding*. 4.
- Ryan, R. M., & Deci, E. L. (2020). Intrinsic and extrinsic motivation from a self-determination theory perspective: Definitions, theory, practices, and future directions. *Contemporary Educational Psychology, 61*, 101860. <https://doi.org/10.1016/j.cedpsych.2020.101860>
- Schulz, L. E., & Bonawitz, E. B. (2007). Serious fun: Preschoolers engage in more exploratory play when evidence is confounded. *Developmental Psychology, 43*(4), 1045–1050. <https://doi.org/10.1037/0012-1649.43.4.1045>
- ScratchJr Studies*. (n.d.). Retrieved October 21, 2022, from <https://sites.bc.edu/codingasanotherlanguage/research/research-studies/scratchjr-studies/>
- Sheridan, K. M. (2020). Constructionism in Art Studios. In *Designing Constructionist Futures: The Art, Theory, and Practice of Learning Designs* (p. 8).

- Strawhacker, A., & Bers, M. U. (2019). What they learn when they learn coding: Investigating cognitive domains and computer programming knowledge in young children. *Educational Technology Research and Development*, 67(3), 541–575. <https://doi.org/10.1007/s11423-018-9622-x>
- Strawhacker, A., Verish, C., Shaer, O., & Bers, M. U. (2020). Designing with Genes in Early Childhood: An exploratory user study of the tangible CRISPEE technology. *International Journal of Child-Computer Interaction*, 26, 100212. <https://doi.org/10.1016/j.ijcci.2020.100212>
- Taranu, M., Andersen, M. M., Bojesen, A. B., & Roepstorff, A. (2022). Trust the process: The effects of iteration in children’s creative processes on their creative products. *Psychology of Aesthetics, Creativity, and the Arts*. <https://doi.org/10.1037/aca0000492>
- Tissenbaum, M. (2020). I see what you did there! Divergent collaboration and learner transitions from unproductive to productive states in open-ended inquiry. *Computers & Education*, 145, 103739. <https://doi.org/10.1016/j.compedu.2019.103739>
- Tomlinson, C. A. (2005). *How to Differentiate Instruction in Mixed-Ability Classrooms*. Association for Supervision & Curriculum Development.
- Unahalekhaka, A., & Bers, M. U. (2021). Taking coding home: Analysis of ScratchJr usage in home and school settings. *Educational Technology Research and Development*. <https://doi.org/10.1007/s11423-021-10011-w>
- Unahalekhaka, A., & Bers, M. U. (2022). Evaluating young children’s creative coding: Rubric development and testing for ScratchJr projects. *Education and Information Technologies*, 21. <https://doi.org/10.1007/s10639-021-10873-w>
- Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. Harvard University Press.
- Vygotsky, L. S. (2004). Imagination and Creativity in Childhood. *Journal of Russian & East European Psychology*, 42(1), 7–97. <https://doi.org/10.1080/10610405.2004.11059210>
- Vygotsky, L. S. (2011). The Dynamics of the Schoolchild’s Mental Development in Relation to Teaching and Learning. *Journal of Cognitive Education and Psychology*, 10(2), 198–211. <https://doi.org/10.1891/1945-8959.10.2.198>
- Wang, L., Sy, A., Liu, L., & Piech, C. (2017). Learning to Represent Student Knowledge on Programming Exercises Using Deep Learning. In *International Educational Data Mining Society*. International Educational Data Mining Society. <https://eric.ed.gov/?id=ED596596>
- Webb, M., Davis, N., Bell, T., Katz, Y. J., Reynolds, N., Chambers, D. P., & Syslo, M. M. (2017). Computer science in K-12 school curricula of the 21st century: Why, what and when? *Education and Information Technologies*, 22(2), 445–468. <https://doi.org/10.1007/s10639-016-9493-x>

## Appendix A

### Data Screening: Descriptive Statistics

**Table A1**

*Descriptive statistics for the outcome and time-varying continuous predictors for research question 3*

Variables	Timepoint 1				Timepoint 2				Timepoint 3			
	<i>n</i>	<i>M (SD)</i>	Skewness	Kurtosis	<i>n</i>	<i>M (SD)</i>	Skewness	Kurtosis	<i>n</i>	<i>M (SD)</i>	Skewness	Kurtosis
Variety	134	10.86 (4.33)	0.13	-0.18	133	8.07 (5.36)	0.20	-0.64	128	9.49 (4.12)	0.02	-0.32
Lessons Duration (min.)	134	26.65 (14.69)	0.79	0.81	133	31.94 (12.55)	-0.04	-0.48	128	32.68 (17.95)	1.05	1.15
Project Iteration (min.)	134	9.08 (9.05)	0.91	0.18	133	5.18 (6.76)	1.26	0.74	128	6.62 (8.47)	1.67	2.66
Coding Exploration (min.)	134	2.23 (3.11)	1.63	2.56	133	1.50 (2.74)	2.21	4.40	128	2.94 (3.63)	1.48	1.66
Long Customization (min.)	134	9.24 (9.33)	1.00	-0.06	133	19.35 (10.84)	0.42	-0.68	128	17.26 (10.87)	0.93	0.71



**Table A2**

*Descriptive statistics for the outcome and time-varying continuous predictors for research question 4*

Variables	Timepoint 1				Timepoint 2				Timepoint 3			
	<i>n</i>	<i>M (SD)</i>	Skewness	Kurtosis	<i>n</i>	<i>M (SD)</i>	Skewness	Kurtosis	<i>n</i>	<i>M (SD)</i>	Skewness	Kurtosis
Adherence	135	0.66 (0.24)	-0.62	-0.37	133	0.33 (0.29)	0.61	-0.65	129	0.19 (0.15)	0.62	-0.29
Lessons Duration (min.)	135	26.60 (14.65)	0.80	0.83	133	31.94 (12.55)	-0.04	-0.48	129	32.94 (18.12)	1.01	1.00
Project Iteration (min.)	135	9.14 (9.05)	0.89	0.14	133	5.18 (6.76)	1.26	0.74	129	6.74 (8.55)	1.61	2.40
Coding Exploration (min.)	135	2.23 (3.10)	1.64	2.60	133	1.50 (2.74)	2.21	4.40	129	2.98 (3.65)	1.44	1.51
Long Customization (min.)	135	9.18 (9.33)	1.01	-0.05	133	19.35 (10.84)	0.42	-0.68	129	17.27 (10.83)	0.93	0.73

**Table A3**

*Descriptive statistics for the outcome and continuous predictors for research question 5*

Variables	<i>n</i>	Timepoint 3		
		<i>M (SD)</i>	Skewness	Kurtosis
Complexity	132	6.34 (5.31)	1.06	0.80
Lessons Duration (min.)	132	34.59 (21.28)	1.13	1.34
Pre-Explore	132	5.42 (5.99)	1.34	1.08
Post-Explore	132	18.98 (12.68)	1.20	1.63

## Appendix B

### Data Screening: Pearson's Correlations

**Table B1**

*Pearson's Correlations between Question 3 and 4 variables before dropping outliers*

	Variety	Adherence	Lessons Duration (min.)	Project Iteration (min.)	Coding Exploration (min.)	Long Customization (min.)
Variety	-					
Adherence	.58***	-				
Lessons Duration (min.)	.40***	.051	-			
Project Iteration (min.)	.44***	.28***	.54***	-		
Coding Exploration (min.)	.42***	.16***	.28***	.079	-	
Long Customization (min.)	-.07	-.29***	.72***	-.01	-.05	-

Note: \*\*\*  $p < .001$ , \*\*  $p < .01$ , \*  $p < .05$

**Table B2***Pearson's Correlations between Question 5 variables before dropping outliers*

	Complexity	Lesson Duration (min.)	Pre-Explore	Post-Explore
Complexity	-			
Lessons Duration (min.)	.45***	-		
Pre-Explore	.28***	.14	-	
Post-Explore	.40***	.05	.41***	-

Note: \*\*\*  $p < .001$ , \*\*  $p < .01$ , \*  $p < .05$

## Appendix C

## Results: Longitudinal Regression Models

Table C1

Results for Models Predicting Variety in Research Question 3

	Empty Model	Uncond. Growth	Cond. Growth with Session Duration	Cond. Growth with Session Duration + LongCus	Cond. Growth with Session Duration + LongCus + Iterate	Cond. Growth with Session Duration + LongCus + Iterate + Explore	Cond. Growth with Session Duration + LongCus + Iterate + Explore + Grade
(Intercept)	9.480***	10.852***	10.860***	13.211***	14.078***	12.418***	12.343***
	[8.981, 9.978]	[10.137, 11.568]	[10.233, 11.487]	[12.477, 13.946]	[12.894, 15.261]	[10.990, 13.845]	[10.888, 13.798]
Timepoint_2		-2.763***	-2.767***	-0.203	-0.056	-0.330	-0.334
		[-3.864, -1.662]	[-3.753, -1.780]	[-1.177, 0.771]	[-1.036, 0.923]	[-1.305, 0.644]	[-1.309, 0.641]
Timepoint_3		-1.348**	-1.348**	0.690	0.840+	0.264	0.260
		[-2.343, -0.353]	[-2.233, -0.463]	[-0.214, 1.595]	[-0.076, 1.757]	[-0.675, 1.202]	[-0.678, 1.198]
TotalMin_Centered			0.134***	0.256***	0.290***	0.229***	0.228***
			[0.108, 0.159]	[0.224, 0.289]	[0.242, 0.337]	[0.173, 0.285]	[0.172, 0.284]
LongCus				-0.255***	-0.292***	-0.226***	-0.226***
				[-0.303, -0.207]	[-0.353, -0.230]	[-0.295, -0.157]	[-0.294, -0.157]
IterateProj					-0.058+	-0.004	-0.004
					[-0.121, 0.005]	[-0.072, 0.063]	[-0.072, 0.063]
ExploreCod						0.253***	0.252***
						[0.123, 0.382]	[0.123, 0.382]
GradeLevel2nd Grade							0.200
							[-0.586, 0.986]
Num.Obs.	395	395	395	395	395	395	395
R2 Marg.	0.000	0.057	0.241	0.408	0.450	0.435	0.436
R2 Cond.	0.068	0.266	0.409	0.490		0.520	0.521
AIC	2360.1	2331.6	2246.9	2163.0	2166.9	2158.3	2160.1
BIC	2372.0	2359.5	2278.7	2198.9	2206.7	2202.1	2207.8
ICC	0.1	0.2	0.2	0.1		0.2	0.2
RMSE	4.46	3.72	3.34	3.17	3.18	3.06	3.06

+ p &lt; 0.1, \* p &lt; 0.05, \*\* p &lt; 0.01, \*\*\* p &lt; 0.001

**Table C2***Results for Models Predicting Adherence Ratio in Research Question 4*

	Empty Model	Uncond. Growth	Cond. Growth with Session Duration + No. Taught Blocks	Cond. Growth with Session Duration + No. Taught Blocks + LongCus	Cond. Growth with Session Duration + No. Taught Blocks + LongCus + Iterate	Cond. Growth with Session Duration + No. Taught Blocks + LongCus + Iterate + Explore	Cond. Growth with Session Duration + No. Taught Blocks + LongCus + Iterate + Explore + Grade
(Intercept)	0.395***	0.663***	0.663***	0.752***	0.759***	0.726***	0.774***
	[0.365, 0.426]	[0.628, 0.697]	[0.630, 0.696]	[0.711, 0.792]	[0.693, 0.826]	[0.644, 0.807]	[0.694, 0.855]
Timepoint_2		-0.337***	-0.360***	-0.279***	-0.278***	-0.284***	-0.309***
		[-0.398, -0.276]	[-0.437, -0.283]	[-0.353, -0.204]	[-0.353, -0.203]	[-0.359, -0.209]	[-0.384, -0.235]
Timepoint_3		-0.476***	-0.486***	-0.414***	-0.413***	-0.426***	-0.433***
		[-0.523, -0.429]	[-0.535, -0.436]	[-0.466, -0.363]	[-0.465, -0.361]	[-0.480, -0.371]	[-0.487, -0.378]
NumBlockTaught_centered			-0.006	-0.011	-0.011+	-0.011+	-0.019**
			[-0.021, 0.008]	[-0.024, 0.002]	[-0.025, 0.002]	[-0.025, 0.002]	[-0.032, -0.005]
TotalMin_Centered			0.004***	0.009***	0.009***	0.008***	0.009***
			[0.003, 0.006]	[0.007, 0.011]	[0.007, 0.012]	[0.005, 0.011]	[0.006, 0.012]
LongCus				-0.581***	-0.601***	-0.520***	-0.543***
				[-0.747, -0.415]	[-0.816, -0.387]	[-0.761, -0.278]	[-0.778, -0.309]
IterateProj					-0.031	0.035	0.025
					[-0.245, 0.183]	[-0.197, 0.266]	[-0.198, 0.248]
ExploreCod						0.310	0.334
						[-0.131, 0.751]	[-0.091, 0.759]
GradeLevel2nd Grade							-0.124***
							[-0.166, -0.082]
Num.Obs.	397	397	397	397	397	397	397
R2 Marg.	0.000	0.417	0.469	0.527	0.527	0.529	0.560
R2 Cond.		0.604	0.642	0.665	0.664	0.672	0.666
AIC	209.4	-10.3	-25.1	-63.3	-58.8	-57.5	-80.0
BIC	221.4	17.6	10.8	-23.5	-15.0	-9.7	-28.2
ICC		0.3	0.3	0.3	0.3	0.3	0.2
RMSE	0.31	0.17	0.16	0.16	0.16	0.15	0.16

+ p &lt; 0.1, \* p &lt; 0.05, \*\* p &lt; 0.01, \*\*\* p &lt; 0.001

## Appendix D

## Assumption Testing Plots

Table D1

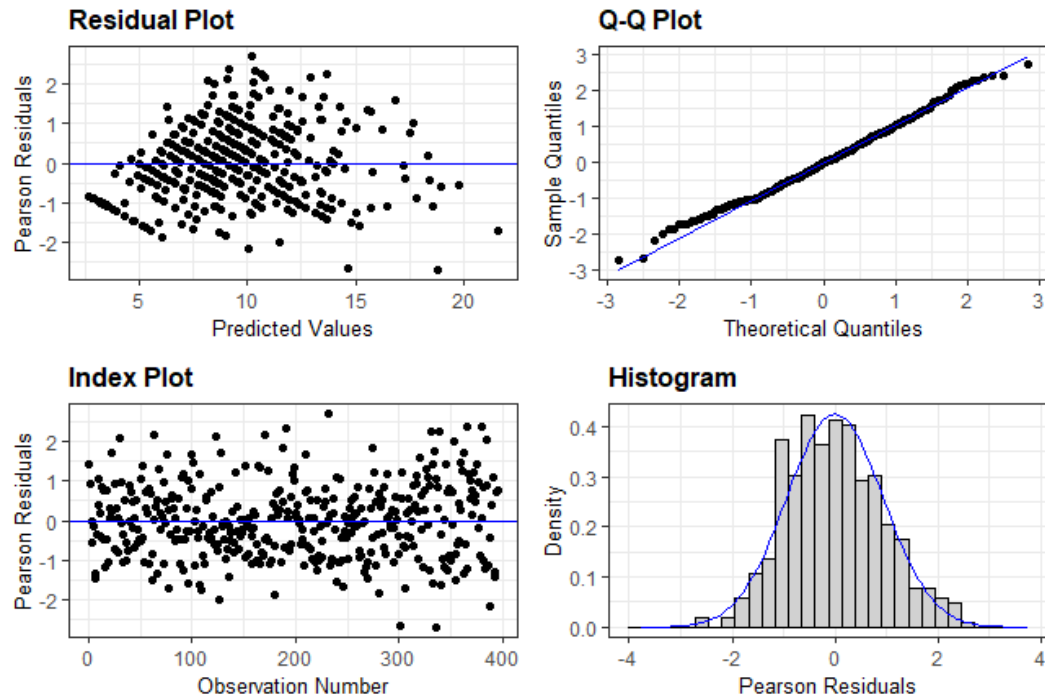
*Assumption Testing for Research Question 3*

Table D2

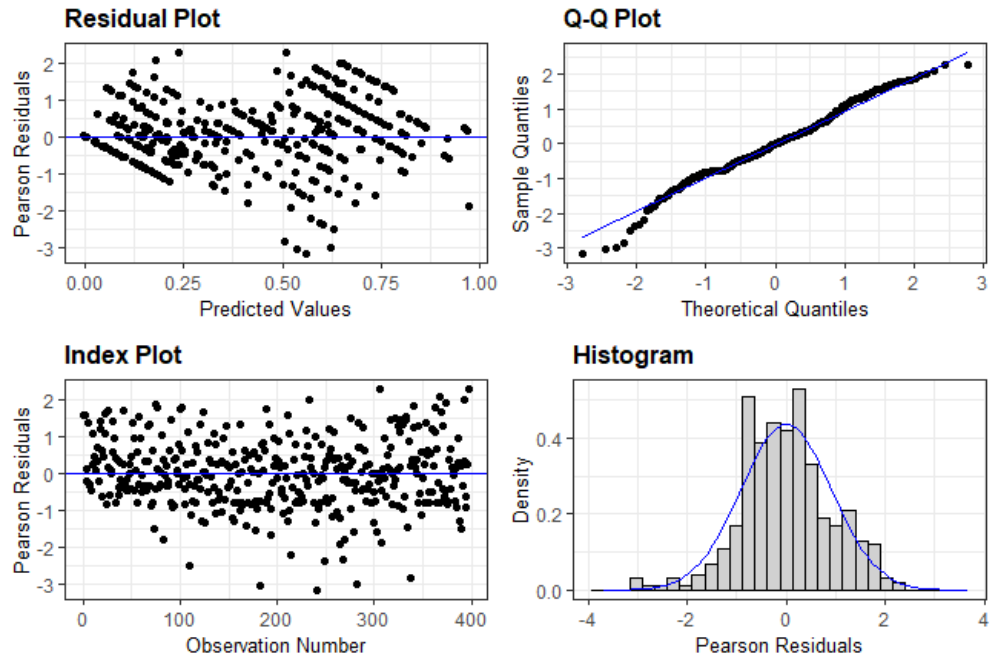
*Assumption Testing for Research Question 4*



Table D3

*Assumption Testing for Research Question 5*