www.ijemst.net

The Impact of Teaching Computational Thinking on the Growth of Students' Algebraic Thinking

James Knutson
California State University, Dominguez Hills, United States of America

Kristen Stagg
California State University, Dominguez Hills, United States of America

To cite this article:

Knutson, J., & Stagg, K. (2025). The impact of teaching computational thinking on the growth of students' algebraic thinking. *International Journal of Education in Mathematics, Science, and Technology (IJEMST), 13*(6), 1615-1631. https://doi.org/10.46328/ijemst.5213

International Journal of Education in Mathematics, Science and Technology

2025, Vol. 13, No. 6, 1615-1631 https://doi.org/10.46328/ijemst.5213

The Impact of Teaching Computational Thinking on the Growth of Students' Algebraic Thinking

James Knutson, Kristen Stagg

Article Info

Article History

Received:
23 April 2025

Accepted:
30 September 2025

Keywords

Algebra 1
Computer science
iReady assessments
Computational thinking
Algebraic thinking

Abstract

This research examines the integration of computational thinking in mathematics education, specifically examining its impact on students' algebraic reasoning. By teaching core computational concepts such as pattern recognition, decomposition, and algorithmic thinking, students develop a deeper understanding of algebraic structures and processes. The study focuses on a classroom intervention where students are concurrently taught Algebra 1 and computer science by the same teacher using interdisciplinary lessons. The results suggest that combining computational thinking with algebra instruction fosters problem-solving skills, builds confidence, and enhances student engagement. A key challenge lies in ensuring that students do not feel obligated to participate, as the teacher serves as both instructor and researcher. To mitigate this, students are fully informed about the study's objectives and provided with consent forms, emphasizing that participation will not influence their grades or offer extra credit. This research provides educators practical insights and strategies for integrating computational thinking into algebra curricula, thereby fostering stronger algebraic reasoning and advancing students' mathematical achievement.

Introduction

In today's rapidly evolving, technology-driven world, mathematics and computer science are foundational disciplines that shape students' readiness for college, careers, and civic life. Algebra, in particular, is considered a "gateway" subject, often determining whether students successfully transition into advanced mathematics courses such as geometry, algebra II, precalculus, and calculus. Unfortunately, many students struggle to master algebraic concepts, leading to frustration, decreased motivation, and, in some cases, long-term disengagement from STEM fields. Research has shown that early struggles with algebra can have a cascading effect on a student's academic trajectory, limiting their opportunities in high-demand careers that require strong quantitative and problem-solving skills (Maudy et al., 2019).

According to the National Assessment of Educational Progress (NAEP), only 26% of eighth-grade students in the United States scored at or above the proficiency level in mathematics in 2023, a sharp decline compared to previous decades. These declining scores have been further exacerbated by disruptions caused by the COVID-19 pandemic, which widened pre-existing achievement gaps among historically underserved student populations, including low-income students, English learners, and students of color. Algebra is particularly challenging because it requires students to transition from concrete arithmetic thinking to abstract symbolic reasoning; a cognitive leap that many students find difficult to navigate. Addressing this persistent problem calls for innovative instructional approaches that bridge students' current understandings with the complex reasoning skills demanded by algebra and higher-level mathematics.

One promising approach is the integration of computational thinking (CT) into mathematics instruction. Computational thinking, defined as "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" (Wing, 2010, p.1), goes beyond simply learning to code. It emphasizes skills such as decomposition (breaking complex problems into manageable parts), pattern recognition, algorithmic design, and abstraction. These skills closely mirror the kinds of reasoning required in algebra, where students must identify patterns, generalize mathematical relationships, and use symbolic language to express and solve problems.

The potential for connecting algebra and computational thinking has gained attention in educational policy. Former President Barack Obama's *Computer Science for All* initiative, launched in 2016, aimed to ensure that every K–12 student in the United States had the opportunity to learn computer science (Office of the Press Secretary, 2016). This movement sparked increased funding, professional development, and curriculum design efforts to bring computer science into schools. However, while computer science has gained momentum as a standalone subject, there has been limited exploration of how it can be meaningfully integrated with core subjects like mathematics. As Hickmott et al. (2017) noted, "studies that explicitly linked the learning of mathematics concepts with computational thinking were uncommon in the reviewed literature" (p. 16). This lack of research represents a critical gap, especially given that the disciplines naturally share overlapping concepts such as variables, functions, and coordinate systems.

For the past fourteen years, I (lead author) have taught both mathematics and computer science to middle and high school students in an urban public school district. Throughout my teaching career, I have observed a consistent pattern: many students struggle to see the relevance of algebra to their lives, viewing it as a series of disconnected procedures rather than a powerful tool for understanding and solving real-world problems. In contrast, when these same students engage in computer science projects, such as designing games or coding simulations, they demonstrate high levels of engagement, creativity, and perseverance. This stark difference in motivation led me to wonder whether computational thinking could serve as a bridge to deeper algebraic understanding.

This study focuses on the concurrent teaching of Algebra 1 and Computer Science to the same group of students, using interdisciplinary lessons designed to highlight shared concepts and reasoning processes. It investigates the research question:

What impact does teaching computational thinking have on the growth of students' algebraic thinking?

The significance of this work lies in its potential to address two pressing educational challenges. On one hand, it seeks to improve algebra readiness and achievement in ways that close gaps in students' mathematical learning. On the other, it expands access to computational thinking, preparing students for participation in a technology-driven workforce where coding and algorithmic reasoning are increasingly essential.

Grounded in constructivist learning theory, the study assumes that students build knowledge through active engagement and by making connections. Integrating algebraic and computational tasks allows students to construct meaning across disciplines rather than compartmentalize their learning. This interdisciplinary approach reflects the priorities of the Next Generation Science Standards (NGSS) and the Common Core State Standards for Mathematics (CCSSM), both of which emphasize problem-solving, reasoning, and the application of knowledge in authentic contexts.

Ultimately, this research aims to contribute to the limited but growing body of scholarship on interdisciplinary teaching, offering practical strategies for educators and insights for curriculum designers and policymakers. By examining how computational thinking can support algebraic reasoning, this study seeks to provide a model for bridging disciplinary boundaries and fostering deeper, more meaningful learning experiences for students.

Literature Review

The purpose of this literature review is to examine research surrounding the integration of computational thinking (CT) and mathematics education, particularly focusing on the role CT plays in developing students' algebraic reasoning. While there is growing interest in computational thinking as a 21st-century skill, studies that directly connect computational thinking to algebraic learning remain relatively rare (Hickmott et al., 2018). This review explores key concepts and definitions, examines overlapping skills between the two domains, and highlights both the potential and challenges of combining these disciplines.

Algebraic Thinking: A Foundational Skill

Algebraic thinking is a critical component of mathematical literacy and is widely regarded as a "gatekeeper" for higher-level mathematics and STEM pathways. Maudy et al. (2019) define algebraic thinking as "an approach to quantitative situations that emphasizes the general relational aspect with tools in the form of a symbol of letters" (p. 1). In other words, algebraic thinking requires students to move beyond arithmetic operations and engage in generalization, abstraction, and reasoning with variables.

Research has consistently demonstrated that students who struggle with algebra often face barriers to later academic success. Kilhamn and Bråting (2019) argue that algebra serves as a "language of mathematics," enabling students to express and manipulate abstract relationships. Without proficiency in algebra, students may find it difficult to engage in advanced science courses, data analysis, or computer science concepts.

However, many students have trouble transitioning from concrete arithmetic thinking to abstract algebraic reasoning. Duval (2006) explains that the cognitive complexity of algebra arises from the need to shift between multiple representations, such as symbols, graphs, and verbal descriptions, and to understand the underlying relationships between them. This complexity often leads to misconceptions. For example, students may misinterpret the equal sign (=) as a command to "compute the answer" rather than as a symbol representing a relationship between two expressions (Kilhamn et al., 2019). Such misunderstandings hinder students' ability to generalize and solve problems effectively.

Efforts to support algebraic thinking often focus on real-world problem-solving and multiple representations. For instance, the Common Core State Standards for Mathematics (CCSSM) emphasize mathematical practices such as reasoning abstractly, constructing arguments, and modeling with mathematics. Integrating computational thinking provides an avenue for achieving these goals by giving students tools to model problems dynamically and test their ideas in interactive, programmable environments.

Defining Computational Thinking

Computational thinking (CT) has emerged as a key skill for 21st-century learners. Wing (2010) defined CT as "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" (p. 1). Kotsopoulos et al. (2017) describe CT as encompassing both core concepts, such as sequences, loops, conditionals, and variables, and core practices, which include abstraction, debugging, and iterative problem-solving. These elements align closely with mathematical practices; for instance, abstraction in CT mirrors algebraic generalization, while debugging parallels the process of checking and revising mathematical reasoning.

In classrooms, CT often manifests through activities such as coding, robotics, and algorithmic problem-solving. Programs like Scratch, Code.org, and Python-based projects provide accessible platforms for students to engage with CT concepts. Importantly, CT is not limited to computer science classrooms; it is increasingly being integrated across disciplines, including science, engineering, and mathematics (Benton et al., 2017).

Connections Between Algebra and Computational Thinking

There are striking overlaps between algebra and computational thinking. Both disciplines involve symbolic reasoning, pattern recognition, and the systematic manipulation of abstract entities. For instance, variables play a central role in both fields. In algebra, a variable represents an unknown quantity or a placeholder within a function, while in computer science, a variable refers to a named location in memory used to store and manipulate data. Functions in mathematics are closely related to functions in programming, where they encapsulate a series of steps or computations. Similarly, coordinate planes are used in graphing algebraic relationships and in designing digital spaces, such as the stage in Scratch or the pixel grid in video game design.

Bråting and Kilhamn (2020) studied the implementation of programming in Swedish mathematics classrooms and found that programming activities enhanced students' algebraic thinking. However, they also observed challenges related to overlapping terminology. For example, the equal sign (=) signifies relational equality in algebra but assignment in programming. Without explicit instructions to address these differences, students may become confused.

To address this challenge, Bråting and Kilhamn (2020) recommended the use of visual supports, such as vocabulary word walls that clearly differentiate terms across disciplines. In practice, this might include side-by-side definitions and examples of shared vocabulary. For example, a word wall entry for "variable" might display a mathematical equation alongside a snippet of code, with annotations explaining the distinct uses of the term. Research also highlights the motivational benefits of combining CT and algebra instruction. Ke (2013) found that when students created computer games to demonstrate mathematical concepts, 41% of their coding activities involved analytical or quantitative reasoning. Students reported greater engagement and a more positive attitude toward mathematics when they were able to apply their knowledge to personally meaningful projects. These findings align with constructivist theories, suggesting that interdisciplinary learning contexts promote deeper understanding and intrinsic motivation.

Global Perspectives and Case Studies

Several international initiatives have explored the integration of programming and mathematics. In the United Kingdom, Benton et al. (2017) investigated primary school classrooms where programming was embedded into math lessons. They found that teachers could adapt computational tasks to meet diverse learning needs, fostering both mathematical understanding and computational fluency. In Sweden, Bråting and Kilhamn's (2020) work highlighted how national curriculum reforms encouraged the integration of CT into mathematics, particularly in grades 1–9. Their findings underscored the importance of teacher preparation and clear communication of disciplinary goals. In the United States of America**,** Ke's (2013) research examined the use of game design in Scratch to help students grow in their mathematical understanding. These global efforts provide valuable lessons for U.S. educators, illustrating both the potential benefits and the practical challenges of interdisciplinary teaching. One consistent theme across contexts is the need for professional development. Teachers must develop confidence in both disciplines and learn strategies for connecting concepts meaningfully, rather than treating programming as a stand-alone skill.

Gaps in Existing Research

Despite growing interest, the literature reveals several key gaps. One is the limited number of empirical studies. While theoretical arguments for integrating computational thinking and mathematics are strong, there are relatively few experimental studies examining the impact on student learning outcomes (Hickmott et al., 2018). Another gap is the lack of focus on algebra**.** Most existing research emphasizes elementary-level math or general problem-solving skills, with less attention given to algebra specifically. A third concern involves equity considerations, as few studies address how interdisciplinary approaches affect diverse student populations, including historically marginalized groups. These gaps highlight the need for studies like the present research, which investigates how concurrent instruction in Algebra 1 and computer science influences students' growth in algebraic thinking. By addressing this gap, the study contributes to a growing body of evidence supporting integrated, 21st-century learning models.

Summary

The literature suggests a strong conceptual alignment between algebraic and computational thinking, with shared emphasis on abstraction, symbolic reasoning, and problem-solving. International case studies demonstrate the feasibility of integrating these disciplines, while classroom-level research highlights potential benefits for student engagement and motivation. However, empirical evidence remains limited, particularly at the secondary level and in U.S. contexts.

This study builds on prior research by implementing a year-long intervention that explicitly connects Algebra 1 and computer science instruction. Through interdisciplinary lessons and targeted vocabulary supports, the study seeks to determine whether computational thinking can enhance students' mastery of algebraic concepts, thereby addressing a critical gap in both practice and scholarship.

Methods

Research Design

This study was conducted using quantitative research methods. Researchers (e.g., Varghese, et al., 2025) assert that quantitative research is most appropriate when the goal is to summarize and describe the characteristics of a specific population or situation without attempting to make predictions or establish cause-and-effect relationships. It is ideal in this study since the situation is one where the researcher(s) gathered data that highlights existing trends or behaviors. Further, quantitative research focuses on numerical data and statistical analysis to measure and quantify variables, aiming for objective and generalizable results (see Scharrer & Ramasubramanian, 2021).

Setting and Participants

The study was conducted in a large, urban public high school within the state of California. The school serves approximately 1,400 students from diverse socioeconomic and cultural backgrounds. Many families in the community face economic hardships, with over 80% of students qualifying for free or reduced-price lunch. The school has been identified as needing targeted support for mathematics achievement due to consistently low standardized test scores in math over the past five years.

Fourteen students participated in the study; all were enrolled in both Algebra 1 and Computer Science Explorations courses taught by the same teacher-researcher. The decision to focus on a single class allowed for a controlled integration of the two disciplines. The students ranged in age from 14 to 15 years old, with the group consisting of 13 male students and 1 female student. In terms of ethnic background, approximately 50% identified as Latino, 35% as African American, and 15% as Filipino. Academically, most students had struggled with mathematics in previous years, as reflected in their placement scores on district assessments. Several had repeated pre-algebra or failed portions of their middle school math curriculum. The small class size was ideal for piloting this interdisciplinary approach, as it allowed for individualized attention and frequent small-group work. While the sample size limits generalizability, it provides an important first step in exploring how computational thinking can support algebraic reasoning.

Instructional Approach and Intervention

The intervention was implemented over the course of the 2023–2024 school year**,** spanning two semesters. The overarching instructional goal was to integrate computational thinking into Algebra 1 lessons through explicit, interdisciplinary connections between mathematical and computer science concepts.

Integration Framework

The integration was guided by several key principles. One was to identify overlapping concepts between Algebra 1 and computer science**,** including variables, functions, graphing, and logic. Another was to use computational thinking practices, such as decomposition, abstraction, algorithmic design, to deepen students' problem-solving in math. A third principle emphasized creating project-based, real-world tasks to increase engagement and highlight the relevance of algebra**.** Finally, explicit vocabulary supports were provided to clarify overlapping terminology and prevent confusion.

Sample Interdisciplinary Lessons

- Variables and Storage:
  In Algebra 1, students learn that variables represent unknown quantities or numbers that can change. In computer science, variables are named locations in memory used to store information.
  - *Activity:* Students compared algebraic equations (e.g., $x + 5 = 12$) to simple code snippets where variables are assigned and manipulated.
  - *Venn Diagram Task:* Working in groups, students created Venn diagrams showing similarities and differences between the two uses of variables.
  - *Outcome:* Students reported increased clarity about how symbols can represent abstract ideas in both math and computer science.
- Cartesian Plane and Graphics:
  Algebraic graphing concepts were connected to computer programming projects using the coordinate plane.
  - *Activity:* Students used Code.org to plot points and design digital characters using coordinate pairs.
  - *Comparison:* They examined the differences between the traditional mathematical Cartesian plane (where the y-axis increases upward) and the coordinate system in coding environments (where the y-axis increases downward).
  - *Outcome:* This activity reinforced graphing skills and spatial reasoning.
- Functions as Algorithms:
  Algebraic functions were explored alongside computer science functions.
  - *Activity:* Students wrote simple functions in Python to represent linear relationships, such as calculating total cost given a fixed price and variable quantity.
  - *Extension:* Students were challenged to write programs that modeled real-world problems, such as determining cell phone data plan costs or predicting savings account balances over time.

Pacing

The integration was not constant but occurred during key Algebra 1 units, approximately every 2–3 weeks. This allowed time for both disciplines to progress through their respective curricula while maintaining meaningful overlap.

Data Collection

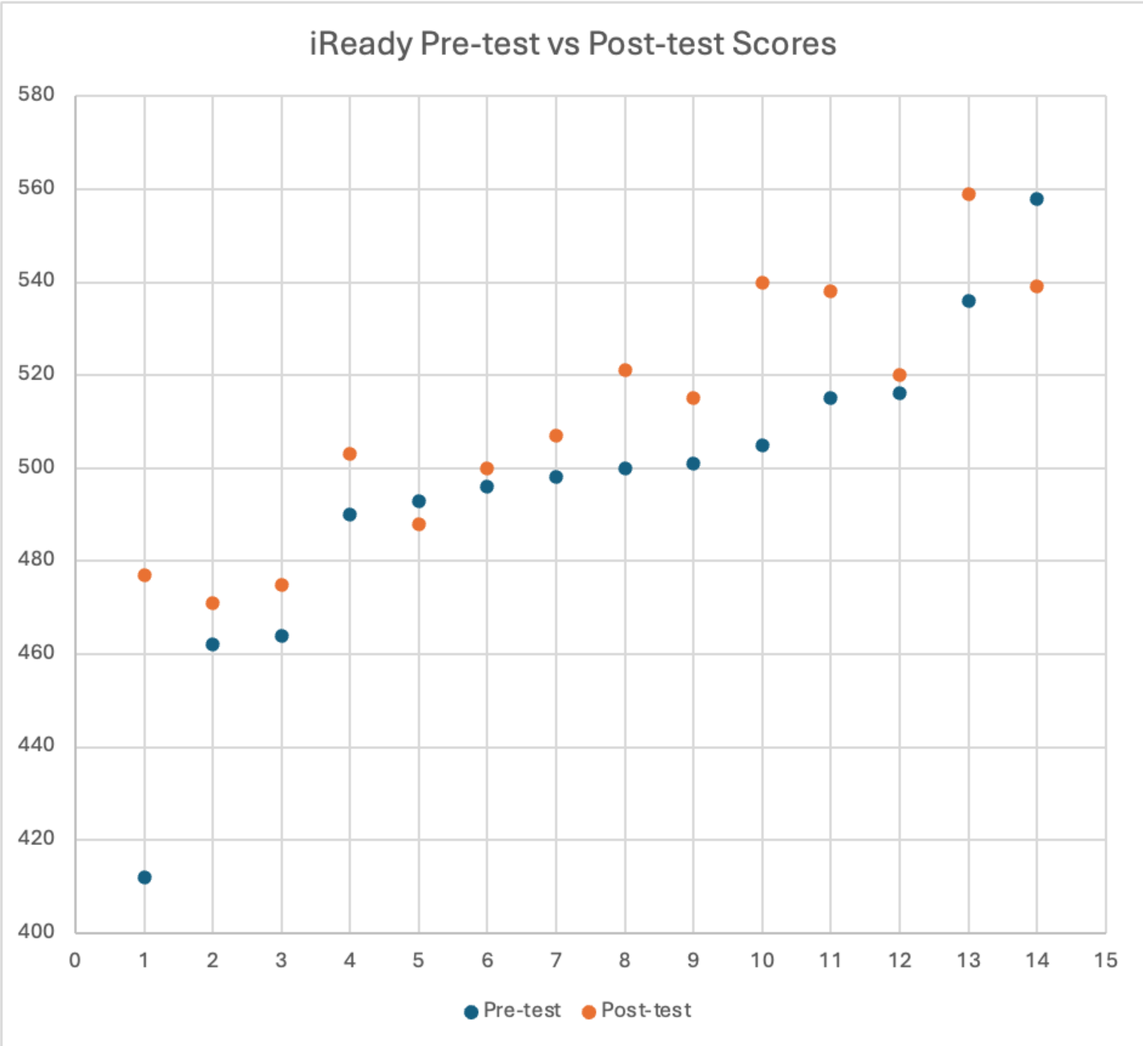
The primary instrument for measuring algebraic growth was the i-Ready Diagnostic Assessment, a computer-adaptive test aligned with Common Core State Standards. Students completed the i-Ready assessment at the beginning of the school year as a pre-test and at the end of the school year as a post-test. The pre-test, conducted in September 2023, established the students' baseline algebraic skills, while the post-test, conducted in May 2024, measured growth after the year-long intervention. The i-Ready system generates scale scores that reflect students' proficiency across multiple mathematical domains, including algebraic reasoning. These scores were used to quantify growth and analyze the effectiveness of the integrated approach.

Ethical Considerations

Ethical integrity was a priority in this study, given that the teacher also served as the primary researcher. Several steps were taken to protect participants and minimize potential bias. Students and their parents/guardians were fully informed about the purpose of the study, the nature of participation, and their right to withdraw at any time. Consent forms were provided in both English and Spanish. Participation was entirely voluntary; students who chose not to participate were still taught the same integrated curriculum but were excluded from data collection. To protect confidentiality, student names were replaced with numerical identifiers, and all data were stored securely in password-protected files. In addition, steps were taken to ensure the neutrality of the teacher-researcher. Students were explicitly told that their participation, or lack thereof, would not affect their grades, class standing, or relationship with the teacher. These measures ensured compliance with ethical standards for educational research, including those outlined by the American Educational Research Association (AERA).

Results

After all data was collected from the students' pre- and post-tests, it was compared to each other to see the students' initial progress. As seen in Graph 1, iReady Pre-test vs. Post-test, out of the 14 students who participated in this action research project, 12 showed growth from their pre-test to post-test. Of the two who had lower scores from the beginning of the year to the end of the year one only dropped 5 points while the other student dropped 23 total points. For the student who had a significant drop in scores from the beginning of the year to the end of the year, it could be concluded that maybe the student did not try as hard as he/she did at the beginning of the year, or that the student may have had a bad day during the end-of-year test. The largest gain in learning was from the student who had the lowest score at the beginning of the year in their pre-test. That student gained 55 points from the pre-test to the post-test by the end of the year. Although it would be nice to assume that this student had grown entirely from this treatment of the action research, there is not enough data to make that conclusion. It is impossible to know if it was the situation where the student did not try as hard as he/she should have during the pre-test or something else. This is one possibility that could contribute to the gains seen from the pre- to post-test. Overall, the initial findings of the data collected seemed to show a movement towards positive growth from the treatment provided.



Graph 1. iReady Pre-test vs Post-test Scores

To better analyze the results from the students' pre- and post-test scores, a paired T-test was conducted. This paired t-test was conducted comparing pre-test vs postest. Looking at Table 1 the average (mean) of the pre-test (InitialScaleScore) is 496.14 with a sample size of 14 students. Whereas the average of the post-test (FinalScaleScore) was 510.93, comparing the same sample size of students. The difference between these mean scores was 14.79 points, which showed that there was growth from the pre- to post-test.

Since the standard deviation of the post-test is lower than that of the pre-test, it suggests that the scores because slightly less variable after students we treated with both classes. It can also be seen that there is a decrease in the Standard Error Mean from pre- to post-test. This indicates that there is a more reliable estimate for the average of the final scores.

Table 1. Paired Sample Statistics

		Mean	N	Std Deviation	Std. Error Mean
Pair 1	Initial Scale Score	496.14	14	34.654	9.262
	Final Scale Score	510.93	14	27.227	7.227

To determine whether the difference in scores is statistically significant, we would typically conduct a paired samples t-test. This test would check if the mean difference (14.79) is significantly different from zero, given the standard deviations and sample size. The paired samples t-test results are as follows:

- t-statistic: -2.83
- p-value: 0.014

Since the p-value (0.014) is less than the typical significance level of 0.05, we can conclude that the difference between the initial and final scores is statistically significant. This helps us to conclude that the use of computer science being taught with Algebra 1 helps students to grow their understanding.

As seen in Table 2 the correlation between the initial and final scores is 0.827. This is a robust positive correlation, suggesting that students' pre-test scores are strongly related to their post-test scores. In other words, students who scored higher initially also tended to achieve higher post-test scores, and similarly for those with lower pre-test scores.

Table 2. Pair Sample Correlations

				Significance	
		N	Correlation	One-Sided p	Two-Sided p
Pair 1	Initial Scale Score & Final Scale Score	14	.827	<.001	<.001

A significantly strong positive correlation ($r = 0.827$, $p < 0.001$) implies that the initial and final scores share a consistent pattern. This suggests that the performance changes were systematic rather than random, and because of this, students' pre-test scores played a role in their post-test scores. Thus, while scores improved significantly on average, individual improvements were still aligned with their pre-test score levels. This correlation analysis adds another layer of understanding to the paired t-test, showing that, beyond the mean improvement, there is a stable relationship between students' initial and final scores. Overall, these results indicate that there was a meaningful and statistically significant increase in scores, with a strong evenness in improvement across the students.

Discussion

The purpose of this study was to explore how integrating computational thinking (CT) into Algebra 1 instruction can influence students' algebraic reasoning. The results reveal that the concurrent teaching of computer science and mathematics, with explicit connections between the two disciplines, resulted in statistically significant growth in students' algebraic understanding. While the sample size was small and limited to a single classroom, the results provide promising evidence that computational thinking can serve as a powerful tool to enhance mathematical learning.

Connecting Results to the Literature

The study's results align closely with prior research that highlights the conceptual overlap between computational thinking and algebraic reasoning. Bråting and Kilhamn (2020) argued that algebra and programming share common cognitive structures, such as symbolic manipulation, generalization, and abstraction. In this study, these overlaps were made explicit through integrated lessons focused on variables, functions, and graphing. Students reported that comparing the two disciplines helped them better understand difficult mathematical concepts.

For example, students initially struggled with the concept of variables, echoing the challenges noted by Kilhamn and Bråting (2019), who observed that the same symbols often carry different meanings across mathematics and computer science. Using side-by-side examples and visual vocabulary supports, students in this study gradually learned to distinguish between the definitions and apply them appropriately. By the end of the year, many students demonstrated fluency in switching between algebraic and computational contexts, suggesting that explicit attention to overlapping vocabulary can reduce confusion and deepen understanding.

The observed gains in engagement are also consistent with the work of Ke (2013), who found that students demonstrated higher motivation and analytical reasoning when engaged in game-based coding projects. In the present study, students expressed excitement about seeing their algebraic knowledge applied to real-world coding tasks, such as designing animations or building video games. These interdisciplinary projects transformed abstract mathematical ideas into tangible, interactive products, fostering a sense of ownership and creativity. This result reinforces the notion that computational thinking can provide a contextual bridge between mathematics and students' personal interests, increasing intrinsic motivation to learn.

The Role of Computational Thinking in Mathematical Growth

The statistically significant improvement in i-Ready post-test scores ($P = 0.014$) suggests that computational thinking was not merely a motivational tool but also a driver of conceptual growth. Several aspects of CT appear to have directly supported algebraic development. Students learned to use decomposition by breaking complex algebraic problems into smaller, manageable steps, mirroring the process of debugging code. For instance, when solving multi-step equations, students approached each step as a discrete "command," similar to how a program executes sequential instructions. Pattern recognition also played a key role. Identifying patterns is a core component of both coding and algebra, and activities such as creating repeating loops in code helped students recognize recurring structures in algebraic expressions, such as factoring or simplifying like terms. Algorithmic thinking further strengthened problem-solving skills, as algebra often requires following a logical sequence of operations. By writing algorithms in computer science, students internalized the importance of precision and order, skills that translated directly to solving equations and modeling functions. These computational thinking practices align with the Common Core State Standards for Mathematical Practice, particularly the emphasis on reasoning abstractly, constructing viable arguments, and modeling with mathematics. The results suggest that integrating CT may provide a natural pathway for meeting these standards while simultaneously preparing students for 21st-century computational literacy.

Practical Classroom Implications

The results of this study offer several practical takeaways for educators seeking to integrate computer science and algebra instruction. One effective approach is to start with shared concepts by focusing on areas of natural overlap, such as variables, functions, and coordinate planes. By highlighting these connections, teachers can help students transfer knowledge between disciplines and reduce cognitive load. Another strategy is to use dual-definition visual supports, such as vocabulary walls or comparison charts, to clarify overlapping terminology. For example, display the mathematical and computational meanings of "variable" side by side with visual examples from equations and code can prevent confusion.

Incorporating project-based learning also proves valuable, as coding projects give students opportunities to apply algebraic concepts in meaningful, creative ways. Examples include designing games that simulate linear motion, building calculators that solve equations, or programming animations that rely on geometric transformations. In addition, encouraging collaborative problem-solving allows students to learn from one another. Group projects, such as collaborative debugging sessions, not only deepen understanding but also build a positive classroom culture. Finally, leveraging free, accessible technology, such as Scratch, Code.org, and Python notebooks, makes it feasible to integrate CT even in resource-limited schools, lowering barriers for students and teachers new to coding. By adopting these strategies, educators can create engaging, interdisciplinary learning experiences that prepare students for both mathematical and computational challenges.

Equity and Access Considerations

This study was conducted in an urban school serving primarily low-income students of color, groups that are historically underrepresented in both advanced mathematics and computer science fields. The results indicate that integrating CT into Algebra 1 may serve as a tool for closing opportunity gaps.

Several students who began the year with below-grade-level math skills demonstrated the most significant growth. For example, one student improved by 55 points on the i-Ready assessment after initially struggling to understand basic algebraic concepts. This finding suggests that computational thinking can provide alternative entry points for students who may not thrive in traditional math classrooms. By framing algebra as a creative, applied discipline connected to technology, educators can help students see themselves as capable problem-solvers and innovators.

However, equity challenges remain. Students with limited access to technology outside of school may require additional support to fully engage with coding projects. Furthermore, female students and students from underrepresented groups may face societal stereotypes that discourage participation in computer science. Addressing these barriers will require intentional strategies, such as mentorship programs, culturally responsive teaching practices, and efforts to ensure equitable access to technology.

Challenges and Limitations in Practice

While the integrated approach yielded positive results, several challenges emerged that align with issues identified in the literature. One challenge was vocabulary Confusion. Students initially struggled with terms that have different meanings in math and computer science, confirming Bråting and Kilhamn's (2020) finding that overlapping language can be a barrier to interdisciplinary learning. Addressing this challenge required explicit teaching and repeated reinforcement. Another challenge was time constraints, as balancing two curricula within a single school year proved difficult. Teachers attempting similar interventions may need to coordinate closely with administrators to ensure adequate instructional time. A third challenge was the imbalance in student motivation. Some students were more motivated by the creative aspects of coding than by the abstract reasoning required in algebra. While this enthusiasm is valuable, it sometimes led to resistance when lessons shifted back to purely mathematical content. Blending the two disciplines consistently helped mitigate this issue.

Limitations, Generalizability, Validity, and Reliability

While the results of this study are promising, it is important to interpret them within the context of the study's constraints. Every research project has limitations that affect the extent to which findings/results can be applied to other settings, populations, or future investigations. This section addresses the limitations of the current study and evaluates its generalizability, validity, and reliability. By critically examining these factors, we can better understand the study's strengths and weaknesses and identify avenues for future research.

Limitations of the Study

Several factors limited the scope and potential impact of this research. The study was conducted with a single class of 14 students; a relatively small group compared to most educational research studies. While the small sample allowed for close observation and individualized support, it significantly limits the statistical power of the results. With only 14 participants, outliers had a disproportionately large effect on the overall averages and statistical tests. For example, one student's dramatic 55-point increase between the pre-test and post-test influenced the class mean more strongly than would be the case in a larger sample. Conversely, a single student's sharp decline in performance could obscure patterns of success. A larger sample would help balance these extremes and provide more stable, generalizable results.

The absence of a control group presents a major limitation. Without a group of students learning Algebra 1 through traditional instruction alone, it is difficult to definitively attribute the observed growth solely to the integration of computational thinking. Other factors, such as natural year-long academic development, improvements in teaching practices, or external tutoring, may have contributed to the gains. Including a control group in future studies would allow for more rigorous comparisons and strengthen the causal claims about the effectiveness of this interdisciplinary approach.

The teacher's dual role as instructor and researcher introduces potential for bias. Despite safeguards such as anonymized data and assurances that participation would not affect grades, the teacher's investment in the project may have influenced student performance. There may have also been subconscious tendencies to interpret results more positively. For example, students may have felt pressure to perform well on the post-test to please the teacher-researcher, despite efforts to emphasize voluntary participation and neutrality.

Finally, the one-year timeframe captured only short-term outcomes. While this period was sufficient to observe short-term growth, it did not allow for examination of long-term effects. It remains unknown whether the gains in algebraic reasoning will persist as students progress to higher-level math courses. Longitudinal studies are needed to determine the long-term impact of integrating computational thinking into mathematics instruction.

Generalizability

Because the study was conducted in a single, urban high school with a small and demographically specific group of students, generalizability is limited. The participating students were predominantly Latino and African American, with a mix of low-income and working-class backgrounds. Schools in rural areas, affluent suburban districts, or international contexts may face different challenges and opportunities in implementing similar programs.

Furthermore, the success of the intervention depended in part on the unique skill set of the teacher-researcher, who was certified to teach both mathematics and computer science. In many schools, these subjects are taught by different teachers with limited collaboration. Replicating the program in settings with multiple instructors may require additional coordination and professional development.

To strengthen generalizability, future studies should involve larger and more diverse samples across multiple schools or districts, test the model in settings with varying technological resources, and explore different instructional approaches such as co-teaching or virtual collaboration. While this study provides valuable insights, its findings should be regarded as preliminary and best understood as a pilot effort rather than definitive evidence for widespread implementation.

Validity

Several factors influence the validity of this research. One such is the internal validity. Because the study lacked a control group, it is not possible to determine if the observed growth in algebraic thinking can truly be attributed to the integration of computational thinking. Other explanations remain possible, such as students may have been improved simply through natural cognitive development over the school year. Gains could also reflect the teacher's instructional effectiveness rather than the interdisciplinary approach itself.

Another factor is the construct validity. The i-Ready assessment was chosen because it aligns with Common Core State Standards and provides detailed data on algebra-related skills. However, standardized tests may not fully capture students' conceptual understanding, creativity, or problem-solving processes. Incorporating qualitative evidence, such as student interviews or performance-based tasks, would provide a more complete picture of their growth and strengthen construct validity.

Lastly, external validity poses another factor. The demographics and conditions of the study limit broad application. For instance, schools with fewer technological resources may face challenges not encountered here, making replication less straightforward.

Reliability

The i-Ready assessment, a nationally normed computer-adaptive test with established reliability metrics, provided a standardized measure of student growth across participants. Instruction was delivered through a structured intervention plan with documented lesson sequences, ensuring that all students experienced the curriculum in a consistent way. Data collection was also systematic, with pre- and post-tests administered under identical conditions to reduce variability.

Even with these strengths, some factors may have affected reliability. The small sample size of 14 students meant that random fluctuations had a greater influence on overall results. In addition, teacher observation logs, while helpful for adding context, were subjective and could reflect researcher bias. Future studies could strengthen reliability by including multiple researchers or external evaluators in the data collection and analysis process, helping to minimize individual bias.

Summary of Strengths and Constraints

In summary, this study provides early evidence that integrating computational thinking into Algebra 1 can improve student outcomes. However, its limitations, including small sample size, lack of control group, and contextual specificity, require caution when interpreting the findings. By acknowledging these constraints, the research establishes a foundation for future investigations that can build on its insights while addressing its weaknesses. The next section will provide recommendations for educators, researchers, and policymakers to expand and refine this innovative approach to interdisciplinary teaching.

Conclusion

This study set out to explore how teaching Computer Science and Algebra 1 concurrently can influence students' development of algebraic thinking. By integrating computational thinking into algebra instruction through interdisciplinary activities, the aim was to support students' conceptual understanding in both domains. The data collected from pre- and post-assessments indicated a statistically significant improvement in students' Algebra 1 scores, thus suggesting that the method of instruction of both disciplines at the same time supported the development of algebraic thinking.

Additionally, the strong correlation between pre- and post-test scores suggests that while individual growth varied, the improvement followed a consistent and reliable pattern. Students appeared to benefit from lessons that connected variable use and coordinate plane graphing in both coding and algebra contexts. Their engagement and deeper understanding were enhanced through the shared language and skills across both subjects.

However, the study also revealed some challenges, such as students' desire to spend more time on computer science projects and the small, non-random sample size, which limits broader application. Since the study lacked a control group, it is difficult to isolate what could have been the exact cause of the student's improved scores.

Recommendations

The first and most important recommendation should be that future research involve larger groups of students. There should also be a control group not exposed to computer science instruction to help with identifying the usefulness of the interdisciplinary teaching. Educators should consider developing units where Algebra 1 and Computer Science content share an overlap. Duval shares that "the root of the troubles that many students have with mathematical thinking lies in the mathematical specificity and the cognitive complexity of conversion and changing representation" (Duval, 2006, p. 127). These can be found in such concepts as variables, graphing, standards for mathematical practices, and problem-solving strategies. Since overlapping terms like "=" or "variables" can lead to confusion, classrooms should incorporate tools such as word walls or dual-meaning charts to distinguish terms used differently across subjects.

Given students' enthusiasm for computer science, lessons could be structured to include some component of coding. Since students are drawn into the games, making use of the game authoring environment combines symbolic and hands-on math representations, helping children connect formal and informal math understanding (Ke, 2013). For example, math problems could be framed within coding challenges or real-world programming scenarios. Educators would benefit from training in computational thinking and interdisciplinary instructional strategies to effectively merge the two content areas. Adding qualitative data, such as open-ended student surveys or observational notes, could provide more insight into how and why these interdisciplinary connections aid learning.

Acknowledgements

The authors would like to gratefully acknowledge funding for this project from the National Science Foundation through the Robert Noyce Master Teacher Fellowship (MTF) Track (Award No. 1949973). As well as Dr. Diana Faatai, Dr. Kathryn Theiss, Dr. Anthony Normore, Dr. Kamal Hamdan, the CISE team, and Erin Barrett. This research was granted IRB approval by the IRB Board at California State University, Dominguez Hills (IRB-FY2024-9).

References

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education, 3*(20), 115–138. https://doi.org/10.1007/s40751-017-0028-x.

Bråting, K., & Kilhamn, C. (2020). Exploring the intersection of algebraic and computational thinking. *Mathematical Thinking and Learning, 23*(2), 170–185. https://doi.org/10.1080/10986065.2020.1779012.

Duval, R. (2006). A cognitive analysis of problems of comprehension in a learning of mathematics, *EducationalStudies in Mathematics*, 2006, Vol. 61, No. 1/2, Semiotic Perspectives in Mathematics Education: A PME Special Issue (2006), pp.103–131. https://doi.org/10.1007/s10649-006-0400-z.

Fisher, L. M. (2016). A decade of ACM efforts contributes to computer science for all. *Communications of the ACM, 59*(4), 25–27., https://doi.org/10.1145/2892740.

Hickmott, D., Holmes, K., & Prieto-Rodriguez, E. (2018). "A scoping review of studies on computational thinking in K–12 mathematics classrooms. *Digital Experiences in Mathematics Education,4*(1),48–69., https://doi.org/10.1007/s40751-017-0038-8.

Ke, F. (2013). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. *Computers & Education, 73,* 26–39. https://doi.org/10.1016/j.compedu.2013.12.010.

Kilhamn, C., & Bråting, K. (2019). *Algebraic thinking in the shadow of programming*. Eleventh Congress of the European Society for Research in Mathematics Education, Utrecht University, Utrecht, Netherlands. HAL-02429028

Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). Apedagogical framework for computational thinking. *Digital Experiences in Mathematics Education, 3*(2), 154–171. https://doi.org/10.1007/s40751-017-0031-2.

Maudy, S. Y., Suryadi, D., & Mulyana, E. (2019). 'Level of student' algebraic thinking. *Journal of Physics: Conference Series, 1157*, 042057. https://doi.org/10.1088/1742-6596/1157/4/042057.

Scharrer, E., & Ramasubramanian, S. (2021). *Quantitative research methods in communication: The power of numbers for social justice*. Routledge

The White House. (2016, January 30). *Fact sheet: President Obama announces Computer Science for All initiative*. Obama White House Archives. https://obamawhitehouse.archives.gov/the-press-office/2016/01/30/fact-sheet-president-obama-announces-computer-science-all-initiative

Varghese, K.A., Ranwah, B.R., Varghese, N., & Varghese, N. (2025). *Research methodology and quantitative techniques: A guide for interdisciplinary research*. Routledge.

Wing, J. (2010, November 17). *Computational thinking: What and why?* Pittsburgh, PA: Carnegie Mellon University, School of Computer Science. Retrieved from https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Author Information

James Knutson
https://orcid.org/ 0009-0000-5040-4116
California State University, Dominguez Hills
1000 E. Victoria St.
Carson, Ca 90747
United States of America

Dr. Kristen Stagg
https://orcid.org/0009-0003-7507-6675
California State University, Dominguez Hills
1000 E. Victoria St.
Carson, Ca 90747
United States of America
Contact e-mail: *kstagg@csudh.edu*