6-28-2024

Benefits of Coding in the Elementary Classroom

Samantha Panek
paneks@csp.edu

Benefits of Coding in the Elementary Classroom: Capstone

Samantha Panek

Concordia University, St. Paul

ED 590: Conducting Research and Completing the Capstone

Dr. Robin Percival

Second Reader: Dr. Nora Flom

June 15, 2024

Table of Contents

Abstract

This capstone project focused on the potential benefits of coding, specifically for elementary-aged children. The literature review analyzed fifteen studies from around the world about benefits that could be gained from coding. Students coded with unplugged methods, block-based programs, and robots. Problem-solving was a common benefit found in research about elementary coding. Research also supported that specific content areas could be positively impacted through coding such as computational thinking and mathematical reasoning. Coding encouraged transferable life skills such as planning, collaboration, and creativity. (Arslan & Çelik, 2022; Murcia et al., 2020; O'Grady-Jones & Grant,2023;Wilson, 2020). Additionally, exposure to coding at a young age increased student interest in potentially lucrative STEM careers (Ogegbo & Aina, 2023; Ragusa & Leung, 2023). While more research was needed about which specific programs or resources had the greatest impact, all coding lessons led to increased skill-building for elementary students and would be a beneficial technology to increase student achievement.

Keywords: coding, computational thinking, elementary, problem-solving

Chapter One

This capstone delved into the benefits of coding for elementary students. According to the K-12 Computer Science Framework Steering Committee (2016), coding helped students gain computational skills and problem-solving skills necessary to "actively participate in a world increasingly influenced by technology" (para. 1). Technology was integral starting in the twenty-first century when the internet, smartphones, and robotics became prevalent in daily lives. Since the strong push for coding in schools in 2016, even more coding-dependent technology was developed including more advanced robotics, artificial intelligence, and virtual reality (Nordrum, 2024). Each new technology integrated into daily life solidified the importance of coding and the need for students to learn how to do it. Coding encompassed written programs for software, hardware, robotics, and other digital tools.

According to the U.S. Bureau of Labor Statistics (2023), jobs in computer and information technology were expected to "grow much faster than the average for all occupations from 2022 to 2032" (para. 1). In addition, about 377,500 openings in the computer and information technology field were predicted across the United State each year from 2022-2032 (U.S. Bureau of Labor Statistics, 2023). Jobs in the computer and information technology field included Computer and Information Research Scientists, Computer Programmers, Information Security Analysts, Web Developers, and Software Developers, among many others (U.S. Bureau of Labor Statistics, 2023). Coding was a necessary skill for all these occupations. Therefore, students who could code had many opportunities available for their future.

Importance of Topic

In 2016, President Obama implemented a K-12 initiative called Computer Science for All that recognized computer science as "a 'new basic skill' necessary for economic growth and social mobility" (Smith, 2016, para. 1). The initiative stated that coding prepares students for careers in technology and can also help students learn problem-solving skills such as breaking problems into smaller steps and prioritizing the steps needed to solve a problem (Smith, 2016). The goal of the initiative was to add coding into the K-12 curriculum. At the elementary level the expectation was "implementation of high-quality curriculum, instruction, and learning opportunities that promote computational thinking and that lay the groundwork for CS and STEM coursework in high school" (Smirth, 2016, para. 16). Coding could encompass robotics or programs such as Code.org (Smith, 2016). Coding recently became an essential job skill and was integrated in the technology field, healthcare, transportation, and financial industries (Smith, 2016). Additionally, coding jobs were lucrative. The median salary in May 2022 for jobs in Computer Science and Information technology was $100,530, compared to the median salary of all occupations at $46,310 (U.S. Bureau of Labor Statistics, 2023).Coding skills gave students many opportunities for different career paths and opportunities for economic success. If students did not receive coding in school, they would not have the skills to join the computer science workforce or to possibly consider a computer science degree.

Additionally, starting coding at a young age was linked to reduced gender stereotypes around STEM fields and increased interest in engineering careers (Bers & Sullivan, 2019). Since many people chose a career based on interest and confidence in their abilities, exposure to coding at a young age introduced students to the options available with coding proficiency. Students without coding lessons were unintentionally barred from a potentially lucrative career.

Scope of Research

The literature review discussed fifteen research studies about coding for elementary students. The studies were conducted in the United States, Italy, Spain, Finland, Australia, Taiwan,Turkey, and Switzerland. A benefit of a global scope was a broader perspective on a field with international importance and relevance. Students ages three to twelve were represented in the research studies since the focus of the research was elementary aged students. Rural, urban, and suburban schools were included in the research analyzed. The differing socioeconomic groups included in the study were intended for consideration of coding benefits across common barriers. Most studies occurred at a school, though a few were completed during extra-curricular programs. The literature reviewed spanned from 2019 to 2024.

The major themes explored in the review included that elementary coding programs developed a variety of student skills from very specific content skills to very broad, transferable skills. Then, the discussion ended with how exposure to coding extended beyond elementary school. The literature review began with the theme that elementary coding supported student problem-solving, since that was a common topic across coding studies (Egbert et al.; O'Grady-Jones & Grant, 2023; Sung et al., 2022; Woo & Falloon, 2022). Next, the research narrowed down to the specific skills of computational thinking and mathematical reasoning gained from coding (Fagerlund et al., 2020; Tran, 2019). After that, the focus broadened to skills students could use beyond coding and included planning, collaboration, and creativity (Arfé et al., 2019; Arslan & Çelik, 2022; Murcia et al., 2020; Somuncu & Aslan, 2022). Finally, the research discussed how students used coding beyond their K-12 education since exposure to coding led to increased interest in STEM careers (Ragusa & Leung, 2023). Throughout the themes, potential benefits of coding were discussed through multiple unique studies.

Definition of Terms

Due to the nature of the literature review, there were a few key terms to know.

- *Coding* refers to activities that promote computational thinking such as unplugged activities, block-based programming languages, programming games, and introductory robotic kits (Bers & Sullivan, 2019, p. 115)
- *Computational thinking* is a problem-solving strategy with 4 components – problem decomposition, abstraction, pattern recognition, and algorithm development (Woo & Falloon, 2022).

  Elementary, in this review, refers to students from preschool (ages 3-4) to sixth grade.
- *Problem-solving* is The ability to achieve a goal while overcoming an obstacle, such as through tinkering and troubleshooting (O'Grady-Jones & Grant, 2023).

Conclusion

In summary, the purpose of the literature review was to examine the benefits of teaching coding at the elementary level. As of June 2024, there was not a standardized method to teach coding across elementary schools. Some schools started some exposure to coding as early as kindergarten (Somuncu & Aslan, 2022), while others have little to no exposure to coding until fourth grade or beyond (Kim et al., 2021). Coding at the elementary level also took many different forms, including "unplugged activities, block-based programming languages, programming games, and introductory robotic kits" (Bers & Sullivan, 2019, p. 115). An example of block-based coding was the program Scratch (O'Grady-Jones & Grant, 2023). Schools have used robotics kits such as Bee-bots (Murcia et al., 2020) or Sphero SPRK+ (Kim, et al., 2021). Since there was not a standardized method for teaching coding, it could have been up for debate whether coding was necessary to add to the elementary curriculum. However, since coding was seen as important to add into elementary schools for a prepared future workforce since the Computer Science for All Initiative of 2016, studies were analyzed about impacts of coding.

The research questions that guided this literature review were what positive impacts were found through coding for elementary students, and in light of what we know about how children learn and education policy and practice, how shall educators best utilize technology to enhance student achievement? The major themes discussed in the review were schools that have used coding at the elementary level have seen the development of their students' problem-solving, creativity, collaboration, planning, and mathematical reasoning skills (Arfé et al., 2019; Arslan & Çelik, 2022; Bers, et al., 2019; Murcia et al., 2020; O'Grady-Jones & Grant, 2023; Wilson, 2020). Additionally, coding led to an increase in interest in STEM careers (Ragusa & Leung, 2023) In the next chapter, 15 studies about coding in elementary schools were explored and analyzed in relation to the research question.

Chapter Two

The literature review focused on 15 peer-reviewed articles related to the benefits of coding for elementary students and how coding was an effective technology for increased student achievement. The studies occurred in many different countries around the world between 2019-2024. The research questions that guided this literature review were what positive impacts were found through coding for elementary students, and in light of what we know about how children learn and education policy and practice, how shall educators best utilize technology to enhance student achievement? The literature review focused on the benefits of coding through several themes. For example, some themes related to specific content areas, such as mathematics and computational thinking developed through coding. Then, the scope was broadened to themes that fell into life skills students could use across multiple disciplines including planning skills, collaboration, and creativity. Finally, the literature review explored the theme of how early exposure to coding was linked to more interest in the STEM field. The first theme discussed was one that occurred frequently throughout studies on coding for elementary students; coding developed student problem-solving skills.

Problem-Solving

The first theme in the exploration of the benefits of coding was that elementary students developed problem-solving skills through coding. Several researchers cited problem-solving as a skill coding developed and there was a broad well of research about the topic (Egbert et al., 2021; O'Grady-Jones & Grant, 2023; Sung, et al., 2022 Woo & Falloon, 2022). Problem-solving skills benefited students because they learned strategies to overcome problems and learned mistakes were okay. Coding was an effective way students practiced problem-solving because they encountered problems when they created their own work, completed specific coding tasks, and fixed work made by other people. The most common problem-solving strategies elementary students used when coding were trial and error, tinkering, and deletion.

One prevalent problem-solving strategy students used when coding was trial and error (Egbert et al., 2021; O'Grady-Jones & Grant, 2023). Researchers Egbert et al. (2021) found trial and error was a strategy students used when coding robots. Egbert et al. (2021) followed 46 second grade students in a rural school in the Pacific Northwest of the United States as they completed a one-week unit on basic coding with Ozobots, small robots that could be controlled through colored pens or a software program. Students were tasked to move the bots using color codes to spin, turn, move a certain way, or go faster and then interviewed about their experiences (Egbert et al., 2021). Students relied heavily on trial and error to problem-solve. For example, one student shared that they felt frustrated when their code did not work, but after they used trial and error to solve their problem, they felt a sense of accomplishment (Egbert et al., 2021). The results supported the research question about the benefits of coding because students' confidence and ability to overcome obstacles improved. Another student shared that they learned it is okay to make mistakes and try another strategy (Egbert et al., 2021). The growth mindset demonstrated by the student showed problem-solving learned by coding could benefit students' perception of mistakes and obstacles.

Similarly, researchers O'Grady-Jones and Grant (2023) found that older students used trial and error as a problem-solving strategy too. Their qualitative study included 46 fourth-grade Gifted and Talented students in a Science, Technology, Engineering, Art, and Mathematics (STEAM) course at Cori Elementary School, a pseudonym for a public elementary school in the southeastern United States and designated STEAM school. The fourth graders created a game with the block-based coding software Scratch for first grade students about light and sound energy during the 9-week spring semester of 2019 (O'Grady-Jones & Grant, 2023). Qualitative data was collected through interviews, recorded observations, in-person observations, and collection journals (O'Grady Jones & Grant, 2023). Results showed that when students encountered a problem, they often used trial and error strategies. One example was when students tinkered, or played around, with sprites (or characters), backdrops, and sounds as a form of problem-solving and created a game they envisioned (O'Grady-Jones & Grant, 2023). Students did not rely on one specific strategy; rather they just played around with the different blocks in a variety of different ways until they found a solution they found satisfactory. Once more, the study answered the research question since coding led to the development of problem-solving because they took time to make a product that made them proud.

Differing from trial and error, some studies supported that sometimes students adapted their project when they couldn't find a solution and built another problem-solving skill. Researchers Woo & Falloon (2022) conducted their qualitative study in Australian metropolitan schools with a group of 40 primary school students in a ⅚ classroom and 26 Year seven students at a secondary school (Woo & Falloon, 2022). These students used Scratch, just like the students in the O'Grady-Jones & Grant (2023) study, but made a 1-2 minute coding animated narrative (CAN) instead of a game (Woo & Falloon, 2022). The ⅚ students had weekly classes of 60-80 minutes during Term 1 and 2 (Woo & Falloon, 2022). The Year seven students had one-hour classes on basics of coding for four weeks, followed by about 6 hours worth of class to create their CAN project (Woo & Falloon, 2022). Qualitative data was gathered through interviews and recorded observations (Woo & Falloon, 2022). An example of coding errors that were bypassed and changed occurred when one group of students could not figure out how to make their Sprite, or character, larger, so they decided to move it to a different part of the frame in order to bypass the problem (Woo & Falloon, 2022). While this strategy may not sound like problem-solving, students effectively problem-solved because they worked around the obstacle to find a solution. Therefore, the study answered the research question because the problem-solving strategy gained by students was a benefit of coding. Another way students worked around the obstacles was through the deletion strategy.

Conversely, another way students worked around the obstacles was through the deletion strategy. In one qualitative study, researchers Sung et al. (2022) asked 51 kindergarten and first grade students in a diverse public school in the northeastern United States to debug, or fix errors, in a program called Scratch Jr. that had four common problems, "...(a) incorrect type of movement, (b) incorrect number of steps regarding object definition, (c) incorrect message send/receive regarding thread synchronization, and (d) missing initiating connection between two characters regarding collision detection" (p. 689-690). Students completed the debugging after a one-day workshop. Scratch Jr. was made for K-2 audiences as opposed to Scratch which was made for an older audience used in previously mentioned studies (O'Grady-Jones & Grant, 2023 ;Woo & Falloon, 2022). Researchers collected qualitative data through recorded think alouds where students shared their problem-solving processes. The researchers then quantified the strategies students used by scoring how many bugs were identified and which strategies they used, including addition, rewriting code, or deleting code (Sung et al., 2022). Additionally, quantitative data was gathered through a programming proficiency post-test, where participants were scored based on how many of the bugs they were able to correct (Sung et al., 2022). The most frequent problem-solving strategy kids used when they debugged others' code was to delete code (Sung et al., 202). Though similar to the adaptation strategy seen by Woo and Falloon (2022) since students avoided the obstacle, deletion was a bit different because students just took away the obstacle. The results answered the research question because the deletion strategy demonstrated a benefit of coding; it showed students that when they were really stuck with a problem, sometimes the best solution was to just start over. When students had a blank slate, they focused on the goal they wanted to achieve as opposed to untangling someone else's work. Additionally, the group that used the delete and rewrite strategy more frequently than the other two groups was able to make the most accurate fixes to the coding (Sung et al., 2022). Therefore, deletion was an effective problem-solving strategy students obtained from coding because they learned an effective way to resolve their issues. Other problem-solving strategies groups used included replacing some of the code or adding more code to compensate for errors in coding, though these were used less frequently (Sung et al., 2022). Therefore, there were some students who developed additional problem-solving strategies through coding as well, which showed students found a problem-solving strategy that worked for them and supported the research question.

Most studies found that the problem-solving skills students used were not always domain-specific. For example, most students did not use domain-specific computational problem-solving skills or even use decomposition, abstraction, pattern recognition, or algorithm development, unless they had previous experience with coding (Woo & Falloon, 2022). Flexible thinking and creative problem-solving were most observed in the studies versus the most efficient or technical strategies associated with coding. Therefore, even if teachers added coding to their instructional practices without explicit instruction, students built problem-solving skills. However, these problem-solving skills were sometimes limited to the task presented and could not be relied on for every problem a student may encounter while coding. For example, if a student was asked to debug a code in a specific way, the adaptation strategy would not be appropriate.

An outlier study showed that coding paired with embodied activities also led to increased problem-solving skills. Researchers Sung et al. (2022) conducted an additional quantitative study with 37 second graders in a six-week summer program at a diverse public school completed tasks with LegoWeDo sets that included a Goal Kicker Robot at a school in the Pacific Northwest (Sung, et al., 2022). Students were split into an embodied group, which physically acted out the code prior to using the programming, and a non-embodied group, which used paper and pencil to plan. Quantitative data was collected through a rubric which scored students on a scale from 1 to 5 on problem-solving skills gleaned from the Engineering Design Process including "...problem analysis, generating a plan, implementing a plan, and evaluation" (Sung et al., 2022, p. 686). Specific results for each problem-solving skill were not included in the study; however, Sung et al. (2022) found students in the embodied group had a higher problem-solving ability and were 8.25 times more likely to gain problem-solving skills. The data collected suggested that problem-solving skills were gained through coding and were built even more through physical movement when planning. Therefore, the results supported the research question that problem-solving was a benefit of coding. Even though no other researchers studied the effect of embodied activities paired with problem-solving, the results concluded that problem-solving was a prevalent skill built through coding.

Some possible limitations of the problem-solving studies were small group sizes which did not always provide a full picture of how the majority of students would develop problem-solving skills through coding (Egbert et al., 2021; O'Grady-Jones & Grant, 2023; Sung et al., 2022; Woo & Falloon, 2022). Short time frames were also listed as limitations for many studies since exposure to coding ranged from a few weeks to a few months (Egbert et al., 2021; However, some researchers felt the short time frame adequately represented how much time the average student spends on coding in a school year (Egbert et al., 2021). The Covid-19 pandemic was cited for shortened research studies as well (Woo & Falloon, 2022).

Through the studies conducted, the research question was answered because the most common problem-solving skills gained through coding were trial and error, adaptation, or starting over. In the broad sense of solving the problem, many students could figure it out, but often without specific coding problem-solving strategies. Free software programs, such as Scratch and Scratch Jr., worked well for students at the elementary level and were used on both iPads and computers (O'Grady-Jones & Grant, 2023; Sung et al., 2022; Woo & Falloon, 2022). The accessibility of the technology paired with the problem-solving skill building made coding an effective technology to increase student achievement. The studies answered the research question because after different problem-solving strategies elementary students used when coding were explored, it was clear coding had a positive impact on student achievement. Students overcame obstacles and created functioning final products through trial and error, adaption, deletion strategies. However, problem-solving was defined as problems solved in any way, while computational thinking involved specific strategies used to solve problems. While these researchers did not find a lot of computational skill growth, other studies found computational skills could be achieved when teachers used a more specific form of instruction. When the focus shifted to computational thinking, additional insight developed into what benefits coding provided for elementary student skill building.

Computational Thinking

As the focus transitioned to another theme, Computational Thinking, or CT, research examined how elementary students attained specific problem-solving strategies related to coding and how that expanded the way they approached coding problems. Though similar to problem-solving mentioned in the previous section, computational thinking was related to specific coding skills that students gained through practice such as algorithms, abstraction, and logic. Computational thinking helped students to solve problems in a specific way instead of trial and error, which broadened their skill set. At the elementary level, computational thinking was basic and most effective when taught through direct instruction.

Computational thinking was developed through coding at a basic level for elementary students (Fagerlund et al., 2020; Tran, 2019). A descriptive case study by researchers Fagerlund et al., (2020) involved 57 fourth grade students at a municipal school in Finland who coded with Scratch. Students completed 13 lessons over the course of four months in 2017 using the program Scratch (Fagerlund et. al, 2020). Tasks students completed included programming a series of instructions, debugging faulty code, and programming a dance performance (Fagerlund et al., 2020). The course ended with students making their own interactive game, story, or animation which was assessed with a rubric that assessed how accurately and often students used CT strategies (Fagerlund et al., 2020). The rubric found that "nearly all (>90%) the students designed complex projects (Abstraction), implemented algorithm control structures and "initialization" (Algorithms), remixed (Collaboration), and utilized logical operators (Logic)" (Fagerlund, et al., 2020, p. 629). These results supported that students gained computational skills through coding tasks; which led to increased skill sets for students. On the other hand, robust CT skills such as Boolean Logic and I/O devices, which were frequently used for more complex, interactive projects, were used by little to no students in their projects (Fagerlund et al., 2020). Therefore, elementary students gained CT skills that were developmentally appropriate for their age group and thus supported the research question that there were benefits of coding.

Similarly, researcher Tran (2019) also found basic computational skills of algorithms and loops were obtained by elementary students in their quantitative study of over 200 third graders across 13 classrooms from a suburban and rural district in Oregon. For 10 weeks, students had 30 minutes of hands-on learning of computer science concepts and 20 minutes of online adaptations from CSUplugged and Blockly programming language from Code.org (Tran, 2019). A main difference in this study was students used CSUplugged and Blockly programming language from Code.org instead of Scratch. Code.org was similar to Scratch because it was a free-to-use program. The results of the Tran (2019) study found students' computational thinking, for algorithms increased between the pre and post test. In addition, Tran (2019) found sequence, loops, and debugging increased between the pre and post test. The greatest growth was seen in the sequence tasks with a growth of 0.23 and 0.17 (Tran, 2019). The results supported the research question because some CT skills were attainable for young coders.

Likewise, researchers Bers et al. (2019) agreed computational skills could be attained by young students. His study gathered quantitative and qualitative data from 172 children ages three to five from three early childhood centers in Tenerife, Spain, who used KIBO robotics to explore early coding skills including sequence, repeats, and conditionals (Bers et al., 2019) From April to June of 2017, students had three coding sessions with the KIBO robots a week (Bers et. al, 2019). Quantitative "Solve-its" assessed computational thinking seen in students' final projects where they programmed a dance for their KIBO robot on a scale from 0-6 (Bers et al., 2019). Results showed the highest mastery level was 5.75 for easy sequencing and the lowest mastery was 1 for ifs (conditionals) (Bers et al., 2019). Once more, the results answered the research questions because they supported that students could benefit from coding at a young age when they practiced developmentally appropriate skills. Essentially, when the K-12 education spectrum was considered, it was logical that elementary students gained baseline knowledge that continued to be developed throughout middle and high school.

CT skills were also found to be best developed when taught through explicit instruction. Deeper knowledge was gained through direct instruction in conjunction with the more popular discovery-based learning in Scratch (Fagerlund et al., 2020). Even though the students in the previously mentioned Egbert et al. (2021) study were older by one to two years, they relied on more general problem-solving strategies since they used a more exploratory method to learn coding. Therefore, research supported that young coders could use computational thinking strategies when the skills were taught explicitly. Though the programs used were different, the studies included explicit coding lessons through the programs used between 10-13 weeks (Fagerlund et al., 2020; Tran, 2019). The research by Fagerlund et al. (2020) and Tran (2019) supported CT skills gained at a young age when students have explicit instruction in coding-specific skills. The studies also supported the more practice a student had with coding, the greater possibility they used CT specific skills (Fagerlund et al., 2020; Tran, 2019). Therefore, when teachers used explicit instruction paired with creative exploration, coding proved to be an effective form of technology to increase student achievement. Conversely, Bers et al. (2019) found that kindergarten students learned computational skills through exploratory coding. The difference in the results could have been because students used robots instead of block-based coding. Additionally, the study mentioned that teachers used songs, games, and storytelling when they taught about the KIBO robot, which could be defined as a developmentally appropriate form of explicit instruction for kindergarten students.

A limitation of the CT skills studies was that the rubrics did not encompass all possible CT skills students can use in Scratch, but rather, core CT-fostering contents (Fagerlund et al., 2020). In addition, only one to two coding programs were included in each study and more research would be needed to see the impact of different software and instructional strategies (Fagerlund et al., 2020; Tran, 2019).

The research showed that students learned CT skills through coding programs such as Scratch and Code.org. When developmentally appropriate CT skills, such as sequence, were taught first and explicitly, the study supported the research question and showed that young students benefited from coding. However, computational thinking skills were only one specific set of skills students needed to be successful. Through the exploration of the next theme, how coding improved mathematical reasoning skills, the amount of academic success students developed through coding expanded into other specific content areas.

Mathematical Skill Development

As the focus pivoted to specific skills beyond computational thinking, another theme was discovered that students developed students' mathematical reasoning skills related to measurement and geometry through coding. Coding was found to be an effective technology because students practiced spatial math skills in a physical manner, which solidified their understanding.

Mathematical skills development was positively impacted through coding projects, specifically related to geometry and measurement. In a five week-long quantitative study conducted in a public kindergarten classroom in Adana,Turkey, 29 students completed four unplugged coding activities and 16 activities with Bee-bots, which practiced mathematical skills such as properties of shapes (Somuncu & Aslan, 2022). Students participated in the activities four days a week over the course of five weeks (Somuncu & Aslan, 2022). Quantitative data was gathered from a pre and post test which consisted of 21 questions that evaluated measurement skills and 19 questions that evaluated probability and data analysis (Somuncu & Aslan, 2022). Researchers Somuncu and Aslan (2022) found the results of their post test showed "mathematical reasoning scores of children in the experimental group significantly increased in comparison with the children in the control group" (p. 887). Therefore, the study answered the research questions because coding positively impacted student math achievement as evidenced by greater mathematics skills growth. While the research did not specifically state which mathematical skills increased in the post-test, graph creation, probability, and sequencing time activities were all taught throughout the Coding Education Program, and therefore, positively contributed to the post test scores (Somuncu & Aslan, 2022). Many of these subjects were spatial, which were embodied by Bee-bots and supported that coding helped with geometry skills.

Likewise, researchers Kim et al. (2021) found increased mathematical reasoning skills were developed with older elementary students too when they taught geometry concepts through coding. Their quantitative study was conducted over the course of a two-week Science, Technology, Engineering, and Math (STEM) summer program in the southwestern United States and used robotics-based coding with twenty-four students in fourth and fifth grade (Kim et al., 2021). Similar to Somuncu and Aslan (2022), this study also focused on geometry since students practiced special pair angles when they coded a Sphero SPRK+ robot to complete three robotic-coding activities: driving, boomerang, and bowling (Kim et al., 2021). Both studies mainly focused on geometry skills that were developmentally appropriate for the ages included in the study. Results of the post test showed an improvement in understanding of special pair angles with an average post test score of 66% compared to 33% on the pretest (Kim et al., 2021). The increased scores showed, once more, coding was an effective technology, especially for spatial mathematical skills that are a part of the geometry branch. The quantitative results supported the research question because they proved coding increased student math scores, and therefore was an effective technology tool for teachers to increase student achievement.

Some possible limitations of the studies were small sample sizes (Kim et al., 2021; Somuncu & Aslan, 2022). Also, the summer STEM program was voluntary (Kim et al., 2021). Therefore, the students involved in the study may be predisposed to an interest or confidence in coding. However, positive results in mathematical reasoning were found in the compulsory kindergarten program, so the program type may not have impacted results. Both studies used robots to practice mathematical skills, which provided a limited scope of coding types (Kim et al., 2021; Somuncu & Aslan, 2022).

Multiple researchers concluded coding improved mathematical reasoning from kindergarten up to fifth grade with developmentally appropriate coding activities. Robots were commonly used for geometric and spatial mathematical skills. Research findings supported that coding boosted mathematical reasoning skills, as evidenced by increased math test scores after robots were used.Research findings supported the research question because math skills necessary for a completed K-12 education were enhanced. However, education was not just limited to math skills and not all students used computational thinking skills daily. Therefore, exploration opened up to a new theme about how coding developed students' planning skills; a transferable skill used across many disciplines.

Planning Skills

When the research broadened to transferable skills, research supported the next theme which was planning skills were also enhanced by coding. Planning skills were used in many academic disciplines including writing when students used a graphic organizer or wrote an outline and mathematics when students chose a problem-solving strategy to solve a world problem. A study by Arfé et al. (2019) was conducted with eighty five and six year olds at the beginning of first grade in an Italian school with Code.org for eight weeks (Arfé et al., 2019). As students progressed through the program, the difficulty of coding tasks increased and included loops, debugging, and a maze (Arfé et al., 2019) Researchers focused on the effect of coding on students' ability to make a plan, defined as an algorithmic procedure to complete a task (Arfé et al., 2019). Researchers used quantitative data gathered through a standardized pre-test, post-test, and delayed post-test to assess student planning skills by having students complete a drawn maze with a set of instructions (Arfé et al., 2019). Arfé et al. (2019) found that even the eight short weeks of coding had a positive impact on students' planning skills. The experimental group who received coding lessons outperformed the control group on the standardized planning tasks (Arfé et al., 2019). Therefore, even though the post test was not specifically related to coding, students exposed to code transferred the skill to other planning tasks better than students who had not used coding. Additionally, as students became more familiar with Code.org, their planning time for tasks decreased (Arfé et al., 2019). For that reason, coding benefitted students because they gained efficiency over time. The findings also supported that the acquired planning skills were retained in the delayed post-test (Arfé et al., 2019). As a result, coding was an effective technology for student achievement because the planning skills gained were prevalent and long-lasting.

Some limitations of the study were the short time frame and lack of follow up by the researchers (Arfé et al., 2019). Other limitations were that researchers did not know if the same results would be obtained with older students as well. Though not many studies focused on planning skills in particular, Arfé et al. (2019) paved a field for further research.

Once more, free coding programs provided accessible tasks that led to student achievement. Since students transferred the planning knowledge gained from coding to standardized tasks, there was potential for student achievement gains in other subject areas. The studies supported the research question because they proved coding was a cost-effective technology for planning skill development in the short-term and retained in the long-term. Hence, planning skills were another skill elementary students gained through coding practice. Further research found other general skills could be practiced through coding that benefited students as well. When the research turned to how students interacted with each other while coding, collaboration stood out as a powerful theme for coding success.

Collaboration

As the attention turned to collaborative coding projects, research highlighted how collaboration, or the ability to work with others, strengthened learner outcomes and was another common theme. Since coding projects were often extensive, such as building a narrative or programming a robot, they often lent themselves to be completed by partners or groups (Bers et al., 2019; Kim et al., 2021; Lin et al., 2022). Collaboration was named as one of the "...the six C's: communication, collaboration, community building, content creation, creativity, and choices of conduct" which have been linked to positive youth development behaviors in several studies (Bers et al., 2019, p. 132). These skills were often cited as transferable skills across multiple disciplines and career paths. Collaboration was found to be both used by students and linked to positive student achievement.

Collaboration was prevalent in coding projects and students took advantage of circumstances where they worked with others. The previously mentioned study by Bers et al. (2019) gathered quantitative and qualitative data from 172 children ages three to five from three early childhood centers in Tenerife, Spain, who used KIBO robotics to explore early coding skills; specifically, coding skills related to the Positive Technological Development (PTD) framework which included the six C's (Bers et al., 2019). Quantitative data was gathered by the PTD checklist and collaboration was rated 1-5 based on whether students ``...helped one another when using the materials"(Bers et al., 2019, p. 139). Results from the PTD checklist showed students scored 4.42 out of 5 in the collaboration category, which meant collaboration was seen almost always during sessions (Bers et al., 2019). Therefore, when students were provided an opportunity to work with a peer, they took advantage of the chance to improve their coding. From qualitative data gathered through teacher interviews and student journals, many teachers referred to collaborative moments that students had such as forming teams and "reasoning together;" and collaboration was mentioned 48 times in interviews or student journals (Bers et al., 2019, p. 141). Once more, collaboration was witnessed frequently and was a strategy students often utilized during coding projects which answered the research question.

Similarly, researchers Lin et al. (2022) found when students were assigned to work with partners during coding projects, they took advantage of the chance. Lin et al. (2022) studied 51 fifth grade students in a school in Taiwan; and 13 groups used collaborative learning through the cognitive apprenticeship model and 25 students completed the course on their own. Students used the program E-game, an online coding education game developed in Taiwan for five weeks

(Lin et al., 2022). Quantitative data tracked how many of the 13 groups used collaboration strategies; and results showed 6 groups mutually collaborated, 3 groups where the higher-scoring student directly taught the lower-scoring student, and 3 groups where lower-scoring students asked the higher-scoring student questions (Lin et al., 2022). Only 1 collaborative group never communicated with each other (Lin et al., 2022). The results proved once more that when paired up, students took advantage of the circumstances and worked together to achieve their goals. The amount of collaboration witnessed in both studies proved that collaborative coding was engaging for students. Since coding projects often lent themselves to working together, coding was an effective technology for increasing student achievement and thus, results supported the research question.

Collaboration through coding projects had positive impacts on student achievement as well. Students felt a greater sense of independence and success with coding when they worked in pairs. Results from the Lin et al. (2019) qualitative interviews found that students had a positive experience with the collaborative group and felt it was helpful. Students shared that a partner helped them to solve problems faster because they did not have to wait for the teacher and could talk through solutions together (Lin et al., 2022). Therefore, coding was beneficial for elementary students because when they collaborated with peers they achieved success in their coding projects without teacher support. Quantitative data gathered between a pretest and posttest called “E-Game CT test” also showed growth in student achievement because “...the mean of the students in Experimental Group 1’s learning efficacy post-test score is 70.71 while that of Experimental Group 2 is 65.96” (Lin et al., 2022, p. 274). Therefore, the collaborative groups ended the course with overall stronger coding skills and higher test scores. When teachers added collaboration to their coding assignments, students had a more positive and successful experience. Additionally, in groups where students communicated effectively, "the higher-scoring student can make progress by 16 points while the lower-scoring student can make progress by up to 40 points" (Lin et al., 2022, p. 281). The results showed that the lower achieving students gained the most from the collaborative grouping and there was a significant difference between low achieving students that worked in pairs versus those who worked on their own. Therefore, increased student success in coding was found in collaborative groups and supported the research question.

One limitation of the collaboration studies were they focused on a specific age group, even though the two studies focused on different grade levels than each other (Bers et al., 2019; Lin et al., 2022). Some suggestions for future research were to extend the study to third and fourth graders to see if there was a difference (Lin et al., 2022). One limitation of the Bers et al. (2019) study was the KIBO robots themselves were sometimes difficult for students to use since their scanners did not always effectively scan the barcodes. Therefore, student achievement was impacted by results tampered by machine errors versus human errors.

Even though Bers et al. (2019) and Lin et al. (2022) used different programs to conduct their experiments, coding worked well for students who collaborated together to complete the activities, which led to academic achievement and communication skill building. Students in groups took advantage of having someone else to work with and asked questions or worked through problems together. Lower-achieving students gained the most from collaborative grouping, but higher-achieving students made progress as well, just in smaller increments. Therefore, where student achievement was concerned, collaborative coding groups had the biggest impact on student success over individuals. Research findings supported the research question because opportunities for collaboration were a strong benefit of coding lessons. Much of the research thus far has focused on coding tasks where specific outcomes were expected or coding was used for science and mathematics courses. However, when focus shifted to more open-ended projects, it was discovered that creativity as a byproduct was a theme of coding as well.

Creativity

When the focus shifted to more open-ended projects,research findings found another theme that student creativity was positively impacted through coding. Open-ended coding projects and external additions to coding projects sparked the most creativity from elementary students.

Open-ended coding projects were one way coding contributed to student creativity. Open-ended coding projects such as creating narratives or designing a game left space for student creativity and personal touch (Arslan & Çelik, 2022). These projects were most commonly used with upper elementary students. In a study conducted by Arslan and Çelik (2022), eight teachers and 16 students between grade 1 and grade 4 were interviewed at a primary school in Amaysa about their views on their weekly hour-long robotics coding program and whether coding had any contribution to students' metacognitive skills, including creativity. The qualitative data gathered through the interviews found most teachers believed the Robotics and Coding course embedded into the curriculum increased student cognition and "improved their sequential thinking skills, creativity, mental development and ability to look at events from different perspectives" (Arslan & Çelik, 2022, p. 183). One respondent believed that creativity was built through coding because students are "making a design in the coding lesson" (Arslan & Çelik, 2022, p. 183). Therefore, since students created something with their own personal touch, they showcased their creativity. In the earlier study by O'Grady and Jones (2022) creativity was developed alongside problem-solving when fourth-grade students "construct[ed} their own original and personally meaningful interpretation.." of the game-based learning assignment (pg. 56). Students exercised creativity when they created the narrative and designed backgrounds that matched their story. Once more, the research question was answered since students had the freedom to express their personal choices into their product and demonstrated creative thinking through coding.

Creativity with coding was more often exhibited through external additions to coding projects for younger elementary students. When students developed stories, homes, and outfits for their robots, their creativity was exhibited (Arslan & Çelik, 2022; Bers et al., 2019; Murcia et al., 2020). At an Australian university's Early Years Centre located in a metropolitan city, Murcia et al. (2020) conducted a qualitative study of eight kindergarten students (ages 3 and 4) who interacted with Bee-bots and the corresponding iPad app. During the two-week unit, students created a grid map and coded a route based on an adventure story they wrote for Willbee, their Bee-bot; and teachers took pictures and wrote reflective learning stories (Murcia et al., 2020). The researchers then analyzed the pictures and the reflective learning stories through the lens of the *A to E of Creativity Framework* that they developed which defined the characteristics of creativity as Agency, Being Curious, Connecting, Daring and Experimenting (Murcia et al., 2020). Within each defined characteristic, there were specific actions a student did to show that skill such as imagining, exploring, learning from failure, and trying out new ideas (Murcia et al., 2020). Students were engaged in creative doing when they coded the route for their Bee-bot based on an original story that they wrote (Murcia et al., 2020). Since the stories were works of fiction, students used creativity when they decided what story they wanted to tell. Students also expressed originality when they designed original houses for their Bee-bot (Murcia et al., 2020). Although this aspect of the project was not specifically related to coding, the project gave students the chance to be creative because they designed homes from their imaginations. Murcia et al. (2020) felt that there were some limitations on creativity with the Bee-bot app since the app was focused on “learning the right solution or sequence of instructions but not on encouraging children’s original thinking and own voice”(p. 1408). So while the process of coding in a vacuum may not have cultivated creativity in this particular unit, the stories that guided the coding and the environments created as obstacles to code around encouraged creative thinking in terms of original thought. The acknowledgement of external creative elements answered the research question because it showed how students can expand their creative thinking in coding projects.

Similarly, Bers et al. (2019) found kindergarten students used creativity with their KIBO robots and scored well on the PTD in creativity. For creativity, students were scored from 1-5 on whether they “...used materials in a divergent, unexpected manner” (Bers et al., 2019, p. 139). The mean score on the PTD for instances of creativity was 3.1 out of 5 (Bers et al., 2019). Therefore, creativity was prevalent in more than half of observed lessons and incorporated into the coding project. Some examples of creative thinking included how students decorated their KIBO robot to complete the folk dance (Bers et al., 2019). Similar to the Bee-bot homes, the creative expression occurred alongside the coding, not directly within the coding. The creative aspect was integral to the final product and built upon the coding. Therefore, depending on the project, coding encouraged creative expression. The research question was answered because when open-ended coding projects were considered, creativity was a positive outcome of coding lessons for elementary students. Creativity gave students the opportunity to exert independence over their code; similar to how they used independence to consider a future career. As the focus shifted to beyond the elementary classroom and toward career aspirations, research supported the final theme which was that exposure to coding led to an increased interest in STEM careers.

Interest in STEM Careers

Finally, when research expanded to outcomes beyond the elementary classroom, a final theme was discovered that exposure to coding encouraged students to consider future STEM careers. Many students were not aware of the job description of computer scientists or roboticists prior to coding lessons, so education and exposure to these professions was vital (Ragusa & Leung, 2023). Coding careers were an in-demand and lucrative career path and encouraged by the K-12 initiative called Computer Science for All (Smith, 2016).

Elementary students were more interested in STEM careers if they had exposure to coding. Over the course of seven years, Ragusa & Leung (2023) gathered quantitative data about career aspirations of 862 students ages seven to ten who participated in Robotics and Coding Academy (RCA) for "...under-resourced city dwelling families in the western United States" (Ragusa & Leung, 2023, p. 5). Quantitative data was collected through a pre and post program career aspiration questionnaire (Ragusa & Leung, 2023). The results showed a 32% increase in interest in coding careers between the pre and post program questionnaires (Ragusa & Leung, 2023). Insight gained from the interviews supported the benefits of coding because students' knowledge of lucrative careers increased and became a possible future for them; thus results answered the research question.

Similarly, another quantitative study by Ogegbo and Aina (2023) found coding exposure was linked to greater interest in STEM careers. This study looked at fifty students, between the ages of 10 and 15 who attended public primary and secondary schools in Gauteng, South Africa who were selected by teachers based on robotics and coding interest and particpated in an after school coding program in October 2022 presented by the TANGIBLE Africa training group (Ogegbo & Aina, 2023). Students worked as teams on 35 levels in the coding mobile app TANKS, where students practiced coding skills such as, '"for loop", "while loop", "if statement", optimization, and algorithm" (Ogegbo & Aina, 2023, p. 6). The study gathered quantitative data through pre and post coding attitude and STEM career interest surveys and qualitative data through a focus group interview (Ogegbo & Aina, 2023). The results also found some students indicated that they would consider a career in the STEM field in their post-survey, though did not provide a specific amount (Ogegbo & Aina, 2023). The results supported the research question that coding could benefit elementary students because their interest in STEM fields could be peaked at a young age and changed attitudes about possible career paths.

Confidence in coding abilities was also linked to interest in STEM careers. Results indicated "a positive relationship between attitude towards coding and STEM career interest," specifically in mathematics (Ogegbo & Aina, 2023, p. 15).Therefore,the greater confidence a student had in their coding abilities, the more likely they were to show interest in STEM careers. Students who completed RCA had an 89% growth in coding skills through the program (Ragusa & Leung, 2023). While there was an increase in both coding skill growth and interest in STEM careers, it was not the same percentage (89% compared to 32%) which meant exposure to coding encouraged some kids to pursue STEM careers, but did not work for all students. However, coding was an effective method to build student confidence in computer science.

Additionally, students who felt that their family members or peers had a positive perception of coding were more likely to show interest in a technology career (Ogegbo & Aina, 2023). Researchers Ogegbo & Aina (2023) concluded that the more exposure and positive framing around coding and the STEM field from teachers, families, and peers could encourage students to pursue STEM careers (Ogegbo & Aina, 2023). Therefore, the results supported the research question since coding was seen as an effective tool for student achievement when paired with positive role models and attitudes around it. Ragusa and Leung did not specifically explore peer influence on whether students' considered coding as a future career, however, they used university students as STEM tutors and near-peer role models, which could have influenced student attitudes towards coding careers.

Even participants who were not specifically interested in computer science as a future career recognized that coding could help them in other fields. In interviews, students shared their learned skills such as problem-solving or thinking critically which are needed to be an engineer or medical doctor (Ogegbo & Aina, 2023). Therefore, students viewed coding as a valuable skill for their futures, which supported the research question.

A limitation of the career aspirations studies were students self-selected if they wanted to participate in the program or were selected by teachers based on interest (Ogegbo & Aina, 2023; Ragusa & Leung, 2023). Therefore, participants showed a possible interest in robotics due to the optional nature which may have resulted in a greater likelihood for students to consider STEM careers. In addition, Ragusa & Leung "could not entirely infer that the results were a consequence of the program" since they were not sure if students had exposure to coding during school or another source (p. 8).

Both Ogegbo & Aina (2023) and Ragusa & Leung (2023) concluded that exposure to coding had a positive impact on student outcomes because students believed that they could learn the skills needed for well-paid careers. Both studies validated earlier exposure to coding led to higher interest and confidence in computer science as a career. Additionally, coding was an effective technology tool even for students not interested in computer science fields because they felt it was useful for other lucrative careers as well. Therefore, the studies answered the research question since coding was seen to have positive impacts while students were in school and believed to have positive effects for student futures.

Conclusion

In summary, research findings have found several positive impacts through coding instruction for elementary students. There were a variety of ways to use coding in an elementary classroom including unplugged methods, block-based coding through programs such as Scratch, and robotics coding with tools such as Bee-bots. All methods used yielded different skills that positively impacted student achievement. Coding had positive impacts on general skills such as student collaboration, creativity, and planning. In addition, several mathematical reasoning skills, including measurement, estimation, and special pair angles, were improved through the use of coding programs. Computational skills began to be developed at the elementary level with the caveat that further education was needed as they continued their K-12 education. Exposure to coding in school encouraged students to explore STEM as a possible career, which has been linked to more in-demand and lucrative careers. Research supported that coding was an effective form of technology to increase student achievement due to its vast skill-building potential. Many coding projects also had students work in groups or pairs, which increased student results, especially for lower achieving students (Lin et al., 2022).After fifteen studies about coding for elementary students were explored, it was determined that coding had many positive impacts and was an effective technology for increased student achievement, therefore, the research question was answered. However, more research was needed about which coding projects and resources were the most effective and whether long term coding implementation created stronger academic achievement. In chapter three, the future of coding for elementary students was considered both through application in elementary schools and through further research needed.

Chapter Three: Discussions, Applications, and Future Studies

After exploring fifteen studies about coding for elementary students, evidence suggests that coding has positive impacts on elementary students in a variety of skills. Though there is not a consistent resource or method for coding implementation, positive results were seen in different ways depending on the project. Insights, applications, and further research will help provide a path forward for elementary teachers and researchers to bolster coding education for the youngest learners.

Insight Gained from the Research

There are many positive impacts for student achievement found in the research on coding for elementary students. A common skill students can learn through a multitude of coding programs and activities is problem-solving. Students apply a variety of strategies to solve problems such as tinkering, deletion, and adaptation when coding. Students apply problem-solving strategies that help them overcome obstacles and find success with coding projects. The development of these strategies leads to student achievement because students gain confidence and methods for solving problems in other content areas as well.

Coding can also help with practicing specific content including mathematical reasoning and computational thinking; though these are the most successful for students when paired with explicit instruction. Geometry skills such as shapes and angles work best with coding because they are spatial tasks. Additionally, basic computational skills such as sequence are attainable for coders as young as three. Teachers must keep in mind that just like with any content, coding needs to be taught at a developmentally appropriate level and built upon as students progress through their K-12 education.

Life skills including collaboration and creativity, two of the six C's found for positive youth development, can also be built on through coding. Many coding projects work best with partners or groups, so collaboration is integral for coding to be successful at the elementary level. Students also use each other frequently as a resource when they are in pairs, which leads to academic gains. Collaboration is especially beneficial for lower achieving students and can provide huge academic gains. While creativity can be a part of coding, at the elementary level, much of the creativity was external additions to the coding projects like a costume for the robot. As students progress through elementary school, they can increase creativity by designing games and narratives. Coding increases students' ability to make a plan efficiently and can then also transfer those planning skills to other tasks. Finally, exposure to coding at a young age can encourage students to pursue future careers in STEM. This benefits students because coding careers are in demand and are often high-paying, which can provide a bright future for students.

Currently, there is not a consistent method for teaching coding at the elementary level. However, Bee-bots were commonly used for robotic coding and seemed to be more developmentally appropriate for K-2 students. As students progress through elementary school, they start developing the capacity for more complex robots and block-based coding programming. Again, considering the wide developmental range of K-6 students, there should be a continuum of skills that start at a basic level and progress in difficulty as students get older. For teachers looking for a place to start with coding, they will find that multiple methods can be used that still yield positive results.

Application

Teachers who are looking to add coding into their lessons have a few pathways they can choose depending on their goal. First, if teachers want students to expand their problem-solving skills, almost any coding program and project will invite some problem-solving practice. Both block-based coding programs and robotics were found to encourage students to use problem-solving skills like trial and error or tinkering. Students were found to use problem-solving skills in creative coding projects and specific goal coding projects. A creative coding project could look like coding a narrative or creating their own game. Specific goal coding projects could be asking a robot to complete a specific task such as spin in a circle or move through a maze.

However, if teachers want their students to gain specific computational thinking skills, they must remember to start with the basics of coding and use explicit instruction so students have a clear understanding of computer science problem-solving strategies. Programs like Scratch Jr. and Code.org have modules students could follow that would explicitly teach computational coding skills. Having elementary students start with the basics will help them gain a strong foundation to build upon as they continue through middle and high school.

When using coding at a young age, collaborative coding projects seem to be the most engaging and have a positive impact on student learning. When teachers use coding in their school, they should use pairs or small groups as much as possible to encourage students to work together and improve their coding and communication skills. Teachers could have students work with a partner to create a game with block-based coding or code a robot to complete tasks.

Coding can also be used to demonstrate mathematical concepts. Robots were commonly used to teach mathematics concepts such as geometry or measurement and can help students visualize math concepts. For example, fourth grade students could code a robot to travel through a maze made up of acute, obtuse, and right angles. Teachers can look at their geometry standards and see how they can intertwine a lesson with robotics.

If a teacher's goal is to use coding to build creativity in their students, they may need to consider projects that lend themselves to creative development such as a narrative story or an original game. While the coding itself may not be considered creative, how the children express their ideas through code and add their personal touch would be the creative piece. These more open-ended creative tasks seemed to be more suitable for upper elementary students, especially for third grade and above. Since younger students are still learning coding basics, they can exhibit their creativity through external additions to their projects like making a costume or home for their robot to supplement the coding.

For K-2 students, Bee-bots seem to be the most popular robot of choice. Older elementary students used a variety of different robots, so more research may be needed to see which robot would best fit the students' goals. Additionally, block-based coding programs such as Scratch were most frequently used with positive results with students in upper elementary.

Programs like Scratch and Code.org are free block-based coding programs and are therefore accessible for the majority of students. Both programs can be accessed through iPads or computers, so students are not limited by the kind of device their school has in their building. These programs are a great place for elementary students to start exploring coding.

Future Research Recommendations

Since there is not a standardized method for teaching coding, the studies covered a wide range of coding programs and project types. First of all, more research could be done to compare the different coding programs to narrow down which ones are the most effective for each skill students want to develop. For example, which coding program or project is the most effective for developing problem-solving? Scratch seems to be a prevalent block-based coding program, but is it the most effective for students? Additionally, more research may need to be done to see if there are significant differences between block-based coding and robotics to teach coding. Are both needed for students to be proficient in coding? If more research is done, educators can feel confident about which programs will be the most effective for their students depending on their goals. This can then guide which resources they will want to make sure they have available in their schools and where professional development funds should be spent.

Secondly, much of the research on coding was limited to a short period of time during short coding units or programs. Most programs ranged from one to six weeks. It would be beneficial for more research to be done to see how coding skills develop over the course of a school year. Then, the studies could expand to cover the continuum of K-6 coding education. In this way, researchers could narrow down exactly which coding skills are developmentally appropriate for each age. Also, further research would need to be done to see how consistent coding throughout K-6 impacts elementary students' potential for coding skills. At the moment, research supports the idea that students can learn basic coding skills at a young age, but that could possibly change if students have more practice with coding.

Thirdly, many of the studies focused on a small group of participants from specific grade levels. If research expands to a larger participant pool, confidence in the results of the studies could be verified. Research expansion could be cross-grade level participants or studies that include more students of the same age. Many of the studies were done with around 40 students, which is a very small sample size. More research studies with larger groups could provide stronger data.

Finally, research suggests that introducing coding at a young age led to more interest in STEM careers. Further research can be done to follow up with students who learned about coding in elementary school and which college majors or careers they actually ended up pursuing. This longitudinal study would solidify if coding at a young age really had an impact on students' career paths. This could then guide elementary teachers to consider how they should add coding to their instruction.

Conclusion

In conclusion, coding has many positive impacts on elementary students. Students can practice life skills such as problem-solving, collaboration, creativity, and planning skills that can transfer beyond coding programs. Academically, coding supports computational thinking and mathematical reasoning, especially in spatial subjects such as geometry. At the elementary level, exploration paired with explicit instruction helps young learners build skills through coding. Finally, exposure to coding has been linked to an increase in interest in the STEM fields, a field that continues to be in-demand for employers even after 2016 when the Computer Science for All initiative began. When considering skills students would benefit from for their future careers, coding can help students in a variety of ways.

When looking for a technology program to add or enhance elementary schools, coding can be an engaging and high-impact way for students to begin computer science education. Elementary teachers can use robots such as Bee-bots or Sphero for a variety of different skills such as problem-solving, computational thinking, and mathematical reasoning.. Block-based programs such as Scratch and Code.org are popular and accessible programs for elementary students as well. These programs work well for students to develop problem-solving, creativity, and computational thinking. Teachers should keep in mind that just like with any subject, they need to start with foundational skills and build upon those skills as students progress through their K-12 education. More research needs to be done on exactly which program is the most effective and whether coding education should include both robotics and block-based coding programs. Additionally, longer studies can provide more insight to how much coding should be included into the curriculum as well as what skills are developmentally appropriate for elementary students. Coding has many positive impacts, but without a systemic way for teaching coding, students may not be reaching their full potential with coding.

Overall, coding has many positive impacts on students and is a technology that can lead to increased student achievement. Studies support the research question that coding has benefits and is a worthwhile technology for teachers to increase student achievement because it builds many skills such as problem-solving, mathematical reasoning, and interest in STEM. Bill Gates, Co-founder of Microsoft, shared that coding helps students in a variety of ways when he said, “Learning to write programs stretches your mind, and helps you think better, creates a way of thinking about things that I think is helpful in all domains” (Code.org, 2024, para. 21). Even if coding doesn’t encourage all students to be computer scientists, it can help encourage students to be well-rounded individuals with strong 21st century skills.

References

Arfé, B., Vardanega, T., Montuori, C., & Lavanga, M. (2019). Coding in Primary Grades Boosts Children's Executive Functions. *Frontiers in Psychology*, *10*, 2713–2713. https://doi.org/10.3389/fpsyg.2019.02713

Arslan, S., & Çelik, Y. (2022). Primary school teachers' and students' views about robotic coding course. *African Educational Research Journal*, *10*(2), 178–189.

Bers, M. U., & Sullivan, A. (2019). Computer science education in early childhood: The case of scratchJr. *Journal of Information Technology Education: Innovations in Practice*, *18*, 113–138.

Bers, M. U., González-González, C., & Armas–Torres, M. B. (2019). Coding as a playground: Promoting positive learning experiences in childhood classrooms. *Computers and Education*, *138*, 130–145. https://doi.org/10.1016/j.compedu.2019.04.013

Egbert, J., Shahrokni, S. A., Abobaker, R., & Borysenko, N. (2021). "It's a chance to make mistakes": Processes and outcomes of coding in 2nd grade classrooms. *Computers and Education*, *168*, 104173-. https://doi.org/10.1016/j.compedu.2021.104173

Fagerlund, J., Häkkinen, P., Vesisenaho, M., & Viiri, J. (2020). Assessing 4th Grade Students' Computational Thinking through Scratch Programming Projects. *Informatics in Education*, *19*(4), 611–640. https://doi.org/10.15388/infedu.2020.27

Code.org. (2024). Leaders and trendsetters all agree on one thing. Code.org. https://code.org/quotes

K-12 Computer Science Framework Steering Committee (2016). K-12 computer science framework. Retrieved from https://k12cs.org/

Kim, Y. R., Park, M. S., & Tjoe, H. (2021). Discovering concepts of geometry through robotics coding activities. *International Journal of Education in Mathematics, Science and Technology*, *9*(3), 406–425.

Lin, C., Yang, S., Lin, K., Looi, C., & Chen, Y. (2022). Explorations of two approaches to learning CT in a game environment for elementary school students. *Journal of Computers in Education, 9*(2), 261-290. https://doi.org/10.1007/s40692-021-00203-x

Murcia, K., Pepper, C., Joubert, M., Cross, E., & Wilson, S. (2020). A framework for identifying and developing children's creative thinking while coding with digital technologies. *Issues in Educational Research, 30*(4), 1395-1417. https://www.proquest.com/scholarly-journals/framework-identifying-developing-childrens/doview/2702191572/se-2

Nordrum, A. (2024, April 15). *10 breakthrough technologies 2024*. MIT Technology Review. https://www.technologyreview.com/2024/01/08/1085094/10-breakthrough-technologies-2024/

Ogegbo, A. A., & Aina, A. Y. (2023). Exploring young students' attitude towards coding and its relationship with STEM career interest. *Education and Information Technologies*. https://doi.org/10.1007/s10639-023-12133-5

O'Grady-Jones, M., & Grant, M. M. (2023). Ready coder one: Collaborative game design-based learning on gifted fourth graders' 21st century skills. *Gifted Child Today*, *46*(2), 84–107. https://doi.org/10.1177/10762175221149259

Ragusa, G., & Leung, L. (2023). The Impact of Early Robotics Education on Students' Understanding of Coding, Robotics Design, and Interest in Computing Careers. *Sensors (Basel, Switzerland)*, *23*(23), 9335-. https://doi.org/10.3390/s23239335

Smith, M. (2016). Computer science for all. Washington, DC: Office of Science and Technology Policy, Executive Office of the President. Retrieved from https://www.whitehouse.gov/blog/2016/01/30/computer-science-all

Somuncu, B., & Aslan, D. (2022). Effect of coding activities on preschool children's mathematical reasoning skills. *Education and Information Technologies*, *27*(1), 877–890. https://doi.org/10.1007/s10639-021-10618-9

Sung, W., Ahn, J., & Black, J. B. (2022). Elementary students' performance and perceptions of robot coding and debugging: embodied approach in practice. *Journal of Research in Childhood Education*, *36*(4), 681–696. https://doi.org/10.1080/02568543.2022.2045399

Tran, Y. (2019). Computational Thinking Equity in Elementary Classrooms: What Third-Grade Students Know and Can Do. *Journal of Educational Computing Research*, *57*(1), 3–31. https://doi.org/10.1177/0735633117743918

Woo, K., & Falloon, G. (2022). Problem solved, but how? An exploratory study into students' problem solving processes in creative coding tasks. *Thinking Skills and Creativity*, *46*, 101193-. https://doi.org/10.1016/j.tsc.2022.101193

U.S. Bureau of Labor Statistics. (2023). *Computer and Information Technology Occupations : Occupational Outlook Handbook.* U.S. Bureau of Labor Statistics. https://www.bls.gov/ooh/computer-and-information-technology/home.htm

FINAL CAPSTONE PAPER RUBRIC: (PASS OR FAIL)				
Criteria (Total points)	**Exemplary** 420-403	**Proficient** 402-361	**Competent** 360-319	**Unsatisfactory** 0
Cover Page; TOC, Abstract, Chapter One (50 points)				
Cover Page; Table of Contents, Abstract, and Chapter One: **Possible Points: ____/50** **Instructor Feedback:** **2nd Reader Feedback:**	The writer has **consistently** utilized the capstone paper template and followed **all guidelines** for the development of the cover page, TOC and Abstract; the writer has developed **a well-organized, succinctly written** chapter one informing the reader of the following: the topic and scope of the research investigation; importance of the topic to the field of education; statement of interest to engage the reader; at least 3 sources cited with a clear connection to the research question; definition of terms; how the scope of the problem investigated will be organized in a logical sequence through the use of subtopics; the research question concludes the chapter connecting to the Essential Question; Chapter ends with a conclusion (chapter summary) paragraph that includes a transition to the following chapter. **48-50 points**	The writer has **usually** utilized the capstone paper template and followed **most of the guidelines** for the development of the cover page, TOC and Abstract; the writer has **mostly** developed chapter one informing the reader of the following: the topic and scope of the research investigation; importance of the topic to the field of education; statement of interest to engage the reader; at least 3 sources cited with a clear connection to the research question; definition of terms; how the scope of the problem investigated will be organized in a logical sequence through the use of subtopics; the research question concludes the chapter connecting to the Essential Question; Chapter ends with a conclusion paragraph that includes a transition to the following chapter. **43-47 points**	The writer has **sometimes** utilized the capstone paper template and followed **some of the guidelines** for the development of the cover page, TOC and Abstract; the writer has **partially** developed chapter one informing the reader of the following: the topic and scope of the research investigation; importance of the topic to the field of education; statement of interest to engage the reader; at least 3 sources cited with a clear connection to the research question; definition of terms; how the scope of the problem investigated will be organized in a logical sequence through the use of subtopics; the research question concludes the chapter connecting to the Essential Question; Chapter ends with a conclusion paragraph that includes a transition to the following chapter. **38-42 points**	The writer has **rarely** met the required components for the criteria in this category resulting in "0" points. **NOTE: Less than 319 points results in "0" for this assignment as it is a "pass or fail" paper representing the successful completion of the MAED program requirements competently.**
Chapter Two: Literature Review (210 points)				
Chapter Two: Literature Review **Possible Points: ____/210**	The writer has **consistently** provided a professionally written narrative which summarizes and synthesizes the information from the selected research studies in order to develop a response and answer to	The writer has **usually** maintained a professionally written narrative which summarizes and synthesizes the information from the selected research studies in order to develop a response and answer to	The writer has **sometimes** maintained a professionally written narrative which summarizes and synthesizes the information from the selected research studies in order to develop a response and answer to	The writer has **rarely** met the required components for the criteria in this category resulting in "0" points. **NOTE: Less than 319 points results in "0" for this**

Instructor Feedback: **2nd Reader Feedback:**	the research question proposed in Chapter One. The narrative **fully** answers the proposed research question. Includes a minimum of 15 scholarly, peer-reviewed qualitative/quantitative/mixed-method **original** research studies. Chapter ends with a research finding summaries and conclusions The writer has **consistently** provided a **succinct and precise** summary of findings includes a review of the proposed problem that was investigated; the importance of this topic; and a paraphrased summary of the main points or themes of the literature review; Chapter ends with a conclusion paragraph (chapter summary) that includes a transition to the following chapter. **202-210 points**	the research question proposed in Chapter One. The narrative **mostly** answers the proposed research question. Includes a minimum of 15 scholarly, peer-reviewed qualitative/quantitative/mixed-method **original** research studies. Chapter ends with a research finding summaries and conclusions The writer has **usually** provided **a mostly developed** summary of findings includes a review of the proposed problem that was investigated; the importance of this topic; and a paraphrased summary of the main points or themes of the literature review; Chapter ends with a conclusion paragraph (chapter summary) that includes a transition to the following chapter. **181-201 points**	the research question proposed in Chapter One. The narrative **partially** answers the proposed research question. Includes a minimum of 15 scholarly, peer-reviewed qualitative/quantitative/mixed-method **original** research studies. The writer has **sometimes provided a partially developed** summary of findings includes a review of the proposed problem that was investigated; the importance of this topic; and a paraphrased summary of the main points or themes of the literature review; Chapter ends with a conclusion paragraph (chapter summary) that includes a transition to the following chapter. **160-180 points**	**assignment as it is a "pass or fail" paper representing the successful completion of the MAED program requirements competently.**

Chapter Three: Discussion / Application / Future Studies (75 points)				
Chapter Three: Discussion/ Application/ Future Studies **Possible Points: ____/75 Feedback:** **2nd Reader Feedback:**	The writer has **consistently** developed a **clear** summary of insights gained from the research that leads to improved instructional practice. The writer provided a **clear** description with examples of how the research is applied to instructional or educational practice; has provided a minimum of three suggestions for possible future studies; and the chapter ends with a powerful conclusion that acts as a conclusion for the entire paper.	The writer has **usually** developed a **mostly clear** summary of insights gained from the research that leads to improved instructional practice. The writer provided a **mostly clear** description with examples of how the research is applied to instructional or educational practice; has provided a minimum of three suggestions for possible future studies; and the chapter ends with a powerful conclusion that	The writer has **sometimes** developed a **partially clear** summary of insights gained from the research that leads to improved instructional practice. The writer provided a **partially clear** description with examples of how the research is applied to instructional practice; has provided a minimum of three suggestions for possible future studies; and the chapter ends with a powerful conclusion that acts as a conclusion for the entire paper.	The writer has **rarely** met the required components for the criteria in this category resulting in "0" points. **NOTE: Less than 319 points results in "0" for this assignment as it is a "pass or fail" paper representing the successful completion of the MAED program requirements competently.**

	72-75 points	acts as a conclusion for the entire paper. **65-71 points**	**57-64 points**	
APA Format & Mechanics (85 points)				
APA format & Mechanics **Possible Points: ____/85** **Instructor Feedback:** **2nd Reader Feedback:**	The writer has **consistently** met the criteria for the following requirements for this paper: APA formatted cover page; Table of Contents right/left justified; clear, half page Abstract – per APA formatting provided; in text citations per APA and included in References page; Reference page formatted per APA guidelines; Correct use of APA level headings; correct use of spelling, grammar, and punctuation; Higher level professional language; third person writing only; correct use of past tense. **82-85 points**	The writer has **usually** met **most** of the criteria for the following requirements for this paper: APA formatted cover page; Table of Contents right/left justified; clear, half page Abstract – per APA formatting provided; in text citations per APA and included in References page; Reference page formatted per APA guidelines; Correct use of APA level headings; correct use of spelling, grammar, and punctuation; Higher level professional language; third person writing only; correct use of past tense. **73-81 points**	The writer has **sometimes** met **some** of the criteria for the following requirements for this paper: APA formatted cover page; Table of Contents right/left justified; clear, half page Abstract – per APA formatting provided; in text citations per APA and included in References page; Reference page formatted per APA guidelines; Correct use of APA level headings; correct use of spelling, grammar, and punctuation; Higher level professional language; third person writing only; correct use of past tense. **65-72 points**	The writer has **rarely** met the required components for the criteria in this category resulting in "0" points. **NOTE: Less than 319 points results in "0" for this assignment as it is a "pass or fail" paper representing the successful completion of the MAED program requirements competently.**
Revised July 2022			**TOTAL POINTS ______/420** **Pass or Fail: 319 points are required to pass**	