

12-2017

# Using App Inventor to Explore Low-Achieving Students' Understanding of Fractions

Lorraine Ann Jacques  
*Clemson University*

Follow this and additional works at: [https://tigerprints.clemson.edu/all\\_dissertations](https://tigerprints.clemson.edu/all_dissertations)

---

## Recommended Citation

Jacques, Lorraine Ann, "Using App Inventor to Explore Low-Achieving Students' Understanding of Fractions" (2017). *All Dissertations*. 2061.  
[https://tigerprints.clemson.edu/all\\_dissertations/2061](https://tigerprints.clemson.edu/all_dissertations/2061)

This Dissertation is brought to you for free and open access by the Dissertations at TigerPrints. It has been accepted for inclusion in All Dissertations by an authorized administrator of TigerPrints. For more information, please contact [kokeefe@clemson.edu](mailto:kokeefe@clemson.edu).

USING APP INVENTOR TO EXPLORE LOW-ACHIEVING STUDENTS'  
UNDERSTANDING OF FRACTIONS

---

A Dissertation  
Presented to  
the Graduate School of  
Clemson University

---

In Partial Fulfillment  
of the Requirements for the Degree  
Doctor of Philosophy  
Learning Science

---

by  
Lorraine Ann Jacques  
December 2017

---

Accepted by:  
Dr. Danielle Herro, Committee Chair  
Dr. Nicole Bannister  
Dr. Jennie Farmer  
Dr. Brian Malloy

## ABSTRACT

A student's understanding of fraction magnitude impacts his/her understanding of algebra (e.g., Booth & Newton, 2012; Siegler et al., 2012), which then influences his/her likelihood of graduating high school (Orihuela, 2006) or succeeding in higher education (Adelman & United States., 2006; Trusty & Niles, 2004). Literature suggests that students gain this understanding when they create and work with various representations of fractions (e.g., Ainsworth, Bibby, & Wood, 2002; Panaoura et al., 2009; Siegler, Fazio, Bailey, & Zhou, 2013), which can occur when students engage in constructivist activities such as developing games (Kafai, 1996, Apr). This study examines an intervention where low-achieving eighth-grade students develop games about fraction magnitude using *App Inventor*, a novice programming environment, to determine what representations students create in their games, how their understanding of fraction magnitude develops when making their games, and what challenges they experience other than challenges concerning fractions. It uses a holistic case study with embedded units to understand the major themes for each research question while considering the influences of individual backgrounds and the various kinds of games each developed. Kolb's (1984) experiential learning theory, which states that ideas are formed by experiences and which occurs when one programs or codes a computer (Robins, Rountree, & Rountree, 2003), grounds the data analysis.

The findings of this study indicate that students primarily use numeric representations and area models to represent fraction magnitude, which are also the most common representations found in textbooks (Zhang, 2012). They developed their

understanding by working with area models, talking about area models, or by developing code to compare two fractions. The way they constructed and critiqued these representations map to the experiential learning cycle, showing that they engaged in concrete experiences with fractions, reflected on the experience, conceptualized their new learning, and experimented with that learning to develop their understanding of fraction magnitude. The challenges they experienced ranged from coding difficulties, such as decomposing their designs into components to code, to non-coding challenges, such as collaborating. Limitations of this study are discussed and implications for practice and future research are delineated.

*For Carl*

## ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Danielle Herro, for her support, patience, and great ideas that influenced not only this dissertation but my professional journey. You invited me to join the Learning Sciences community and have shown me how to incorporate my various interests into an exciting research trajectory. I have greatly appreciated your critical friendship these years.

I would also like to acknowledge my committee members, Dr. Nicole Bannister, Dr. Jennie Farmer, and Dr. Brian Malloy. I chose you for my committee because of your high expectations and I appreciate your guidance. Special thanks belong to Dr. Bannister for the support you have provided me beyond this dissertation.

Dr. Jacqueline Malloy and Dr. Phillip Wilder, thank you for being my sounding boards throughout this process, and for everything else.

Daniel Alston, Jennifer Raasch, Kathy Li, Heidi Cian, and Andrea Miller, thank you for your friendship as we navigated doctoral studies together.

John Keogh, my many experiences in education began with your encouraging me to explore non-traditional ways of teaching diverse students and to share my discoveries with other teachers. The opportunities you gave me then and the support you have given me since continue to influence my professional goals. Thank you.

For my parents, gratitude is insufficient. I am at this point because of you.

## TABLE OF CONTENTS

TITLE PAGE .....	ii
ABSTRACT .....	ii
DEDICATION .....	iv
ACKNOWLEDGEMENTS .....	v
TABLE OF CONTENTS .....	vi
LIST OF TABLES AND FIGURES .....	viii
CHAPTER 1: INTRODUCTION .....	1
Problem Statement and Research Questions.....	4
Definitions.....	5
Conclusion .....	6
CHAPTER 2: LITERATURE REVIEW .....	9
Secondary Mathematics Achievement in America .....	10
The Importance of Understanding Fractions .....	11
The Effectiveness of Intervention Methods Other Than Direct Instruction .....	17
Theoretical Framework for This Study.....	22
The Use of Computer Programming/Coding to Learn Mathematics.....	25
Challenges Faced by Students Learning to Code .....	31
Summary .....	35
CHAPTER 3: METHODOLOGY .....	39
Intervention Design.....	39
Setting and Participants.....	43
Research Questions.....	46
Research Design.....	46
Data Collection .....	48
Data Analysis .....	52
The Trustworthiness of This Study.....	59
Summary .....	63
CHAPTER 4: FINDINGS .....	65
RQ1: How Do Low-achieving Middle School Math Students Represent Fraction Magnitude as They Design and Develop Games About Fractions Using <i>App</i> <i>Inventor</i> ? .....	67
RQ2: How Do Low-achieving Middle School Math Students Develop an Understanding of Fraction Magnitude When Developing Games About Fractions Using <i>App</i> <i>Inventor</i> ? .....	73
RQ3: What challenges, other than with fractions, do low-achieving secondary math students experience in designing and developing games using <i>App Inventor</i> ? .....	96
Summary .....	128
CHAPTER 5: DISCUSSION.....	130
Relationship of Prior Research to the Study’s Findings .....	131
Limitations of the Study.....	138
Implications of the Study’s Findings .....	140
Final Reflections .....	146

REFERENCES .....	149
APPENDICES .....	165
Appendix A: Instrument for Pre- and Posttest .....	1656
Appendix B: Scoring the Instrument .....	168
Appendix C: Permission to Use Instrument.....	169
Appendix D: Math Games .....	171
Appendix E: Student Game Analysis Sheet.....	172
Appendix F: Resources for Fraction Assistance .....	173
Appendix G: Student Reference Guide for App Inventor.....	174
Appendix H: App Inventor Design Template .....	177
Appendix I: Coding Plan Template.....	178
Appendix J: Strategies to Support Students Coding .....	179
Appendix K: Student Log Template .....	181
Appendix L: Observation Protocol .....	182
Appendix M: Interview Protocol .....	183
Appendix N: Challenges Other Than with Fractions (RQ3).....	184
Appendix O: Directions for Sharing Projects .....	186
Appendix P: Sample Code Created for Students .....	187



## LIST OF TABLES AND FIGURES

Figure 2.1: The experiential learning cycle. ....	23
Figure 2.2: A novice programming environment (MIT, 2017) .....	26
Figure 2.3: Students as instructional technology designers (Israel et al., 2013).....	28
Figure 3.1: Connecting a device to a project for testing .....	40
Figure 3.2: Movable desks in the classroom.....	43
Figure 3.3: Inside a student collaboration room .....	44
Table 3.1: Data collection and purpose.....	49
Table 3.2: List of possible codes from the literature for RQ1: Representing fractions .....	53
Table 3.3: Theme development for RQ2: Developing an understanding of fractions .....	55
Table 3.4: Codes and resulting themes for RQ3: Challenges other than with fractions .....	57
Table 3.5: Matrix of findings and sources for data triangulation.....	62
Figure 4.1: A simple quiz game using buttons for answer choices.....	68
Figure 4.2: A simple quiz game using the ListPicker component.....	69
Figure 4.3: A game where the player answers the question then gets to shoot the basketball. ....	69
Figure 4.4: A game that does not ask questions.....	70
Figure 4.5: Area models from participants’ experiences. ....	71
Figure 4.6: Representing fractions as division in the code. ....	72
Figure 4.7: Examples of the questions participants created for their games. ....	73
Figure 4.8: A participant using manipulatives to create fraction magnitude questions. ....	76
Figure 4.9: Sarah and Kala’s pizza example.....	79
Figure 4.10: The problem, and resolution, Brandy and Ariel discussed.....	83
Figure 4.11: Original game design for Justin and Daniel .....	84
Figure 4.12: The instructions Justin found and copied for comparing fractions. ....	86
Figure 4.13: Justin’s code with the erroneous division expressions.....	88
Figure 4.15: The experiential learning cycle. ....	89
Figure 4.16: A student’s sketch of an area model question.....	89
Figure 4.17: Experiential learning cycle for working with area models.....	90
Table 4.1: Working with area models data mapped to the experiential learning cycle.....	90
Figure 4.18: Experiential learning cycle for talking about area models. ....	92
Table 4.2: Talking about area models data mapped to the experiential learning cycle.....	93
Figure 4.19: Experiential learning cycle for developing code.....	94
Figure 4.20: The lines Destini and Chris were trying to move. ....	100
Figure 4.21: Expressing frustration when debugging. ....	102
Figure 4.22: Matthew’s code with the correct parameters for the “Heading” property...106	106
Figure 4.23: A game with several moving components.....	108
Figure 4.24: Incorrect use of event handlers.....	112
Figure 4.25: Four game designs with features not addressed in the tutorials. ....	116
Figure 4.26: Possible events for the screen component.....	118
Figure 5.1: Justin (background) smiling as Daniel plays the working game.....	148

## CHAPTER 1: INTRODUCTION

Starting in 1969, the National Assessment of Educational Progress (NAEP) has measured what American students know and can do in various subjects; since 1978, the percent of 13-year-olds achieving a rating of “proficient” or higher in mathematics has never exceeded 35% (National Center for Education Statistics, 2015). Research suggests that middle school students who have difficulties in mathematics, specifically in understanding fractions, greatly impacts their ability to understand algebra 1 (e.g., Booth & Newton, 2012; Brown & Quinn, 2007; Siegler et al., 2012), which then negatively influences their ability to take a math course beyond algebra 2 (Sciarra, 2010), graduate high school (Orihuela, 2006) or succeed in higher education (Adelman & United States., 2006; Trusty & Niles, 2004).

There are two dominant theories on what the connection is between understanding fractions and understanding introductory algebra. The first suggests that the connection is symbolic and procedural. Algebra frequently uses fraction notation to indicate a quotient (Rotman, 1991), involves algebraic fractions when solving equations (Laursen, 1978), and uses algorithms similar to fraction arithmetic (Kieren, 1980; Wu, 2001). The second and more recent theory suggests that the connection stems from one’s understanding of fraction magnitude. Booth and Newton (2012) found that middle school students’ understanding fraction magnitude, especially unit fractions (fractions with a numerator of one), was highly correlated with algebra readiness measures. Similarly, Mou et al. (2016) found that eighth and ninth grade students’ understanding of fraction magnitude predicted their algebra achievement, even when results were controlled for the participants’ seventh

grade math achievement. Other studies have found that a student's understanding of fraction magnitude influences his/her ability to catch algebraic errors (Brown & Quinn, 2006) and helps identify students with a math learning disability (Mazzocco, Myers, Lewis, Hanich, & Murphy, 2013). Fraction magnitude is a conceptual understanding which involves (a) understanding their properties, such as the principle of equivalent fractions, (b) understanding how the numerator and the denominator determine magnitude, and (c) the ability to work with and create various ways to represent fraction magnitude, such as ordering on a number line (Gabriel et al., 2012; Jordan et al., 2013; Siegler, Fazio, Bailey, & Zhou, 2013; Vamvakoussi & Vosniadou, 2004). Understanding fractions is not easy for young learners, and the United States Department of Education (2008) recognizes it as a difficult and pervasive problem.

To understand a mathematical concept, students need to learn how to construct, interpret, and connect various representations (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000; Panaoura, Gagatsis, Deliyanni, & Elia, 2009). For fraction magnitude, The Common Core State Standards (NGA, 2010) suggest that these representations include number lines, fraction models, partitioning into equal parts, and as addition or multiplication of unit fractions. Many students, however, learn to represent fraction magnitude primarily through using area models, a specific type of fraction model in which the fraction is shown as a shaded portion of a two-dimensional figure (Zhang, 2012), which poses difficulties for transferring knowledge to other representations (Zhang, Clements, & Ellerton, 2015). Simply providing learners with multiple representations, however, is not as effective as having them construct meaning with those

representations or construct their own representations (Ainsworth, Bibby, & Wood, 2002; Greeno & Hall, 1997; Rau, Alevan, & Rummel, 2015; Zhang, Clements, & Ellerton, 2015). One way that has been proposed to allow learners to construct their own representations of mathematical concepts is game design (Kafai, 1995, April). When students are challenged to design a game about fractions, they can create and integrate various ways of representing fractions in their games (Kafai, Franke, Ching, & Shih, 1998). Another way is programming; when students develop code about fractions, they construct their own experiences and representations of fractions in the code (Feurzeig & Papert, 2011; Kafai, 1995).

Programming once required learning a formal programming language, but the advent of novice programming environments (NPEs) have made creating computer programs and apps more accessible (Peppler & Kafai, 2007). Modern NPEs utilize graphics and visual blocks of code so users can learn programming concepts without simultaneously learning syntax. They have been used to teach mathematical concepts like fractions (Harel & Papert, 1990; Kafai, 1995), proportional reasoning (Psycharis & Kynigos, 2011), and properties of infinite number sets (Kahn, Sendova, Sacristán, & Noss, 2011).

This study examined an intervention that asked low-achieving middle school students to create games about fraction magnitude using *App Inventor* (MIT, 2017), a NPE. After a brief introduction to *App Inventor* and basic game design, participants worked in groups of two or three to design and develop a game that would teach players something about fraction magnitude. The participants determined what part of fraction

magnitude the games focused on and what representations of fractions appeared in the games. Participants spent two to three days designing their games and creating a coding plan then the remainder of this ten-day intervention creating their games in *App Inventor*.

The intent of this study was to examine what representations of fractions low-achieving students used in the games they created, how they developed an understanding of fraction magnitude while developing their games, and what challenges they had beyond working with fractions as they developed their games. The literature on representing fractions and the challenges students with learning disabilities have when learning computer science or mathematics was used to understand the representations participants used in their games and the challenges they experienced other than with fractions. To investigate how their understanding developed, this study used experiential learning theory (Kolb, 1984) as a lens for the interactions participants had with fractions. Experiential learning theory states learning occurs as a cycle of four phases: concrete experience, reflective observation, abstract conceptualization, and active experimentation. Learners enter this cycle when they encounter a challenging experience and progress through the phases as they think critically about this experience (Matsuo, 2015). This study demonstrates that the way participants interacted with fractions maps to the experiential learning cycle to show how they developed their understanding of fraction magnitude during the intervention.

### **Problem Statement and Research Questions**

NPEs such as *App Inventor* are relatively new and little evidence exists on how they may be used in academics. Some studies have examined the role that using NPEs for

game design can have in science, but very few have addressed other academic subjects like mathematics. Additionally, many of these studies focus on elementary school students instead of secondary students (e.g., Kafai, Franke, Ching, & Shih, 1998; Calder, 2010). This study adds to the literature by exploring how NPEs and game design help develop and demonstrate math understanding at the secondary level and will ask the following research questions:

**RQ1:** How do low-achieving middle school math students represent fraction magnitude when developing games about fractions using *App Inventor*?

**RQ2:** How do low-achieving middle school math students develop an understanding of fraction magnitude when developing games about fractions using *App Inventor*?

Because this study specifically targeted students who struggle in mathematics, it is also important to understand what challenges these students may have when working with NPEs during a math intervention. Understanding these challenges may help identify and explain any factors that may have limited the students' development of fraction understanding (Allsopp, McHatton, & Farmer, 2010). Therefore, an additional question was investigated during this study:

**RQ3:** What challenges, other than with fractions, do low-achieving secondary math students experience in designing and developing games using *App Inventor*?

### **Definitions**

**Coding vs. programming:** Dictionary.com (coding, n.d.; programming, n.d.) defines both as the act of creating computer code. This paper will differentiate them as

follows: Programming is the formal act of creating computer code; coding represents the beginning steps of programming or programming using a tool intended for beginners (Prottzman, 2015).

**Fraction magnitude:** The size of a fraction, determined by the fraction's numerator and denominator and some object, collection, length, or position on a number line representing one "whole."

**Fraction representations:** Objects, language, symbols, or images (Lesh, Post, & Behr, 1987) used to represent fraction magnitude, including number lines, fraction models, spoken/written language, and real-world applications (NGA, 2010; Zhang, Clements, & Ellerton, 2015). For this study, representations will also include those expressed in the students' code as a form of written language.

**Low-achieving middle school math students:** Students in grades six through eight who demonstrated low achievement in prior math classes or on state assessments and are enrolled in a math assistance class in addition to their grade-level math course.

**Novice programming environment (NPE):** A computer coding environment that utilizes graphics and visual blocks of code to create programs.

## **Conclusion**

Research suggests that understanding fraction magnitude can positively influence math achievement in secondary (e.g., Booth & Newton, 2012; Brown & Quinn, 2007; Siegler et al., 2012) and post-secondary education (Adelman & United States., 2006; Trusty & Niles, 2004). Examining how low-achieving middle school students develop

and demonstrate their understanding of fraction magnitude is therefore an area worthy of study.

The second chapter reviews literature that demonstrates how interventions other than direct instruction may be an effective way to help low-achieving students develop their understanding and that creating games for mobile devices can support this learning. The review also shows that research in this area is limited, not only concerning middle school and/or low-achieving students' use of NPEs to learn mathematics but also concerning what challenges low-achieving students face when coding.

To investigate the use of NPEs as a tool for learning fractions, the third chapter describes methodology used in this study. In this chapter, the intervention is described in greater detail. This study used a holistic case study with embedded units to examine each of the research questions. The holistic approach enabled examination of the representations (RQ1) and development (RQ2) of fraction magnitude knowledge as well as the challenges faced when creating their games (RQ3), while the embedded units enabled the researcher to consider the influences of individual backgrounds and the various kinds of games each developed. Details including the role of the researcher, selection of participants, data collection/analysis, and trustworthiness issues are included in this third chapter.

The fourth chapter details the findings of this study. It begins with a description of the games that participants developed and what representations of fractions they used in their games and game designs (RQ1). The chapter then describes how participants developed their understanding of fraction magnitude during the intervention (RQ2) by (a)



presenting the results of the pre- and posttest, (b) describing the three ways participants showed their developing understanding, which were working with area models, talking about area models, and developing code to compare fractions, and (c) mapping these methods to the experiential learning cycle. The chapter then presents the findings for the challenges participants had when creating their games other than with fractions (RQ3), what supports were offered to help with these challenges, and what challenges were common to participants who did not complete their games.

The final chapter summarizes and discusses the findings. It begins by situating the findings for each research question in the relevant literature. The limitations of the study follow this section, including occurrences or details that could impact this study's transferability and credibility. The chapter then discusses the implications this study will have for practitioners, especially those wanting to use NPEs in their instruction, and for future research. The chapter concludes with a final reflection on this study.

## CHAPTER 2: LITERATURE REVIEW

This study is grounded in literature by discussing the need for students to understand fractions, the effectiveness of alternative instructional techniques in mathematics, and the effectiveness of programming or coding to learn mathematics. This chapter will begin by providing a foundation for investigating fraction magnitude understanding with low-achieving middle school students and includes (a) a brief discussion concerning secondary math achievement in America, (b) the importance of understanding fractions for secondary math achievement, (c) what it means to understand fractions, and (d) a discussion of the effectiveness of math interventions that do not use direct instruction. Following these sections, this chapter will consider the appropriateness of using novice programming environments in a secondary math intervention by discussing (a) the theoretical framework for this study, (b) the use of programming or coding to learn mathematics, and (c) the challenges faced by students learning to program or code.

Academic Search Premier, Computer Source, Education Full Text and ERIC were used to identify relevant studies. Search terms included game-based learning, novice programming environments, app development, computer programming, students designing games, education, middle school, high school, mathematics, fractions, algebra, problem solving, intervention, experiential learning, *Scratch*, and *App Inventor*. The search returned 746 articles, of which 93 were considered for this study. Other relevant articles were found using *Google Scholar* and by reviewing the references of previously found articles. Articles were rejected if they addressed non-academic learning such as

empathy, the creation of a program or tool (other than by students), working with teachers instead of students, cognitive strategies such as self-explanation, low-incidence disabilities or preschool children, using technology for non-instructional tasks such as data mining, editorials or literature reviews about related articles, or were not available in English. Five additional articles were removed as they addressed enhanced-reality programs, which is beyond the scope of this study.

### **Secondary Mathematics Achievement in America**

Research conducted in the United States demonstrates that student achievement in mathematics declines during middle school. The 2011 National Assessment of Educational Progress (NAEP) results show that 40% of fourth-graders were proficient or better in mathematics (National Center for Education Statistics, 2015). In 2015, when these students were in eighth grade, the NAEP results showed that only 33% of eighth-graders were proficient or better. Similar declines are apparent with eighth-graders who were tested in 2013 and 2011. In each of those tested years, 35% of eighth-graders were proficient or better in mathematics, while the scores when they were in fourth-grade showed 39% were proficient or better.

Helping students achieve mathematical proficiency during their early secondary school years will impact the educational opportunities these students will have as young adults. Students who fail algebra 1 in high school are more than four times as likely to not graduate as those who pass (Orihuela, 2006). More than two-thirds of students who do graduate high school enroll in college right away (United States Department of Labor, 2015), but one-quarter of them will not return to college after their first year and most

will not complete a 2- or 4-year degree (ACT, 2015). A strong predictor of college completion is high school math achievement: Students who take at least one math course beyond algebra 2 in high school are much more likely to complete a four-year college degree (Adelman & United States., 2006; Trusty & Niles, 2004), and math achievement scores and grades are the most significant variables for predicting if a high school student will take a math course beyond algebra 2 (Sciarra, 2010). These studies suggest that increasing mathematical proficiency will help students to graduate high school and complete college.

### **The Importance of Understanding Fractions**

In order to understand high school algebra, the National Mathematics Advisory Panel (United States Department of Education, 2008) recommends that students have a strong understanding of fractions first, and research supports this recommendation. A longitudinal study by Siegler et al. (2012) involving 4,276 children in both the United Kingdom and the United States compared students' mathematical understandings at age 10 and 16. They found that a student's understanding of fractions at age 10 was a better predictor of algebraic understanding at age 16 than other numeracy skills, general intellectual ability, or family background. Brown and Quinn (2007) measured 191 students' understandings of fractions and compared those scores to the students' final algebra exam grades. They found that students who struggled in algebra also struggled with fractions and those that performed well in algebra also understood fractions. Zientek, Younes, Nimon, Mittag, and Taylor (2013) measured fraction and algebra 1 skills in 573 K-8 preservice teachers. They determined that participants who could not

multiply an improper fraction by a whole number were more than five times as likely to solve algebra equations incorrectly as those who could, and that those who could not add and divide fractions or could not reduce mixed numbers, convert mixed numbers to improper fractions, and divide fractions were more than seven times as likely to be unable to solve algebra equations. In a qualitative study, Hackenberg and Lee (2015) found that students who had difficulties drawing pictures representing improper fractions also had difficulties writing algebraic equations for simple word problems involving multiplicative relationships.

Research has found two possible explanations for this connection between fractions and algebra. One line of reasoning suggests that this connection is due to the prevalence of fractions and fraction notation found in algebra. Algebra frequently uses fraction notation to indicate a quotient (Rotman, 1991), involves algebraic fractions when solving equations (Laursen, 1978), and often uses similar algorithms as arithmetic with fractions uses (Kieren, 1980; Wu, 2001). These researchers suggest that fluency with fraction manipulation would simplify a student's learning of algebra. Other researchers, however, have found a more abstract link between fraction understanding and algebra readiness. Booth and Newton (2012) studied middle school students who were registered to take algebra 1 the following school year. Students were measured on their understanding of fraction and whole number magnitude, foundational algebra knowledge (such as defining an equal sign), simple algebraic equation solving, and simple algebraic word problems. They found that understanding fraction magnitude, especially unit fractions (fractions with a numerator of one), was highly correlated with all three algebra

readiness measures. Similarly, Mou et al. (2016) compared 122 eighth and ninth grade students' fraction knowledge and algebra achievement. This study also determined that understanding fraction magnitude strongly predicted algebra achievement, even when results were controlled for the participants' seventh grade math achievement. Brown and Quinn (2006) performed an error analysis on a math skills instrument given to high school students in algebra 1. This instrument included fraction arithmetic, fraction magnitude, and one-step algebra equations that each included one fraction. The error analysis showed that students generally did not understand fraction magnitude or were not able to apply their understanding to determine the reasonableness of their solutions. For example, when asked what half of two-thirds was, over a quarter of the students gave an answer that was larger than two-thirds. They determined that this lack of understanding of fraction magnitude causes students to incorrectly apply procedures to fraction and algebraic equations and theorized that it is because students cannot determine the reasonableness of the procedure they are using.

Three longitudinal studies examined this relationship between understanding fractions and math achievement and concluded that understanding fractions, especially fraction magnitude, impacts future math achievement. Bailey, Hoard, Nugent, and Geary (2012) studied students from first through seventh grade and measured them on IQ, math achievement, and specific mathematical tasks, including fraction concepts and skills. They found that scores in sixth grade on fraction concepts and skills predicted seventh grade math achievement but sixth grade math achievement did not predict seventh grade scores on fraction concepts and skills. Siegler, Thompson, and Schneider (2011)

presented a series of problems and tasks to sixth and eighth graders that measured knowledge of fraction magnitude and fraction arithmetic skills then compared those results to the students' state exam scores. To determine if a general understanding of fractions was related to general mathematics achievement, or if specific fraction knowledge was, they conducted a regression analysis. The analysis showed that understanding fraction magnitude when controlling for fraction arithmetic skills was a strong predictor of state exam scores but the reverse, understanding fraction arithmetic when controlling for fraction magnitude understanding, was not. Mazzocco, Myers, Lewis, Hanich, and Murphy (2013) measured students in grades four through eight who were identified as typical-achievers (TA), low-achievers (LA), or as having a math learning disability (MLD) on their general mathematics achievement and their understanding of fraction magnitude. In addition to confirming that the fractions measure accurately identified students in each group, the researchers found that the MLD group showed a significant grade-level delay in understanding what the fraction "one-half" represents. Fraction comparisons that included one-half were significantly easier for TA starting in fourth grade, for LA starting in fifth, but not until grade seven for MLD. Further examination showed that this "one-half advantage" was a precursor to understanding fraction magnitude problems that did not include one-half. These three studies suggest that understanding fraction magnitude significantly impacts achievement in future math courses.

A student's knowledge of fractions during middle school effects his/her educational outcomes as young adults. These studies show that understanding fraction

magnitude and being comfortable with fraction notation impacts what a student will understand and be able to do in high school algebra. As the previous section demonstrated, failure in algebra may decrease a student's chance of graduating high school and success may increase a student's chance of completing college. Strengthening a student's readiness for algebra by increasing their understanding of fractions should help them succeed in algebra 1.

### **Understanding Fractions**

A full understanding of fractions involves understanding them on both a conceptual and a procedural level, with conceptual knowledge impacting procedural knowledge (Fuchs et al., 2013). Understanding fractions conceptually includes understanding (a) properties of rational numbers, such as the principle of equivalent fractions, (b) the relationship between the numerator and the denominator and how together they determine magnitude, and (c) various ways to represent fraction magnitude, such as ordering on a number line (Gabriel et al., 2012; Jordan et al., 2013; Siegler, Fazio, Bailey, & Zhou, 2013; Vamvakoussi & Vosniadou, 2004).

In order to gain conceptual understanding for a mathematical topic, such as fraction magnitude, research suggests that students need to learn to work with and convert between various representations of that mathematical topic (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000; Panaoura, Gagatsis, Deliyianni, & Elia, 2009). Such representational knowledge supports complex problem solving, the transfer of learning to new situations, and the understanding of more difficult concepts (Greeno & Hall, 1997; Hiebert & Carpenter, 1992; Niemi, 1996; Puttnam, Lampert, & Peterson,



1990). Mathematical representations may consist of objects, language, symbols, or images (Lesh, Post, & Behr, 1987) and, through the middle grades, come from the student's concrete experiences (NCTM, 2000, p. 68). The Common Core State Standards (NGA, 2010) and the National Council of Teachers of Mathematics (2000) suggest that students should be able to use the following representations of fraction magnitude: number lines, fraction models, as partitioning into equal parts, as the quotient of integers, and as addition or multiplication of unit fractions. The most common representations used in textbooks, however, are area models, a specific type of fraction model in which the fraction is shown as a shaded portion of a two-dimensional figure (Zhang, 2012), with circles being the recommended figure for these area models (Bray & Abreu-Sanchez, 2010; Cramer & Henry, 2002) Fractions may also be represented as portions of perimeters, capacities, lengths of objects, collections, and real-world applications in addition to the representations suggested by the Common Core State Standards, however, many students who understand area models still have difficulty transferring their knowledge to these other representations (Zhang, Clements, & Ellerton, 2015).

Simply providing learners with multiple representations, however, is not as effective as having them construct meaning with those representations or construct their own representations (Ainsworth, Bibby, & Wood, 2002; Greeno & Hall, 1997; NCTM, 2000; Rau, Aleven, & Rummel, 2015; Zhang, Clements, & Ellerton, 2015). Activities such as game design allow learners to construct their own representations of mathematical concepts (Kafai, 1995, April).

## **The Effectiveness of Intervention Methods Other Than Direct Instruction**

To help low-achieving students to succeed in mathematics, educators often use direct instruction, a method recommended for students with learning disabilities in which the instructor demonstrates a procedure then the student copies the procedure on similar problems (Gersten, Chard, Jayanthi, Baker, Morphy, & Flojo, 2009). This method has been shown to be highly effective with elementary students and students with learning disabilities because it reduces the cognitive load on working memory, but older learners without a learning disability may not need the same instructional support (Kirschner, Sweller, & Clark, 2006), especially when problem-solving (Kuhn, 2007). As an alternative to direct instruction, some researchers have explored more constructivist approaches for mathematics intervention. They include having students designing an item, exploring problems with real-life connections and data, and encouraging students to reason mathematically. The following section discusses a few of those interventions.

Having students design a real-life object has been shown to help students increase their skills in mathematics. Bottage and Haselbring (1999) conducted a study asking middle-school students with disabilities to design a cage for a pet using materials that were within a given budget. The students used a provided video for the information they required and used resources other than the teacher to learn how to perform necessary calculations. They then presented their designs and explained their reasoning. A related study asked middle-school students to design a skateboard ramp then had them build their ramps during a technology education class (Stephens, Bottge, & Rueda, 2009). In each case, the students showed improvements in computation skills afterwards, especially

when working with fractions. These studies were recently expanded to twenty-five inclusion mathematics classrooms in twenty-four middle schools (Bottge et al., 2015). Two-hundred forty-eight students, 29% of whom were identified by their districts as having a learning disability, received math instruction that was typical for their school while 223 students, 28% of whom were identified as having a learning disability, received instruction that blended video, virtual interactives, and hands-on projects. These projects focused on fractions, proportional reasoning, and budgeting and included the pet cage design and skateboard ramp building from the previous studies as well as a roll-over cage for a hovercraft and a model racecar track. Students in the experimental group showed higher gains than those in the control group on researcher-developed measures of fraction skills and problem solving, but both groups had similar gains on standardized tests for computation and problem solving. This result was consistent for both students with and without a learning disability.

A 3-year longitudinal study in Texas, however, compared three high schools that integrated project-based learning (PBL) throughout the curriculum with two high schools that had not (Han, Caparo, & Caparo, 2015), with the students in the PBL schools experiencing at least two PBL lessons every six weeks. This study found that students in the PBL schools showed greater gains on the state mathematics assessment than students in the other schools, with the highest gains shown by students who had not met proficiency levels in mathematics on previous state assessments. These studies suggest that a project-based learning approach can effectively increase students' math achievement.

To increase students' abilities in problem-solving and reasoning, two studies had students explore real-life scenarios with authentic data. Mousoulides, Christou, and Sriraman (2008) investigated the effect that mathematical modeling with authentic data had on sixth- and eighth-graders' mathematical achievement. Over three months, these students with low modeling abilities, as measured by a pretest, participated in six modelling activities, including determining which city to move to, developing a procedure for calculating how much paint it takes to paint a car, and ranking medications based on quantitative data, while a control group received traditional mathematics instruction on word problems. Problem-solving skills were measured before the intervention, after the intervention group completed three activities, and at the end. The rate of change over these three measures showed that sixth-graders increased in their problem-solving abilities two and a half times more than the control group and eighth-graders increased three times more. Van Dooren, de Bock, Hessels, Janssens, and Verschaffel (2004) studied an intervention for eighth-graders of varying math achievement levels addressing non-proportional reasoning. Students in the intervention group participated in hands-on explorations of proportional and non-proportional scenarios in geometry, such as the quadratic growth of area when enlarging two-dimensional objects, while students in the control group worked on traditional word problems. Post-testing showed that both groups performed similarly on proportional-reasoning tasks, but the intervention group answered twice as many non-proportional reasoning items correctly as they did on the pre-test while the control group showed no change. Another study explored the kind of help provided by the teacher when students

worked on complex problems (Dekker & Elshout-Mohr, 2004). Students who partially understood transformations, as measured by a pretest, in both the intervention and control groups were given identical geometry problems to solve collaboratively, but the control group received explicit help with the mathematics and the intervention group received help on working collaboratively. The intervention group scored significantly higher than the control group on a post-test addressing the geometry concepts targeted during the experiment.

These studies demonstrate the effectiveness of non-traditional instructional methods for secondary mathematics interventions, with two studies (Bottge et al., 2015; Han, Caparo, & Caparo, 2015) demonstrating the effectiveness of these methods on low-achieving students. Whether students design something, work with real scenarios, or consider abstract ideas, allowing them the time to explore and experiment with mathematics can increase their skills and problem-solving abilities.

### **The Challenge of Intervention Methods Other Than Direct Instruction**

Although the previous studies demonstrate that approaches other than direct instruction have benefits for all learners of mathematics, research has shown that direct instruction is highly effective for students with learning difficulties, especially those with learning disabilities (Gersten et al., 2009). These students often have working memory deficits and visual-spatial difficulties (Cai, Li, & Deng, 2013; Geary, 2013; Swanson & Zheng, 2013). Working memory is the system that allows one to complete complex tasks such as reasoning and problem-solving (Baddeley, 2010), and is often limited in students with learning disabilities because they have difficulties retrieving information from long-

term memory (Swanson & Zheng, 2013), which is what reduces the cognitive load on working memory (Kirschner, Sweller, & Clark, 2006). Visual-spatial processing is a component of working memory that allows one to manipulate or recall spatial information (Swanson & Zheng, 2013). Deficits in working memory, including visual-spatial processing, negatively affect one's ability to learn mathematics (Barnes & Raghobar, 2014; Cai, Li, & Deng, 2013; Geary, 2013; Swanson & Zheng, 2013), including fraction magnitude (Jordan, Resnick, Rodrigues, Hansen, & Dyson, 2016).

Direct instruction techniques reduce the cognitive load on working memory by directing the learner's attention to the key characteristics of the problem being solved (Kirschner, Sweller, & Clark, 2006; Likourezos & Kalyuga, 2017) and by presenting information sequentially and in smaller amounts (Adams & Carnine, 2003). Direct instruction has been found to be effective for teaching fractions when students learn how to draw accurate models (Sharp & Shih-Dennis, 2017) and to make connections between concrete, representational, and abstract representations (Kim, Wang, & Michaels, 2015),

Research has found that, for students with learning disabilities, the most effective instruction is a combination of direct instruction and strategy instruction, which is instruction on how to process a problem and design a potential solution process (Fuchs, Fuchs, Schumacher, & Seethaler, 2013; Gersten et al., 2009; Swanson, 2001). Strategy instruction can address mathematical problem-solving directly, which is effective when the strategy itself does not place extra burdens on working memory (Swanson, Orosco, & Lussier, 2014; Zhu, 2015). It can also address working memory directly by teaching students to say the important information in a problem aloud and repeatedly; although

this form of strategy instruction was shown to improve performance, it did not improve the actual working memory capacity of the participants (Peng & Fuchs, 2017; Swanson, Kehler, & Jerman, 2010). Either approach to strategy instruction, when used with direct instruction, reduces the burden on working memory by focusing the learner's attention on key characteristics of the problem (Fuchs, Fuchs, Schumacher, & Seethaler, 2013; Gersten et al., 2009; Swanson, 2001).

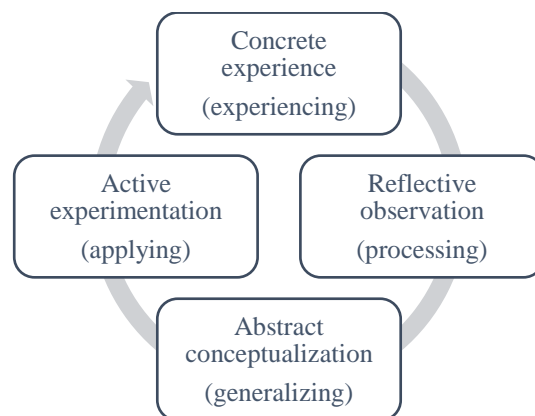
Other instructional approaches, such as constructivism, are challenging for a learner with working memory deficits because the pre-requisite knowledge is not readily available or easily retrievable from long-term memory, which can cause errors, as the working memory is unable to distinguish between important and irrelevant information, and frustration, as the working memory is unable to contain the information needed for problem-solving (Swanson & Zheng, 2013). Thus, a student with learning disabilities would likely require additional supports to be successful when direct instruction is not employed (Godino, Batanero, Cañadas, & Contreras, 2017). These supports include allowing students to use concrete or semi-concrete supports, such as counting on fingers, prompting to help them articulate their thinking, explicitly demonstrating connections between similar problems (Moscardini, 2010; Xin, Liu, Jones, Tzur, & Si, 2016), and employing direct instruction techniques when providing guidance for the student (Ding & Li, 2014).

### **Theoretical Framework for This Study**

Experiential learning theory states that ideas are formed and re-formed through experience (Kolb, 1984). Education has traditionally used direct instruction, an

instructional model where the material is explicitly taught to students (NIFDI, 2015), but John Dewey (1938/1998) suggests direct instruction prevents students from being active participants because there exists a difference between the adult-created products that form the basis of instruction and the experiences of the children who are trying to learn. As an alternative, many educators have advocated that children should learn through experience. John Dewey (1938/1998) describes learning through experience as the connection one makes between what a person does and what happens because of the person's action. Sanford, Hopper, and Starr (2015) state that learning occurs when the learner engages in building, creating, and interacting to create their own experiences.

David Kolb (1984) defines experiential learning as a cyclic process with four stages: concrete experience, reflective observation, abstract conceptualization, and active experimentation. During concrete experience, a learner engages in an activity. Then the learner reflects on that activity or experience during reflective observation. The learner gains knowledge or skills from the experience during the abstract conceptualization stage. The learner then tries out or tests their learning through active experimentation. These stages can also be thought of as experiencing, processing, generalizing, and applying.



*Figure 2.1: The experiential learning cycle.*



While most of the research on the experiential learning cycle has focused on the learner's preferences within the cycle, recently there has been consideration of the cycle holistically as an idealized learning cycle (Kolb, Boyatzis, & Mainemelis, 2001). This learning cycle models what occurs in the classroom when students are given a complex problem (Georgio, Zahn, & Meira, 2008). The "concrete experience" and "active experimentation" phases of the cycle occur when one has a challenging experience, such as those that occur when solving a complex problem, and thinking critically about that experience is when "reflective observation" and "abstract conceptualization" occur (Matsuo, 2015). For experiential learning to be effective, however, there must be a manageable gap between what the learner can presently do and what the learner wants to do; additionally, what is to be learned needs to connect to what the learner values; the learner must believe that what he/she needs to learn will help achieve his/her goal (Burns & Gentry, 1998).

Experiential learning has been applied to mathematics education. It has been found to increase students' mathematical skills (Stone, Alfeld, & Pearson, 2008) and understanding of mathematical concepts (Fest, Hiob, & Hoffkamp, 2011). Experiential learning environments allow students to express their concerns and beliefs about mathematics (Skehill, 2013), which may also impact math achievement (Wilhelm, She, & Morrison, 2011). Learning to program a computer (coding) allows experiential learning to occur because it is a process that involves regular re-examination of the problem (Robins, Rountree, & Rountree, 2003). For these reasons, this study uses experiential

learning as its theoretical framework because students created games about fractions by coding in a novice programming environment.

### **The Use of Computer Programming/Coding to Learn Mathematics**

Seymour Papert believes that programming a computer “fosters an experimental approach towards solving problems” (Feurzeig & Papert, 2011, p490). “When composing lessons on the computer, the designer combines knowledge of the computer, knowledge of programming, knowledge of computer programs and routines, knowledge of the content, knowledge of communication, human interface, and instructional design. The communication between the software producers and their medium is dynamic” (Harel & Papert, 1990, p28). He also found that situating knowledge in internalized, mental environments acted similarly to those situated in external, physical environments (Feurzeig & Papert, 2011), allowing the abstract to become concrete (Turkle & Papert, 1990). Additionally, Papert believed that programming encouraged students to reflect upon their errors. Students often view wrong answers as things to be disposed of, but when programming they focus on trying, fixing, and improving their work (Papert, 1980). When errors occur, students study them instead of ignoring them (Papert, 1980) because a program that does not work still does something that can be observed, reflected upon, and understood (Feurzeig & Papert, 2011).

In Papert’s work with teaching students to program in *Logo*, he found that programming built a relationship between the learner and the content, making the content relevant to the learner (Papert, 1980). This relationship increased their willingness to learn the content, even if previously the content was uninteresting to the student (Harel &

Papert, 1990). Papert attributes this relationship-building to the creativity of software design; students he worked with found programming to be a tool for personal expression and creativity despite the formality inherent to programming (Feurzeig & Papert, 2011). He found that “the computer is an expressive medium that different people can make their own in their own way” (Turkle & Papert, 1990).

Creating computer programs once required learning a formal programming language, but in the late 1960’s the *Logo* programming language and environment was developed (Feurzeig & Papert, 2011). Logo was designed to provide a conceptual foundation to teach mathematical and logical ways of thinking. Papert (1980) wrote of programming that it transformed the accessibility of knowledge from formal processes only into a concrete experience. Since the development of *Logo*, we have seen novice programming environments (NPEs) emerge. These NPEs utilize graphics and visual blocks of code to make software development accessible to more people; users can learn programming concepts without simultaneously learning syntax.

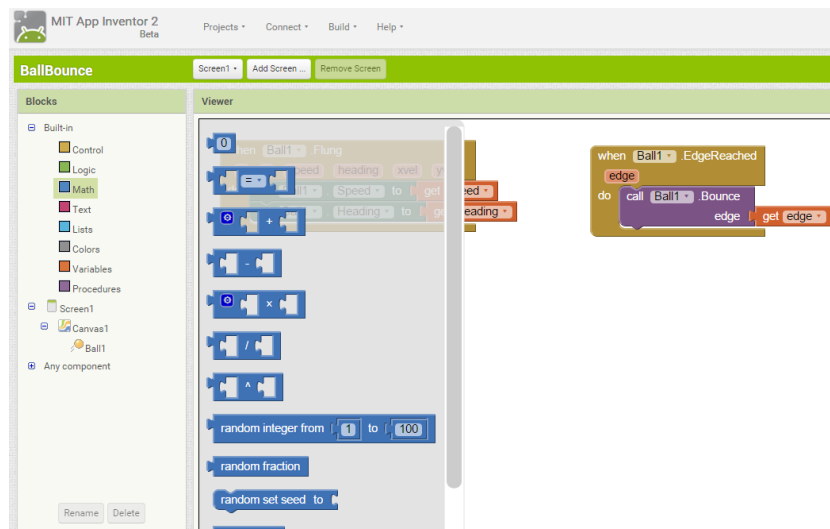
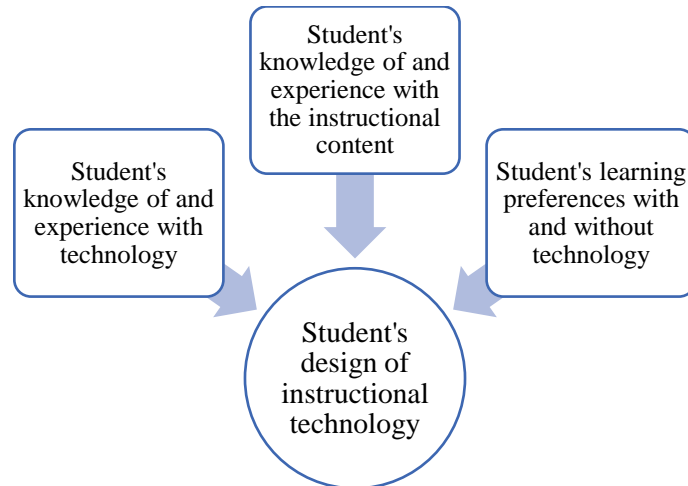


Figure 2.2: A novice programming environment (MIT, 2017)

The formal syntax of programming languages makes learning through programming difficult because they inadvertently distract novices from creativity and problem-solving (Dekhane, Xu, & Tsoi, 2013). NPEs provide a natural environment for multimedia education because they have low barriers to artistic expression and civic engagement (Pepler & Kafai, 2007). In this qualitative study, Pepler and Kafai (2007) found students who used NPEs for multimedia education were exploring independently, closely analyzing text, and expressing their cultures through the games they created. Asking students to create games for younger students allows them to transform traditional methods of instruction, which they have likely experienced for themselves, into more contemporary forms (Prensky, 2008). Designing games and models using NPEs has also been shown to help students develop narrative and journalism skills (Robertson & Good, 2005; Wolz, Stone, Pearson, Pulimood, & Switzer, 2011), visualize social studies content (An, 2016; Ioannidou, Repenning, Lewis, Cherry, & Rader, 2003), and explain scientific ideas (Baytak & Land, 2011; Ioannidou, Repenning, Lewis, Cherry, & Rader, 2003; Israel, Marino, Basham, & Spivak, 2013; Khalili, Sheridan, Williams, Clark, & Stegman, 2011; Yang & Chang, 2013).

When students design computer games for learning, they incorporate knowledge from three areas: (1) what they understand and have experienced with technology of any kind, (2) what they understand and have experienced with the educational content, and (3) their personal learning preferences, both general learning preferences and technology-specific (Israel et al., 2013). They use multiple means of expression to demonstrate their understanding of the content (Israel et al., 2013; Khalili et al., 2011) and independently

find ways to fill any gaps in their understanding (An, 2016; Khalili et al., 2011; Savignano, Williams, & Holbrook, 2014). They try to make the content accurate in their games (An, 2016; Khalili et al., 2011), but even when they do not, they are able to identify the misconceptions they represented (An, 2016).



*Figure 2.3: Students as instructional technology designers (Israel et al., 2013)*

Programming a computer to learn mathematics is not a new idea; a study from the 1970's showed that students who developed algebra programs using the BASIC programming language improved their algebra skills (Tilford, 1979). Similarly, Harel and Papert (1990) had fourth-graders develop software using *Logo*. One group developed programs that taught something about fractions and one group simply learned how to program using *Logo*. Compared to a control group that did not learn to program, both groups scored higher on the state mathematics exam of basic skills. Additionally, a fraction skills pre- and post-test measure showed that the fraction-lesson programmers had almost twice the gains than the other two groups had. Papert found that children working with *Logo* provided them with a framework, vocabulary, and experience for discussing mathematics (Feurzeig & Papert, 2011), a culture that promoted active

learning of mathematics (Papert, 1987), and “a context that mobilized creativity, personal knowledge, and a sense of doing something more important than just getting a correct answer” (Harel & Papert, 1990). Yasmin Kafai (1995) conducted a similar study where fourth-graders developed fraction games using *Logo*. She also found that these students increased their understanding of fraction concepts and skills between the pre- and post-test. Most notably, she found that students showed increased flexibility in translating between different representations of fractions. She suggested that this was because students could create their own representations of fractions in their programs.

Computer programming also builds reasoning and problem-solving skills while supporting abstraction and conceptual understanding in mathematics (Aydin, 2005). For example, Kahn, Sendova, Sacristán, and Noss (2011) had students aged nine through thirteen work with a scripting language embedded in a graphical environment where the students “trained” a virtual robot to perform computational tasks to discover concepts concerning infinity. Students were asked questions such as “Are there more natural numbers than even ones?” and created programs to discover properties about infinite number sets. At the conclusion of the study, students could reason about infinite sets and support their reasoning with what they had experienced programming. Psycharis and Kynigos (2011) used programming to explore proportional reasoning. In a *Logo*-like environment, they asked seventh graders to write programs that would shrink or enlarge characters on the screen without distorting them. They found that students could then apply their experiences to formal proportional reasoning and were better able to recognize when they needed to use such reasoning.

Convergent cognition theory (Rich, Bly, & Leatham, 2014) suggests that the gains in mathematical achievement found in these studies are due to the similarities found between computer science and mathematics. Convergent cognition happens when new knowledge in one domain is built from prior knowledge in another domain and vice-versa. This reciprocal effect happens because both domains share core attributes, but learners find that one is more abstract and the other is more applied. Jeanette Wing (2006) explains this relationship as, “Computer science inherently draws on mathematical thinking” (p. 35), but Rich, Bly, and Leatham (2014) suggest mathematics and computer science are a convergent pair because both work with variables, functions, and procedures, but mathematics is more abstract and computer science is more applied, making the relationship more reciprocal. Their research has found that students who learn to program show significant gains in mathematics understandings, especially when given enough time to explore the programming environment and when connections between the two subjects are shown to the learner. While this theory may account for the increase in mathematical skills shown in the studies described earlier in this section, other studies have found additional benefits for learning mathematics through programming.

When students design math games, they can engage students in significant thinking about mathematics (Kafai, 1996). Students tend to begin by making quiz-style games so that the math content and the game narrative are separate, resulting in traditional representations of fractions, but will integrate various representations of fractions with the game narrative when challenged to create a game that doesn't ask any questions (Kafai, Franke, Ching, & Shih, 1998). Another qualitative study found that

students engaged in spatial reasoning, problem solving, and reasoning about mathematics (Calder, 2010). In this study, students used *Scratch*, an NPE, to create games for younger students on math topics of their choosing. Because of the visual nature of their games, students could explore geometry concepts such as angles and expand their understanding of the coordinate system in addition to the mathematics that their game addressed. Both Kafai and Calder worked with late-elementary students.

Ke (2014) investigated if creating math games using NPEs fostered mathematical thinking and positive attitudes towards mathematics in middle school students. Sixty-four students were asked to create a game using *Scratch* that would teach a math idea to a younger student. Most of the resulting games addressed integer arithmetic, which the students reported as being useful math to know and math they were most comfortable with, although students also recognized that they needed to use basic algebra and geometry skills to create their games. After the experience, students' attitudes towards mathematics increased significantly, including in areas of self-confidence and motivation.

### **Challenges Faced by Students Learning to Code**

Learning to code involves developing computational thinking skills (Wing, 2008), which are “the thought processes involved in expressing solutions as computational steps or algorithms that can be carried out by a computer” (K–12 Computer Science Framework, 2016, p. 68). Grover and Pea (2013) summarize these skills as: a) Abstractions and pattern generalizations, b) systematic processing of information, c) symbol systems and representations, d) algorithmic notions of flow of control, e) structured problem decomposition, f) iterative, recursive, and parallel thinking, g)



conditional logic, h) efficiency and performance constraints, and i) debugging and systematic error detection (p. 39 – 40).

Coding and developing computational thinking skills has limited literature, however, concerning the challenges faced by students with learning difficulties (Santi & Baccaglini-Frank, 2015); this review only found three such studies. The first (Ratcliff & Anderson, 2011) explored the use of a LOGO-like environment with fourth graders with learning disabilities, including ADHD, visual-spatial disabilities, and learning disabilities affecting reading and/or math. The main challenge students in this study faced concerned manipulating the graphics, such as drawing a shape on the screen, because determining the attributes of the graphic, such as lengths or angles, was difficult for the students. Students in this study also found remembering what they learned the previous lesson and fixing a mistake in the code difficult. The second study (Santi & Baccaglini-Frank, 2015) was a case study about a high school student with math and reading learning disabilities, also using a LOGO-like environment. This study reported that the student had difficulty translating what he was thinking into computer code, even when encouraged to plan ahead using paper, employed trial-and-error strategies frequently, and had difficulty transferring what was learned in a previous task to a new task. The third study (Snodgrass, Israel, & Reese, 2016) was a comparative case study of two late-elementary students, both identified with learning disabilities that affected their reading, communication, and writing skills. The challenges reported in this study were of the adult actions towards the students: Teachers and aides did the tasks for the students when the students expressed frustration and they significantly lowered expectations for the students

to the point where they could not determine what, if anything, the student was learning. A fourth study (Israel et al., 2015) did not report what challenges students faced when learning to code, but did find that students from low-income households had more difficulties than students with learning difficulties because they had limited experience with computers. Instead, this study found that students with learning difficulties preferred coding to other instructional activities because they found it to be a safer environment for learning.

Because computer science and mathematics share core attributes so that the learning of one affects the learning of the other (Rich, Bly, & Leatham, 2014), the challenges students with learning disabilities have when learning mathematics may help explain the challenges they have when learning to code. One such challenge may be working memory deficits; problems with working memory affect one's ability to complete complex tasks and to ignore irrelevant information but do not affect one's ability to plan, such as the planning required to complete the Towers of Hanoi puzzle (Swanson & Zheng, 2013). This difficulty directly and negatively affects problem-solving skills because the student may not be able to retrieve needed information, manipulate the information to solve the problem, or transfer learning from a past problem to the current one (Allsopp, Kyger, & Lovin, 2007; Geary, 2013; Kirschner, Sweller, & Clark, 2006; Lyon & Weiser, 2013; Swanson & Zheng, 2013). Transferring learning was a challenge identified in two of the studies concerning students with learning difficulties and coding (Ratcliff & Anderson, 2011; Santi & Baccaglini-Frank, 2015). Another challenge affecting math achievement that is related to coding is learned helplessness, which is the

reluctance to try something new and the reliance on others to assist, and affects not only the learning of mathematical content but also the use of the mathematical process skills of problem, solving, reasoning and proof, communication, and making connections (Allsopp, Kyger, & Lovin, 2007). The math process skills of problem solving, reasoning, and making connections are also skills used when coding (Calao, Moreno-León, Correa, & Robles, 2015).

More studies have investigated effective strategies for supporting diverse learners than examining challenges they face. The most commonly reported effective strategy was collaboration, specifically pair programming, where two people work together on a shared computer to complete one task (Braught, Wahls, & Eby, 2011; Carver, Henderson, He, Hodges, & Reese, 2007; Cao & Xu, 2005; Chang, Thorpe, & Lubke, 1984; Denner, Werner, Campe, & Ortiz, 2014; Israel et al., 2015; McDowell, Werner, Bullock, & Fernald, 2003; Nosek, 1998; Van de Grift, 2004). Pair programming is when:

One programmer (the driver) operates the keyboard and concentrates on lower-level details of the task at hand, such as language, syntax, and control structures. The other programmer (the navigator) observes and offers suggestions, but is primarily concerned with higher level issues, such as overall program design and integration. These roles are exchanged at regular intervals, and in practice both programmers share responsibility for all aspects of the program (Braught, Wahls, & Eby, 2011, p. 1).

Pair programming is similar to structured cooperative learning groups, a strategy that allows low-achieving students improve their understanding of mathematics by working together using structured procedures and clear goals (Allsopp, Kyger, & Lovin, 2007). In computer science, having peer support increased perseverance and enjoyment of computing tasks (Carver et al., 2007; Denner et al., 2014; Israel et al., 2015;

McDowell et al., 1993; Nosek, 1998; Van de Grift, 2004). Students in pair programming environments asked for advice, requested and gave explanations, critiqued each other's approach, and summarized just completed tasks, activities that promote deeper thinking about a topic (Cao & Xu, 2005). The learning benefits of pair programming were especially significant for females and students with lower academic achievement (Braught, Wahls, & Eby, 2011). Other effective strategies for students with learning difficulties included modeling, scaffolding, having common tasks (e.g., downloading an image) explained and easily referenced, having the student "act out" what (s)he wants the computer to do, and asking probing questions (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Snodgrass, Israel, & Reese, 2016). With the limited literature, however, it is difficult to know what, if any, challenges remain for students with learning difficulties when they code. Understanding these challenges may help identify and explain any factors that may have limited the students' development of fraction understanding (Allsopp, McHatton, & Farmer, 2010).

### **Summary**

The research shows that understanding fractions are a critical component for high school and college completion. Students who understand fractions, especially fraction magnitude and notation, are better able to understand algebra 1 (Brown & Quinn, 2007; Siegler et al., 2012; Zientek et al., 2013; Hackenberg & Lee, 2015), which in turn improves a student's chance for high school completion (Orihuela, 2006). Additionally, success in algebra 1 increases the likelihood that a student will complete math courses beyond algebra 2 (Sciarra, 2010), which in turn increases the likelihood that the student

will complete college (Adelman & United States, 2006; Trusty & Niles, 2004). Studies have also shown that achievement in mathematics is more dependent on understanding fractions than it is on general mathematics ability (Siegler, Thompson, & Schneider, 2011; Bailey, Hoard, Nugent, and Geary; 2012). Specifically, it is the conceptual understanding the magnitude of fractions that is highly correlated with algebra readiness indicators (Booth & Newton, 2012; Brown & Quinn, 2006; Mou et al., 2016). Therefore, addressing students' conceptual understanding of fraction magnitude while they are in middle school is important for their future achievement. Research suggests that a student would demonstrate an understanding of fraction magnitude concepts by generating and working with various representations of fractions, including text, images, and symbols (e.g., Ainsworth, Bibby, & Wood, 2002; Lesh, Post, & Behr, 1987; Panaoura et al., 2009; Siegler, Fazio, Bailey, & Zhou, 2013). Activities such as game design would enable learners to construct their own representations of mathematical concepts (Kafai, 1995, April).

Although direct instruction is a common approach for helping students who struggle with mathematics, it may not be as effective for secondary students who already have a basic understanding of the topic (Kirschner, Sweller, & Clark, 2006) or who are developing conceptual understanding (Kuhn, 2007). More constructivist approaches for older students appear to be a more viable option. Studies conducted in middle and high schools show that students gain mathematical skills, including skills with fractions and related topics, when they design and build objects (Bottge et al., 2015), experience project-based curricula (Han, Caparo, & Caparo, 2015), work with authentic data (Van

Dooren et al., 2004; Mousoulides, Christou, & Sriraman, 2008), or receive help on collaborating instead of mathematics when problem-solving (Dekker & Elshout-Mohr, 2004). Having students design and develop games about mathematics could create such a constructivist environment.

Programming a computer is a natural environment for experimentation and reflection, key components for experiential learning (Robins, Rountree, & Rountree, 2003; Feurzeig & Papert, 2011). Experiential learning is a cyclic process of concrete experience, reflective observation, abstract conceptualization, and active experimentation (Kolb, 1984), and has been shown to increase students' understanding of mathematics (Fest, Hiob, & Hoffkamp, 2011; Wilhelm, She, & Morrison, 2011). Programming also helps make the content relevant to the learner (Papert, 1980), which allows experiential learning to be more effective (Burns & Gentry, 1998). Having students program mathematical processes and ideas transforms the content from abstract to concrete (Papert, 1980; Rich, Bly, & Leatham, 2014) and has been shown to increase students' skills in several areas of mathematics, including fractions (Tilford, 1979; Harel & Papert, 1990; Kafai, 1995; Psycharis & Kynigos, 2011).

Because this study will involve students with learning difficulties, it anticipates that the students will have challenges when working with the novice programming environment. The literature on understanding these challenges is limited, however. Challenges that have been reported include difficulties coding the graphics, coding the computer to emulate what one has in mind, and applying problem-solving strategies (Ratcliff & Anderson, 2011; Santi & Baccaglioni-Frank, 2015). These challenges are

similar to difficulties students with learning disabilities have learning mathematics (Allsopp, Kyger, & Lovin, 2007; Geary, 2013; Lyon & Weiser, 2013; Swanson & Zheng, 2013) because the math process skills of problem solving, reasoning, and making connections apply to coding (Calao, Moreno-León, Correa, & Robles, 2015). Pair programming, modeling, and scaffolding techniques have been shown to reduce these challenges (e.g., Braught, Wahls, & Eby, 2011; Cao & Xu, 2005; Israel et al., 2015).

This literature review identified three significant gaps in the literature. First, none of the studies concerning programming or coding and fractions addressed secondary students who were low-achievers in mathematics. The studies that addressed the learning of fractions involved elementary students (Harel & Papert, 1990; Kafai, 1995) and the studies that involved secondary students did not address fractions (Tilford, 1979; Psycharis & Kynigos, 2011). Second, novice programming environments are a relatively new tool with little research on their potential applications in core academic subjects or with diverse populations. Finally, there is limited research concerning the challenges that students with learning difficulties face when using a coding environment (Santi & Baccaglini-Frank, 2015). This study aims to extend the literature by examining how secondary students who are low achievers in mathematics develop and demonstrate their understanding of fraction magnitude and what challenges they still face after research-supported instructional techniques for coding are enacted.

## CHAPTER 3: METHODOLOGY

The goal of this study was to explore how low-achieving students develop their understanding of fractions when creating games about fractions. This chapter describes the intervention, the setting and participants, the research question and design used to examine the effects of the intervention, the role of the researcher, data collection and analysis, and the trustworthiness of the study.

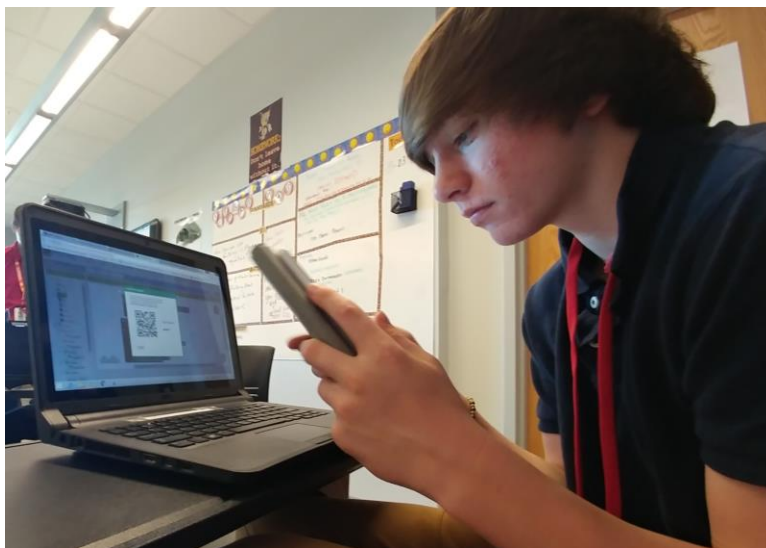
### **Intervention Design**

According to the literature, coding and programming each create an experiential learning environment (Feurzeig & Papert, 2011; Robins, Rountree, & Rountree, 2003), which increases students' understanding of mathematical concepts (Fest, Hiob, & Hoffkamp, 2011). Since the 1970's, research has shown that students who code mathematical algorithms gain a deeper understanding of the skills and concepts concerning the mathematics they coded (e.g., Harel & Papert, 1990; Kafai, 1995; Psycharis & Kynigos, 2011; Tilford, 1979). When students design games about math, they can work with multiple representations of the math while engaging in deep reasoning about the mathematical ideas (Calder, 2010; Kafai, Franke, Ching, & Shih, 1998). With the advent of novice programming environments (NPEs), students can create more complex programs, such as games, without also having to learn the syntax and complexities of a formal programming language (Peppler & Kafai, 2007). The literature suggests that understanding fraction magnitude has a significant impact on a student's ability to succeed in high school algebra (Brown & Quinn, 2007; Booth & Newton, 2012;



Mou et al., 2016); therefore, this study asked students to create games addressing fraction magnitude.

This study used *App Inventor* (MIT, 2017) for creating the games. *App Inventor* is a free NPE that allows users to create apps for the *Android* operating system, which runs on many tablets and smartphones. Like other NPEs, *App Inventor* users design the user interface by placing components on the screen then create functionality using code blocks that fit like puzzle pieces. This work is done on the *App Inventor* website. To test the app, users connect their device to their project (see figure 3.1) using *MIT AI2 Companion* (MIT, 2017), a testing environment app, or they may use an emulator on their computers, which is available on the *App Inventor* website. In addition to the coding environment, the *App Inventor* website includes thirty-one sample projects with step-by-step tutorials, a gallery of user-created apps that includes their source code, and resources for teachers.



*Figure 3.1: Connecting a device to a project for testing.*

Two pre-intervention days, 90-minutes each, were used to introduce students to *App Inventor* and game design and to conduct a pretest on their knowledge of fraction

magnitude (see appendix A for the instrument, appendix B for the scoring protocol, and appendix C for permission to use the instrument). The first of these days, students took the pretest then spent the remainder of the class period playing various math games that are freely available online (see appendix D), completing an information sheet about what they enjoyed and did not enjoy about each game (see appendix E), and engaging in a researcher-led discussion about what makes a game more or less enjoyable. The games that they played addressed whole number mathematics and included a variety of genres: puzzle, action, quiz, and sandbox. This activity helped students identify elements that they might want to consider when making their own games, such as including math help, allowing players to choose their avatar, or what genre their game should be.

The second of these days, students received an introduction to *App Inventor* and created two simple apps from its tutorials. The tutorials *Paint Pot* and *Ball Bounce* were chosen because they contained interactive graphics and used components students would likely want in their own games, such as buttons and sprites, yet could be completed in the time allocated. Two of the students had prior experience with *App Inventor* and used this day to re-familiarize themselves with the environment by following tutorials of their choice: *Magic 8-ball* and *Mole Mash*.

The intervention itself consisted of ten sessions conducted during normal class time in which the students designed and created their own games about fraction magnitude using *App Inventor*. Eight of the sessions were ninety minutes in length and two were fifty minutes. Students were placed in groups of two or three based on having similar pretest scores and similar opinions on what makes a game enjoyable. Some

adjustments were made by the classroom teacher because the pair did not get along or the pair had a history of socializing rather than working, but in each group, the students' pretest scores were within three points of each other. There were four groups of three students, nine groups of two, and one student working alone after his partner was removed from the class on the third day of the intervention.

Students coded their games following the pair programming model, in which two students share one computer to create their game (Hanks, Fitzgerald, McCauley, Murphey, & Zander, 2011). Pair programming has been found to be an effective means of reducing the challenges faced by students learning to program or code (e.g., Braught, Wahls, & Eby, 2011; Cao & Xu, 2005; Israel et al., 2015). This study used pair programming to mitigate the effects of learning to code while learning the mathematics. This study also provided students with resources to help with the mathematics (see appendix F), a brief reference guide for *App Inventor* (see appendix G), and a binder to store and organize their materials.

When designing their games, students used either a template specifically created for designing apps in *App Inventor* (appendix H; Herro, Gardner, & Boyer, 2015), graph paper, or both. When the group was satisfied with their design, they then listed the objects in their design and what action each does on a coding plan (appendix I). This coding plan was then shown to the researcher to ensure completeness. Most groups took two sessions to complete this process; two groups took three sessions. The remaining sessions, students created and coded their games. It was anticipated that students would need assistance creating their games, so a list of anticipated difficulties and what the teacher's

or researcher's response would be was created and shared with the classroom teacher (see appendix J). At the end of each session, students uploaded the day's work to *Google Classroom* and completed a daily work log (see appendix K). On the school day after the conclusion of the intervention, students completed a posttest identical to the pretest to determine if there was any change in their understanding of fraction magnitude.

### **Setting and Participants**

The setting for this study was a middle school with a focus on STEAM (Science, Technology, Engineering, Arts, and Mathematics) education, located in a city in southeastern United States. The school was designed to support student collaboration and transdisciplinary instruction by including movable desks in each classroom (see figure 3.2), collaboration rooms for the students (see figure 3.3), and open or movable space for classes to work together. The school provides each student with a laptop and has class sets of *Android* tablets available.



*Figure 3.2: Movable desks in the classroom*



*Figure 3.3: Inside a student collaboration room*

The course in which this intervention occurred was an assistance class for eighth-grade students who had low achievement in mathematics during prior grades; students in this course also attended a grade-level math course. Most of the students were recommended for this course by their seventh-grade math teacher due to low grades; two asked to take the course because they were concerned about their mathematical abilities. Two sections of this course, taught by the same instructor, were used in this study. The course met on alternate school days, usually for ninety minutes. The teacher of this course was a mathematics teacher and former database programmer. Although she had prior coding experience, she had not worked with an object-oriented programming language, graphics programming, or a novice programming environment prior to this study.

Thirty-five students, nineteen in one section and sixteen in the other, were invited to participate in the study. Each student had demonstrated some understanding of fraction magnitude by scoring at least ten points, out of twenty-four, on the pretest. Although all initially agreed and had permission to participate, three later discontinued participation;

one for disciplinary reasons not connected to this study, one for security reasons not connected to this study, and one because she was self-conscious about her ability to speak English. Of the remaining thirty-two participants, twelve identified as female, twenty identified as Black, twelve identified as Caucasian, eleven received free or reduced lunch, and fifteen received special education services. These participants differ from the school's student demographics by having a higher proportion of students identifying as Black and students receiving special education services, but they are representative of students taking low-level or remedial secondary math courses (Archbald & Farley-Ripple, 2012). All participants except one worked in groups of two or three to create their apps; one participant chose to work alone after his partner was removed from the class. Nine participants were also selected to interview after the intervention was completed. These students were chosen to represent the types of games created, the demographics of the participants, and the degree in which their group was able to complete their game.

All participants had engaged in the *Hour of Code* day (Code.org, 2017) earlier in the school year, but only three had prior coding experience beyond that. Two of the participants had taken a coding course the previous school year and worked with *App Inventor* in addition to two other novice programming environments. One of the participants belonged to an after-school club that used a novice programming environment to code functionality in robots.

Because this study used pair programming, extended participant absences could have posed a threat to implementation of the intervention. Twenty-four participants attended every session, six missed one session, and two missed two sessions. During a

student's absence, the remaining partner continued working on his or her game and received additional support from the teacher or another classmate. This additional support was to mitigate the potential of absences significantly affecting the study.

### **Research Questions**

This study examined the following research questions using data collected during the intervention:

**RQ1:** How do low-achieving middle school math students represent fraction magnitude when developing games about fractions using *App Inventor*?

**RQ2:** How do low-achieving middle school math students develop an understanding of fraction magnitude when developing games about fractions using *App Inventor*?

**RQ3:** What challenges, other than with fractions, do low-achieving secondary math students experience in designing and developing games using *App Inventor*?

### **Research Design**

This study is a holistic case study with embedded units to examine each of the research questions. A holistic case study enables the researcher to consider the global nature of a project or program (Yin, 2014, p. 55) and is appropriate when the case itself is unique (Baxter & Jack, 2008; Rowley, 2002). It allows for a broad perspective on the case, such as examining a process within a software development cycle (Runeson, Höst, Rainer, & Regnell, 2012). Because this study explored how students develop an understanding of fraction magnitude (RQ2), it is examining a process. Additionally, it seeks to understand this development, from a broader perspective, rather than how

individuals each develop their understanding. Similarly, this study seeks a more global understanding of the challenges students face when developing their games (RQ3), rather than the issues that the individual students have. A holistic approach also allows for the general classification of the ways students represent fractions in their games (RQ1). Research suggests that students will use concrete experiences (NCTM, 2000, p. 68) and area models (Zhang, 2012) primarily in the visual portions of their games; a holistic approach enables the possibility of supporting that theory (Yin, 2014, p. 55). Therefore, a holistic approach appropriately allows this study to answer all three research questions, with the primary unit of analysis being the math support class in which this study took place.

A holistic approach alone, however, may create a level of abstraction that is too vague to be useful (Yin, 2014, p. 55). Including embedded sub-units, which would be the individual participants, enabled this study to consider the influences of individual backgrounds and the various kinds of games each develops on the overall case (Baxter & Jack, 2008). Baxter and Jack (2008) further suggest:

The ability to look at sub-units that are situated within a larger case is powerful when you consider that data can be analyzed within the subunits separately (within case analysis), between the different subunits (between case analysis), or across all of the subunits (cross-case analysis). The ability to engage in such rich analysis only serves to better illuminate the case. (p. 550)

The embedded sub-units allowed this study to examine the similarities and differences among the participants while still focusing on the three research questions holistically, rather than focusing on the individuals themselves (Yin, 2014, p. 55-56).



The holistic approach with embedded sub-units was chosen over a multiple-case study for two reasons. First, the embedded sub-units are in the same context, the math support class, which supports a holistic single-case more than a multiple-case study (Baxter & Jack, 2008). Second, this study is more revelatory in nature rather than looking for replication, which supports the use of a single-case more than multiple cases (Yin, 2014).

### **Role of the Researcher**

The role of the researcher was that of a participant-observer. This role allowed interaction with the participants within the classroom culture to gain a better understanding of the setting, participants, and their behavior (Glesne, 2011). The benefit of this approach is that it enabled the researcher to question participants about what they were doing or thinking as the event occurred rather than relying on their memory during the concluding interview or as written in their daily logs (Yin, 2014). The risk involved was that the teacher or researcher might inadvertently or intentionally influence students' development of fraction understanding or how they represent fractions by offering mathematical help. This risk was reduced by limiting the researcher's and teacher's role in such discussions to those that encouraged collaboration with a peer or finding the answer they seek on their own. The researcher and teacher wore recording devices to ensure fidelity.

### **Data Collection**

This study collected the following kinds of data: observations, interviews, student work, student work logs, and the games that the participants create. Table 3.1 describes

how each was collected and for what purpose. The student work log template can be found in appendix K and the observation and interview protocols can be found in appendices L and M respectively.

Table 3.1

*Data collection and purpose*

<u>Data source</u>	<u>How collected</u>	<u>When collected</u>	<u>Connection to research question(s)</u>
Observations	Field notes Audio recordings of participants Digital photos of student work	Each class session	RQ2: Reveals how the game development process develops participants' understanding of fractions RQ3: Reveals the challenges participants faced when creating their games
Student work and student work logs	Students uploaded their work to <i>Google Classroom</i> and recorded events or challenges in a notebook after each session.	At the end of the game development course; 1 per student group	RQ1: Reveals what representations students intended to use in their games RQ2: Reveals how the game development process develops participants' understanding of fractions RQ3: Reveals the challenges participants faced when creating their games
Interviews	Audio recording	At end of the game development course; 9 participants	RQ2: Allows participants to explain what they learned about fractions and how they learned it RQ2: Allows participants to describe what they understand about fractions as a result of making their game. RQ3: Allows participants to discuss challenges they faced when developing their game

Student apps	Copied to portable memory device	At end of the intervention; 1 per student group	RQ1: Reveals what representations students used in their games RQ2: Front-end and back-end analysis reveals how participants demonstrate their understanding of fractions. The games will also be used as a tool during the interviews to give participants a focus for the discussion.
--------------	----------------------------------	---	--

Ratcliff and Anderson (2011) found that students learning to code would engage in self-talk, verbalize frustrations, and voluntarily help others. Two other studies (Cao & Xu, 2005; Israel et al., 2015) also found that students would collaborate, especially when working with a partner towards a common goal, and would also verbally summarize just-completed tasks. The participants in this study worked in groups of two or three, following the pair-programming protocol, and so were expected to engage in the verbalizations described in these studies. For this reason, the audio recordings of participants were used to illuminate what they understood or found challenging with this project.

Interviewing select participants individually after the intervention helped clarify what they learned as a result of the intervention. The interviews were conducted in a manner similar to photo elicitation, which is the use of photographs during a semi-structured interview to elicit comments from the participant (Glesne, 2010, p. 82; Torre & Murphy, 2015). Photo elicitation helps the participant to remember and reflect on the experiences related to each photograph (Torre & Murphy, 2015). This technique has been shown to be especially effective with children as it gives them something other than the

interviewer to focus on (Glense, 2010, p. 82; Leonard & McKnight, 2015; Torre & Murphy, 2015). In this study, instead of photographs, the researcher showed each interviewed participant her/his game, sections of the code (s)he has written, and work completed on paper so the participant could reflect on the representations used in the game as well as the challenges faced when creating the game. Member checking occurred at the end of the interviews by rephrasing their responses and asking what might have been misunderstood or omitted. Glesne (2011) defines member checking as "sharing interview transcripts, analytical thoughts, and/or drafts of the final report with research participants to make sure you are representing them and their ideas accurately" (p. 49). Member checking occurred during the final interviews to reduce the interruption during the school year and because of the ages of the participants (Simpson & Quigley, 2016) and before final interpretations could be made by the researcher (Angen, 2000; Carlson, 2010).

Because the participants created games, a content analysis of each game was also conducted. This analysis is a common approach used in media studies and communication (Macnamara, 2005) and allowed the researcher to understand how each participant has communicated their understanding of fractions. Therefore, the games themselves served as a fourth data source. The analysis will be described further in the following section.

## **Data Analysis**

### **Coding the First Two Research Questions**

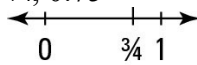
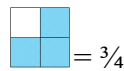

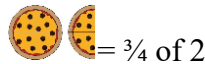
Because the first two research questions concern students' understanding or representation of fraction magnitude, the observation recordings, field notes, student work, and interviews were first analyzed to identify where students discussed, researched, or worked with fractions. The initial coding was a simple separation of the fraction data, with "representation" identifying data that described or demonstrated a fraction representation, such as numeric representations used in a game, and "understanding" identifying when participants were interacting with representations, because students develop and demonstrate their understanding of mathematics when working with or converting between representations (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000; Panaoura, Gagatsis, Deliyianni, & Elia, 2009), and where they demonstrated a change in their thinking regarding (a) the properties of rational numbers, (b) the relationship between the numerator and the denominator, and/or (c) how to represent fraction magnitude (Gabriel et al., 2012; Jordan et al., 2013; Siegler, Fazio, Bailey, & Zhou, 2013; Vamvakoussi & Vosniadou, 2004).


To answer the first research question, how do low-achieving middle school math students represent fraction magnitude as they design and develop their games, the games themselves were analyzed using content analysis on both the front-end (what the user sees) and the back-end (the code itself) and triangulated using the initial designs students created, discussions they had with their groups concerning fraction representation, and final interviews. For incomplete games, the game design was used as the primary data

source and triangulated with what participants did complete in their games as well as their discussions and interviews. A directed content analysis was used because the mathematics in each game could be analyzed according to existing theories (Hsieh & Shannon, 2005). The Common Core State Standards (NGA, 2010) and the National Council of Teachers of Mathematics (2000) suggest the following representations for fractions: numeric representations including decimals, number lines, fraction models such as area models or collections, as partitioning into equal parts, as the quotient of integers, and as addition or multiplication of unit fractions. Fractions may also be represented as portions of perimeters, capacities, lengths of objects, collections, and real-world applications (Zhang, Clements, & Ellerton, 2015). Thus, each representation participants used in their games and/or game designs, such as a drawing of an area model in the sketch of a game screen, was analyzed and coded according to these fraction representations from the literature (see Table 3.2).

Table 3.2

*List of possible codes from the literature for RQ1: Representing fractions*

<u>Code</u>	<u>Definition</u>	<u>Example(s)</u>
Numeric	Fractions in the form $a/b$ or as a decimal.	$\frac{3}{4}$ , 0.75
Number line	Fractions represented as a position on a number line.	
Area model	Fractions represented as the shaded area of a two-dimensional figure.	 = $\frac{3}{4}$
Collection	Fractions represented as a portion of individual objects.	 = $\frac{3}{4}$
Partitioning	Fractions represented as the division of one or more objects into equal parts.	 = $\frac{3}{4}$ of 2 pizzas
Quotient	Fractions represented as the division of two integers.	$3 \div 4 = \frac{3}{4}$

Unit fractions	Fractions represented as addition of like unit fractions or multiplication of a unit fraction and an integer.	$\frac{1}{4} + \frac{1}{4} + \frac{1}{4}$ $= 3 * \frac{1}{4}$ $= \frac{3}{4}$
Other	Fractions represented as portions of perimeters, capacities, lengths of objects, or in real-world applications (Zhang, Clements, & Ellerton, 2015).	 $= \frac{3}{4}$ cup

To answer the second research question, table 3.3 shows that data coded as “understanding” was then coded to identify the representation(s) participants were using, since representations are a key part of developing mathematical understanding (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000; Panaoura, Gagatsis, Deliyianni, & Elia, 2009). Then the same data was coded using process coding, which uses gerunds (“-ing” words) to identify human action in the data as a means of discovering participants’ actions and interactions in response to a problem or when trying to achieve a goal (Saldana, 2013, p. 96). Because data was initially coded as “understanding” when participants were interacting with fraction representations, the gerunds were chosen as codes to describe how this interaction was occurring. Two codes emerged during this phase: “Working” identifies identified when participants were creating representations, such as drawing an area model, or critiquing a representation another participant created, and “talking” identifies when participants were discussing representations that they did not create, such as one found in a book, or how a representation might appear for a given scenario but without creating that representation. These codes were then combined with the codes for the representations used in these data segments, which the literature suggests is how students develop and demonstrate their understanding (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000;

Panaoura, Gagatsis, Deliyianni, & Elia, 2009), resulting in the themes for how participants developed their understanding of fractions magnitude: Working with area models, talking about area models, and developing code to compare fractions.

Table 3.3

*Theme development for RQ2: Developing an understanding of fractions*

<u>Phase</u>	<u>Code</u>	<u>Criteria</u>
1: Identifying when students were developing their understanding of fractions.	Understanding	Data shows participants working with or converting between various representations of fractions (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000; Panaoura, Gagatsis, Deliyianni, & Elia, 2009). Data shows a change in participant’s thinking regarding (a) the properties of rational numbers, (b) the relationship between the numerator and the denominator, and/or (c) representing fraction magnitude (Gabriel et al., 2012; Jordan et al., 2013; Siegler, Fazio, Bailey, & Zhou, 2013; Vamvakoussi & Vosniadou, 2004).
2: Identifying the fraction representation used or referred to by the participant(s).	Same codes as RQ1.	Same criteria as RQ1.
3: Identifying how participant(s) interacted with that representation.	Working	Participant(s) created, adjusted, or manipulated a representation.
	Talking	Participants discussed a representation without the representation being present in some form or without creating, adjusting, or manipulating a representation.
4: Combining phases 2 and 3 to describe the process participants used to develop their understanding.	Working with area models	Participant(s) created, adjusted, or manipulated and area model on paper, physically, or as a digital image. (Codes “working” plus “area model”)
	Talking about area models	Participants discussed an area model without the model being present (on paper, physically, or digitally) or discussed an area model found



in one of the provided resources. (Codes “talking” plus “area model”)

Developing  
code for  
comparing  
fractions

Participant created or adjusted the code in the game that represented fractions as the division of integers. (Codes “working” plus “quotient”)

### **Coding the Third Research Question**

A similar data analysis approach was used to answer the third research question. First, the data was analyzed to identify where students experienced challenges other than with the fractions. In this study, challenges are defined as difficulties affecting all members of a group and preventing the group from progressing with their work independently or later creating difficulties that impeded independent progress. Examples of challenges include not knowing how to develop an algorithm, which prevented the group from coding, or designing a complex game, which later prevented the group from completing their game. Difficulties that did not prevent independent work were not coded as challenges, such as a vocabulary term that one group member found difficult but another member could explain.

Table 3.4 shows that instances of challenges were then coded using structural coding, which uses a content- or concept-based phrase to label or index the data, as a means of identifying the kinds of challenges students faced when creating their games as this approach allows an exploratory investigation to collect and create a topic list, which then can be used for more in-depth analysis (Saldana, 2013, p. 84). For codes that were terms used in other literature, such as decomposition or learned helplessness, the definitions or descriptions of those terms was compared to the data to ensure the code

was being used in a manner consistent with the literature. After the data were coded, the instances within each code were reexamined to ensure they met this study's definition of a challenge and were not better described by another code. Five codes were eliminated during this process because further investigation showed they did not meet the definition of a challenge or because each instance within that code was better described by another code.

Table 3.4

*Codes and resulting themes for RQ3: Challenges other than with fractions*

<u>Code</u>	<u>Definition</u>	<u>Resulting Theme</u>
Algorithm development	Participants are unable to independently create an appropriate algorithm or adjust a similar algorithm from another source.	Prior research
Debugging	Participants are unable to independently identify and fix errors in their code.	
Transfer	Participants are unable to independently recognize that their current problem is like another problem or to apply prior learning to their current problem.	
Working with angles	Participants are unable to independently identify when angle measurements are required or what angle measurement would produce the desired result in their graphics.	
Design	Participants designed games with several components on the screen that were difficult to code and/or that did not relate to one another from a coding perspective.	Specific to coding
Decomposition	Participants are unable to independently separate their game design, or elements in their design, into the required components.	
Concepts/skills	Participants are unable to independently understand coding concepts or skills relevant to <i>App Inventor</i> coding, such as components requiring code to function or choosing the correct component.	
Limitations	Limitations in the <i>App Inventor</i> environment that impeded groups from working independently.	

Vocabulary	Participants are unable to independently understand or recognize the terms used on some of the coding blocks.	
Collaboration	Participants are unable to work with their group members or follow the pair programming protocol without support from an adult.	Not specific to coding
Learned helplessness	Participants do not attempt to problem-solve independently and consistently request assistance.	
Support	Participants request additional support without attempting to problem-solve independently.	<i>Recoded - learned helplessness</i>
Syntax	Participants are unable to correctly code because of difficulties with the syntax.	<i>Recoded – concepts/skills</i>
Organization	Participants have difficulties managing time or resources	<i>Removed – did not meet</i>
Software	Participants have difficulties using software other than <i>App Inventor</i>	<i>definition of a challenge</i>
Hardware	Participants experiencing problems with a laptop or tablet.	

Within each code, the instances were organized by participants' groups and when they occurred to identify instances describing the same event. These events were then reexamined to determine if consecutive events within a group described unique challenges or a continuation of an unresolved challenge; events identified as continuations were merged with the initial event for that challenge. The groups that appeared within each code was used to determine the number of groups or participants affected and the number of unique events within each code was used to determine the frequency of the code. The frequency that each code occurred was then used to determine dominant themes (Saldana, 2013). Challenges that were addressed in prior research were separated so that how they presented in this study could be discussed with the literature. A thematic analysis then identified the implicit topics that organized the remaining challenges identified by these dominant themes, which were challenges specific to coding

and challenges not exclusive to computer science. The result was a descriptive summary of the challenges students faced during the intervention that were not related to fractions. A sample of the data, coding, and resultant themes was reviewed by a peer researcher to strengthen the credibility of this process (Creswell & Miller, 2000).

### **The Trustworthiness of This Study**

#### **Impact of Using Resources for Fraction Assistance on the Intervention and Findings**

Experiential learning is similar to the inquiry process (Kolb, 1984), which also consists of generating a hypothesis, pursuit of possible solution paths, mentally testing one of the possibilities, and making a decision (Goldman, 1983). Cognitively, a person in an inquiry cycle gathers information before exploring possible solution paths (Zhong, Wang, & Chiew, 2010). It was anticipated that the online and text-based resources provided to the students would support this first cognitive aspect of the inquiry cycle, gathering information, which would in turn support experiential learning because of the similarities between these two processes.

Assistance with fractions could have been provided by the researcher or other adult(s) in the room, but such instruction would introduce a significant threat to the trustworthiness of this study. The adult would understand the context in which the participant wants assistance because she would know the game that the participant is trying to develop and thus might target instruction to fit within that context. This instruction might then inadvertently direct or influence what representations and/or algorithms the participants are trying to develop. By having participants learn specific skills through online or text-based resources instead, the participants will need to transfer

what they have learned into the context of their game. It was anticipated that there would still be some influence on what the participants are creating, but learning in a context unconnected to the games would require that participants apply their learning to their code and/or representations of fractions, which supports deeper understanding (Spiro, 1988). To help identify the influence these resources will have on the participants' work and the impact this has on the findings of this study, participants recorded each resource in their daily log when they used them and the researcher recorded such use in field notes and recordings of the students working. Students primarily used the books provided by the researcher or Google images; appendix F lists the resources students used and their frequency. Most of the resource use occurred when students were designing their games. The researcher then compared the representations and algorithms used in these resources with the participants' games to identify areas of similarity. Nine of the games used representations similar to those found in these resources, but these representations are also the ones most commonly used to teach fraction magnitude (Zhang, 2012). Because of when the resources were used, however, it is more likely that the resources influenced what representations students used rather than their prior knowledge.

Although it was expected that some students would have difficulties applying the information from these tutorials to their algorithms (Santi & Baccaglini-Frank, 2015), the literature suggests that some difficulties will be mitigated because they will be working with a peer (e.g., Israel et al., 2015). Other recommended instructional strategies for coding, such as using probing questions about their algorithm and having the students "act out" what they wish to code (e.g., Ratcliff & Anderson, 2011), were also employed

and did not likely affect the outcomes of this study, as these techniques addressed coding knowledge rather than fraction knowledge. Recordings of these conversations between the researcher or teacher and the students were analyzed for fidelity to the intervention and no threats were identified.

### **Impact of the Tutorials on Credibility and Transferability**

Credibility was addressed by considering an alternate theory (Patton, 1999), that it was these resources and not the game design experience that had an impact on developing fraction understanding. Participants' work logs, interviews, and researcher's observations were used to triangulate the data gathered from the game analysis to address this alternate theory. While there is evidence suggesting that the resources influenced what representations students used in nine of the games, the student work for each day, student logs, and audio recordings suggest that students used the resources as a tool for exploring fraction magnitude concepts, which supports the theory that students used the resources as a part of the inquiry cycle (Zhong, Wang, & Chiew, 2010). To support the transferability of the study, the resources that participants used are included in appendix F with frequency of use to provide a more complete description of contextual factors impacting the study (Anney, 2014; Shenton, 2004).

### **Credibility of the Study Overall**

Case studies have been used to understand issues regarding NPEs (e.g., Kafai et al., 1998) and constructivist approaches to math instruction (e.g., Bottge et al., 2015). Therefore, a case study approach was a credible method for examining the use of NPEs in math instruction. The data collection and analysis methods used in this study also reflect

the techniques and artifacts used in these related case studies as well as using accepted standards for analyzing the mathematical content, such as the Common Core State Standards (NGA, 2010). Credibility was also supported through the use of multiple embedded subjects and data sources, as table 3.5 demonstrates.

Table 3.5

*Matrix of findings and sources for data triangulation*

<u>RQ</u>	<u>Finding</u>	<u># Occurrences*</u>	<u>Data Source</u>			
			<u>O</u>	<u>W</u>	<u>I</u>	<u>A</u>
1	Numeric representation	15	X	X	X	X
	Area model	10	X	X	X	X
	Division of integers	1	X	X		X
2	Working with area models	31	X	X	X	
	Talking about area models	14	X		X	
	Developing code to compare fractions	4	X			X
3	Challenges identified by prior research	86	X	X	X	X
	Challenges specific to coding	104	X	X	X	X
	Challenges not specific to coding	30	X		X	

*Note: O = Observational data, W = Student work, I = Interview, A = Students' apps*  
*\* # Occurrences = the number of unique occurrences after triangulation*

Using multiple embedded subjects helped corroborate individual experiences while using multiple data sources helped verify details that emerge during this study (Shenton, 2004).

Recording participants as they develop their games also helped credibility as this data source will capture information “in the moment” rather than relying on memory. The most significant threats to the credibility of this study, the researcher, teacher, or resources may influence participants’ understanding about fractions or design of their game, were addressed earlier in this chapter.

## **Transferability of the Study**

Shenton (2004) states that transferability can occur if practitioners can relate their situation to that described in the study. Towards that end, this study described the context in which it occurred in enough detail that similar contexts can be identified by interested parties but not so much that the participants' identities are at risk. Transferability is also strengthened when similar studies are conducted in different settings (Shenton, 2004). This study extends the work of research conducted with NPEs and mathematics, especially that of Seymour Papert and Jasmine Kafai, and thus may have greater transferability based on those prior findings, most of which involved younger students and did not specifically target those who were low-achievers in math.

## **Summary**

This study investigated the developing understanding of fraction magnitude of low-achieving middle school students as they created games about fractions using *App Inventor*, a novice programming environment. Literature suggests that students understand a math topic well, such as fraction magnitude, when they can create and work with various representations (e.g., Ainsworth, Bibby, & Wood, 2002; Lesh, Post, & Behr, 1987; Panaoura et al., 2009; Siegler, Fazio, Bailey, & Zhou, 2013), which game development encourages (Kafai, 1996, Apr). Therefore, this study employed a holistic case study with embedded units to examine each of the research questions. The holistic approach enabled examination of the representation (RQ1) and development (RQ2) of fraction magnitude understanding as well as the challenges faced when creating their



games (RQ3), while the embedded units enabled the researcher to consider the influences of individual backgrounds and the various kinds of games each develops.

## CHAPTER 4: FINDINGS

The results of this study show that participants created three kinds of representations for fractions and used these representations to develop their understanding of fraction magnitude. All participants used numeric representations and most also used area models, which are the most common representations found in math textbooks (Zhang, 2012). The results of the pre- and posttest given to the participants suggest that the participants who scored less than 60% on the pretest were the ones who developed their understanding of fraction magnitude during the intervention; most of them also created more than one kind of representation in their games and had several instances of working with or talking about fractions in the qualitative data. Participants who scored higher on the pretest or who worked only with numeric representations did not show gains on the posttest and had few conversations or artifacts concerning fractions, which suggests that these participants may not have developed their understanding of fractions during the study.

Experiential learning theory explains how the participants developed their understanding of fraction magnitude as they interacted with fractions while developing their games. Experiential learning theory is a cyclic process of four stages: concrete experience, reflective observation, abstract conceptualization, and active experimentation (Kolb, 1984). This cycle maps to the data showing how the participants worked with area models, talked about area models, and developed code for comparing fractions. to develop their understanding of fraction magnitude.

The results of this study also show that participants experienced several challenges other than with fractions when developing their games. As stated in the literature review, previous studies have found that students with learning disabilities have specific challenges when learning to code: algorithm development, debugging, transferring learning from one task to another, and working with angles in graphics (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Santi & Baccaglioni-Frank, 2015). Each of these challenges appeared in this study and were not restricted to participants who had an identified learning disability. Participants in this study also experienced additional challenges when coding and challenges that are not specific to computer science activities. These additional challenges participants had coding were challenges concerning their game designs, decomposing their game designs into components to code, coding concepts and skills, limitations in the *App Inventor* environment, and some of the vocabulary used in the coding blocks. The challenges participants had that are not exclusive to computer science were challenges collaborating and learned helplessness. Understanding these challenges may help identify and explain any factors that may have limited the participants' development of fraction understanding (Allsopp, McHatton, & Farmer, 2010).

This chapter will begin with the findings concerning the first research question, "How do low-achieving middle school math students represent fraction magnitude as they design and develop their games," by first describing the types of games that participants developed then presenting the results of the analysis of the participants' games and game designs. Following this section, this chapter will address the second

research question, “How low-achieving middle school math students develop an understanding of fraction magnitude when developing games about fractions using *App Inventor*,” by first presenting the results of the pre- and posttest, followed by the themes and data resulting from the qualitative analysis, then connecting these findings with experiential learning theory. The chapter will then address the third research question, “What challenges, other than with fractions, do low-achieving secondary math students experience in designing and developing games using *App Inventor*,” by presenting the findings for the challenges identified in prior research, the challenges concerning coding, and the challenges not exclusive to computer science. This chapter will conclude with a summary of the findings. All names of participants are pseudonyms.

**RQ1: How Do Low-achieving Middle School Math Students Represent Fraction Magnitude as They Design and Develop Games About Fractions Using *App Inventor*?**

All fifteen games that participants developed were included to determine how participants represented fraction magnitude in their games. Content analysis on both the front-end (what the user sees) and the back-end (the code itself) was used to analyze the data according to existing theories on fraction representations (Hsieh & Shannon, 2005) and triangulated using the initial designs participants created, discussions they had with their groups concerning fraction representation, and the final interviews with participants. For incomplete games, the game design was used as the primary data source for the front-end analysis and triangulated with what participants did complete in their games as well as their discussions and final interviews. This section will begin by describing the kinds

of games participants developed to provide context for the types of representations participants used in them, then it will present the findings for the first research question.

### The Types of Games Participants Developed

**Simple quiz games.** Six of the fifteen games were simple quiz games; players answered a question about fractions and a correct answer allowed the player to answer another question about fractions. All of these games had hardcoded questions (question and answer choices were predetermined rather than randomly generated) and most of these games displayed the answer choices as buttons, as shown in figure 4.1, with players' selections changing the appearance of the buttons to indicate right or wrong.

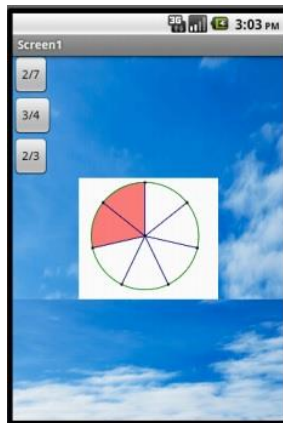


Figure 4.1: A simple quiz game using buttons for answer choices.

One game, *Masterdoom*, used a list for the answer choices. In *AppInventor*, using the *ListPicker* component instead of buttons causes the answer choices to show on a different screen and not on the screen with the question, as shown in figure 4.2. When asked why they decided to use the *ListPicker* component, Walt, one of the two boys who worked on this game, said, “Well, we didn’t mean to have the answers show up like that. But we kind of liked that [the players] had to figure [the question] out before they saw

their [answer] choices.” All but one of the simple quiz games was completed during the intervention.

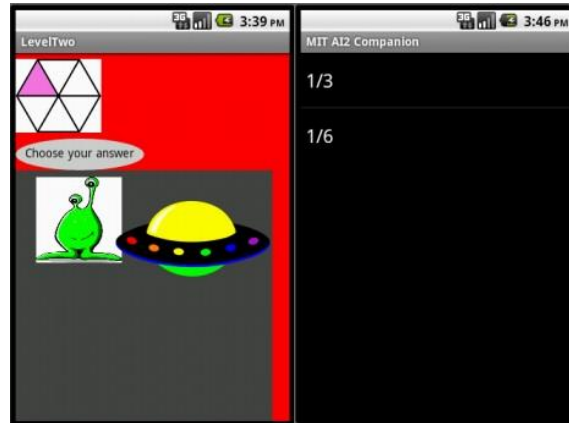


Figure 4.2: A simple quiz game using the ListPicker component.

**Games with quiz-like questions.** Eight of the fifteen games also used quiz-like questions, but in these games answering a question correctly allowed the player to do something else, such as shoot a basketball or fight zombies. Like the simple quiz games, these games hardcoded the question and answer choices. Each of these games used buttons for the answer choices, as shown in figure 4.3. None of these games were completed during the intervention; possible reasons will be described in the next section.

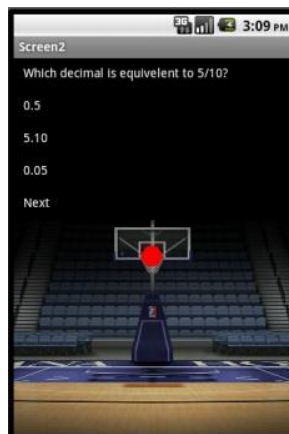
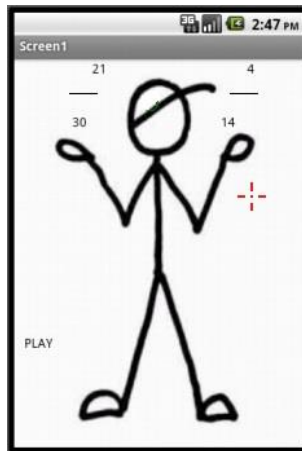


Figure 4.3: A game where the player answers the question then gets to shoot the basketball.

**A game without questions.** One game did not ask any questions. Entitled *FractionMasters*, a stick figure appears holding two randomly generated fractions in its hands and the object of the game is to “shoot” the larger fraction by dragging the crosshairs sprite to the player’s choice, as shown in figure 4.4. When the player releases the crosshairs, an image appears saying “boom” and another image appears on the stick figure’s hat indicating if the choice was correct or incorrect. Because the fractions were randomly generated, determining which fraction was the correct answer had to be calculated in the code itself. This game was completed during the intervention.



*Figure 4.4: A game that does not ask questions.*

### **Representations of Fraction Magnitude in the Games**

**Representations found in the front-end analysis.** Ten of the fifteen games used numeric representations and area models in their front-end (what the player sees), such as the games shown in figures 4.1 and 4.2. Eight of these games that used circles for their area models and one game used both circles and hexagons. Only one game used objects from participants’ experiences for their area models, pizza (see figure 4.5), although another game that used circular area models related the models to an object from his

experiences by naming the game *Space Pies* because, as he said in the final interview, “When fractions are like that it's like a pie, and there's like a spaceship [in the game], and they're in space.” An examination of the participants’ initial paper designs revealed that these representations were what they intended to create. These findings match the prior research on the representations students use to understand fraction magnitude (NCTM, 2000, p. 68; Zhang, 2012).



Figure 4.5: Area models from participants’ experiences.

Five of the fifteen games used only numeric representations in their front-end. The games depicted in figures 4.3 and 4.4 are examples of games using only numeric representations. Three of these games involved comparing fractions, one of which was completed, and two asked players to match fractions with a decimal equivalent, one was completed and one was completed enough to be a working prototype for what the participants intended. Two participants who were partners chose to use fractions with decimal equivalents because it would relate to the players’ lives and help in understanding money. As one of them, Kassidy, said in her final interview:

So three-fourths was a good example because we would talk like three-fourths as using quarters. And so the full would always equal one-hundred, so like one-



hundred minus twenty-five is seventy-five, so. We decided to do small things like that, you know, that they could think about it. Not in like a fraction way but you know like if they learn it this way, then they could use it in life, too.

Kassidy was also considering fractions as the division of two integers. When I asked if she found converting fractions to decimals easy or hard, she replied:

I found it easy because some of it you know was just dividing or basic things like if it was one-half it would be point five. And so some of it was a lot, lot easier than the others. Example, like one of the hard ones would be four-fifths, which you couldn't really relate that one to money a whole lot, so you kind of had to think about it more.

The designs that these participants sketched on paper show that they planned on using only numeric representations in their games, although one game used another representation in their code, as the back-end analysis shows.

**Representations found in the back-end analysis.** Because only one game, *FractionMasters* (figure 4.4), randomly generated the fraction scenarios instead of predetermining the problems, it was the only game that included fraction representations in the back-end (code). In this game, Justin had the game randomly generate fractions by randomly generating the numerators and denominators separately. Then as figure 4.6 shows, he represented the fractions as division so the code could compare the values.



Figure 4.6: Representing fractions as division in the code.

Unlike how Kassidy thought her players would use fractions when they tried to identify their decimal equivalents, Justin did not initially intend to represent fractions as the division of two integers. Instead, he founded he needed to use this representation so his game could compare the numeric representation of the fractions he used in the front-

end. Representing fractions two ways in their games, either with area models or the division of two integers, contributed to participants' understanding of fractions, as the next section demonstrates.

**RQ2: How Do Low-achieving Middle School Math Students Develop an Understanding of Fraction Magnitude When Developing Games About Fractions Using *App Inventor*?**

Although only six of the fifteen games were completed during the intervention, all participants worked with fractions at least during the design phase of their projects, which happened during the first two or three days of the intervention. During this phase, participants discussed what they wanted the fraction portion of their games to be; for the quiz-like games, this often included creating the questions their game would ask (see figure 4.7). All but three of the student teams revisited fractions near the end of the intervention as they completed or tried to complete their games. Thus, most participants worked directly with fractions for five or six days out of the ten allowed for this project.

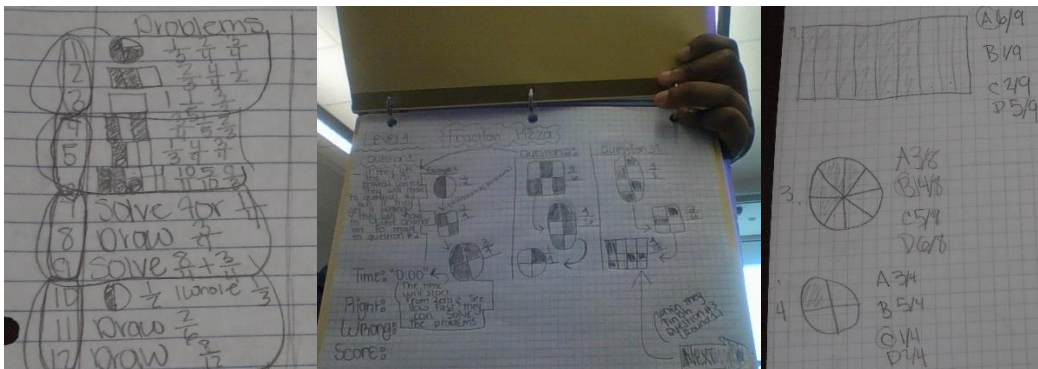


Figure 4.7: Examples of the questions participants created for their games.

A paired t-test on the pre- and posttest suggested that only the participants who scored less than 60% on the pretest developed their understanding of fraction magnitude

during this study. Examination of the transcripts, observational data, and student work supported this finding and revealed three main themes on how participants developed their understanding of fraction magnitude: They worked with area models, they talked about area models, or they developed code for comparing fractions. This section will first present the results of the pre- and posttest then present the findings for each of the qualitative themes and connect them to the four phases of Kolb's (1984) experiential learning cycle: concrete experience, reflective observation, abstract conceptualization, and active experimentation.

To connect the findings to the experiential learning cycle, the data within each theme was first examined to identify where participants entered the cycle and at what phase by identifying a challenging experience participants had with fractions and mapping it to "concrete experience" when the participant was working with fractions on paper or "active experimentation" when the participant was trying to verbalize an idea about fractions (Matsuo, 2015). This entry phase was then compared across the data to determine if it was consistent for that theme. Next, the data was mapped to the four phases of the cycle to identify where participants experienced, reflected upon, conceptualized, and experimented with fraction magnitude (Matsuo, 2015). Finally, the data for each phase was examined to create a generalized description of what occurred within that phase. Because the phase "abstract conceptualization" often occurs within one's mind (Kolb, 1984), it was directly observed in only one instance where the participant was thinking aloud. Thus, the data was re-examined to identify instances where the participant entered the next phase, "active experimentation," to determine if

“abstract conceptualization” could be inferred from the participant’s actions. When the participant said or did something demonstrating a change in their thinking, “abstract conceptualization” was determined to have occurred but not been observable (Matsuo, 2015); otherwise, it was determined that there was no evidence for this phase.

### **Results of the Pre- and Posttest**

Participants in both classes ( $n = 32$ ) took the pretest one week prior to the start of the intervention and the posttest three days after the intervention, with scores on each test ranging from 10 to 20 out of a possible 24 points. A paired t-test revealed that there was not a significant difference between the pretest ( $M = 14.15$ ,  $SD = 2.83$ ) and the posttest ( $M = 14.7$ ,  $SD = 2.43$ ). Observation of the raw scores, however, suggested that there could be a difference between the pre- and posttest for participants who scored less than 60% (14 points or lower) on the pretest; a paired t-test on this subset ( $n = 18$ ) confirmed that there was a significant difference ( $\alpha = .05$ ) between the pretest ( $M = 11.94$ ,  $SD = 1.39$ ) and the posttest ( $M = 13.67$ ,  $SD = 2.2$ ),  $t(17) = -2.62$ ,  $p = .02$ . Although these results should not be used for generalizations because the sample size is small, they do suggest who in this study developed an understanding of fraction magnitude. Analysis of the qualitative data supports this finding because participants who scored above 60% on the pretest had few conversations or artifacts addressing fractions, none of which could be identified as a challenging experience with fractions (Matsuo, 2015). The findings in this section will use these eighteen participants.

## Findings for the Three Themes

**Theme 1: Working with area models.** Although math textbooks were available for student use, the five participants who were interviewed and who used area models in their games stated that they created their own questions. Examination of the student work and discussions revealed that all the games that used area models used questions that their designers created. For five of the games that used area models, this creation process involved participants drawing fraction magnitude representations, sometimes with the aid of manipulatives that the teacher made available (see figure 4.8). Working with area models to make their game questions developed their understanding of fraction magnitude.



*Figure 4.8: A participant using manipulatives to create fraction magnitude questions.*

In the first days of the intervention, participants looked at math textbooks for third graders to see what kinds of fraction magnitude questions they could ask in their games and created similar questions on paper. During this process, participants realized they had misunderstandings or knowledge gaps concerning fraction magnitude. For example, as Keith was looking at problems in a book, he said to his partner, “Third grade fractions, one half equals, what, two fourths. One half equals two fourths. One third equals what

over six... I don't know, these are not even, how are these third-grade problems?"

Sometimes encountering a difficulty led the participant to rewrite the question rather than work to find the answer, such as Matthew did when he talked to himself as he created his questions, "Let's see, what about this one. One fourth equals blank 8? No, too hard. One half equals what?" These difficulties led many participants to use area models because, as Brian explained in the post-interview, "It's kind of like an easy way to start off by looking at pictures and kind of just count. You can count and get your answer." Choosing to use area models did not eliminate participants' difficulties, however. For example, Greg was sharing the questions he created with his partner, Katherine, when she found a problem with one:

Katherine: What's number 4?

Greg: Where's, what graph has  $\frac{1}{4}$  shaded?

Katherine: I just don't know. Both graphs have  $\frac{1}{4}$  shaded.

Greg: No, only one does.

Katherine: No, both do. Count!

[Greg counts on his area models.]

Greg: Oh, right. Ok, this one has two answers then.

In most of these teams, one partner initially took responsibility for creating the questions. This person was not always the one who understood area models best, as the Greg and Katherine discussion above demonstrates, but even when the question creator was the better student with area models, he or she found ways to involve the other partner in learning, such as how Matthew involved Rhianna:

Matthew: Our graph has three fourth shaded in minus one fourth equals what?

Rhianna: That's too hard, that would be like what is that?

Matthew: Three fourths minus one fourths, Rhianna. Two fourths.

Rhianna: I don't know what that is.

Matthew: Draw a graph that's two fourths shaded in.

By having her draw this area model, Matthew was giving Rhianna a chance to work with the representation, too. In another group, Walt involves John by asking him the questions he has prepared for their game.

Walt: Well, what is shaded in this picture?

John: Three-fourths.

Walt: What is shaded in this graph?

John: I thought we did that.

Walt: No, it's different.

John: Oh, okay. This one has two-fourths shaded.

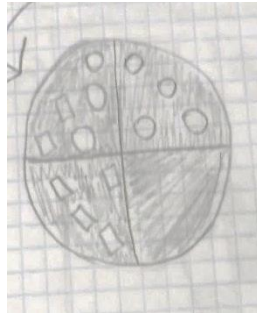
Towards the end of the intervention, more partnerships showed both participants working with area models equally. For example, in the beginning of the intervention, Sarah drew the area models her team thought they would use in their game. On the last day of the intervention, Sarah and her partner, Kala, were telling me their game was not going to be finished and they explained to me what they intended their game to do:

Kala: Like what, okay, shade in one half of the pizza.

Sarah: [Draws an example.] Like one half pepperoni.

Kala: Like this, and one half onion and leave one quarter cheese. [Helps with the drawing.]

As they talked, both girls worked together to represent the fraction scenario shown in figure 4.9.



*Figure 4.9: Sarah and Kala's pizza example.*

On the same day, Greg, Katherine, and Ian were finishing the digital images for their questions and critiquing them together:

Greg: [Looking at the image he created] Yeah, no, that's, that's not two thirds.

Ian: Yeah, it is.

Greg: Oh, yeah, it is.

Katherine: Well, I will finish this. I'm going to get one of those fraction circles that will help us.

In his interview, Greg mentioned this cooperation when asked why their game was special or unique:

Greg: It's unique because like we kind of thought of the fractions off the top of our head. Kind of designed some of the pictures on Google Images. Like that one that you actually have right there, Katelyn drew that out. We decided that we were going to draw the pictures out so we could make them unique.

Me: So you guys drew the pictures and came up with the problems yourself.

Greg: [nods]



Me: Uh, huh. [Switching to a different screen.] Who did that one?

Greg: Ian.

Me: Ian did that one? So, which one did you do?

Greg: I did the first one.

By working with area models using drawings or manipulatives to create questions for their games, nine participants developed their understanding of fractions.

**Theme 2: Talking about area models.** Four teams did not create area models on paper or use manipulatives, choosing instead to put the representations directly into their game, and yet still showed gains on the posttest. In these cases, the evidence of learning appeared in the transcripts, since these participants used talk to experience and reflect on fraction magnitude. For example, Brandy and Ariel, decided to interview each other on the recording device to ensure each understood what they intended for their game before they began making it:

Brandy: So, Ariel, how do you think this fraction game, called the *Fraction Machine*, is going to help the kids learn fractions?

Ariel: It's going to show them step by step how to do fractions. And it's going to, you know, like, it's going to help them.

Brandy: Basically, what it's going to do is it's going to, for example,  $\frac{3}{4}$ , and there's like a little pizza and it has 3 of them are gone and it's only 4 slices and there's one left, so things are going to be colored in and show them, you know what I mean? Did I explain that right? Is that right?

Also during the design phase, participants used talk to explain concepts they saw in the resources. In this example, Chris and Destini are looking at a textbook for ideas when Chris has a question:

- Chris: What the opposite of the numerator?
- Destini: Oh! It's on this picture, oh! [Points to an area model in the book.] It tells how many of those equal parts for the fraction stands for.
- Chris: How many equal parts there are?
- Destini: Yeah, look! [Points to picture.] Count them!
- Chris: Oh, ok.

Later in the intervention, when participants were putting the fractions in their games, they used talk to express difficulties and help their partner. For example, Zach and his partner, Keandra, used talk to help him understand how to represent an improper fraction with area models:

- Zach: How is it possible to do twenty over five shaded in? Twenty over five? That means there is... only five are there and twenty shaded in. How is that?"
- Keandra: You make more groups of five.
- Zach: You can do that?
- Keandra: Yeah, some fractions are more than one.
- Zach: So it would be like five and five and five until I can shade in twenty, right?
- Keandra: Yeah.

Similarly, Keith used talk to help his partner, Sarah, understand a subtraction problem by verbalizing a similar problem for her:

- Keith: Do you mean, what is three fourths minus three fourths, is that too hard?
- Sarah: Actually, I don't know.
- Keith: What is three fourths minus three fourths?

Sarah: I don't know. If it's too hard for me then it's too hard for them.

Keith: Three fourths minus, think, Sarah, you can have three dollars add four dollars, right?

Sarah: Yeah.

Keith: You got three dollars, you subtract three dollars, equals what?

Sarah: Zero.

Keith: Exactly. Three fourths minus three fourths?

Sarah: Well then, it's not that hard. It's just in a harder version.

In each of these cases, the participants did not include these fraction scenarios in their games. Instead, they included simpler problems so they could have more of their game completed before the end of the intervention.

Brandy and Ariel, who had the highest and second highest gains on the posttest, talked throughout the intervention but put very little on paper. Approximately half of the time, this talk was about the game they were making. As they were finishing their game, they used talk to resolve a disagreement they had about one of their problems (see figure 4.10):

Ariel: Ok, three-fifths is done.

Brandy: I don't think that's three-fifths.

Ariel: Sure it is, girl! It's got three shaded and five not!

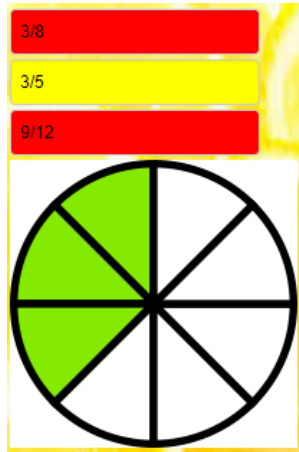
Brandy: But that don't mean three-fifths. That's like three-eighths or something.

Ariel: How you mean?

Brandy: Doesn't the bottom number have to be, like, the whole thing?

Ariel: Oh, yeah.

Brandy: But leave three-fifths. See if anyone else picks it.



*Figure 4.10: The problem, and resolution, Brandy and Ariel discussed.*

Brandy and Ariel, like the other teams in this section, used talk to experience and reflect upon area models to develop their understanding of fractions as they developed their games.

**Theme 3: Developing code for comparing fractions.** Justin and Daniel were the only coding team that did not use area models in their game, did not ask questions in their game, and yet completed their game during the intervention. Instead, their game idea was to display two fractions on stick figures and have the player “shoot” the larger fraction (see figure 4.11). Theirs was the only game, therefore, that developed code for working with fractions.

Justin appeared young for his age, liked to please his teacher, was accustomed to asking for help whenever faced with a new situation, and was diagnosed with a learning disability. Daniel was Justin’s opposite; he was loud, argumentative with authority, and spent most of the classes trying to distract other students. Daniel did not participate in the

project often, but when he did he provided key insights or ideas. It was up to Justin, however, to develop those ideas. For example, figure 4.11 shows the original game idea that Daniel drew on the first day while Justin watched.

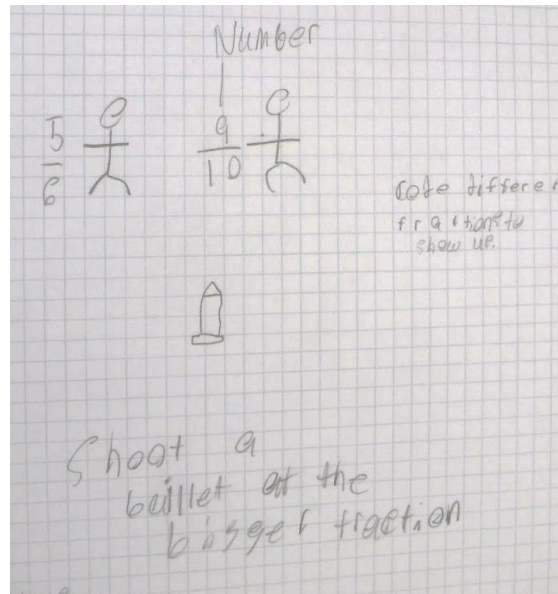


Figure 4.11: Original game design for Justin and Daniel

Afterwards, Daniel rarely participated, even when it was his day to code, and instead limited his contributions to approving or criticizing what Justin did. Thus, it was Justin who developed an understanding of fraction magnitude by creating and testing his code. As the following exchange from the second day of the intervention demonstrates, this responsibility was not one he accepted willingly:

- Teacher: You know what the tricky part it's going to be? Having your game figure out which fractions are bigger one so it knows whether it's right or wrong.
- Justin: How are we going to do that?
- Teacher: You going to have to starting thinking about that one. I'm not giving that one away.
- Justin: Oh, come on!

Justin chose to ignore his dilemma until everything else in the game was completed, such as choosing the images and having random fractions appear on the screen. Near the end of class on the seventh day of the intervention, however, the only thing he had left to develop was an algorithm for comparing the fractions so the game could tell the player if the selection was correct or not. Justin then called the teacher over for help.

Justin: So you said we had to have 2 fractions, like this. [Writes one-half and five-thirtieths on paper.]

Teacher: Right, so how do you know which one is bigger?

Justin: Well.

Daniel: You look at it.

Justin: You look at it.

Teacher: Well, which one is bigger?

Justin: That one? [indicates one-half]

Teacher: Why?

Daniel: Or five-thirty.

Justin: Yeah. 'Cause like the numbers are bigger in the other one, so like this one [indicates five-thirtieths] is bigger than the numbers are there [indicates one-half].

Teacher: Is that always the case with fractions?

Justin: No.

Teacher: So, how do I know which one is a bigger?

Justin: I don't know.

At this point, Daniel went to distract another team while Justin tried to find out how to compare fractions from the Internet. Figure 4.12 shows what Justin found and copied before the bell rang.

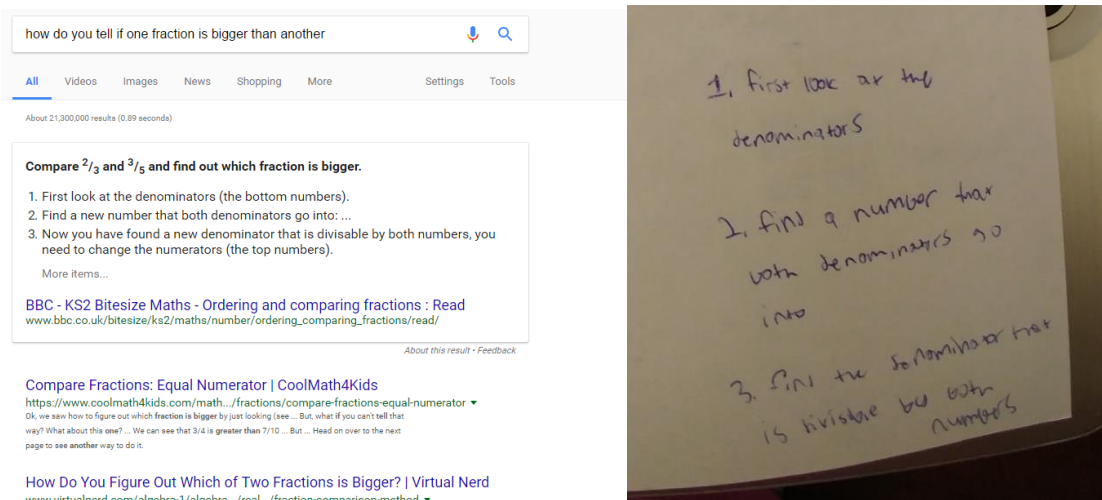


Figure 4.12: The instructions Justin found and copied for comparing fractions.

Justin was absent the next time that class met, so it was five calendar days before he revisited his notes, the ninth day of the intervention. Finding the notes confusing, he asked Daniel for help.

Justin: I had wrote down the steps that were on Google, how do you make the fraction... Is that the only way to do this?

Daniel: No, you can do another decimal. Change it to a decimal.

Justin: How do you change a fraction to a decimal?

Daniel: You divide them. Bottom divided by top.

Justin proceeded to code Daniel's suggestion. When he tested it, however, he called me over and was visibly agitated. Daniel was outside the classroom at this time.

Justin: Ms. J, it's messed up!

Me: Tell me.

Justin: Well, it was working but then it says this is wrong and it ain't!  
[Shows me the screen. It has  $4/9$  on the left and  $23/1$  on the right. Justin had selected  $23/1$  as the largest, which the game marked as wrong.]

Me: How do you know the program's wrong and not you?

Justin: 'Cause this [points to the fraction on the right] is twenty-three!

Me: How'd you know that was twenty-three?

Justin: 'Cause it's over one.

Me: So if it's not you, it must be your code.

Justin: [Indicates at code.] Yeah, but where?

Me: Well, where'd you deal with the fractions?

Justin: Right here. [Points to code showing the division (see figure 4.13)]

Me: So try doing exactly what your code says in this line. [Points to same line of code.] Use a calculator with the same fractions you have and see what happens.

Justin: [Calculates one divided by twenty-three.] Wait, that ain't right.  
[Calculates twenty-three divided by one.] That's right.

Me: What did you do?

Justin: I did twenty-three divided by one.

Me: Is that what your code did?

Justin: No. Should I change it?

Me: Probably.



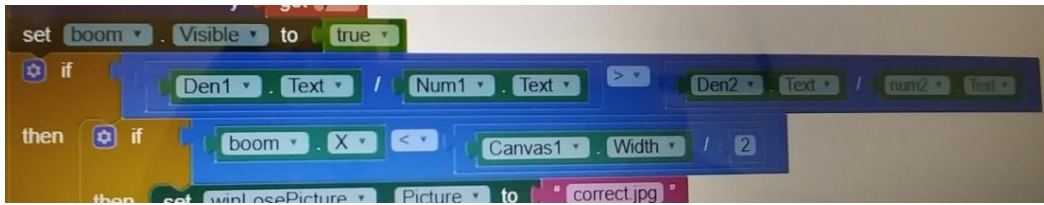


Figure 4.13: Justin's code with the erroneous division expressions.

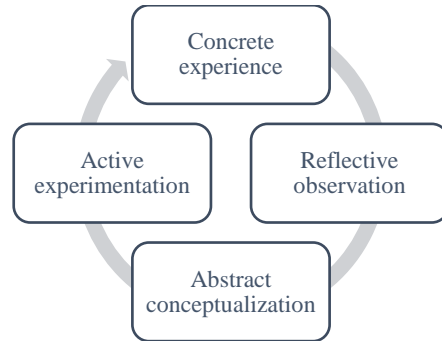
Justin corrected his code and was very pleased to have a working game. In the post-interview, I asked Justin about the directions he found online, which were for finding a common denominator. He said the directions looked familiar because of previous math classes, but he did not remember what the method was called and he said he would not have thought of it on his own. He also said he decided to use Daniel's suggestion of turning the fractions into decimals because he felt it would be easier, although, in the interview, he said he did not know how to do that before he made his game. When asked what he felt he learned during the project, the first thing he said was, "I learned how to be better with fractions."

### **How Each Theme Connects to Experiential Learning Theory**

Participants interacted with fractions while creating their games in three ways: (a) working with area models, (b) talking about area models, and (c) developing code. These methods map to the four phases of Kolb's (1984) experiential learning theory (see figure 4.15): concrete experience, reflective observation, abstract conceptualization, and active experimentation.

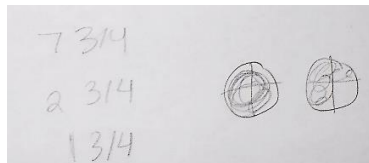
During concrete experience, a learner engages in an activity. Then the learner reflects on that activity during reflective observation. The learner gains knowledge from the experience during the abstract conceptualization stage. The learner then acts on the knowledge through active experimentation. This section will connect the three themes

from the findings of this study with the experiential learning cycle by mapping each theme to the cycle and describing the evidence from the findings for this mapping.



*Figure 4.15: The experiential learning cycle.*

**Theme 1: Working with area models.** Participants who worked with area models entered the learning cycle at “concrete experience” because they were creating questions on paper before they added them to their games. In this study, participants demonstrated they were in this phase of the cycle by sketching area models, like the example seen in figure 4.16.



*Figure 4.16: A student's sketch of an area model question.*

The participants then shared their questions with their partners and received feedback. They entered the “reflective observation” phase by considering the feedback as it related to their area models and the “abstract conceptualization” phase as they accepted or rejected the feedback. These phases are not easily observed, as they typically occur during silent thought (Kolb, 1984), but may be inferred by a longer than usual pause in the conversation followed by the student entering the “active experimentation” phase

(Matsuo, 2015), where he or she applied the acceptance or rejection of the partner's feedback to the original area model. The full cycle as it applies to participants who worked with area models may be seen in figure 4.17.

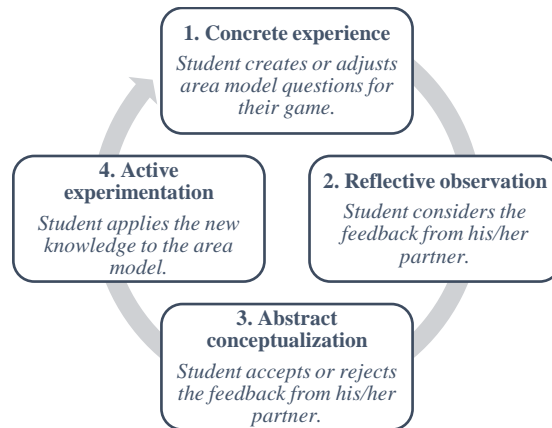


Figure 4.17: Experiential learning cycle for working with area models.

Table 4.1

*Working with area models data mapped to the experiential learning cycle*

<u>Phase</u>	<u>Greg and Katherine</u>	<u>Matthew and Rhianna</u>	<u>Walt and John</u>
Concrete experience	[Greg has drawn questions using area models and is sharing them with Katherine.] Greg: Where's, what graph has 1/4 shaded?	Matthew: Draw a graph that's two fourths shaded in. [Rhianna draws the area model.] Matthew: Now draw one for three fourths. [Rhianna draws the area model.]	Walt: What is shaded in this graph?
Reflective observation	Katherine: I just don't know. Both graphs have 1/4 shaded. Greg: No, only one does. Katherine: No, both do. Count!	Matthew: What's the difference?	John: I thought we did that. Walt: No, it's different.

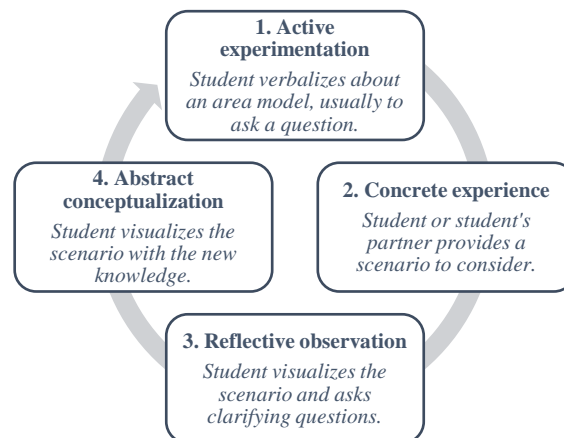
Abstract conceptualization	[not observable]	[not observable]	[not observable]
Active experimentation	[Greg counts on his area models.] Greg: Oh, right. Ok, this one has two answers then.	Rhianna: Oh, this one has like one less.	John: Oh, okay. This one has two-fourths shaded.

Table 4.1 shows how participants working with area models map to the experiential learning cycle. In each case, questions posed by their partners allowed participants to enter the reflective observation phase because the questions challenged their thinking (Matsuo, 2015). The successful resolution of those challenges suggests that abstract conceptualization occurred because the participants each revised their thinking (Matsuo, 2015).

**Theme 2: Talking about area models.** When a participant talked about area models instead of working with the models, she or he was observed to enter the learning cycle at the “active experimentation” phase. The participant had a prior understanding about fraction magnitude that she or he was trying to articulate, usually to ask a question; this verbalization demonstrated that the participant was acting on hers or his knowledge. The “concrete experience” phase then happened when the participant or, more often, the participant’s partner provided a scenario to consider. The participant then engaged in “reflective observation,” which, unlike the participants who worked with area models, was easier to identify because the participant usually asked clarifying questions of their partner concerning the scenario. “Abstract conceptualization” occurred silently, but could be inferred because the participant would enter another “active experimentation” phase

by applying hers or his new understanding to the scenario given by hers or his partner. The full cycle as it applies to participants who talked about area models may be seen in figure 4.18.

Table 4.2 shows how the conversations participants in this study had about area models map to the experiential learning cycle. Zach’s and Chris’s clarifying questions show engagement in reflective observation because each is challenging the visualization his partner suggested. Brandy, however, engages in reflective observation by asking her partner for confirmation. Brandy and Ariel exit the experiential learning cycle at this point, making it unclear if they engaged in abstract conceptualization, but the others re-enter the active experimentation phase in their dialogs by applying new knowledge, suggesting that abstract conceptualization occurred to revise their thinking (Matsuo, 2015).



*Figure 4.18: Experiential learning cycle for talking about area models.*

Table 4.2

*Talking about area models data mapped to the experiential learning cycle*

<u>Phase</u>	<u>Zach and Keandra</u>	<u>Brandy and Ariel</u>	<u>Destini and Chris</u>
Active experimentation	Zach: How is it possible to do twenty over five shaded in? Twenty over five? That means there is... only five are there and twenty shaded in. How is that?"	Brandy: So, Ariel, how do you think this fraction game, called the <i>Fraction Machine</i> , is going to help the kids learn fractions?	Chris: What the opposite of the numerator?
Concrete experience	Keandra: You make more groups of five.	Ariel: It's going to show them step by step how to do fractions. And it's going to, you know, like, it's going to help them. Brandy: Basically, what it's going to do is it's going to, for example, $3/4$ , and there's like a little pizza and it has 3 of them are gone and it's only 4 slices and there's one left, so things are going to be colored in and show them, you know what I mean?	Destini: Oh! It's on this picture, oh! [Points to an area model in the book.] It tells how many of those equal parts for the fraction stands for.
Reflective observation	Zach: You can do that? Keandra: Yeah, some fractions are more than one.	Brandy: Did I explain that right? Is that right?	Chris: How many equal parts there are?
Abstract conceptualization	[not observable]	[no evidence]	[not observable]

Active experimentation

Zach: So it would be like five and five and five until I can shade in twenty, right?

Destini: Yeah, look! [Points to picture.] Count them!  
Chris: Oh, ok.

**Theme 3: Developing code for comparing fractions.** Seymour Papert and Wallace Feurzeig described programming for learning using terms similar to how Kolb (1984) described the experiential learning cycle when they said, “Program descriptions are open to reflection and discussion, and procedures that fail can be examined, analyzed, and repaired” (Feurzeig & Papert, 2011, p. 488). Although only one participant in this study, Justin, interacted with fractions in the code, his experience with his code when the game produced an error followed both the experiential learning cycle and what Feurzeig and Papert (2011) wrote. As figure 4.19 demonstrates, this cycle begins at the “concrete experience” phase when the program responds incorrectly to input then continues through the other phases as the student attempts to find, understand, and repair the error in the code.

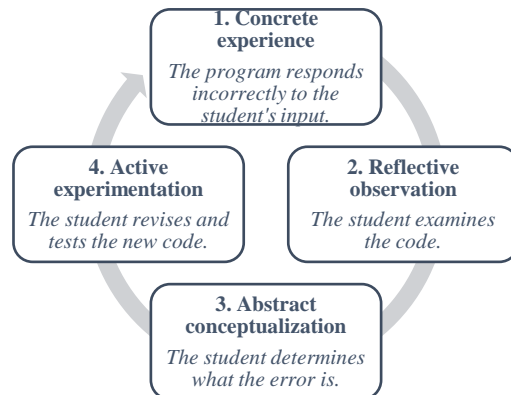


Figure 4.19: Experiential learning cycle for developing code.

The following conversation between Justin and myself the error in his game was discovered shows how he moved from one phase to the next in the experiential learning

cycle. Because Justin's partner had deserted him at this point and because this was the first Justin had encountered a coding error, I used leading questions to help him through the reflective observation phase so he could transfer from what he could do at that time to what he wanted to accomplish, which is needed for experiential learning to be effective (Burns & Gentry, 1998).

Concrete experience:

Justin: Ms. J, it's messed up!

Me: Tell me.

Justin: Well, it was working but then it says this is wrong and it ain't!

[Justin shows me the screen. It has  $4/9$  on the left and  $23/1$  on the right. Justin had selected  $23/1$  as the largest, which the game marked as wrong.]

Reflective observation:

Me: How do you know the program's wrong and not you?

Justin: 'Cause this [points to the fraction on the right] is twenty-three!

Me: How'd you know that was twenty-three?

Justin: 'Cause it's over one.

Me: So if it's not you, it must be your code.

Justin: [Indicates at code.] Yeah, but where?

Me: Well, where'd you deal with the fractions?

Justin: Right here. [Points to code showing the division.]

Me: So try doing exactly what your code says in this line. [Points to same line of code.] Use a calculator with the same fractions you have and see what happens.



Abstract conceptualization:

Justin: [Calculates one divided by twenty-three.] Wait, that ain't right.  
[Calculates twenty-three divided by one.] That's right.

Me: What did you do?

Justin: I did twenty-three divided by one.

Me: Is that what your code did?

Justin: No. Should I change it?

Me: Probably.

Active experimentation:

[Justin changes and tests his code.]

Within the “reflective observation” phase, Justin engages in a smaller experiential learning cycle similar to that experienced by participants who talked about area models. My leading question, “How do you know the program’s wrong and not you?” provided him with a scenario which he considered then challenged with an explanation. Overall, however, Justin’s experience fits the experiential learning cycle because fixing the mathematical algorithm in his code allowed him to re-examine his initial math problem of comparing fractions (Robins, Rountree, & Rountree, 2003)..

**RQ3: What challenges, other than with fractions, do low-achieving secondary math students experience in designing and developing games using *App Inventor*?**

In addition to understanding how students’ understanding of fraction magnitude developed, this study also examined the challenges they had when working with an NPE during a math intervention. In this study, challenges were defined as difficulties affecting all members of a group and preventing the group from progressing with their work

independently or later creating difficulties that impeded independent progress.

Understanding these challenges may help identify and explain any factors that may have limited the students' development of fraction understanding (Allsopp, McHatton, & Farmer, 2010). Thus, this section will present the findings of the third research question: What challenges, other than with fractions, do low-achieving secondary math students experience in designing and developing games using *App Inventor*? For this research questions, data from all fifteen groups (thirty-two participants) will be used because all participants experienced challenges when making their games, with some challenges affecting most or all participants.

The available literature suggested some of challenges the participants in this study might face when creating their games and how to assist them, so this section will begin with a brief summary of those challenges that presented themselves and how the research-supported strategies helped. The section will then present findings showing that participants encountered additional challenges coding as well as two challenges that are not exclusive to computer science activities: collaboration and learned helplessness. The section concludes by summarizing these challenges and identifying the challenges common to the groups that did not complete their games during the intervention. The challenges, support offered, number of participants affected, average number of times the challenge presented per group, and the data sources that revealed the challenge are listed in appendix N.

## **Challenges Identified by Prior Research**

As stated in the literature review, previous studies have found that students with learning disabilities have specific challenges when learning to code. Three studies (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Santi & Baccaglini-Frank, 2015) identified algorithm development, debugging, and transferring learning from one task to another as difficult for their participants; Ratcliff and Anderson (2011) also found that participants found working with graphics, especially angles in graphics, challenging. Each of these challenges appeared in this study. Although these studies specified students with learning disabilities as having these challenges, this study found that they also affected participants without identified learning disabilities.

**Algorithm development.** Algorithms are, according to Wing (2008), “an abstraction of a step-by-step procedure for taking input and producing some desired output” (p. 3718). The data from the final interviews suggested that many of the participants in this study found algorithm development challenging; analysis of the audio recordings of participants and student work revealed this challenge affected eleven of the fifteen groups. In the interviews, each of the nine participants said that making their games was difficult. When asked if that difficulty was because they could not “picture what to do in your mind” or if they did not know what code to use, seven replied that they could not even picture what they needed to do. Prior research has shown that algorithm development is difficult for new coders and suggests helping students by helping them plan on paper before coding (Chang, Thorpe, & Lubke, 1984; Santi & Baccaglini-Frank,

2015) or encouraging them to look at other code and ask questions (Ratcliff & Anderson, 2011).

Because most of the algorithms participants were trying to develop were addressed, at least partially, in the tutorials available on the *App Inventor* website, participants were encouraged to examine the tutorials and ask questions. These thirty-one tutorials provide descriptions for what the example app does, step-by-step instructions for creating the user-interface and code, and descriptions for how each of the components in the app functions. This support created another challenge for the participants: Choosing or adjusting an appropriate algorithm from the tutorials. Keith and Sarah, for example, were trying to have a cannon shoot a ball along a path, which was similar to the way objects move in the tutorial that recreated a classic arcade game, *Space Invaders*, but were trying to use the code found in a tutorial that recreated the game *Mole Mash* because, Keith explained, “It says ‘MoveTo’ and we want the ball to move.” They did not recognize that the algorithm in *Mole Mash* moved objects differently than the way they wanted their ball to move. Destini and Chris were also trying to move objects along the screen to give the illusion of lines moving along a road (see figure 4.20). They implemented an algorithm from a tutorial to make the lines move but found that all of the lines stopped at the top of the screen. When the researcher gave them the suggestion to move the lines to the bottom when they reach the top, they then had difficulties developing that algorithm even on paper, which involved using a conditional (“if” statement) and detecting edges on the screen. Choosing or adjusting appropriate algorithms from the tutorials was a challenge for nine of the fifteen groups; developing

algorithms on paper was a challenge for two groups. In each case, these groups could create or modify an algorithm to achieve at least partial functionality in their code after receiving help. Four groups did not have difficulties with algorithm development.



*Figure 4.20: The lines Destini and Chris were trying to move.*

**Debugging.** The debugging process is a cycle of identifying the error, finding the error in the code, changing the code to hopefully fix the error, then testing to determine if the error is gone (Rouse, 2016). Debugging is a challenge for new coders (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Santi & Baccaglini-Frank, 2015); research suggests encouraging students to “act out” the code on paper by writing down what happens with each line of code (Chang, Thorpe, & Lubke, 1984). Analysis of the observational data, specifically the audio recordings and field notes, revealed that identifying the error was the first challenge some participants faced, once even to the point of recognizing that the code had an error:

Travone: Did I do it right?

Teacher: Is it working the way you want it?

Travone: Nope.

The audio recordings, code, and student work logs revealed that identifying the error was a common challenge for the groups who were trying to make objects move on

the screen. As the following exchange between Destini and Chris demonstrates, participants found it challenging to recognize what the object was actually doing when they observed it moving incorrectly:

Destini: Still not working.

Chris: Kind of worked, and kind stopped working. It started moving then disappeared, so I did see something moving then disappear. Did you see that, the lines split weird then went back where they belong?

Destini: Kind of.

Recognizing that the error was a coding error was also a challenge for a few groups. For example:

Katherine: Just like the picture. It's not doing anything.

Greg: That means someone set it wrong in our coding.

Katherine: But we didn't do anything.

Greg: Is the code, we got it.

Katherine: That's not the coding page, the coding page is where you would do the blocks.

Greg: No, this is the same thing. Yes, so, I knew we made a mistake and...

Katherine: Ms. K, I'm confused. I don't know how to code, it's hard.

Greg: Yeah, this is hard.

Once an error was identified, specifying what the actual error was so it could be found in the code was the next challenge for participants. Destini and Chris, for example,

needed guiding questions to recognize that one of their errors was that the object moved in the wrong direction:

- Teacher: Okay, did you notice which direction it moved?
- Chris: It went that way, it was like it was moving backwards.
- Teacher: So, it moved to the left? To the right?
- Chris: It went right.
- Destini: Oh! But we want to move up, right?

For eight groups, being specific about the error was enough for them to identify what part of their code contained the error. Fixing the error remained a challenge, but the challenge was reduced after the teacher or researcher taught them a few debugging skills, such as using trial and error to determine what values to use or getting one object at a time to work correctly, in addition to having them “act out” the code on paper (Chang, Thorpe, & Lubke, 1984). Debugging remained a source of frustration for participants, however, as Tyrone, Clayton, and Ken expressed in their log one day (figure 4.21) after trying to identify the error that prevented their ball from moving when “flung” by the player; with assistance from the teacher the next session, they recognized that they had mistakenly set the component’s speed to zero, which prevented movement. Only three participants could debug their code without assistance, two of whom had prior experience using *App Inventor*.

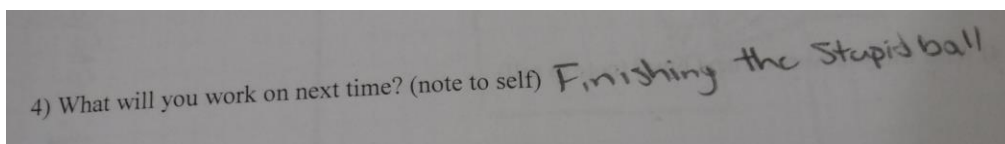


Figure 4.21: Expressing frustration when debugging.

**Transferring learning.** Transfer is an individual’s ability to apply prior knowledge, skills, and strategies to new scenarios (Fuchs et al., 2003). Recognizing that the current problem is related to a previously solved problem is one requirement for transfer to occur successfully (Cooper & Sweller, 1987), and has been identified as a challenge for students with learning disabilities when coding (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Santi & Baccaglini-Frank, 2015). The audio recordings and field notes revealed that ten groups did not identify that they had solved a similar problem without prompting by the teacher or researcher. For example, Keandra and Zach’s game included having the player “shoot” a basketball in a manner similar to the functionality found in the *Ball Bounce* tutorial that they had completed in the first week. As the following exchange demonstrates, they did not recognize the similarities between these tasks on their own:

Keandra:        Okay, so we got a basketball; how do we put it on there?

Me:                Seems to me that we made a game with a ball on it once, right?

Zach:             What, it’s the same?

Keandra:        Oh, yeah! We can use that!

In Brandy and Ariel’s group, they asked the teacher for help because they did not realize that the code they already created for one button would be similar to the code they needed for another button:

Brandy:        I need help. I just need to get... so this button, and then I do...

Teacher:        Honey, go look at the other button you created. 'Cause aren't you just doing the same thing over again?



Brandy: Yeah.

Teacher: Look at the other button.

Ariel: What other button?

Teacher: You have another button already done. You're doing the same thing.

Brandy: Oh, right.

Not recognizing the similarities between a previously solved coding task and the current one occurred in ten of the groups. In seven groups, participants began identifying similarities and transferring knowledge after prompting by the teacher or researcher. For the other three groups, such as Brandy and Ariel, the teacher or researcher needed to identify the similarities explicitly before the participants recognized how to transfer that previous coding task to the current problem.

**Working with angles.** The “heading” property, which gives the sprite component its direction to move, uses angle measurements (in degrees) for its parameter. Ratcliff and Anderson (2011) found that students struggled with using angles when coding; analysis of the audio recordings and daily work revealed that participants in this study also failed to recognize when they needed to use angles. My conversation with Matthew when his partner was absent demonstrates how not recognizing this parameter as an angle and working with angle measurements both presented challenges:

Matthew: Okay I've got something there all right and we put in zero. I'm going to click left, it's moving, but it moved right, didn't it?

Me: Well, at least we know how to move things to the right. Okay, and you tried using a negative number, too. What did you pick?

Matthew: Minus three.

Me: Did that change anything?

Matthew: No.

Me: Okay, but in the tutorial, [a sprite is] moving down. What did they use?

Matthew: Minus ninety.

Me: Try that.

[Matthew changes the parameter and tests the code. The sprite moves down.]

Me: So negative ninety moved it down. What about positive ninety?

Matthew: That moved it up.

Me: Good, we've already figured out 3 of your buttons not just the button we are working on, right? So, you're going to want to write this down.

Matthew: So, zero is right, ninety moved it up, minus ninety down.

Me: Where do we see zeros and nineties in math?

Matthew: On triangles.

Me: On triangles. Why? Because what is it describing?

Matthew: The angles.

Me: The angles, and how do we measure angles? Which tools do we use?

Matthew: Protractor

Me: The protractor, right. So, you know what? Maybe you want to take a look at a picture of a protractor to see what numbers you should put in for left.

Matthew needed a little assistance afterwards on how to read a protractor but then could enter the correct parameters for his game's directional buttons, as figure 4.22

demonstrates. The other four groups with this difficulty had similar conversations with the teacher, researcher, or Matthew to activate their prior knowledge of angles and to understand how they applied to the “heading” property in the code.

The image shows five blocks of App Inventor code. Each block starts with a 'when' event (Button\_click) and a 'do' block containing 'set' actions for ImageSprite1. The 'Heading' property is set to true, 0, 180, 90, and -90 degrees in the five blocks respectively. The 'Speed' property is set to 6 in four of the blocks. The first block also sets the 'Visible' property to true.

```
when Button_avatar .Click
do set ImageSprite1 . Visible to true

when Button_Right .Click
do set ImageSprite1 . Heading to 0
set ImageSprite1 . Speed to 6

when Button_up .Click
do set ImageSprite1 . Heading to 90
set ImageSprite1 . Speed to 6

when Button_left .Click
do set ImageSprite1 . Heading to 180
set ImageSprite1 . Speed to 6

when Buttondown .Click
do set ImageSprite1 . Heading to -90
set ImageSprite1 . Speed to 6
```

Figure 4.22: Matthew’s code with the correct parameters for the “Heading” property.

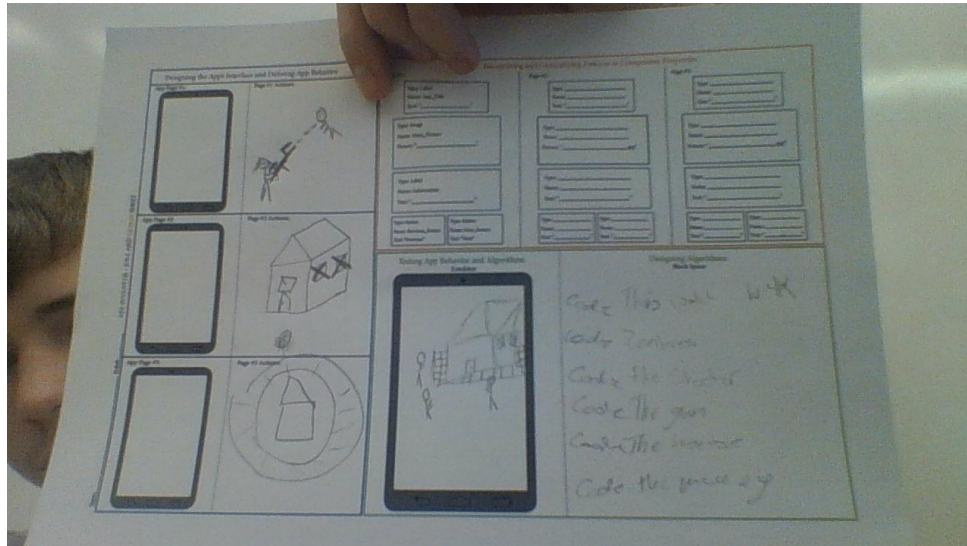
### Challenges Specific to Coding

Although all the participants in the study had participated in the “Hour of Code” event (Code.org, 2017) two months prior to the intervention, only three participants had additional coding experience, two of whom with *App Inventor*. The two participants with prior *App Inventor* experience worked together and did not have challenges coding that they were unable to resolve independently; all the remaining participants had several challenges. In addition to the challenges identified in the literature and discussed earlier this chapter, participants had challenges concerning their game designs, decomposing their game designs into components to code, coding concepts and skills, limitations of the *App Inventor* environment, and challenges with some of the vocabulary and angle use in the coding blocks.

**Game design issues.** The eight games that combined quiz-like questions with other game elements, such as shooting a basketball or running a kitchen, were not completed during the intervention. Analysis of the artifacts, specifically the design sketches and coding plans, revealed that these groups designed games of a greater coding complexity than did the other groups; audio recordings, field notes, student work, and incomplete final products showed that this issue was a continuous challenge for participants throughout the intervention. For six of these groups, plus one group that later changed to a simple quiz game, the games they designed had several components on screen, many of which moved. In figure 4.23, for example, the object of the game was to survive a zombie attack by shooting them and building defenses; materials to do so could be purchased with currency earned from answering questions correctly. As the participants identified on their design sheet, this game would require several components to be coded: zombies, shooters, guns, building materials for the house, money, and the fraction questions.

Additionally, these components have little in common with each other, so coding knowledge gained from one component may not transfer to another. In comparison, the simple quiz games typically had four components on each screen, none of which moved, and each screen was similarly designed so the knowledge gained from coding one screen directly transferred to the next. Other game designs with several components on the screen included running a kitchen, building a hotel, and car racing, as well as two other shooting games. The participants, however, were unaware of the complexity of their designs. As Katherine said to her partners when they completed their initial design, “It’s

going to be easier than I thought it was going to be.” Katherine was a member of the group that completely changed their design to a simple quiz game later.



*Figure 4.23: A game with several moving components.*

Two of the eight games that were not completed during the ten sessions had similar functionality: Players could shoot a basketball when they answered a fraction question correctly. These games had only a few more components than the simple quiz games, but these components had more complex functionality than the quiz games’ components had, even those quiz games that included animation. Specifically, the basketball games needed code recognizing when the player is allowed to “shoot” the ball, how the player “shoots” the ball, and a win condition. The complexity of these design elements in addition to coding the fraction questions resulted in both games being incomplete at the end of the intervention, although they each had most of the functionality completed.

**Decomposition.** Decomposition involves taking a complex task, separating it into smaller tasks, and organizing those tasks by the order each should be completed (Wing,

2008). Problem decomposition in computer science also includes the defining of objects and methods (Barr & Stephenson, 2011, p. 117). Decomposition was identified as a challenge when the data showed participants were unable to independently decompose their game designs into a list of components on their coding plans, created incomplete coding plans and were unable to identify what was missing without assistance, or needed to create something that is normally considered to be one object but actually requires three components to replicate on the screen, such as a fraction in  $a/b$  form.

Nine groups had challenges with decomposition, which was first identified in the audio recordings from when participants designed their games and the first drafts of the coding plans participants created. The audio recordings from when participants started creating their games and the interview data confirmed that these participants were not able to identify one task to begin with even with their coding plans unless the teacher or researcher assisted. As one participant said, “It was just really confusing just how to start off, like where do I get these pictures, how do I code it? So it was kind of sort of overwhelming with all the blocks and all the, especially for my first time not knowing how to code.” That feeling of being overwhelmed was also articulated when one team asked the teacher for help:

Travone: I don't understand really how we are going to be able to do this.

Teacher: Okay, like what? What's one thing?

Travone: Put the guns in there, making the person, just staying around him.

Cary: Being able to move.

Travone and Cary were unable to answer with “one thing” as the teacher requested

because, as they later said, “There were so many things we had to do!” Guiding questions from the teacher or myself helped participants to identify the individual objects in their games, however, as the following exchange demonstrates:

Teacher: Okay, so settings screen, okay, player choses what level. So, now I'm in the game, what I'm I seeing on the screen? Give me one thing I'm seeing on the screen.

Destini: A car.

Teacher: A car. What does the car do?

Destini: Sitting there.

Chris: The car and then you have like the gears.

Teacher: Okay, so what's the car going to do?

Destini: It's going to go.

Teacher: Okay, so if I could get my question right, it goes, if I don't it just sits there looking pretty?

Destini: Like it starts off, it goes 5 miles per hour, if you get it right it goes like the 7, you get it right it goes to 15. If you get it wrong, you slower.

Chris: You like go back.

Teacher: Oh! Okay, so, really what's changing is not the car itself, from the player perspective, but the speed of the car. So, what shows that?

Destini: There's a little speedometer there.

In later sessions, three groups encountered additional challenges with decomposition when they were trying to make a random fraction appear on the screen. In *App Inventor*, they discovered the “random fraction” function displayed values in decimal form, but they were trying to create a fraction in  $a/b$  form. In each case, these groups

needed help recognizing that they would have to use three separate components to represent the numerator, denominator, and fraction bar. Guiding questions from the teacher or researcher resolved this challenge.

**Coding concepts and skills.** Coding was new to all but three of the participants. As one participant explained in the post-interview, “I never really knew how to like, what a code was. I always thought like a game, you didn't have to code it. Now I know there's stuff behind it.” Student work, audio recordings, and field notes revealed that eight groups encountered challenges with one or more of the following coding concepts or skills: the relationship between a component and its code, working with event handlers, choosing the correct component, and naming components meaningfully. For each challenge, explicit instruction given once on the concept or skill resolved the issue, which was not experienced by that group again during the intervention.

*The relationship between a component and its code.* After following two of the tutorials as an introduction to coding, three of the groups did not understand the relationship between the visual component and the code making it function until it was explicitly told to them. In some cases, the participants tried coding a component they had not created:

Teacher:       Where is the fraction? There aren't fractions here.

Ken:            What do you mean?

Teacher:       You've got to get it on the screen before you can make code for it.

In other cases, participants created the component but did not assign any functionality to it:

Chris:          I thought that they were going to like move?



Teacher: But how are they going to move? You need code to make them move.

In each case, explaining the relationship between a component and its code resolved the challenge.

**Working with event handlers.** Another coding challenge that appeared in two of the groups was in understanding how to code for event handlers. Event handlers in *App Inventor* are “when” statements that contain code to follow in response to certain input, such as pressing a button (MIT, 2017). Each event handler may appear only once in the code, even when multiple actions occur in response to the event. As figure 4.24 shows, participants tried using event handlers multiple times in their code to distinguish between the different actions that were to occur; again, explicit instruction given once resolved this challenge.

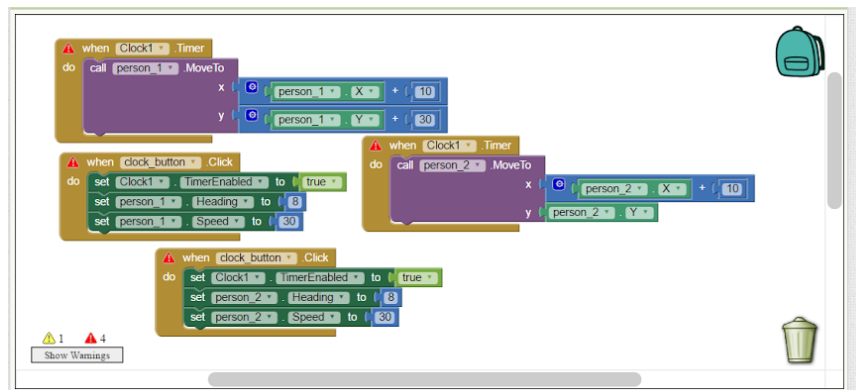


Figure 4.24: Incorrect use of event handlers.

**Choosing the correct component.** Participants also found distinguishing between the various components that can contain images challenging. Several components in *App Inventor* can contain a static image, including one called “Image,” but if the image will move, like a spaceship traveling across the screen, the coder needs to use two

components, a “sprite” for the moving image and a “canvas” to contain the sprite. Both tutorials participants completed in the beginning used the canvas component and one used the sprite component, but participants still had difficulties recognizing when to use which component for images, as the following exchange demonstrates:

Me: Okay, now you are... that’s an actual image [component]. That thing is going to move, right?

Jayla: Yes.

Me: You know what you might want to do? Instead of using an image, use a sprite.

Alexis: Yeah, I saw that in the animations.

Seven of the groups had difficulties recognizing which component to use for images, although most of them could use the image components after being directed to the correct type or a tutorial using the correct type. One group, however, changed their game design to a simple quiz game because, as Greg said in the post-interview, “We couldn't figure out how to get a block [sprite] to move.”

*Naming components meaningfully.* Eight of the groups had multiple instances of the same component on the screen, such as multiple buttons or multiple sprites. Three of these groups did not rename their components and so found keeping track of which was to do what challenging:

Ian: So, then where do you want me to go first?

Katherine: Do the room button, if you can find it on the blocks.

Ian: The button, all the blocks, all the buttons have numbers beside them. I don’t know which one is it.

One of the groups did rename their components, but not in a meaningful way:

Sarah: We named the buttons.

Brian: But I don't know which one to use.

Keith: We named them already: Brian, Sarah, Keith.

Although renaming components was addressed before the intervention, it needed reinforcing with these four groups.

**Limitations of *App Inventor*.** *App Inventor* allows users to create apps of varying complexity (MIT, 2017). Although it allows more functionality in the apps one can create than other NPEs do, the participants in this study still found three limitations that challenged their ability to create the games they designed: allowing collaboration on a project, finding relevant tutorials, and allowing dynamic memory allocation. Each of these challenges required that the researcher provide participants with instructions or sample code to bypass these limitations.

***Allowing students to collaborate on a project.*** The first limitation of *App Inventor* participants encountered was that it did not easily allow collaboration on a single project. Projects in *App Inventor* may only belong to one email address. Prior to beginning this study, the researcher and the teacher determined that participants would have to download their projects then upload them to the course's cloud service to allow pair programming to occur and to compensate for when a partner was absent. Creating new email addresses for this project was considered but rejected as a possible security issue. Directions for how to share the projects via the cloud service were given to each group in their binders (see appendix O) and guided instruction was provided at the

beginning of the intervention to ensure all students could follow them. However, this “work-around” created challenges for the participants that persisted throughout the study, especially downloading the project from the cloud service then importing it into *App Inventor*, as the following exchange demonstrates:

- Katherine: How did you get it, do the same thing right here on Greg’s computer?
- Greg: Yeah, I don’t even know how we got it.
- Ian: So all I had to do is go to ‘projects.’
- Greg: Projects, we already have that.
- Ian: Then go to ‘import projects from my computer.’
- Greg: Yeah, I already did that.
- Ian: Even though, sometimes it takes, well for me it took a couple of tries but eventually went in.

Difficulties with this process included following the sequence of steps and renaming files that had special characters added during download. In many cases, participants simply uploaded their projects at the end of class but then swapped computers, rather than downloading and importing, the next day to allow the other partner to code. As the audio recordings and field notes revealed, this challenge affected every group except the one participant who worked without a partner.

***Finding relevant tutorials.*** The second limitation of *App Inventor*, which audio recordings and the game designs showed four groups encountered, was that there were not tutorials demonstrating functionality that they wanted in their games. Participants preferred using the tutorials to learn from, rather than the apps other users had uploaded,

because, as Keandra explained in her interview, “Those other games just give you the code and don’t tell you what it does.” These game designs, as shown in figure 4.25, had functionality that were not addressed in any of the tutorials: looping an image on the screen (two games) and building (two games). For each group, the researcher created example code demonstrating similar functionality (see appendix P) then explained how the code worked to them.

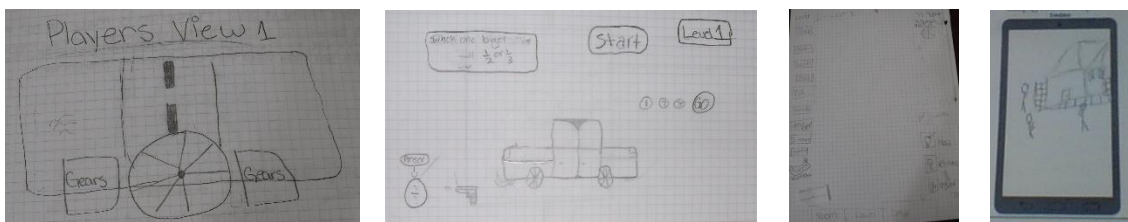


Figure 4.25: Four game designs with features not addressed in the tutorials.

**Allowing dynamic memory allocation.** The final limitation participants encountered with *App Inventor* was that it does not allow dynamic allocation of memory. Two groups designed building games where part of the functionality was to create an object, such as a room or a brick, then place it where the user wants it on the screen. In a standard object-oriented programming environment, the coder would create an abstraction or template for the desired object then call for instances of that object to be created as needed; *App inventor* does not have this or any similar functionality (Italo, 2017). Although the participants did not recognize that they were trying to allocate memory dynamically, the audio recordings and student work revealed that they were trying to create code that would make an object when a button was clicked. Sample code showing how to simulate this functionality (see appendix P) was given and explained to

the groups; after trying to implement the code in their own games, one group chose to design a simpler game.

**Difficulties with vocabulary.** The vocabulary terms used in the coding blocks challenged seven of the groups, with audio recordings, field notes, and student work showing two terms being especially difficult for them. Although twenty-two participants had difficulties with a vocabulary term at some point during the intervention, it was identified as a challenge only when all members of the group did not understand or misunderstood a term and could not progress until the teacher or researcher intervened. In the other instances where vocabulary was a difficulty, another member of the group explained the term; since progress was not impeded, these instances were not identified as a challenge.

The first term that posed a noticeable challenge, “initialize,” was needed by three groups to have something happen when a screen first appears. In each case, the groups knew they needed an event listed under the screen component (see figure 4.26), but either asked for assistance when they saw the choices or tried using some of the other events because, as one student explained, “I knew some of the words in them.” For example, Destini was trying to have lines move along the screen as soon as the game began, but did not know which event to use:

- Me:                   Okay, so we want them to start moving right away, correct? Right when the screen first shows up? When the screen first shows up... what does that mean? Which one [event] do you think that is?
- Destini:            I tried this one [“OtherScreenClosed”] and this one [“ScreenOrientation Changed”].

Teacher: Huh. What does “initialize” mean?

Destini: I don’t remember.

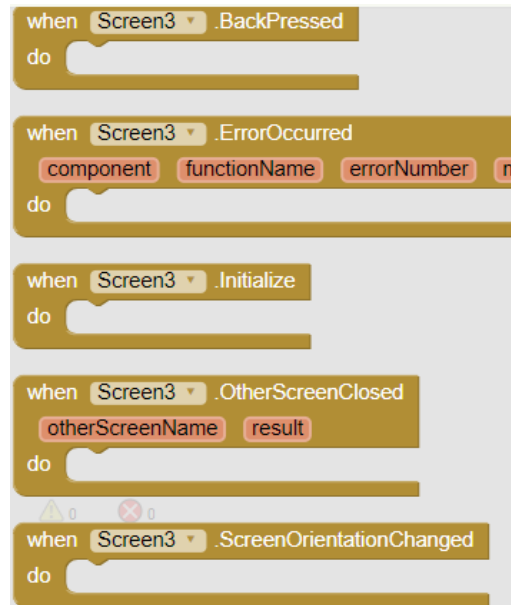


Figure 4.26: Possible events for the screen component.

The second vocabulary term which was a challenge for five of the groups was “heading.” These groups saw the term used in several of the tutorials, but when asked if they knew what it meant, one student replied, “It’s the top of a *Word* doc, right?” Unlike “initialize,” not knowing this term did not stop participants from using the code block because they saw its use in the tutorials, although they did have difficulty using the block correctly. For both “initialize” and “heading,” the teacher or researcher provided participants with their meaning and how these terms are used in the code. In addition to these terms, “logic” and the division symbol (“/”) was also a difficulty for individual participants, but in each case the participant’s partner explained the term.

## **Challenges Not Exclusive to Computer Science**

**Collaboration.** Collaboration in a learning activity is “students working together in small groups towards a common goal” (Kuo, Hwang, Chen, & Chen, 2012, p. 320) and can allow low-achieving students improve their understanding of mathematics when they work together using structured procedures (Allsopp, Kyger, & Lovin, 2007). Pair programming, a structured procedure for students learning to code by working together, was used in this study and has been found to encourage collaboration (Braught, Wahls, & Eby, 2011; Cao & Xu, 2005; Denner, Werner, Campe, & Ortiz, 2014). Additionally, Ratcliff and Anderson (2011) found that students with learning disabilities collaborated on their own when learning to code. In this study, however, audio recordings, field notes, and interview data showed collaboration was a challenge for nine of the groups even when the pair programming structure was enforced and when participants were encouraged to work together and seek peer support, an additional strategy for helping diverse learners learn to code (Israel, Pearson, Tapia, Wherfel, & Reese, 2015).

All of the groups collaborated well when designing their games, but on the first day of coding, the non-coding partner, who was supposed to be telling the coding partner what to do, was not included in the coding partner’s thought processes in all but three groups. The coding partner in each group was also the more dominant partner on this first day. When the coding partner had difficulties, (s)he would ask the teacher or researcher for assistance rather than the non-coding partner. These adults responded by re-enforcing pair programming and including both partners, such as when the teacher responded to one group, “You know what he is saying yet? You know what he is trying to do? So right off



the bat, I see the first problem: Your partner doesn't even know what you are trying to do." In some cases, this lack of collaboration was because the group misunderstood pair programming:

Teacher: Where is your computer? How can you be looking up stuff for him if your computer is not out?

Tyrone: I got it, it was just one person that's coding.

Teacher: One coding, but the other two should be looking up different things and saying this is how you do it.

On the second day of coding, when the partners switched roles, I observed in my field notes:

Today was the first time the non-dominant person coded. There was definitely resistance, both by that person and by the dominant partner. Sometimes because the less dominant didn't want to do the work (Matthew), sometimes because the more dominant didn't trust the other (Brandy). It didn't take long, however, for them to figure out how to work together. Often, I saw the dominant person lead the other into starting to code and I saw that person gain confidence.

This increase in collaboration did not happen easily, however. In this exchange, for example, Daniel wants the teacher's help because he does not have confidence in his partner:

Daniel: Okay, now what do I do?

Teacher: Now Justin, help Daniel figure out the next step.

Daniel: Show me how to put it in here.

Teacher: Justin is going to do that because he has the tutorial out.

Daniel: Or he obviously doesn't know how to do it.

In other cases, such as with Cary when he returned from an absence, the less dominant partner was reluctant to code because (s)he lacked confidence:

Cary: I don't know, I don't know what coding is, no, I'll be honest with you, I don't know what coding is.

Teacher: You don't have to worry about it, you really don't have to worry about it, you're still the one who's coding.

Cary: What is coding?

Teacher: Making a program work, but Travone is going to show you what he figured out last time and he's the one who's telling you what to do and he's very good at that. He did an excellent job helping out another group last time.

Although collaboration improved for some groups after the second day, eight groups continued to find collaboration challenging. In a few instances, the lack of collaboration was a minor interruption, like when Greg annoyed Katherine by echoing everything she said for five minutes, but for five of these groups, this lack of collaboration contributed to their not completing their games. As Destini explained in her interview why she was unhappy with the progress they made:

Destini: Well, if we worked better, then [the game] would have turned out like we wanted it, but it's not, it's not all that great.

Me: What do you mean if you worked better?

Destini: If we like put more effort in and actually like cooperated, I guess, then it would have been likely better, but it didn't turn out how we wanted.

In seven groups, at least one member participated so little that the other member(s) of the group stopped collaborating with them. As Cassidy explained in her interview when asked why she completed the game on her own, "I worked with Chalise before and she's really smart, but like if she doesn't get something she doesn't want to try as hard. So I guess this was just one of those things where like she didn't know a lot and

she just didn't want to try.” All but one of the groups with three members had a member who stopped participating after the design phase, even when the teacher or researcher suggested meaningful ways for that person to contribute.

**Learned helplessness.** Most groups asked the teacher or researcher for help once per session in the beginning of the intervention, every-other session, on average, after the third session, and engaged in conversations concerning coding or fractions when no adult was near. These groups initially expressed their lack of confidence in coding but gained confidence with reassurance and encouragement to try various approaches (Israel et al., 2015). Four groups, however, asked for assistance at least twice per session once they began coding, which remained consistent throughout the intervention, and rarely, if ever, discussed coding or fractions unless someone else was helping them. Allsopp, Kyger, and Lovin (2007) describe these behaviors as learned helplessness and further explain, “Students who experience continuous failure in mathematics expect to fail; resulting both in reticence to try something new and reliance on others to help them” (p. 46). They further explain, “Students with learned helplessness often resist trying new strategies in problem solving situations” (p. 50) and affect not only the learning of mathematical content but also the use of the mathematical process skills of problem, solving, reasoning and proof, communication, and making connections. The math process skills of problem solving, reasoning, and making connections are also skills used when coding (Calao, Moreno-León, Correa, & Robles, 2015). This code was used when the audio recordings and field notes showed instances of participants expressing reluctance to solve the problem they had identified, such as stating “I give up” after realizing they needed to use

a component they had not used before, followed by the group asking for assistance rather than seeking a solution independently; groups that decreased this behavior after the third session were then removed from this code's data because their initial behavior likely indicated a lack of confidence in coding rather than learned helplessness.

The participants in these four groups regularly made statements to the teacher or researcher that expressed defeat, such as "I don't want to do this no more" and "I give up." The conversations that these groups had preceding such statements to the teacher or researcher revealed that one member of the group would share a problem or frustration with the others, but then another member of the group would respond with a statement that encouraged the rest of the group to quit. For example, Katherine was attempting to get an object to move on the screen when she asked her group for help:

Katherine: I told you, so for all the work that I do today, I need help.

Greg: We need Jesus.

Ian: Yes.

After Greg's and Ian's responses, Katherine stopped working and the group engaged in a conversation about a social event until the teacher walked near and they asked for help.

Allsopp, Kyger, and Lovin (2007) suggest helping students overcome learned helplessness by decomposing tasks into smaller ones and monitoring their progress (p. 50). During the intervention, the teacher or researcher applied these strategies by identifying one task for the group to work on, monitoring their progress, then identifying the next task. For example, after Amy, Kala, and Sarah had completed the visual part of their game, they immediately asked the researcher for help:

- Amy: I'm confused because this don't make no sense.
- Me: Okay, what part are you working on?
- Amy: A majority of things to work, but we don't know.
- Sarah: How to get it.
- Me: Okay, well, I would start with the buttons. Let's do one together.

When they completed coding their buttons, they again asked for help and again the researcher suggested a task from their coding plan. This pattern continued throughout the intervention and was sufficient for two of the groups to make progress on their games.

Justin and Daniel also benefited from the teacher or researcher identifying smaller tasks for them to complete, but they often stopped halfway through the task to ask for additional assistance. An additional strategy, to encourage and reassure students' attempts (Israel et al., 2015), helped them, as the following exchange demonstrates:

- Justin: Ma'am, we need some help. I don't know how to get it, we almost got it over his hand.
- Teacher: Oh! My goodness, you're almost there! Okay, how did you get it so close?
- Daniel: We kept using bigger numbers.
- Teacher: That was a good idea. Why did you stop trying that?
- Justin: I don't know. It didn't seem to work.
- Teacher: But it almost worked, so maybe just keep trying it?
- Justin: Okay.

One group would not try to code unless someone explicitly helped them, even after the above strategies were attempted. Their game design contained several

components, but most of these components had the same functionality, so completing one component successfully would provide them with a template for completing many of the others. By the eighth session, however, they only had code for one of these components completed, and that was done with the researcher. During this session, another student had completed his game and volunteered to help others. He worked with this group for approximately thirty minutes when the teacher suggested he help another group for a while:

Teacher: All right, are you okay for a while without Brian?

Katherine: No.

Greg: No, we can't do it.

Greg, Katherine, and Ian continued to work only when someone sat with them helping for the remainder of the sessions, which happened more often as others completed their games. When asked in the post-interview why he felt he needed this level of support, Greg replied, “I thought it was too complicated to like code something to ... but when people came over and did step by step with me and showed me how to do this, that wasn't too bad.”

### **Summary of Challenges**

The thirty-two participants in this study each experienced one or more challenges when making their games beyond the challenge of working with fractions. Most of these challenges directly concern coding: algorithm development, debugging, working with angles, complexities in the game design, decomposition, coding concepts and skills, vocabulary used in the coding blocks, and limitations found in *App Inventor*. Three of the

challenges, however, are not exclusive to coding or computer science: transferring learning from one task to another, collaboration, and learned helplessness. For all but one challenge, complexities in the game design, the researcher or teacher provided supports which helped most participants continue progress on their games. These challenges, how many participants experienced them, and the supports provided are listed in appendix N.

Four of these challenges have also been identified in the literature, with three studies (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Santi & Baccaglini-Frank, 2015) identifying algorithm development, debugging, and transferring learning from one task to another as difficult for their participants and Ratcliff and Anderson (2011) also identifying graphics, especially angles in graphics, as a challenge. These challenges affected twenty-nine of the participants in this study, with debugging challenging each of the twenty-nine, algorithm development affecting twenty-three participants, transferring learning affecting twenty-three participants, and working with angle measurements affecting eleven participants. The literature suggested ways to support students with these challenges, but with each challenge additional supports were needed.

All participants also experienced challenges that were specific to coding their games. The primary challenge, identified before the intervention began, was that *App Inventor* does not support collaborative development. Thirty-one participants needed to download and share their projects through another environment, *Google Classroom*, so more than one student could work on the game directly; the one unaffected participant was working alone after his partner left the study. *App Inventor* had other limitations that

affected participants' coding efforts. The researcher created sample code for the affected groups and explicit instruction on how the code works; difficulties after this instruction were classified as "transfer" or "algorithm development" challenges.

Twenty-four participants encountered four additional challenges specific to coding other than challenges associated with *App Inventor*. These challenges were: (1) designing games with complex features or functionalities (21 participants), (2) decomposing their designs or elements in their designs into smaller tasks (19 participants), (3) understanding coding concepts or skills (18 participants), and (4) understanding the vocabulary in the coding blocks (15 participants). Guiding questions and explicit instruction was sufficient support for three of these challenges, but no support was provided to address participants with complex game designs. The researcher determined that encouraging participants to simplify their designs could influence their work with fractions, which would threaten the trustworthiness of this study, and instead chose to encourage these participants to create a prototype, a version of their game with some of the features functional.

Two challenges, collaboration and learned helplessness, are not exclusive to coding or computer science yet affected twenty-one participants. Encouragement and splitting tasks into smaller parts helped all nine participants who demonstrated learned helplessness, but collaboration remained a challenge for fifteen participants throughout the study. Additional encouragement and re-enforcing the pair-programming protocol was attempted but only helped six participants collaborate effectively.



Nine of the fifteen games were not completed during the intervention and one group dramatically changed their game design to a simple quiz game during the intervention. Two challenges were common to each of these groups: complex game design and collaboration challenges. Four of these groups produced a mostly-functional prototype, which included some of their fraction problems and at least half of their additional features, by the end of the intervention. Three of these four groups, each with two participants, had resolved their collaboration challenges. Of the six completed games, three groups had unresolved collaboration challenges; each was a group of two participants and completed their games after one participant decided to work without the other's assistance.

### **Summary**

Participants created three kinds of representations for fractions and used these representations to develop their understanding of fraction magnitude. All participants used numeric representations and most also used area models, which are the most common representations found in math textbooks (Zhang, 2012). The ways participants interacted with their fraction representations developed their understanding of fraction magnitude and maps to the experiential learning cycle (Kolb, 1984). Thus, experiential learning theory explains how participants developed their understanding of fraction magnitude, which occurred when participants worked with area models, talked about area models, and developed code for comparing fractions.

Participants also experienced several challenges other than with fractions when developing their games. Some of these challenges have been identified in previous

studies concerning students with learning disabilities and coding: algorithm development, debugging, transferring learning from one task to another, and working with angles in graphics (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Santi & Baccaglini-Frank, 2015). In this study, these challenges were not restricted to participants who had an identified learning disability. Participants in this study also experienced additional challenges when coding and challenges that are not specific to computer science activities. These additional challenges participants had coding were challenges concerning their game designs, decomposing their game designs into components to code, coding concepts and skills, limitations in the *App Inventor* environment, and some of the vocabulary used in the coding blocks. The challenges participants had that are not exclusive to computer science were challenges collaborating and learned helplessness. These challenges may help explain why only six games were completed during the intervention and may help identify factors that may have limited the participants' development of fraction understanding (Allsopp, McHatton, & Farmer, 2010).

## CHAPTER 5: DISCUSSION

This study asked low-achieving eighth-grade students to create games about fraction magnitude using an NPE, *App Inventor*, to address gaps in their understanding. The research is based on the work of Seymour Papert (Feurzeig & Papert, 2011; Harel & Papert, 1990) and Yasmin Kafai (1995) and extends their work by using a different NPE and by working with older students who have demonstrated low achievement in mathematics. It asked what representations of fractions the participants used in their games, how they developed their understanding of fraction magnitude, and what challenges they experienced other than with fractions.

The findings suggest that participants with a minimal understanding of fraction magnitude, as measured by the pretest, developed their understanding of fraction magnitude during the intervention. These participants also included two representations in their games, one numeric in the form  $a/b$  and one non-numeric. Most of the non-numeric representations were area models, which participants worked with or talked about during the intervention. One participant, however, represented fractions in his code as the division of two integers; he demonstrated his developing understanding when he encountered an error in his code. Each of these ways of interacting with fractions mapped to the experiential learning cycle, demonstrating that participants engaged in a concrete experience with fractions, reflected on what they observed, conceptualized their understanding, and experimented with their new understanding (Kolb, 1984; Matsuo, 2015). Participants also experienced several challenges when creating their games. Many of these challenges have been identified in prior research concerning students with

learning disabilities and computer science or mathematics education, but in this study, the challenges were found to affect participants with and without identified learning disabilities.

This chapter will begin by situating the findings for each research questions with the relevant literature. It will then describe the limitations of this study and the implications for practitioners and researchers. The chapter concludes with a final reflection.

### **Relationship of Prior Research to the Study's Findings**

The purpose of this study was to examine what representations of fractions low-achieving students use in the games they create, how they develop an understanding of fraction magnitude while developing their games, and what challenges they have beyond working with fractions as they develop their games. This study adds to the literature on the use of NPEs by extending prior research to the secondary school level and by working with low-achieving students. This section will situate the findings of this study into the existing body of research.

#### **RQ1: Representing Fraction Magnitude in Games**

Ten of the fifteen games used area models, a specific type of fraction model in which the fraction is shown as a shaded portion of a two-dimensional figure. Area models are the most common non-numeric representation of fractions in textbooks (Zhang, 2012) and in teaching (Zhang, Clements, & Ellerton, 2015), so the participants in this study would likely have been more familiar with area models than other representations and thus would have chosen them to represent fractions in their games. Students in

elementary and middle school grades also tend to represent mathematics using objects from their concrete experiences (NCTM, 2000, p. 68), but only one of these games used a real-world object, pizza, as an area model. The remaining nine games used basic geometric figures for their area models; eight used circles and one used both circles and hexagons. Basic geometric figures are the most common form of area models in textbooks (Zhang, 2012), with circles being the recommended figure for teaching fractions (Bray & Abreu-Sanchez, 2010; Cramer & Henry, 2002), which again suggests that the students in this study would have seen or used circle area models more than other representations in their previous math instruction.

One game represented fractions as the division of two integers. This representation is one that the Common Core State Standards (NGA, 2010) and the National Council of Teachers of Mathematics (2000) recommend students should be able to use to represent fraction magnitude. The remaining four games only used numeric representations of fractions. Although these can be valid representations for fraction magnitude (Lesh, Post, & Behr, 1987) and may have been effective learning experience for these students because they constructed the representations themselves (Ainsworth, Bibby, & Wood, 2002; Greeno & Hall, 1997; NCTM, 2000; Rau, Aleven, & Rummel, 2015; Zhang, Clements, & Ellerton, 2015), their learning may have been limited because they did not convert between various representations like the other participants did (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000; Panaoura, Gagatsis, Deliyianni, & Elia, 2009). Only one of these seven participants who only used numeric representations showed gains on the posttest.

## **RQ2: Developing an Understanding of Fraction Magnitude**

Participants developed their understanding of fraction magnitude when creating their games by working with area models, talking about area models, and developing code for comparing fractions. The data showed several instances where students changed their thinking regarding the properties of rational numbers, the relationship between the numerator and the denominator, or how to represent fraction magnitude, which the literature suggests shows a development of understanding (Gabriel et al., 2012; Jordan et al., 2013; Siegler, Fazio, Bailey, & Zhou, 2013; Vamvakoussi & Vosniadou, 2004). In each case, students constructed a way to represent fractions (verbally, representatively, or physically); the research suggests that students interacting with representations is required to develop their understanding (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000; Panaoura, Gagatsis, Deliyianni, & Elia, 2009) and is especially effective when they create their own representations (Ainsworth, Bibby, & Wood, 2002; Greeno & Hall, 1997; NCTM, 2000; Rau, Alevan, & Rummel, 2015; Zhang, Clements, & Ellerton, 2015). The participants in this study who demonstrated that they developed their understanding of fraction magnitude, as evidenced in the qualitative and quantitative data, created area models or developed code to represent fractions as the division of two integers and converted between these representations and numeric representations of fractions.

Asking participants to develop a game about fraction magnitude using an NPE created a catalyst for experiential learning because learning to code or program fosters an experiential learning environment (Feurzeig & Papert, 2011; Robins, Rountree, &

Rountree, 2003), as does providing students with a problem case to work (Georgio, Zahn, & Meira, 2008). Experiential learning theory explains how participants developed their understanding of fraction magnitude because they created and interacted with fraction representations while designing and developing their games (Sanford, Hopper, & Starr, 2015), and the ways they did so map to the four phases of the experiential learning cycle: concrete experience, reflective observation, abstract conceptualization, and active experimentation (Kolb, 1984; Matsuo, 2015).

When participants created their area models or verbally posed a question or scenario about area models to their partners, they demonstrated that these were challenging experiences for them because the creations, questions, and scenarios exposed their misconceptions about fraction magnitude (Matsuo, 2015). These challenging experiences map to the “concrete experience” and “active experimentation” phases of the experiential learning cycle (Matsuo, 2015). Receiving and considering the feedback from their partners maps to the “reflective observation” and “abstract conceptualization” phases because “feedback provides the basis for a continuous process of goal-directed action and evaluation of the consequences of that action” (Kolb, 1984, p. 22) and encouraged participants to think critically about their experience (Matsuo, 2015).

Developing code for comparing fractions, the third way participants developed their understanding of fraction magnitude, also maps to the experiential learning cycle. Working with code, either creating new code or fixing existing code, maps to the “concrete experience” phase because transferring knowledge into code creates a concrete experience for the person coding (Turkle & Papert, 1990). Fixing an error in the code

then maps to the remainder of the cycle because a program that does not work still does something that can be observed, reflected upon, and understood (Feurzeig & Papert, 2011). Like the others who worked with or talked about area models, the participant who developed code for comparing fractions encountered challenging experiences when his game did not work as intended and when he tried fixing his code and thought critically about his experiences as he tried to determine the cause of the error and a possible solution, which are evidence for how his interaction with fractions maps to the experiential learning cycle (Matsuo, 2015).

### **RQ3: Challenges Experienced When Designing and Developing Games**

Three studies (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Santi & Baccaglini-Frank, 2015) identified algorithm development, debugging, and transferring learning from one task to another as difficult for students with learning disabilities when they learn to code; Ratcliff and Anderson (2011) also found working with graphics, especially angles in graphics, challenging for them. This study confirmed these findings and furthermore found that these challenges affected participants with and without identified learning disabilities. The supports identified in these studies were also found to be effective supports for helping participants through these challenges.

Two studies (Israel et al., 2015; Ratcliff & Anderson, 2011) found that students would work together on their own to overcome coding difficulties. This study did not find evidence of participants voluntarily helping their peers, although in three instances a participant willingly helped another when the teacher or researcher invited her or him to do so. Other studies (e.g., Braught, Wahls, & Eby, 2011; Denner, Werner, Campe, &



Ortiz, 2014; Van de Grift, 2004) suggested implementing a pair programming protocol so students would have shared but equal responsibilities coding and would therefore support each other's learning. This study used the pair programming protocol because, in addition to the benefits written about it, structured procedures for working together can allow low-achieving students to improve their understanding of mathematics (Allsopp, Kyger, & Lovin, 2007). Nine of the fifteen groups in this study, however, had challenges that affected their ability to work independently rooted in their inability to work together even when encouraged to collaborate (Israel et al., 2015) and when the protocol was re-enforced (Braught, Wahls, & Eby, 2011).

Participants experienced other challenges when developing their games that were not identified in the literature. Some of these challenges can be attributed to the participants' inexperience when coding, such as not understanding computer science concepts, skills, or vocabulary, or to limitations of the NPE, *App Inventor*. The remainder, however, have connections with the literature concerning students with learning disabilities. One issue identified in this literature that helps explain participants' challenges, working memory deficits, is a common issue for students with learning disabilities (Cai, Li, & Deng, 2013; Geary, 2013; Swanson & Zheng, 2013), negatively impacts problem-solving skills (Allsopp, Kyger, & Lovin, 2007; Baddeley, 2010; Geary, 2013; Kirschner, Sweller, & Clark, 2006; Lyon & Weiser, 2013; Swanson & Zheng, 2013) and is related to reasoning ability (Baddeley, 2010; Kyllonen & Christal, 1990). The computational thinking skill of decomposition is a part of problem solving (Selby & Woollard, 2013) and thus would be affected by working memory deficits because

decomposition asks the individual to identify the key characteristics of the problem and disassemble it into smaller components (Grover & Pea, 2013), both are skills that interventions for students with working memory deficits address (Adams & Carnine, 2003; Kirschner, Sweller, & Clark, 2006; Likourezos & Kalyuga, 2017). Working memory deficits may explain why decomposition was a challenge experienced by participants in this study as well as possibly explaining the challenges previous studies identified that are related to problem solving, such as algorithm development and transfer. Another challenge identified in this study as well as by Ratcliff and Anderson (2011), working with angles, may also be related to working memory deficits because working with angles requires visual-spatial reasoning (Hegarty & Kozhevnikov, 1999), which is a component of working memory (Swanson & Zheng, 2013).

Allsopp, Kyger, & Lovin (2007) also identify learned helplessness as a behavior common to students with learning difficulties that affects problem solving, reasoning, and making connections in mathematics. This study identified four groups that had at least one participant displaying behaviors consistent with learned helplessness to such a degree as it prevented the group from working without assistance. Decomposing tasks into smaller ones and monitoring their progress, strategies identified by Allsopp, Kyger, and Lovin (2007) to help students exhibiting this behavior, supported the groups experiencing this challenge.

Participants required additional supports when experiencing challenges, which is common for students with learning disabilities when using approaches other than direct instruction (Godino, Batanero, Cañadas, & Contreras, 2017). Appendix N shows the

supports provided for each challenge. Supporting debugging, decomposition, and transferring knowledge challenges occurred by prompting students to help them articulate their thinking and explicitly demonstrating connections between similar problems, which are suggested strategies for supporting students with learning disabilities when direct instruction is not used (Moscardini, 2010; Xin, Liu, Jones, Tzur, & Si, 2016).

### **Limitations of the Study**

There were two items regarding the participants in this study which may limit its transferability. First, fifteen participants were identified as having a learning disability, but information about their disabilities was not available to this researcher, and these participants were in thirteen of the fifteen groups. Because of this, distinguishing how participants with learning disabilities, or participants with specific learning disabilities, represented fractions, developed their understanding of fractions, and experienced challenges when creating their games could not be distinguished from participants without identified disabilities. Thus, the findings of this study only apply to its intended population, secondary students with low-achievement in mathematics. Second, participants' test scores and grades from the previous year were also not available to this researcher, which not only limited the description of the participants but also prevented the research from understanding their previous understanding of fraction magnitude.

The credibility of the findings for the first research question, how students represented fractions, could be questioned because of the impact that the resources available to the participants may have had on their representations. Nine of the games used area models, which are the most common non-numeric representation found in math

textbooks (Zhang, 2012). An examination of the textbooks made available to the participants showed that area models were the predominant non-numeric representation they used. The data shows that all groups referred to the provided textbooks when designing their games, and although the data did not reveal any direct evidence suggesting the influence of these books (e.g., a participant stating “Let’s do it like this.”), when they used the books increases the likelihood that the representations in the books influenced their thinking.

The pre- and posttest used in this study ensured that all participants had at least a basic understanding of fraction magnitude and identified who likely developed their understanding of fraction magnitude during the intervention. Although quantitative methods were used to analyze this data, the sample size is too small to generalize the results to a population outside of this study.

The use of multiple data sources, member checking during the interviews, and peer review of the findings were used to minimize confirmatory bias of the researcher (Rabin & Schrag, 1999; Shenton, 2004). Still, the researcher’s experiences and epistemological beliefs influenced the data collection and analysis (Shenton, 2004; Whitemore, Chanse, & Mandle, 2001). For the third research question especially, this researcher’s prior experience in computer science and computer science education influenced the codes used to identify the challenges participants experienced. Prior research was used to ensure that these codes were consistent with the literature, but since not all of them could relate to the literature, this researcher used her experiences to identify and define the remaining challenges.

## **Implications of the Study's Findings**

### **Implications for Practitioners**

With increasing demand to bring computer science education to all K-12 learners (Krueger, 2017), finding ways to integrate these concepts and skills with existing curricula could help more schools include computer science education in their already packed schedules (Mehta, 2013; Sniegowski, 2017). This integration would especially help low-achieving students who cannot take as many electives as their peers because they are enrolled in additional math or reading classes (Williams, 2014), such as the participants in this study, and thus would not have equal opportunities to learn computer science. This study demonstrated one possible way to integrate computer science with a core subject area, mathematics, to provide opportunities in computer science to low-achieving students.

This study also presents practitioners with a viable intervention for middle school students struggling in mathematics. Kirschner, Sweller, and Clark (2006) suggest that direct instruction benefits learners who do not have enough knowledge stored in their long-term memory, but Deanna Kuhn (2007) suggests that constructivist approaches to instruction are more effective than direct instruction when teaching problem-solving and conceptual understanding, especially to older students; other studies have since found that constructivist approaches are effective for all learners at the secondary level to gain mathematical understanding (e.g., Bottge et al., 2015; Han, Caparo, & Caparo, 2015). In this study, participants who scored lowest on the pretest demonstrated a developing understanding of fraction magnitude during the intervention, suggesting that having

students create games about fractions, a constructivist approach to instruction, would be an effective activity to help low-achieving secondary students improve their mathematical understanding, at least with fraction magnitude.

The third implication for practitioners concerns the challenges participants in this study faced when creating their games. The literature is limited concerning the challenges faced by students with learning difficulties as they learn to code (Santi & Baccaglini-Frank, 2015), which could present problems as schools try to implement the new K-12 Computer Science Framework (2016) because teachers would be unable to prepare for the difficulties their learners might encounter. This study confirmed what challenges have been identified in the literature (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011; Snodgrass, Israel, & Reese, 2016), identified other challenges that participants had when coding, and described the supports used during the intervention to help participants through these challenges. Such knowledge could support practitioners as they teach computer science to a diverse student population.

### **Implications for Research**

This study extends the work of Seymour Papert (Feurzeig & Papert, 2011; Harel & Papert, 1990; Papert, 1987) and Yasmin Kafai (Kafai, 1995; Kafai, Franke, Ching, & Shih, 1998), who worked with elementary students, by demonstrating that coding to learn fractions is a viable intervention for secondary students with low achievement in mathematics to develop their understanding of fraction magnitude. This study demonstrates that participants changed their thinking regarding fraction magnitude and constructed representations of fractions, which the literature suggests shows a

development of understanding (e.g., Ainsworth, Bibby, & Wood, 2002; NCTM, 2000; Siegler, Fazio, Bailey, & Zhou, 2013; Zhang, Clements, & Ellerton, 2015).

This study also extends the literature regarding students with learning disabilities by demonstrating that four of the challenges (decomposition, algorithm development, transfer of knowledge, and working with angles) students experienced when creating their games are like those experienced by students with learning disabilities in other educational settings. These challenges relate to problem solving or visual-spatial reasoning (Grover & Pea, 2013; Selby & Woollard, 2013), which are negatively affected by working memory deficits (Baddeley, 2010), a common characteristic of students with learning disabilities (Cai, Li, & Deng, 2013; Geary, 2013; Swanson & Zheng, 2013), and impact one's ability to learn mathematics (Barnes & Raghobar, 2014; Cai, Li, & Deng, 2013; Geary, 2013; Swanson & Zheng, 2013). Another challenge that presented in this study, learned helplessness, is also a challenge experienced by students with learning difficulties and affects their problem-solving, reasoning, and making connections (Allsopp, Kyger, & Lovin, 2007), skills used when coding (Calder, 2010).

### **Future Research**

The nation currently faces a shortage of computer science teachers (Maio, 2016; United States Department of Education, 2017), so realizing the vision of computer science education across all grades and with all learners may require preparing current and prospective non-computer science educators to include it in their instruction (K-12 Computer Science Framework, 2016). If teachers are going to use this instruction, further

research will be needed to understand how best to train and support them in this work (Grover & Pea, 2013).

Additional research is also needed to understand how English-language learners develop their understanding of fraction magnitude when developing games about fractions using an NPE. Although there was a student identified as an English-language learner in one of the classes for this study, she chose to not participate in the research. Her behaviors during the intervention, however, suggest that there are specific challenges and supports needed to help this population participate in computer science activities and develop an understanding of fraction magnitude using a non-traditional approach such as this study's intervention.

Because only participants who earned less than 60% of the possible points on the pretest demonstrated they developed an understanding of fraction magnitude during this study, further research may help identify why the remaining participants did not. The findings of this study suggest that their use of only numeric representations of fractions contributed to this lack of development, but it is also likely that a ceiling effect occurred with the instrument used for the pretest or that their developing understanding was not detected in the qualitative analysis. Continuing research on this intervention would identify if and how students with a stronger understanding of fraction magnitude continue to develop in their understanding.

Another one of the findings of this study, the impact collaboration challenges had on participants' completing their games, is an area for further research. Prior research suggested that students would work together to overcome coding difficulties (Israel et al.,



2015; Ratcliff & Anderson, 2011) and, to support such collaboration, the pair programming protocol (Braught, Wahls, & Eby, 2011) was used during the intervention. The findings of this study suggest that this support was insufficient; further research may help identify what support would increase collaboration during the intervention.

Finally, further research can study the effectiveness of having low-achieving secondary students create computer games to learn mathematics. Although a body of research exists suggesting coding is a viable tool for learning mathematics (e.g., Calder, 2010; Harel & Papert, 1990; Kafai, 1996), analyzing the effectiveness of this approach has been limited. Is an intervention such as the one used in this study an effective approach for learning fraction magnitude? Are there constraints or conditions on the effectiveness of this approach, such as the age of the student or their prior experience coding? And finally, how does this approach compare to other methods for teaching fractions to low-achieving students?

### **Post-Mortem**

I believe that, overall, this intervention was successful in helping low-achieving middle school students develop a better understanding of fraction magnitude, but there are a few things I would do differently to maintain student motivation throughout the intervention and, possibly, improve the benefits to students. While motivation was not generally an issue with this project, participants demonstrated less on-task behavior during the middle of the intervention (sessions 4 through 7 out of 10). During these sessions, several participants commented on how they had until the end of the month to complete their games, and these comments were said without a feeling of urgency, which

suggests that they felt no need to work diligently during these sessions. To help maintain student motivation, I would include benchmarks with due dates. For example, the design and coding plan would need to be finished by the end of session 2, all components would need to be placed on the front-end by the end of session 4, and then benchmarks for completing and testing sections of the code would be determined on a game-by-game basis so each group would have a checklist of deliverables specific to their game. These benchmarks would help students feel a sense of urgency to complete tasks, since the due dates would be near, and may minimize some of the challenges they experienced by providing a more organized structure to their game development process.

In this study, not every group created multiple representations of fractions in their games, which is what research suggests is the best practice for developing an understanding of fraction magnitude (e.g., Ainsworth, Bibby, & Wood, 2002; Panaoura et al., 2009; Siegler, Fazio, Bailey, & Zhou, 2013). The predominance of simple quiz games and games with quiz-like questions likely contributed to this limitation, since quiz-like questions can be created using only numeric representations. When students are challenged to design a game about fractions that does not ask questions, however, they will create and integrate various ways of representing fractions in their games (Kafai, Franke, Ching, & Shih, 1998). Justin's game is a good example of this: He did not intend to use multiple representations, but the only way he could make his code compare the numeric representations displayed on the front-end of his game was to represent them as the division of integers in his code. Therefore, the other modification I would do to the intervention would be to have participants create games that did not ask questions. This

change may need to be preceded by having them create a simpler app so they may develop coding skills and confidence before they develop this more challenging game, but it would likely ensure that students work with multiple representations of fractions, which would increase the benefits for the students.

### **Final Reflections**

The reason this study specified low-achieving middle school students and their fraction understanding is because research suggests students who have difficulties in mathematics in middle school, specifically in understanding fraction magnitude, will have difficulties understanding algebra 1 (e.g., Booth & Newton, 2012; Brown & Quinn, 2007; Siegler et al., 2012), be less likely to take a math course beyond algebra 2 (Sciarra, 2010), and be less likely to graduate high school (Orihuea, 2006) or succeed in higher education (Adelman & United States., 2006; Trusty & Niles, 2004). Fraction magnitude is a conceptual understanding which involves (a) understanding their properties, such as the principle of equivalent fractions, (b) understanding how the numerator and the denominator determine magnitude, and (c) the ability to work with and create various ways to represent fraction magnitude, such as ordering on a number line (Gabriel et al., 2012; Jordan et al., 2013; Siegler, Fazio, Bailey, & Zhou, 2013; Vamvakoussi & Vosniadou, 2004). To understand fraction magnitude, students need to learn to work with and convert between various representations (Duval, 2006; Even, 1998; Lesh, Post, & Behr, 1987; NCTM, 2000; Panaoura, Gagatsis, Deliyianni, & Elia, 2009). Asking students to construct their own representations is the most effective way for them to gain this understanding (Ainsworth, Bibby, & Wood, 2002; Greeno & Hall, 1997; Rau,

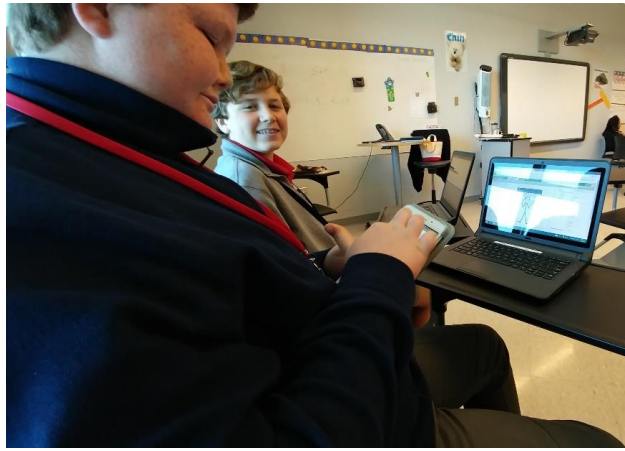
Aleven, & Rummel, 2015; Zhang, Clements, & Ellerton, 2015) and was the primary aim of this intervention.

Using *App Inventor* and game design as a means of getting students to construct their own representations of fractions are methods supported in the literature, but this researcher also hoped that students would find game design and/or coding to be a motivating experience. A student who experiences difficulties with secondary mathematics has likely experienced difficulties since learning fractions in elementary school (Booth & Newton, 2012); such long-term difficulty decreases motivation (Nicholls, 1979). Coding can increase students' willingness to learn a topic even if they found that topic uninteresting beforehand (Harel & Papert, 1990). Although data on motivation was not deliberately included in this study, observational and interview data suggest that creating games about fractions did motivate participants, at least in the beginning. The challenge of learning to code, however, caused some participants to lose motivation. As Cassidy said of her partner:

I worked with Chalise before, like she's in my math class and she's really smart, but like if she doesn't get something, she like doesn't want to try as hard. So I guess this [coding] was just one of those things where like she didn't know a lot about it and that she just didn't want to try.

For other participants, the challenges they experience became a source of pride. In the final interviews, participants regularly cited one of their coding challenges as what they were most proud of in their games. Brian mentioned learning how to change screens when a button was clicked, Destini discussed getting her image sprites to move correctly, Matthew recounted how he learned the “heading” block used angle measurements, and both Cassidy and Justin shared how fixing the errors in their games were what they were

most proud of. Justin's pride was also evident when he shared his newly-working game with Daniel (see figure 5.1). Justin was one of the participants who displayed learned helplessness behaviors, so to see him smiling and sharing his working game and to hear him say he was proud of how he fixed his code's error was an additional benefit for this researcher.



*Figure 5.1: Justin (background) smiling as Daniel plays the working game.*

Creating games using *App Inventor* to develop an understanding of fraction magnitude is a viable intervention for low-achieving eighth grade students, as the findings of this study demonstrated. Further research will determine the effectiveness of the intervention, the issues concerning student populations not represented in this study, and what preparations teachers will need to use this or similar interventions in their classrooms. Importantly, the findings of this study may inform researchers and practitioners wanting to work with NPEs and low-achieving students, especially in mathematics, because it adds to the literature on NPEs, using fraction representations, and developing an understanding of fraction magnitude.

## References

- ACT. (2015). 2015 Retention/Completion summary tables. *Research and Policy Issues: College student retention and graduation rates from 2000 to 2015*. Retrieved from <http://www.act.org/research/policymakers/reports/graduation.html>.
- Adams, G., & Carnine, D. (2003). Direct instruction. *Handbook of learning disabilities*, 403-416.
- Baddeley, A. (2010). Working memory. *Current Biology*, 20(4), R136-R140. doi:10.1016/j.cub.2009.12.014
- Adelman, C. & United States. (2006). *The toolbox revisited: Paths to degree completion from high school through college*. Washington, D. C.: Office of Vocational and Adult Education, U.S. Dept. of Education. Retrieved from <http://catalog.hathitrust.org/Record/005568101>; <http://hdl.handle.net/2027/mdp.39015069291808>
- Ainsworth, S., Bibby, P., & Wood, D. (2002). Examining the effects of different multiple representational systems in learning primary mathematics. *Journal of the Learning Sciences*, 11, 25-61.
- Allsopp, D. H., McHatton, P. A., & Farmer, J. L. (2010). Technology, mathematics PS/RTI and students with LD: What do we know, what have we tried, and what can we do to improve outcomes not and in the future? *Learning Disability Quarterly*, 33(4).
- Allsopp, D. H., Kyger, M., & Lovin, L. (2007). *Teaching Mathematics Meaningfully: Solutions for Reaching Struggling Learners*. Baltimore, MD: Paul H. Brookes Publishing Co.
- An, Y. (2016). A case study of educational computer game design by middle school students. *Educational Technology Research and Development*, 64(4), 555-571. doi:10.1007/s11423-016-9428-7
- Angen, M. J. (2000). Evaluating interpretive inquiry: Reviewing the validity debate and opening the dialogue. *Qualitative Health Research*, 10(3), 378-395.
- Anney, V. (2014). Ensuring the quality of the findings of qualitative research: Looking at trustworthiness criteria. *Journal of Emerging Trends in Educational Research and Policy Studies*, 5(2).
- Archbald, D. & Farley-Ripple, E. (2012). Predictors of Placement in Lower Level Versus Higher Level High School Mathematics. *The High School Journal*, 96(1), 33-51.

- Aydin, E. (2005). The use of computers in mathematics education: A paradigm shift from “computer aided instruction” towards “student programming.” *The Turkish Online Journal of Educational Technology*, 4(2).
- Bailey, D. H., Hoard, M., Nugent, L., & Geary, D. (2012). Competence with fractions predicts gains in mathematics achievement. *Journal of Experimental Child Psychology*, 113, 447-455.
- Barnes, M. A., & Raghubar, K. P. (2014). Mathematics development and difficulties: The role of visual–spatial perception and other cognitive skills. *Pediatric Blood & Cancer*, 61(10), 1729-1733. doi:10.1002/pbc.24909
- Baxter, P. & Jack, S. (2008). Qualitative case study methodology: Study design and implementation for novice researchers. *The Qualitative Report*, 13(4).
- Baytak, A. & Land, S. (2011). An investigation of the artifacts and process of constructing computers games about environmental science in a fifth grade classroom. *Educational Technology Research & Development*, 59(6), 765-782. doi:10.1007/s11423-010-9184-z
- Booth, J. L. & Newton, K. J. (2012). Fractions: Could they really be the gatekeeper's doorman? *Contemporary Educational Psychology*, 37(4), 247-253.
- Bottge, B. A. & Hasselbring, T. S. (1999). Teaching mathematics to adolescents with disabilities in a multimedia environment. *Intervention in School & Clinic*, 35(2), 113-116. doi:10.1177/105345129903500208
- Bottge, B., Toland, M., Gassaway, L., Butler, M., Choo, S., Griffen, A., & Ma, X. (2015). Impact of enhanced anchored instruction in inclusive math classrooms. *Exceptional Children*, 81(2).
- Brought, G., Wahls, T., & Eby, L. M. (2011). The case for pair programming in the computer science classroom. *ACM Transactions on Computing Education*, 11(1), article 2.
- Bray, W. S. & Abreu-Sanchez, L. (2010). Using Number Sense to Compare Fractions. *Teaching Children Mathematics*, 17(2), 90-97.
- Brown, G. & Quinn, R. J. (2006). Algebra Students' Difficulty with Fractions: An Error Analysis. *Australian Mathematics Teacher*, 62(4), 28-40.
- Brown, G. & Quinn, R. J. (2007). Investigating the relationship between fraction proficiency and success in Algebra. *Australian Mathematics Teacher*, 63(4), 8-15.

- Burns, A. & Gentry, J. (1998). Motivating students to engage in experiential learning: A tension-to-learn theory. *Simulation & Gaming, 29*(2), 133.
- Cai, D., Li, Q. W., & Deng, C. P. (2013). Cognitive processing characteristics of 6th to 8th grade chinese students with mathematics learning disability: Relationships among working memory, PASS processes, and processing speed. *Learning and Individual Differences, 27*, 120. doi:10.1016/j.lindif.2013.07.008
- Calao, L. A., Moreno-León, J., Correa, H. E., & Robles, G. (2015). Developing mathematical thinking with scratch. In *Design for Teaching and Learning in a Networked World* (pp. 17-27). Cham: Springer. DOI: 10.1007/978-3-319-24258-3 2
- Calder, N. (2010). Using Scratch: An integrated problem-solving approach to mathematical thinking. *Australian Primary Mathematics Classroom, 15*(4), 9-14.
- Carlson, J. A. (2010). Avoiding traps in member checking. *The Qualitative Report, 15*(5), 1102.
- Carver, J. C., Henderson, L., He, L., Hodges, J., & Reese, D. (2007, July). Increased retention of early computer science and software engineering students using pair programming. In *20th Conference on Software Engineering Education & Training (CSEET'07)* (pp. 115-122). IEEE.
- Cao, L. & Xu, P. (2005). Activity patterns of pair programming. In *HICSS '05: Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05)*, 88a. Los Alamitos, CA: IEEE Computer Society.
- Chang, B., Thorpe, H., & Lubke, M. (1984). LD students tackle the LOGO language: Strategies and implications. *Journal of Learning Disabilities, 17*(5), 303-304.
- Code.org (2017). *Hour of Code*. Retrieved August 26, 2017 from <https://hourofcode.com/us>
- coding. (n.d.). *Dictionary.com Unabridged*. Retrieved October 20, 2016 from Dictionary.com website <http://www.dictionary.com/browse/coding>
- Cooper, G. & Sweller, J. (1987). Effects of schema acquisition and rule automation on mathematical problem-solving transfer. *Journal of Educational Psychology, 79*(4), 347-362. doi:10.1037/0022-0663.79.4.347
- Cramer, K. & Henry, A. (2002). Using manipulative models to build number sense for addition and fractions. In B. Litwiller (Ed.), *Making Sense of Fractions, Ratios, and Proportions* (pp. 41-48). Reston, VA: The National Council of Teachers of Mathematics.



- Creswell, J. W. & Miller, D. L. (2000). Determining Validity in Qualitative Inquiry. *Theory Into Practice*, 39(3), 124.
- Dekhane, S., Xu, X., & Tsoi, M. Y. (2013). Mobile app development to increase student engagement and problem solving skills. *Journal of Information Systems Education*, 24(4), 299.
- Dekker, R. & Elshout-Mohr, M. (2004). Teacher interventions aimed at mathematical level raising during collaborative learning. *Educational Studies in Mathematics*, 56(1), 39-65. doi:10.1023/B:EDUC.0000028402.10122.ff
- Denner, J., Werner, L., Campe, S., & Ortiz, E. (2014). Pair Programming: Under what conditions is it advantageous for middle school students? *Journal of Research on Technology in Education (Routledge)*, 46(3), 277-296.
- Dewey, J. (1938/1998). *Experience and education: The 60th anniversary edition*. Indianapolis, IN: Kappa Delta Pi Press.
- Ding, M., & Li, X. (2014). Facilitating and direct guidance in student-centered classrooms: Addressing “lines or pieces” difficulty. *Mathematics Education Research Journal*, 26(2), 353-376. doi:10.1007/s13394-013-0095-2
- Duval, R. (2006). A cognitive analysis of problems of comprehension in learning of mathematics. *Educational Studies in Mathematics*, 61, 103–131.
- Even, R. (1998). Factors involved in linking representations of functions. *The Journal of Mathematical Behavior*, 17(1), 105–121.
- Fest, A., Hiob, M., & Hoffkamp, A. (2011). An interactive learning activity for the formation of the concept of function based on representational transfer. *Electronic Journal of Mathematics & Technology*, 5(2), 169-176.
- Feurzeig, W. & Papert, S., with a preface by Bob Lawler (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments*, 19(5), 487-501. DOI: 10.1080/10494820903520040
- Fuchs, L. S., Fuchs, D., Prentice, K., Burch, M., Hamlett, C. L., Owen, R., Hosp, M., & Jancek, D. (2003). Explicitly teaching for transfer: Effects on third-grade students' mathematical problem solving. *Journal of Educational Psychology*, 95(2), 293-305. doi:10.1037/0022-0663.95.2.293
- Fuchs, L. S., Schumacher, R. F., Long, J., Namkung, J., Hamlett, C. L., Cirino, P. T., Jordan, N., Siegler, R., Gersten, R. & Changas, P. (2013). Improving at-risk

- learners' understanding of fractions. *Journal of Educational Psychology*, 105(3), 683-700.
- Gabriel, F., Coche, F., Szucs, D., Carette, V., Rey, B., & Content, A. (2012). Developing children's understanding of fractions: An intervention study. *Mind, Brain, and Education*, 6(3), 137-146.
- Geary, D. (2013). Learning disabilities in mathematics: Recent advances. In H. L. Swanson, K. Harris and S. Graham (Eds.), *Handbook of Learning Disabilities, second edition* (pp. 239 – 255). New York, NY: The Guilford Press.
- Georgio, I., Zahn, C., & Meira, B. J. (2008). A systematic framework for case-based classroom experiential learning. *Systems Research and Behavioral Science*, 25. DOI: 10.1002/sres.858
- Gersten, R., Chard, D., Jayanthi, M., Baker, S., Morphy, P., & Flojo, J. (2009). Mathematics instruction for students with learning disabilities: A meta-analysis of instructional components. *Review of Educational Research*, 79(3), 1202-1242.
- Glesne, C. (2011). *Becoming Qualitative Researchers: An Introduction (4<sup>th</sup> ed.)*. Boston, MA: Pearson Education, Inc.
- Godino, J. D., Batanero, C., Cañadas, G. R., & Contreras, J. M. (2017). Linking inquiry and transmission in teaching and learning mathematics and experimental sciences. *Acta Scientiae*, 18(4).
- Goldman, A. I. (1983). Epistemology and the theory of problem solving. *Synthese*, 55(1), 21-48.
- Greeno, J. G. & Hall, R. P. (1997). Practicing representation: Learning with and about representational forms. *Phi Delta Kappan*, 78, 361–367.
- Grover, S. & Pea, R. (2013). Computational Thinking in K—12: A Review of the State of the Field. *Educational Researcher*, 42(1), 38-43.
- Hackenberg, A. & Lee, M. (2015). Relationships between students' fractional knowledge and equation writing. *Journal for Research in Mathematics Education*, 46(2), 196-243.
- Han, S., Caparo, R., & Caparo, M. (2015). How science, technology, engineering, and mathematics (STEM) project-based learning (PBL) affects high, middle, and low achievers differently: The impact of student factors on achievement. *International Journal of Science and Mathematics Education*, 13.

- Hanks, B., Fitzgerald, S., McCauley, R., Murphy, L., & Zander, C. (2011). Pair programming in education: A literature review. *Computer Science Education, 21*(2), 135-173. DOI: 10.1080/08993408.2011.579808
- Harel, I. & Papert, S. (1990). Software design as a learning environment. *Interactive Learning Environments, 1*, 1-32.
- Hegarty, M., & Kozhevnikov, M. (1999). Types of visual–spatial representations and mathematical problem solving. *Journal of Educational Psychology, 91*(4), 684.
- Herro, D., McCune-Gardner, C., & Boyer, D., (2015). Perceptions of coding with MIT App Inventor: Pathways for their future. *Journal for Computing Teachers, winter 2015*, p. 30.
- Hiebert, J. & Carpenter, T. (1992). Learning and teaching with understanding. In D. A. Grouws (Ed.), *Handbook of research on mathematics teaching and learning* (pp. 65–97). New York: Macmillan.
- Hsieh, H. & Shannon, S. (2005). Three approaches to qualitative content analysis. *Qualitative Health Research, 15*(9).
- Ioannidou, A., Repenning, A., Lewis, C., Cherry, G., & Rader, C. (2003). Making constructionism work in the classroom. *International Journal of Computers for Mathematical Learning, 8*(1), 63-108. doi:10.1023/A:1025617704695
- Israel, M., Marino, M. T., Basham, J. D., & Spivak, W. (2013). Fifth graders as app designers: How diverse learners conceptualize educational apps. *Journal of Research on Technology in Education, 46*(1), 53. doi:10.1080/15391523.2013.10782613
- Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education, 82*, 263-279.
- Italo (2017, February 28). Instantiate dynamic number of image objects. Message posted to [https://groups.google.com/forum/#!msg/mitappinventortest/cB\\_Ws1KObA4/TnVEuFTWAAAJ;context-place=forum/mitappinventortest](https://groups.google.com/forum/#!msg/mitappinventortest/cB_Ws1KObA4/TnVEuFTWAAAJ;context-place=forum/mitappinventortest)
- Jordan, C. L., Hansen, N., Fuchs, L., Siegler, R., Micklos, D., & Gersten, R. (2013). Developmental predictors of conceptual and procedural knowledge of fractions. *Journal of Experimental Child Psychology, 116*, 45–58.

- Jordan, N. C., Resnick, I., Rodrigues, J., Hansen, N., & Dyson, N. (2016). Delaware longitudinal study of fraction learning: Implications for helping children with mathematics difficulties. *Journal of Learning Disabilities*. doi: 0022219416662033.
- K-12 Computer Science Framework. (2016). Retrieved from <http://www.k12cs.org>.
- Kafai, Y. (1995). *Minds in Play: Computer Game Design as a Context for Children's Learning*. New Jersey: Lawrence Erlbaum Associates.
- Kafai, Y. (1995, April). Making game artifacts to facilitate rich and meaningful learning. Paper presented at the annual meeting of the *American Educational Research Association* annual conference, San Francisco, CA.
- Kafai, Y. (1996). Software by kids for kids. *Communications of the ACM*, 39(4), 38-39.
- Kafai, Y., Franke, M. L., Ching, C. C., & Shih, J. C. (1998). Game design as an interactive learning environment for fostering students' and teachers' mathematical inquiry. *International Journal of Computers for Mathematical Learning*, 3(2), 149-184. doi:10.1023/A:1009777905226
- Kahn, K., Sendova, E., Sacristán, A. I., & Noss, R. (2011). Young students exploring cardinality by constructing infinite processes. *Technology, Knowledge and Learning*, 16(1), 3-34.
- 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.
- Khalili, N., Sheridan, K., Williams, A., Clark, K., & Stegman, M. (2011). Students designing video games about immunology: Insights for science learning. *Computers in the Schools*, 28(3), 228. doi:10.1080/07380569.2011.594988
- Kieren, T. E. (1980). The rational number construct: Its elements and mechanisms. In T. E. Kieren (Ed.), *Recent Research on Number Learning* (pp. 125-149). Columbus: Ohio State University.
- Kim, S. A., Wang, P., & Michaels, C. A. (2015). Using explicit C-R-A instruction to teach fraction word problem solving to low-performing Asian-English learners. *Reading & Writing Quarterly*, 31(3), 253-278. doi:10.1080/10573569.2015.1030999
- Kirschner, P., Sweller, J., & Clark, R. (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, 75–86.
- Kolb, D. A. (1984). The process of experiential learning. *Experiential learning: Experience as the source of learning and development* (pp. 20-38). Englewood Cliffs, NJ: Prentice Hall.
- Kolb, D. A., Boyatzis, R. E., & Mainemelis, C. (2001). Experiential learning theory: Previous research and new directions. *Perspectives on Thinking, Learning, and Cognitive Styles*, 1, 227-247.
- Krueger, N. (2017, September 20). How can school leaders leverage computer science resources? [Blog post]. Retrieved September 20, 2017, from <https://www.iste.org/explore/articleDetail?articleid=1056&category=Computer-Science&article>
- Kuhn, D. (2007). Is direct instruction an answer to the right question? *Educational Psychologist*, 42(2), 109-113.
- Kuo, F.-R., Hwang, G.-J., Chen, S.-C., & Chen, S. Y. (2012). A Cognitive Apprenticeship Approach to Facilitating Web-based Collaborative Problem Solving. *Educational Technology & Society*, 15(4), 319–331.
- Kyllonen, P. C. & Christal, R. E. (1990). Reasoning ability is (little more than) working-memory capacity?!. *Intelligence*, 14(4), 389-433.
- Laursen, K. W. (1978). Errors in first-year algebra. *Mathematics Teacher*, 71(3), 194–195.
- Leonard, M. & McKnight, M. (2015). Look and tell: Using photo-elicitation methods with teenagers. *Children's Geographies*, 13(6), 629-14. doi:10.1080/14733285.2014.887812
- Lesh, R., Post, T., & Behr, M. (1987). Representations and translations among representations in mathematics learning and problem solving. In C. Janvier (Ed.), *Problems of representation in the teaching and learning of mathematics* (pp. 33–40). Hillsdale, NJ: Lawrence Erlbaum Associates.
- Likourezos, V., & Kalyuga, S. (2017). Instruction-first and problem-solving-first approaches: Alternative pathways to learning complex tasks. *Instructional Science*, 45(2), 195-219. doi:10.1007/s11251-016-9399-4
- Lyon, G. R. and Weiser, B. (2013). The state of science in learning disabilities: Research impact on the field from 2001 to 2011. In H. L. Swanson, K. Harris and S. Graham

- (Eds.), *Handbook of Learning Disabilities, second edition* (pp. 118 – 154). New York, NY: The Guilford Press.
- Macnamara, J. R. (2005). Media content analysis: Its uses, benefits and best practice methodology. *Asia-Pacific Public Relations Journal*, 6(1), 1.
- Maio, P. (2016, August 23). New computer science course's challenge is finding qualified teachers to teach it. *EdSource*. Retrieved September 23, 2017, from <https://edsource.org/2016/new-computer-science-courses-challenge-is-finding-qualified-teachers-to-teach-it/568081>
- [MIT] Massachusetts Institute of Technology. (2017). *App Inventor*. Retrieved April 12, 2017, from <http://AppInventor.mit.edu/explore/>
- Matsuo, M. (2015). A framework for facilitating experiential learning. *Human Resource Development Review*, 14(4), 442-461. doi:10.1177/1534484315598087
- Mazzocco, M., Myers, G., Lewis, K., Hanich, L. & Murphy, M. (2013). Limited knowledge of fraction representations differentiates middle school students with mathematics learning disability (dyscalculia) versus low mathematics achievement. *Journal of Experimental Child Psychology*, 115, 371-387.
- McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2003). The impact of pair programming on student performance, perception and persistence. In *Proceedings of the International Conference on Software Engineering (ICSE 2003)*, 602–607. Washington, DC: IEEE Computer Society.
- Mehta, H. (2013, February 21). Schools encouraged to teach computer science, coding. [Blog post]. *Faronics Blog*. Retrieved September 23, 2017, from <http://www.faronics.com/news/blog/schools-encouraged-to-teach-kids-computer-science-coding/>
- Moscardini, L. (2010). "I like it instead of maths": How pupils with moderate learning difficulties in scottish primary special schools intuitively solved mathematical word problems. *British Journal of Special Education*, 37(3), 130.
- Mou, Y., Li, Y., Hoard, M. K., Nugent, L. D., Chu, F. W., Rouder, J. N., & Geary, D. C. (2016). Developmental foundations of children's fraction magnitude knowledge. *Cognitive Development*, 39, 141-153.
- Mousoulides, N. G., Christou, C., & Sriraman, B. (2008). A modeling perspective on the teaching and learning of mathematical problem solving. *Mathematical Thinking and Learning: An International Journal*, 10(3), 293-304.

- National Center for Education Statistics. (2015). *The Nation's Report Card*. Retrieved July 15, 2016, from <http://nces.ed.gov/nationsreportcard/>.
- [NCTM] National Council of Teachers of Mathematics. (2000). *Principles and standards for school mathematics*. Reston, VA: Author.
- [NGA] National Governors Association Center for Best Practices, & Council of Chief State School Officers. (2010). *Common Core State Standards*. Washington D.C.: National Governors Association Center for Best Practices, Council of Chief State School Officers.
- Nicholls, J. G. (1979). Quality and equality in intellectual development: The role of motivation in education. *American Psychologist*, Retrieved from <http://search.ebscohost.com/login.aspx?direct=true&db=eric&AN=EJ218804>
- Niemi, D. (1996). Assessing conceptual understanding in mathematics: Representations, problem solutions, justifications, and explanations. *The Journal of Educational Research*, 89(6), 351-363.
- [NIFDI] National Institute for Direct Instruction (2015). DI vs. di: The term “direct instruction.” Retrieved September 21, 2017, from <https://www.nifdi.org/what-is-di/di-vs-di>.
- Nosek, J.T. (1998). The case for collaborative programming. *Communications of the ACM*, 41, 105–108.
- Orihuela, Y. R. (2006). *Algebra I and other predictors of high school dropout* (Order No. 3249717). Available from ProQuest Dissertations & Theses Global. (304924276). Retrieved from <http://search.proquest.com.libproxy.clemson.edu/docview/304924276?accountid=6167>
- Panaoura, A., Gagatsis, A., Deliyianni, E., & Elia, I. (2009) The structure of students’ beliefs about the use of representations and their performance on the learning of fractions. *Educational Psychology*, 29(6), 713-728.
- Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books.
- Papert, S. (1987). Computer criticism vs. technocentric thinking. *Educational Researcher*, 16(1), 22-30.
- Patton, M. (1999). Enhancing the quality and credibility of qualitative analysis. *Health Services Research*, 34(5).

- Peng, P., & Fuchs, D. (2017). A randomized control trial of working memory training with and without strategy instruction: Effects on young children's working memory and comprehension. *Journal of Learning Disabilities, 50*(1), 62-80.  
doi:10.1177/0022219415594609
- Peppler, K. A. & Kafai, Y. B. (2007). From SuperGoo to scratch: Exploring creative digital media production in informal learning. *Learning, Media & Technology, 32*(2), 149-166. doi:10.1080/17439880701343337
- Prensky, M. (2008). Students as designers and creators of educational computer games: Who else? *British Journal of Educational Technology, 39*(6), 1004-1019.
- programming. (n.d.). *Dictionary.com Unabridged*. Retrieved October 20, 2016 from Dictionary.com website <http://www.dictionary.com/browse/programming>
- Prottzman, K. (2015, April 12). Coding vs. programming – Battle of the terms! *The Huffington Post*. Retrieved October 20, 2016 from [http://www.huffingtonpost.com/kiki-prottzman/coding-vs-programming-bat\\_b\\_7042816.html](http://www.huffingtonpost.com/kiki-prottzman/coding-vs-programming-bat_b_7042816.html)
- Psycharis, G. & Kynigod, C. (2011). Normalising geometrical figures: Dynamic manipulation and construction of meanings for ratio and proportion. *Research in Mathematics Education, 11*(2), 149-166.
- Putnam, P. T., Lampert, M., & Peterson, P. L. (1990). Alternative perspectives on knowing mathematics in elementary schools. In C. B. Cazden (Ed.), *Review of research in education* (Vol. 16, pp. 57–150). Washington, DC: American Educational Research Association.
- Rabin, M. & Schrag, J. L. (1999). First impressions matter: A model of confirmatory bias. *The Quarterly Journal of Economics, 114*(1), 37-82.
- Ratcliff, C. & Anderson, S. (2011). Reviving the turtle: Exploring the use of LOGO with students with mild disabilities. *Computers in the Schools, 28*, 241-255.
- Rau, M., Aleven, V., & Rummel, N. (2015). Successful learning with multiple graphical representations and self-explanation prompts. *Journal of Educational Psychology, 107*(1), 30-46.
- Rich, P., Bly, N., & Leatham, K. (2014). Beyond cognitive increase: Investigating the influence of computer programming on perception and application of mathematical skills. *Journal of Computers in Mathematics and Science Teaching, 33*(1), 103-128.



- Robertson, J. & Good, J. (2005). Children's narrative development through computer game authoring. *TechTrends: Linking Research & Practice to Improve Learning*, 49(5), 43-59.
- Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2), 137-172.
- Rotman, J. W. (1991). *Arithmetic: Prerequisite to Algebra?* Lansing, MI: Annual Convention of the American Mathematical Association of Two-Year Colleges.
- Rouse, M. (2016). Debugging. *TechTarget: Search Software Quality*. Retrieved August 26, 2017, from <http://searchsoftwarequality.techtarget.com/definition/debugging>
- Rowley, J. (2002). Using case studies in research. *Management research news*, 25(1), 16-27.
- Runeson, P., Höst, M., Rainer, A., & Regnell, B. (2012). Chapter 3: Design of the case study. *Case Study Research in Software Engineering: Guidelines and Examples*. Hoboken, NJ: John Wiley & Sons.
- Saldana, J. (2013). *The Coding Manual for Qualitative Researchers (2<sup>nd</sup> ed.)*. Sage Publications.
- Sanford, K. J., Hopper, T. F., & Starr, L. (2015). Transforming teacher education thinking: Complexity and relational ways of knowing. *Complicity: An International Journal of Complexity & Education*, 12(2), 26-48.
- Santi, G. & Baccaglini-Frank, A. (2015). Forms of generalization in students experiencing mathematical learning difficulties. *PNA*, 9(3), 217-243.
- Savignano, M., Williams, M. K., & Holbrook, J. (2014). Yes, your students can create games that land in the apple app store. *Learning & Leading with Technology*, 41(5), 26.
- Sciarra, D. T. (2010). Predictive factors in intensive math course-taking in high school. *13*, 196+.
- Selby, C. & Woollard, J. (2013). Computational thinking: The developing definition. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education, SIGCSE 2014*. ACM.
- Shenton, A. (2004). Strategies for ensuring trustworthiness in qualitative research projects. *Education for Information*, 22.

- Sharp, E., & Shih-Dennis, M. (2017). Model drawing strategy for fraction word problem solving of fourth-grade students with learning disabilities. *Remedial and Special Education, 38*(3), 181-192. doi:10.1177/0741932516678823
- Siegler, R. S., Duncan, G. J., Davis-Kean, P. E., Duckworth, K., Claessens, A., Engel, M., Susperreguy, M. & Chen, M. (2012). Early predictors of high school mathematics achievement. *Psychological Science, 23*, 691–697. doi:10.1177/0956797612440101
- Siegler, R. S., Fazio, L. K., Bailey, D. H., & Zhou, X. (2013). Fractions: the new frontier for theories of numerical development. *Trends In Cognitive Sciences, 17*(1), 13-19
- Siegler, R. S., Thompson, C. A., & Schneider, M. (2011). An integrated theory of whole number and fractions development. *Cognitive Psychology, 62*, 273–296.
- Simpson, A. & Quigley, C. F. (2016). Member checking process with adolescent students: Not just reading a transcript. *The Qualitative Report, 377+*.
- Skehill, K. (2013). Making sense of math: Changing perspectives on math through experiential learning. *Education Canada, 53*(5), 21.
- Sniegowski, S. (2017, February 27). Will this school year be when you learn to code? [Blog post]. *National Consortium of Secondary STEM Schools*. Retrieved September 23, 2017, from <http://ncsss.org/publications/ncsss-blog/item/22-will-this-school-year-be-when-you-learn-to-code>
- Snodgrass, M. R., Israel, M., & Reese, G. C. (2016). Instructional supports for students with disabilities in K-5 computing: Findings from a cross-case analysis. *Computers & Education, 100*, 1-17.
- Spangler, D. (2011). *Strategies for Teaching Fractions*. Thousand Oaks, CA: Corwin. 36-88.
- Spiro, R. J. (1988). Cognitive flexibility theory: Advanced knowledge acquisition in ill-structured domains. *Technical Report No. 441*.
- Stephens, A. C., Bottge, B. A., & Rueda, E. (2009). Ramping up on fractions. *Mathematics Teaching in the Middle School, 14*(9), 520-526.
- Stone, J. R., Alfeld, C., & Pearson, D. (2008). Rigor and relevance: Enhancing high school students' math skills through career and technical education. *American Educational Research Journal, 45*(3), 767-795. doi:10.3102/0002831208317460

- Swanson, H. L., Kehler, P., & Jerman, O. (2010). Working memory, strategy knowledge, and strategy instruction in children with reading disabilities. *Journal of Learning Disabilities, 43*(1), 24-47. doi:10.1177/0022219409338743
- Swanson, H. L., Orosco, M. J., & Lussier, C. M. (2014). The effects of mathematics strategy instruction for children with serious problem-solving difficulties. *Exceptional Children, 80*(2), 149-168. doi:10.1177/001440291408000202
- Swanson, H. L. & Zheng, X. (2013). Memory difficulties in children and adults with learning disabilities. In H. L. Swanson, K. Harris and S. Graham (Eds.), *Handbook of Learning Disabilities, second edition* (pp. 214 – 238). New York, NY: The Guilford Press.
- Tilford, M. P. (1979). Achievement in algebra II using computer programming. *SIGCUE Outlook, 13*(2), 9-14.
- Torre, D. & Murphy, J. (2015) A different lens: Changing perspectives using Photo Elicitation Interviews. *Education Policy Analysis Archives, 23*(111), <http://dx.doi.org/10.14507/epaa.v23.2051>
- Trusty, J., & Niles, S. G. (2004). Realized potential or lost talent: High school variables and bachelor's degree completion. *The Career Development Quarterly, 53*(1), 2-15. doi:10.1002/j.2161-0045.2004.tb00651.x
- Turkle, S. & Papert, S. (1990). Epistemological pluralism: Styles and voices within the computer culture. *Signs, 16*(1), 128-157.
- United States Department of Education. (2008). *Foundations of success: The final report of the National Mathematics Advisory Panel*. Washington, DC: Author.
- United States Department of Education (2017). *Teacher Shortage Areas: Nationwide Listing 1990-1991 through 2017-2018*. Washington, DC: Author.
- United States Department of Labor, Bureau of Labor Statistics. (2015). College enrollment and work activity of 2014 high school graduates [Press release]. Retrieved from <http://www.bls.gov/news.release/hsgec.nr0.htm>.
- Vamvakoussi, X. & Vosniadou, S. (2004). Understanding the structure of the set of rational numbers: A conceptual change approach. *Learning and Instruction, 14*, 453–467.

- Van de Grift, T. (2004). Coupling pair programming and writing: Learning about students' perceptions and processes. In *Proceedings of the Thirty-fifth SIGCSE Technical Symposium on Computer Science Education*, 2-6. New York, NY: ACM.
- Van Dooren, W., De Bock, D., Hessels, A., Janssens, D., & Verschaffel, L. (2004). Remediating secondary school students' illusion of linearity: A teaching experiment aiming at conceptual change. *Learning & Instruction*, 14(5), 485-501. doi:10.1016/j.learninstruc.2004.06.019
- Wilhelm, J., She, X., & Morrison, D. C. (2011). Differences in math and science understanding between NSF GK-12 participant groups: A year long study. *Journal of STEM Education: Innovations & Research*, 12(1), 55-68. Retrieved from <http://search.ebscohost.com/login.aspx?direct=true&db=aph&AN=67407996>
- Williams, S. (2014). Brokering instructional improvement through response to intervention. *Journal of School Public Relations*, 35(2), 271-297.
- Wing, J. M. (2006). *Computational thinking*. New York: ACM. doi:10.1145/1118178.1118215
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717-3725. doi:10.1098/rsta.2008.0118
- Whittemore, R., Chase, S. K., & Mandle, C. L. (2001). Validity in qualitative research. *Qualitative Health Research*, 11(4), 522-537. doi:10.1177/104973201129119299
- Wolz, U., Stone, M., Pearson, K., Pulimood, S. M., & Switzer, M. (2011). Computational thinking and expository writing in the middle school. *ACM Transactions on Computing Education*, 11(2).
- Wu, H. (2001). How to prepare students for algebra. *American Educator*, 25(2), 10-17.
- Xin, Y. P., Liu, J., Jones, S. R., Tzur, R., & Si, L. (2016). A preliminary discourse analysis of constructivist-oriented mathematics instruction for a student with learning disabilities. *The Journal of Educational Research*, 109(4), 436-447. doi:10.1080/00220671.2014.979910
- Yang, Y. C. & Chang, C. (2013). Empowering students through digital game authorship: Enhancing concentration, critical thinking, and academic achievement. *Computers & Education*, 68, 334-344. doi:10.1016/j.compedu.2013.05.023
- Yin, R. (2014). *Case Study Research: Design and Methods (5<sup>th</sup> ed.)*. Sage Publications.

- Zhang, X. (2012). Enriching fifth-graders' concept images and understandings of unit fractions. Illinois State University, IL: Unpublished PhD dissertation.
- Zhang, X., Clements, M. A., & Ellerton, N. (2015). Conceptual mis(understandings) of fractions: From area models to multiple embodiments. *Mathematics Education Research Journal*, 27, 233-261.
- Zientek, L. R., Younes, R., Nimon, K., Mittag, K. C., & Taylor, S. (2013). Fractions as a Foundation for Algebra within a Sample of Prospective Teachers. *Research in the Schools*, 20(1), 76-95.
- Zhong, N., Wang, Y., & Chiew, V. (2010). On the cognitive process of human problem solving. *Cognitive Systems Research*, 11, 81-92.
- Zhu, N. (2015). Cognitive strategy instruction for mathematical word problem-solving of students with mathematics disabilities in china. *International Journal of Disability, Development and Education*, 62(6), 608-627.  
doi:10.1080/1034912X.2015.1077935

## APPENDICES

**Diagnostic Test  
Fraction Concepts**

Multiple Choice: Circle the correct answer. If your answer is not given, circle "Not here."

**Part 1**

1. How much of the circle is shaded?



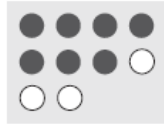
- A.  $\frac{3}{5}$       B.  $\frac{2}{3}$   
 C.  $\frac{2}{5}$       D.  $\frac{5}{2}$   
 E. Not here

2. How much of the rectangle is *not* shaded?



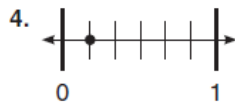
- A.  $\frac{3}{7}$       B.  $\frac{4}{7}$   
 C.  $\frac{4}{3}$       D. 4  
 E. Not here

3. What part of the group of circles is shaded?



- A.  $\frac{7}{10}$       B.  $\frac{7}{3}$       C.  $\frac{10}{7}$       D. 7      E. Not here

In items 4 and 5, which number names the point shown on the number line?

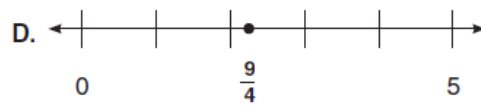
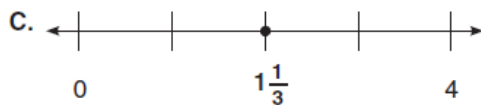
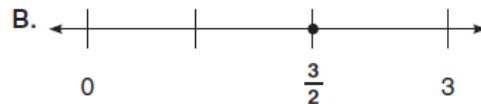


- A.  $\frac{1}{5}$       B.  $\frac{1}{6}$       C.  $\frac{1}{7}$       D.  $\frac{2}{7}$   
 E. Not here



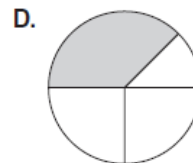
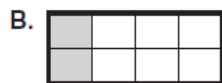
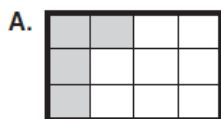
- A.  $\frac{3}{4}$       B.  $\frac{7}{8}$       C.  $1\frac{7}{8}$       D.  $1\frac{3}{4}$   
 E. Not here

6. On which number line is the point correctly named?



- E. Not here

7. Which diagram shows  $\frac{1}{4}$  shaded?



E. Not here

8. Which tells about how much is shaded?



A. less than  $\frac{1}{4}$

B. about  $\frac{1}{2}$

C. about  $\frac{2}{3}$

D. more than  $\frac{3}{4}$

E. Not here

9. Which is correct?

A.  $\frac{2}{3} > \frac{3}{4}$

B.  $2\frac{1}{2} < 1\frac{9}{10}$

C.  $\frac{12}{18} < \frac{5}{6}$

D.  $\frac{3}{5} > \frac{5}{8}$

E. Not here

10. Which of these numbers is the greatest, or are they all equal?

$$2\frac{2}{6}, 2\frac{1}{3}, \frac{7}{3}$$

A.  $2\frac{2}{6}$

B.  $2\frac{1}{3}$

C.  $\frac{7}{3}$

D. They are all equal.

11. Which is correct?

A.  $\frac{2}{5} > \frac{2}{4} > \frac{2}{3}$

B.  $\frac{11}{12} < \frac{7}{12} < \frac{5}{12}$

C.  $\frac{1}{2} > \frac{1}{3} > \frac{1}{4}$

D.  $\frac{6}{10} < \frac{9}{20} < \frac{1}{5}$

E. Not here

12. Which means the same as  $\frac{3}{4}$ ?

A.  $3 \times 4$

B.  $4 \div 3$

C.  $3 \div 4$

D. 3.4

E. Not here



## Appendix B: Scoring the Instrument

### *Answer choices and their types of errors*

<u>Question</u>	<u>Choice A</u>	<u>Choice B</u>	<u>Choice C</u>	<u>Choice D</u>
1	Careless	Part/Whole	Correct	Part/Whole
2	Careless	Correct	Part/Whole	Part/Whole
3	Correct	Part/Whole	Part/Whole	Part/Whole
4	Number Line	Correct	Part/Whole	Number Line
5	Careless	Part/Whole	Part/Whole	Correct
6	Careless	Part/Whole	Part/Whole	Correct
7	Arithmetic	Correct	Part/Whole	Part/Whole
8	Careless	Part/Whole	Careless	Correct
9	Representation	Arithmetic	Correct	Representation
10	Arithmetic	Representation	Representation	Correct
11	Part/Whole	Careless	Correct	Representation
12	Representation	Representation	Correct	Representation

### *Scoring*

<u>Error Type</u>	<u>Points</u>	<u>Description</u>	<u>Example</u>
Correct	2	Student answered the question correctly.	
Careless	2	Student may have misread the problem.	Student chose the fraction that represented the shaded portion when the question asked for the unshaded portion.
Number Line	1	Student did not read the number line correctly but, based on how he/she did interpret the line, chose a viable fraction.	Student counted the tick marks on the line and used that value as the denominator.
Arithmetic	1	Student did not do the required arithmetic correctly or read the inequality wrong.	Student simplified a fraction wrong.
Part/Whole	0	Student does not recognize a fraction as representing a part of a whole.	Student chose an answer with the numerator and denominator reversed.
Representation	0	Student was unable to create a representation of a fraction or use benchmark fractions to answer the question.	Student could not accurately compare two fractions with different denominators.

## Appendix C: Permission to Use Instrument



Lorraine Jacques <lorraj@g.clemson.edu>

---

### seeking permission to republish

4 messages

---

Lorraine Jacques <lorraj@g.clemson.edu> Tue, Jan 24, 2017 at 9:57 AM To: info@corwin.com

Good morning,

I am a PhD candidate at Clemson University who would like to use an assessment from one of your books in my dissertation. Please inform me what I should do to obtain permission.

The assessment in question is "Fraction Concepts" from the following:  
Spangler, D. (2011). *Strategies for Teaching Fractions*. Thousand Oaks, CA: Corwin. 36-88.  
Please note that I intend to use a shortened version of this item, but will not make changes to any of the wording or images.

Thank you for your help,

Lorraine Jacques  
PhD Candidate, Learning Sciences  
Eugene T. Moore  
College of Education  
Clemson University

---

permissions (US) <permissions@sagepub.com>  
To: "lorraj@g.clemson.edu" <lorraj@g.clemson.edu>

Wed, Jan 25, 2017 at 2:42 PM

Dear Lorraine Jacques,

Thank you for your request. In order to proceed, you will need to tell us how much material you are requesting to use. Are you requesting to use multiple pages from the book (36-88) or are you requesting to use one page or excerpt.

If you are requesting to use pages 36-88, please clarify how you will be using that much material. Once we you provide clarification, we can further review your request.

Best regards,

Michelle Binur

Contract Administrator

SAGE Publishing

2455 Teller Road

Thousand Oaks, CA 91320

USA

[www.sagepublishing.com](http://www.sagepublishing.com)

Los Angeles | London | New Delhi

Singapore | Washington DC | Melbourne

---

Lorraine Jacques <lorraj@g.clemson.edu> Wed, Jan 25, 2017 at 3:05 PM To: "permissions (US)" <permissions@sagepub.com> Good afternoon,

I plan on using the diagnostic test from pages 36-38, minus a few questions, which I have attached here for your review. I will be referencing the remainder of that chapter (up to page 88) in my dissertation when I explain how I assess students' understanding of fraction concepts before and after an intervention addressing fraction magnitude. Specifically, I will be using the error analysis descriptions in that chapter to identify students' needs.

Please let me know if you would like more detailed information or anything further from me. And thank you for your time!

Lorraine

[Quoted text hidden]



fraction

---

permissions (US) <permissions@sagepub.com>  
To: Lorraine Jacques <lorraj@g.clemson.edu>

Thu, Jan 26, 2017 at 2:09 PM

Dear Lorraine,

Thank you for that information. You can consider this email as permission to use the material as detailed below in your upcoming dissertation. Please note that this permission does not cover any 3<sup>rd</sup> party material that may be found within the work. You must properly credit the original source, Strategies for Teaching Fractions. Please contact us for any further usage of the material.

Best regards,

Michelle Binur

## Appendix D: Math Games

<u>Name</u>	<u>Location</u>	<u>Topic</u>	<u>Genre</u>
Sum Shapes	<a href="http://www.mathplayground.com/sum_shapes.html">http://www.mathplayground.com/sum_shapes.html</a>	Addition	Puzzle
Factor Fracture	<a href="http://www.funbrain.com/brain/games/factor-fracture/index.html#game">http://www.funbrain.com/brain/games/factor-fracture/index.html#game</a>	Factoring	Action
Theme Hotel	<a href="http://www.hoodamath.com/games/themehotel.html">http://www.hoodamath.com/games/themehotel.html</a>	Money	Sandbox
Mather-piece	<a href="http://mrnussbaum.com/matherpiece/">http://mrnussbaum.com/matherpiece/</a>	Arithmetic	Quiz
Black Order of Operations	<a href="http://www.xpmath.com/forums/arcade.php?do=play&amp;gameid=100">http://www.xpmath.com/forums/arcade.php?do=play&amp;gameid=100</a>	Arithmetic	Quiz
Place Value Game	<a href="http://education.jlab.org/placevalue/gamepage.html">http://education.jlab.org/placevalue/gamepage.html</a>	Place Value	Puzzle
Integers in Space	<a href="http://www.mathwarehouse.com/games/our-games/arithmetric-games/integers-in-space/">http://www.mathwarehouse.com/games/our-games/arithmetric-games/integers-in-space/</a>	Ordering	Action
Connect 10	<a href="http://www.mindgames.com/game/Connect+10">http://www.mindgames.com/game/Connect+10</a>	Addition	Puzzle
Math Tower Defense	<a href="http://www.mathnook.com/math/mathtowerdefense.html">http://www.mathnook.com/math/mathtowerdefense.html</a>	Arithmetic	Action

Appendix E: Student Game Analysis Sheet

Name:

<b>Game</b>	<b>What parts of this game did you like?</b>	<b>What parts of this game did you <u>not</u> like?</b>

## Appendix F: Resources for Fraction Assistance

<u>Resource provided to students</u>	<u>How often used</u>
Cavanagh, M. (2006). <i>Math to Know: A Mathematics Handbook</i> . Wilmington, MA: Great Source Education Group.	7
Charles, R., Caldwell, J., Cavanagh, M., Copley, J., Crown, W., Fennell, F., Murphy, S., Sammons, K., Schielack, J., & Tate, W. (2012). <i>enVision Math: Common Core, Grade 3</i> . Upper Saddle River, NJ: Pearson Education, Inc.	3
Treff, A. & Jacobs, D. (2003). <i>Basic Math Skills</i> . Circle Pines, MN: AGS Publishing.	3
University of Chicago Mathematics Project (2012). <i>Everyday Mathematics: Student Reference Book</i> . Chicago, IL: McGraw-Hill Education.	5
Khan Academy ( <a href="https://www.khanacademy.org/math/arithmetic-home/arithmetic/fraction-arithmetic">https://www.khanacademy.org/math/arithmetic-home/arithmetic/fraction-arithmetic</a> )	0
PurpleMath ( <a href="https://www.purplemath.com/modules/index.htm">https://www.purplemath.com/modules/index.htm</a> )	1
Help with Fractions ( <a href="http://www.helpwithfractions.com/">http://www.helpwithfractions.com/</a> )	1
Math Goodies ( <a href="http://www.mathgoodies.com/lessons/toc_unit14.html">http://www.mathgoodies.com/lessons/toc_unit14.html</a> )	0
Review of Fraction Concepts ( <a href="https://www.youtube.com/watch?v=7Wrde6iFVcA">https://www.youtube.com/watch?v=7Wrde6iFVcA</a> )	0
Math Is Fun ( <a href="https://www.mathsisfun.com/fractions.html">https://www.mathsisfun.com/fractions.html</a> )	1
Fraction circles (manipulative)	3
<u>Resources students found independently</u>	<u>How often used</u>
Google search for “fractions”	8
Google search for “comparing fractions”	1

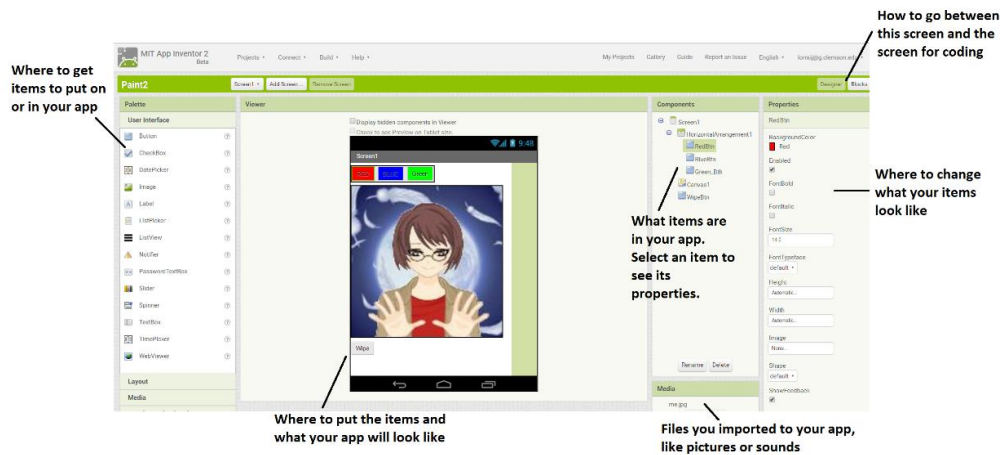
## Appendix G: Student Reference Guide for *App Inventor*



# Cheat Sheet for Making Your Game!

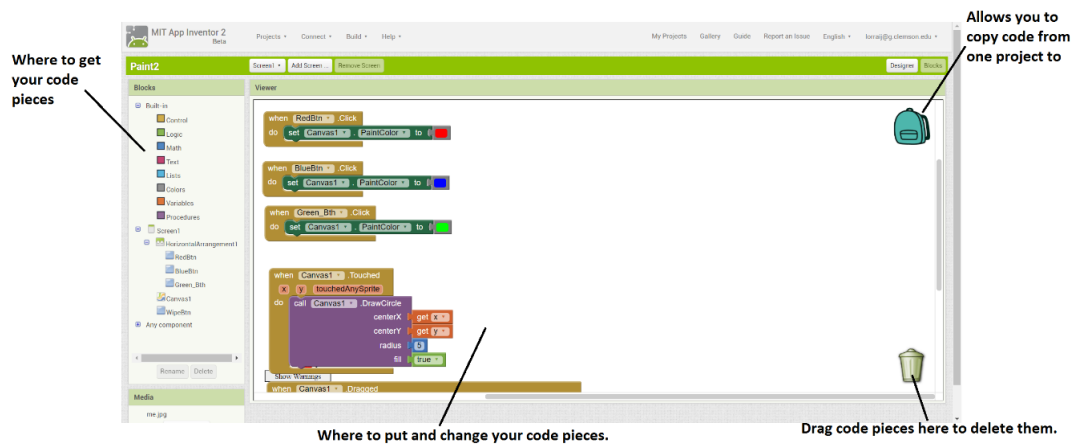
## The Designer Screen

This screen is where you will create the “look” of your app.



## The Blocks Screen

This screen is where you will create your code.



## A Few Common Code Blocks for Games

Clicking buttons:

```
when Button1 .Click
do call Player1 .Start
```

Flinging an image:

```
when Ball1 .Flung
do
  set Ball1 .Speed to get speed
  set Ball1 .Heading to get heading
```

Bouncing off the edge:

```
when Ball1 .EdgeReached
edge
do call Ball1 .Bounce
  edge get edge
```

Moving (dragging) an image:

```
when RocketSprite .Dragged
startX startY prevX prevY currentX currentY
do set RocketSprite .X to get currentX
```

Reacting to collisions:

```
when Bullet .CollidedWith
other
do set Bullet .Visible to false
  set ScoreLabel .Text to ScoreLabel .Text + 1
```

Random choice from a list: (in the List section)

```
set Label2 .Text to pick a random item list
make a list
  It is certain
  Without a doubt
  As I see it, yes
  Ask again later
  Reply hazy try again
  Don't count on it
  My sources say no
  Outlook not so good
```

Random number: (in the Math section)

```
x random integer from Hole . Radius
to Canvas1 . Width - Hole . Radius
y random integer from Hole . Radius to 75
```

## More Items Help

<http://ai2.appinventor.mit.edu/reference/components/>

*Another name for "items"*

Components App Inventor for Android

Component Reference

Component Types

This document describes the components you can use in App Inventor to build your apps.

Each component can have methods, events, and properties. Most properties can be changed by apps — these properties have blocks you can use to get and set the values. Some properties can't be changed by apps — these only have blocks you can use to get the values, not set them. In this document, read-only properties are shown in *italics*. A few properties are only available in the Designer.

- User interface components
- Layout components
- Media components
- Drawing and Animation components
- Sensor components
- Social components
- Storage components
- Connectivity components

*The groups you got your items from. Select the group to find the item.*

Each item will show two things: **properties** and **events**.

**Properties** are the way the item looks. You can change those in the Design screen or in the code.

**Events** are what the item does. You only use those in the code.







## Fractions and Other Math Help

**Remember:** If you look at any of these or any other math website, list it in your log!

- Khan Academy (<https://www.khanacademy.org/math/arithmetic-home/arithmetic/fraction-arithmetic>)
- PurpleMath (<https://www.purplemath.com/modules/index.htm>)
- Help with Fractions (<http://www.helpwithfractions.com/>)
- Math Goodies ([http://www.mathgoodies.com/lessons/toc\\_unit14.html](http://www.mathgoodies.com/lessons/toc_unit14.html))
- Review of Fraction Concepts (<https://www.youtube.com/watch?v=7Wrde6iFVcA>)
- Math Is Fun (<https://www.mathsisfun.com/fractions.html>)

*There are also books in the room that can help you!*

Appendix H: *App Inventor* Design Template (Herro, McCune-Gardner, & Boyer, 2015)

<p><b>Designing the App's Interface and Defining App Behavior</b></p> <p>App Page #1:</p>  <p>Page #1 Actions: When the NEXT button is clicked,</p> <p>When the PREVIOUS button is clicked,</p>	<p><b>Recognizing and Generalizing Patterns in Component Properties</b></p> <p>Page #1:</p> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Label Name: App_Title Text: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Image Name: Main_Picture Picture: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Label Name: Information Text: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Button Name: Previous_Button Text: "Previous"</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Button Name: Next_Button Text: "Next"</p> </div> <p>Page #2:</p> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Label Name: App_Title Text: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Image Name: Main_Picture Picture: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Label Name: Information Text: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Button Name: Previous_Button Text: "Previous"</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Button Name: Next_Button Text: "Next"</p> </div> <p>Page #3:</p> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Label Name: App_Title Text: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Image Name: Main_Picture Picture: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Label Name: Information Text: "</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Button Name: Previous_Button Text: "Previous"</p> </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> <p>Type: Button Name: Next_Button Text: "Next"</p> </div>	<p><b>Testing App Behavior and Algorithms</b></p> <p>Emulator</p>  <p><b>Designing Algorithms</b></p> <p>Block Space</p>
<p>APP INVENTOR - IDEA AND DESIGN SHEET</p>		
<p>App Page #2:</p>  <p>Page #2 Actions: When the NEXT button is clicked,</p> <p>When the PREVIOUS button is clicked,</p>		
<p>App Page #3:</p>  <p>Page #3 Actions: When the NEXT button is clicked,</p> <p>When the PREVIOUS button is clicked,</p>		

Appendix I: Coding Plan Template

Partners: \_\_\_\_\_

Object	Action	Date Completed	Notes
"Start" Button	Starts the game when pressed		
"Win" text	Appears when the player wins		
"Lose" text	Appears when the player loses		

## Appendix J: Strategies to Support Students Coding

<u>Issue</u>	<u>Teacher Action</u>
Students need help working with fractions.	Direct students to the resource list for fraction help.
Students need help transferring their fraction knowledge to their game's code.	Have students "act out" what they would like the computer to do (Chang, Thorpe, & Lubke, 1984; Ratcliff & Anderson, 2011). Help students plan on paper before coding (Chang, Thorpe, & Lubke, 1984; Santi & Baccaglioni-Frank, 2015).
Students need help with math other than fractions (i.e., coordinates).	Teachers will help students with the math.
Students lack confidence in coding.	Encourage students to try different things. Reassure them that they will not break anything (Israel et al., 2015).
Students need help developing an algorithm for a part of their game.	Help students plan on paper before coding (Chang, Thorpe, & Lubke, 1984; Santi & Baccaglioni-Frank, 2015). Encourage students to look at other classmates' code and ask questions (Ratcliff & Anderson, 2011).
Partners have trouble collaborating equitably.	Re-enforce the pair programming protocol: One partner decides what to do and the other finds and places the code to do the task (Braught, Wahls, & Eby, 2011).
Students need help with <i>App Inventor</i> . Students know they want an object to do something, but do not know where to start.	Teachers will help the student as needed. Teachers will help the students find similar actions in the tutorials and/or in the <i>App Inventor</i> library. Teachers will help the student transfer this knowledge to their own games (Snodgrass, Israel, & Reese, 2016).
Students are trying to code the entire game at once.	Students will be encouraged to code the action for one object at a time then test the code to see that the object behaves as intended.
Students are having difficulties debugging their code.	Students will be encouraged to "follow" the code on paper by writing down what happens with each line of code; teachers will likely need to demonstrate or assist

Students have working memory and visual-spatial deficits that are causing difficulties in coding the graphics.

students with this process a few times (Chang, Thorpe, & Lubke, 1984). Provide students with materials (e.g., graph paper) to model the graphics coding goals and stands to hold the models and reduce working memory strain.

## Appendix K: Student Log Template

Name: \_\_\_\_\_ Date: \_\_\_\_\_

1) Take a picture or a screen shot of everything you worked on today – paper design, your app, your code. Put those pictures in a folder.

2) List any books, websites, or things you used to learn something about fractions. (If you did not use any today, say “none.”)

3) Did you make any changes to your game design? If so, why?

4) What will you work on next time? (note to self)

Appendix L: Observation Protocol

Observer: \_\_\_\_\_

Date: \_\_\_\_\_

<u>Time</u>	<u>Who Involved</u>	<u>Observation/Quote</u>	<u>Notes</u>

## Appendix M: Interview Protocol

- I. Display the game that the subject created.
  - a. Tell me what's special or cool about your game.
  - b. How will your game help someone understand fractions?
  - c. Why did you decide to do it this way?
  
- II. Display the *App Inventor* code that the subject created.
  - a. Tell me about writing this code.
  - b. What part are you most proud of? <Allow subject to show as well as tell.>
  - c. <Bring up the section of code directly concerning fractions. If more than one section does this, do one at a time.>
    - i. How did you learn to do this?
    - ii. Was it hard or easy to do? Why?
      1. Was it hard to picture what you needed to do or could you picture it but couldn't find the code you needed?
  
- III. Display the game again.
  - a. What would you say about your game to convince someone to download it?
  - b. Does it do everything you hoped it would do? <If not, ask for details.>
  - c. What did you learn from doing this project? <If nothing about fractions is mentioned, follow with "What did you learn about fractions from doing this project?">
  - d. What challenges did you experience when making your game?
  - e. What was the best thing about doing this project?
  - f. What was the worst thing about doing this project?
  - g. Overall, how did this project compare with the other things you do in school?

Is there anything else you would like to tell me about your experience in this project?



## Appendix N: Challenges Other Than with Fractions (RQ3)

<u>Challenge</u>	<u>Support offered</u>	<u># Affected*</u>	<u>Avg. # Occurrences**</u>	<u>Data Sources</u>
Algorithm development	Helped students to plan on paper before coding (Chang, Thorpe, & Lubke, 1984; Santi & Baccaglioni-Frank, 2015) or encouraged them to look at other code and ask questions (Ratcliff & Anderson, 2011).	23	1.6	Audio recordings, student work, interviews
Debugging	Encouraged and helped students to “act out” the code on paper by writing down what happened with each line of code (Chang, Thorpe, & Lubke, 1984).	29	1.7	Audio recordings, field notes, code for the games, student work logs
Transfer	Prompted students to remember they had solved a similar problem.	23	1.3	Audio recordings, field notes
Angles in graphics	Activated prior knowledge of angles.	11	1	Audio recordings, student work
Design	<i>No support offered. Affected participants either changed their game design or did not complete their games.</i>	21	<i>This challenge affected most of the work participants did during each coding session.</i>	Game designs, coding plans, audio recordings, student work, field notes, games at end of study
Decomposition	Asked students to identify components in their designs on a coding plan.	19	2.3	Audio recordings, coding plans, interviews
Coding concepts or skills	Explicit instruction on the concept or skill.	18	2.5	Student work, audio recordings, field notes
<i>App Inventor</i> limitations	Provided students with directions or sample code to	31	2.9	Audio recordings, field notes,

	compensate for each limitation.			game designs, student work
Coding vocabulary	Defined the vocabulary and explained its use in coding.	15	1.1	Audio recordings, field notes, student work
Collaboration	Encouraged students to work together and seek peer support (Israel, Pearson, Tapia, Wherfel, & Reese, 2015); re-enforced the pair-programming protocol (Braught, Wahls, & Eby, 2011).	21	6.3	Audio recordings, field notes, interviews
Learned helplessness	Decomposed tasks into smaller ones and monitored their progress (Allsopp, Kyger, & Lovin, 2007).	9***	17	Audio recordings, field notes, student work

\* # Affected = Total # of participants in the groups experiencing challenge

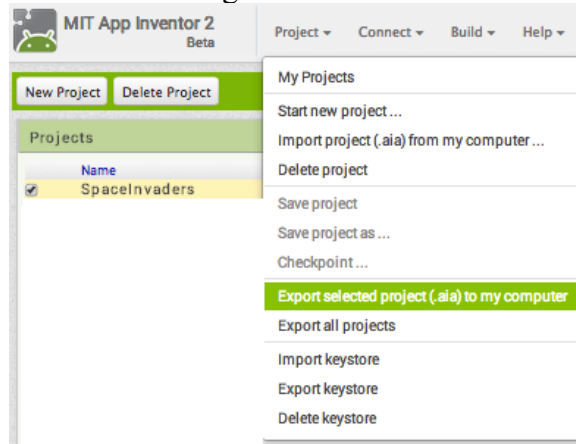
\*\* Avg. # Occurrences = Average number of times challenge presented per group during the study (data from the interviews were not used in this calculation)

\*\*\* This number does not reflect the number of participants displaying learned helplessness behaviors but the number of participants affected by at least one member of their group displaying such behaviors.

## Appendix O: Directions for Sharing Projects

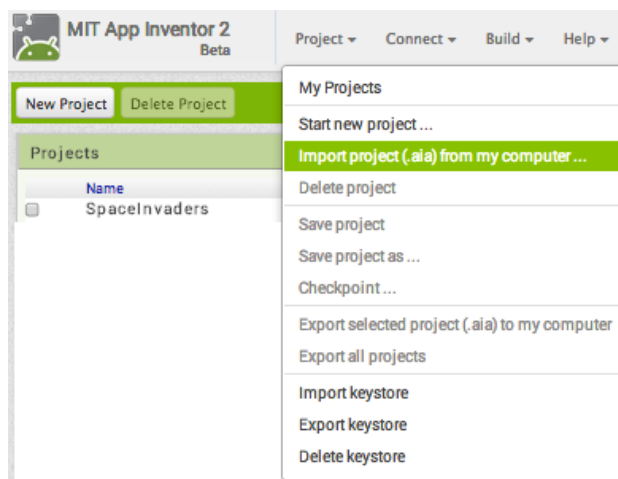
### At the End of EVERY Class

- 1) Go to “Projects” then “My Projects”
- 2) Select the checkbox next to your app
- 3) Go to “Projects” then “Export selected project (.aia) to my computer”
- 4) Put the downloaded file in our Google Drive folder



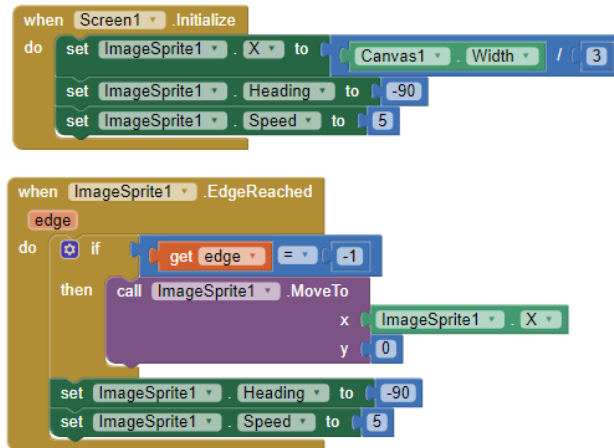
### If Your Partner is Absent and You Need the Code

- 1) Go to your shared Google Drive folder and download the newest copy of your code to your computer
- 2) In App Inventor, go to “Projects” then “My Projects”
- 3) Go to “Projects” then “Import project (.aia) from my computer”
- 4) Choose the .aia file you just downloaded (it should be in your Downloads folder) and hit OK.



## Appendix P: Sample Code Created for Students

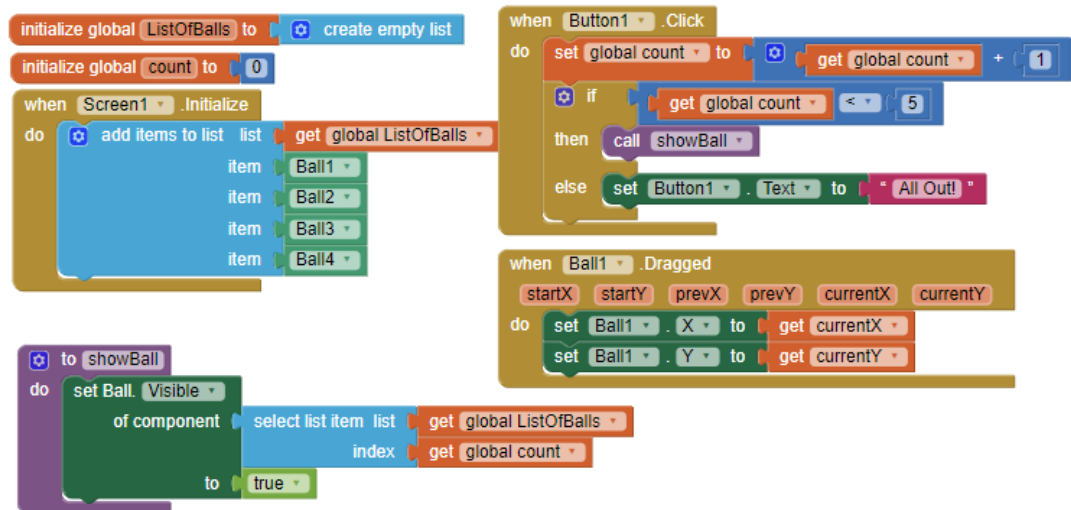
### Scrolling a Sprite



```
when Screen1.Initialize
do
  set ImageSprite1.X to Canvas1.Width / 3
  set ImageSprite1.Heading to -90
  set ImageSprite1.Speed to 5

when ImageSprite1.EdgeReached
edge
do
  if get edge = -1
  then
    call ImageSprite1.MoveTo
    x ImageSprite1.X
    y 0
  set ImageSprite1.Heading to -90
  set ImageSprite1.Speed to 5
```

### Creating Objects from a List



```
initialize global ListOfBalls to create empty list
initialize global count to 0

when Screen1.Initialize
do
  add items to list list get global ListOfBalls
  item Ball1
  item Ball2
  item Ball3
  item Ball4

when Button1.Click
do
  set global count to get global count + 1
  if get global count < 5
  then
    call showBall
  else
    set Button1.Text to "All Out!"

when Ball1.Dragged
startX startX prevX prevY currentX currentY
do
  set Ball1.X to get currentX
  set Ball1.Y to get currentY

to showBall
do
  set Ball.Visible of component select list item list get global ListOfBalls
  index get global count
  to true
```