Liu, R., Gonczi, A., & Maeng, J. (2026). Learning, implementation, and outcome: A case study of an elementary teacher's integration of scratch programming in science instruction. *Contemporary Issues in Technology and Teacher Education, 26*(1), 116-145.

Learning, Implementation, and Outcome: A Case Study of an Elementary Teacher's Integration of Scratch Programming in Science Instruction

Ruohan Liu
Seattle University

Amanda Gonczi
Michigan Technological University

Jennifer Maeng
University of Virginia

The Scratch programming language makes computing accessible for young students and provides valuable support for non-computer-science instruction. Guided by the Interconnected Model of Professional Growth (IMPG), this case study explored an elementary teacher's learning, implementation, and outcomes in Scratch integration after participating in a year-long science, technology, engineering, and mathematics professional development (PD) program. By analyzing data from multiple sources, the findings suggest that the PD program was effective in fostering positive Scratch-related beliefs, as well as technological and pedagogical knowledge. Following PD participation, the teacher was able to implement concrete strategies to engage students in Scratch-integrated science learning. However, a significant challenge for the teacher, and an evident influence on student-created Scratch projects, was limited programming content knowledge. Based on the findings, the authors offer actionable implications for future research and PD design.

To fully participate in a technology-driven world, children need early opportunities to engage with computer programming to develop a foundational understanding of how digital technologies work (Febrian & Lawanto, 2018; Rich et al., 2018). One avenue to enable equitable access to programming in elementary classrooms is by integrating programming into elementary instruction (Mladenovic et al., 2016). Programming provides opportunities for creative expression, communication, and collaboration. Students can express ideas through the creative design of programming artifacts and collaborate on problem-solving (Barr & Stephenson, 2011; Bers et al., 2010; Kafai & Burke, 2014).

Of the variety of programming tools aiming to engage K-12 students in learning computer programming, Scratch is a powerful tool to introduce programming and connect programming with other elementary curricula (Smith & Burrow, 2016). Designed with accessible affordances and a novice-friendly interface, Scratch enables students to explore disciplinary concepts and demonstrate understanding in an engaging, creative, and collaborative way (Resnick et al., 2009).

For example, in elementary mathematics, students can create Scratch projects to model solutions to math problems (Rodríguez-Martínez et al., 2020). Students can demonstrate English language arts knowledge and computer science (CS) skills by using Scratch to build stories (Pektas & Sullivan, 2021). Scratch is a particularly popular tool to support science learning, allowing students to explore concepts by building animations, interactive games, and simulation-based models (Aksit & Wiebe, 2020; Ogegbo & Ramnarain, 2021; Tucker-Raymond et al., 2019).

Despite the value of learning programming for students, many elementary teachers remain uncertain as to why it is important to integrate programming into their instruction and how they can effectively integrate it to support instruction in their subject area (Broza et al., 2023). Particularly, elementary teachers often hesitate to include programming in their instruction due to limited knowledge of programming, uncertainty about how programming can meaningfully augment their existing instructional practices and the potential benefits of programming for students' learning of other subjects (Sentance & Csizmadia, 2017).

Low teacher buy-in would ultimately constrain students' opportunities to engage in integrated programming activities in class. As such, the present study implemented a year-long science, technology, engineering, and mathematics (STEM) professional development (PD) program with a central focus on building elementary teacher capacity to integrate Scratch to enhance disciplinary instruction. This study used a case study design to gain an in-depth understanding of one elementary teacher's Scratch-related beliefs, knowledge, instructional practices, and student outcomes following PD participation.

Scratch is a powerful tool for engaging elementary students in programming and an accessible instructional resource for elementary teachers with limited programming experience. With its block-based design, Scratch offers several unique affordances that make it particularly well-suited for elementary instruction. First, Scratch is *accessible and scaffoldable*. Anyone can learn programming with Scratch, regardless of age and background, and learners can start from the basics of

programming and gradually build increasingly complex projects (Dasgupta & Hill, 2018; Resnick et al., 2009).

Second, Scratch supports *interdisciplinary learning*. It can be used to connect computational knowledge with other disciplines. Using Scratch, students can model solutions to disciplinary problems and create animations to demonstrate disciplinary understanding (e.g., see Burke & Kafai, 2010, 2012; Pektas & Sullivan, 2021).

Third, Scratch is *interactive and shareable*.It offers an online community where users can share, comment, and collaborate on projects. This community-based approach creates an environment that allows students to showcase their creations and receive peer feedback.

Fourth, Scratch *promotes self-expression and creativity*. The multimedia features of Scratch allow learners to create personally meaningful games, stories, and animations to express ideas and demonstrate creativity (Brennan & Resnick, 2012; Kafai & Burke, 2014). However, despite the well-recognized affordances of Scratch, knowledge regarding ways Scratch can be meaningfully integrated in elementary instruction to support student learning of other subjects and ways to build elementary teachers' capacity to fully leverage its potential is limited.

Relevant Works

The existing literature has explored the potential of Scratch as an interdisciplinary teaching tool. Lopez and Hernandez (2015) highlighted the value of Scratch as a computational modeling tool for teaching physics. They demonstrated sample Scratch programs to model the phenomenon of free fall and simple harmonic motion and argued that Scratch language is accessible for elementary and secondary students to use to express their own models, make predictions, and evaluate the results of models.

Rodríguez-Martínez et al. (2020) examined the potential of Scratch in supporting elementary students' math learning through a quasi-experimental design with sixth-grade students. This study identified an improvement in participants' math problem-solving performance after engaging in programming activities using Scratch.

Parsazadeh et al. (2021) implemented a quasi-experimental design to examine the effectiveness of using Scratch-based digital storytelling for English language teaching for fifth-grade students. Results of this study indicated that Scratch-based digital storytelling can effectively motivate participants' English language learning and enhance their performance.

While the existing literature has highlighted the benefits of integrating Scratch into elementary education, to successfully integrate Scratch in disciplinary teaching, elementary teachers need professional support to build their knowledge and skills in using Scratch as an interdisciplinary tool. Most current studies on Scratch-related professional support focus on its use as a general programming tool (e.g., Montiel et al., 2021; Rich et al., 2021), with limited attention to the specific knowledge and support teachers need to connect Scratch programming to subject content. To

address this gap, the present study explored an elementary teacher's development in using Scratch as a tool for teaching science concepts.

Conceptual Framework

This study is guided by the Technology, Pedagogy, and Content Knowledge (TPACK) model and the Interconnected Model of Professional Growth (IMPG). TPACK provides a valuable lens to understand the type of knowledge educators need to be equipped with to effectively integrate technology into teaching. It emphasizes the interactions between content knowledge (CK), pedagogical knowledge (PK), and technological knowledge (Koehler & Mishra, 2009).

In the context of science education, TPACK involves the knowledge regarding the characteristics, functions, and use of technology to support science teaching. It also includes the ability to leverage the affordances of technology to engage and enhance students' learning and evaluate their science understanding (Jang & Tsai, 2012). For the scope of this study, we focused on two specific components of TPACK in the context of an elementary teacher's use of Scratch for science instruction: (a) technological content knowledge (TCK), the teacher's understanding and beliefs regarding ways Scratch can be used to enhance the teaching of science concepts and (b) technological pedagogical knowledge (TPK), the teacher's ability to use Scratch to make science content engaging and accessible to students.

The IMPG served as the analytical model for this study. IMPG delineates the domains and pathways of teacher change in professional learning (Clarke & Hollingsworth, 2002). It considers teacher development as a nonlinear process and identifies four distinct domains related to teacher development: (a) External domain - information, resources and support teachers obtain outside of the classroom, such as PD programs and workshops; (b) Personal domain - teacher knowledge, beliefs, and attitudes; (c) Domain of practice - teacher professional experimentation and teaching practice; and (d) Domain of consequence - outcomes realized as a result of implementing new pedagogies (e.g., student learning outcomes, motivation, classroom management, etc.). IMPG postulates that teacher changes in one domain can influence changes in another domain through the mediating processes of "reflection" and "enaction" (Figure 1).

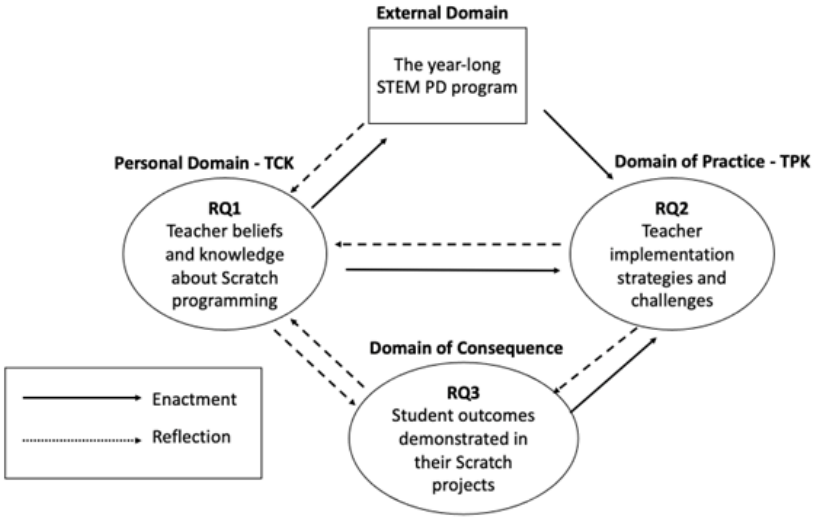
Guided by TPACK and IMPG, we examined an elementary teacher's development in Scratch integration across three domains after participating in a year-long STEM PD program: (a) Personal domain, focusing on the teacher's TCK, particularly the beliefs and knowledge about using Scratch programming in science teaching; (b) Domain of practice, exploring the teacher's PCK, specifically the strategies for implementing Scratch in science teaching, and the challenges encountered; and (c) Domain of consequence, examining student science learning outcomes, as demonstrated in their Scratch projects. The following research questions (RQs) guided this study:

1. What were the elementary teacher's Scratch-related beliefs and knowledge following PD?

2. Following PD, how did the elementary teacher implement Scratch programming into science instruction, and what challenges were encountered?

3. After learning science with Scratch, how did students use Scratch to express their science understanding?

Figure 1
The Operationalization of TPACK and IMPG in the Present Study



Methodology

This study adopted Stake's (1995) single instrumental case study design. Specifically, we examined a single bounded case — an elementary teacher integrating Scratch into science instruction — to gain insights into the broader issue: the processes, challenges, and outcomes associated with elementary teachers' integration of Scratch into disciplinary instruction following PD. Following Stake's case study guidelines, we collected and analyzed multiple sources of data to obtain a holistic case understanding (Table 1).

Study Context and Case Selection

The present study was situated in a year-long PD program that aimed to support elementary teachers to implement technology-enhanced STEM instruction. One of the central foci of the PD was to help teachers effectively integrate Scratch programming to support elementary instruction. The PD was composed of two phases: a 5-day in-person summer institute implemented in summer 2022, followed by school-year coaching support provided from fall 2022 through spring 2023.

Table 1
Alignment of the IMPG Model With Research Questions and Data Sources

IMPG Model	Research Questions	Data Sources
Personal Domain	1. What were the elementary teacher's Scratch-related beliefs and knowledge following PD?	- Teacher Interview - Teacher's Scratch project and lesson plan
Domain of Practice	2. Following PD, how did the elementary teacher implement Scratch programming into science instruction, and what challenges were encountered?	- Lesson plan - Videotaped lesson recordings (n = 5) - Teacher Interview
Domain of Consequence	3. After integrating Scratch programming in science instruction, how did students use Scratch to express their science understanding?	Student Scratch projects (n = 13)

The summer institute emphasized hands-on and active learning. Three PD facilitators worked with a small cohort of four elementary teachers to explore strategies to integrate technology and engineering effectively into elementary instruction. The PD highlights the interdisciplinary value of Scratch, demonstrating how the computational concepts underlying Scratch code blocks can be communicated to students, while also introducing ways to use Scratch as a tool for representing and assessing disciplinary understanding. Specifically, PD facilitators explicitly introduced the Scratch platform, modeled its interface and use, explained the Scratch coding blocks and the associated computational concepts, and demonstrated techniques to integrate Scratch to support teaching of elementary subjects.

Following the demonstration, teachers had the opportunity to reflect, share, and discuss ideas about how to use Scratch in their lessons. As part of the expectation for participating in the summer institute, teachers were encouraged to create a Scratch project that they could use to introduce a concept in their subject and build a Scratch-integrated lesson plan that they could implement during the school year. For the school-year coaching, facilitators maintained consistent individual communication with teachers through emails and meetings using the Zoom video-conference platform to address their questions, review their lesson plans, provide feedback and classroom support.

According to Stake (1995), an instrumental case is studied to provide insights into the broader issue. For this study, one of the four teachers, Mr. Miller (pseudonym), was selected as the instrumental case because his experiences of learning and implementing Scratch in science teaching could shed light on the wider issue of how the PD program impacted elementary teacher integration of programming in elementary instruction.

Mr. Miller was selected because his case met the following criteria (Stake, 1995): (a) Relevance. Mr. Miller attended the entire PD program and used Scratch programming in lessons as expected; (b) Data richness. Compared with other participants, we had access to a more extensive dataset about

Mr. Miller, which allowed us to extract evidence from multiple data sources; and (c) Accessibility. Mr. Miller was willing to share his experience and perspectives. We were able to maintain regular contact throughout the year, which allowed us to collect sufficient data about his case and conduct member checks on his data.

At the time of the study, Mr. Miller was a fifth-grade teacher with 8 years of teaching experience. He worked in a rural elementary school in a mid-Atlantic state in the U.S. and taught fifth-grade science and geometry. He reported having minimal experience with Scratch before the PD.

Data Collection and Analysis

Data collection took place in spring 2023, as participants implemented their Scratch-integrated lesson plans in classroom settings. We collected data from multiple sources: (a) Five videotaped lesson recordings (273 minutes); (b) a semistructured interview with the teacher; (c) instructional artifacts, including a Scratch project and lesson plan; and (d) students' Scratch projects (Table 1).

Data were analyzed using a five-step inductive approach (Thomas, 2006):

1. Reviewed the data preliminarily to build familiarity;

2. Examined data to identify specific texts/segments related to the RQs;

3. Created codes to denote the essence of the texts/segments related to RQs;

4. Compared and collapsed codes to construct themes relevant to each RQ;

5. Continued to review, validate, and refine the themes through peer debriefing and member checks.

Following these procedures, for RQ1, we coded the teacher's interview data to develop themes regarding his Scratch-related beliefs, and we coded the Scratch project and lesson plan he built during PD to explore his demonstration of Scratch-related knowledge. For RQ2, we coded the teacher's interview data, lesson plan, as well as lesson recordings to understand how he implemented a Scratch-integrated lesson unit and the challenges he encountered. Each dataset was coded independently before merging and consolidating the codes into overarching themes.

For RQ 3, we coded the Scratch projects students created at the end of the lesson unit. Given Scratch's recognized capabilities in fostering knowledge expression and creativity (Kobsiripat, 2015), in data analysis we focused particularly on how students designed their Scratch project, how they communicated science ideas, and what their creative endeavors looked in Scratch projects.

Results

RQ1. Scratch-Related Beliefs and Knowledge After PD

We identified three themes regarding Mr. Miller's beliefs and knowledge about Scratch programming:

1. Expressed positive beliefs about integrating programming in science instruction;

2. Recognized the multidimensional value of Scratch in science instruction; and

3. Developed technological and pedagogical knowledge about Scratch in science instruction.

These themes and supporting evidence are elaborated in the following section.

Positive Beliefs About Integrating Programming in Science

This theme is related to Mr. Miller's beliefs that it is essential to include programming in elementary instruction. He discussed the necessity of integrating programming and expressed a genuine willingness and keen interest in including programming instruction in his lessons. When asked his opinion about integrating programming, Mr. Miller stated, "I'd love to explore more about it [programming], like, how can I encourage students to program? It is more kind of a universal thing across the board."

He viewed programming as an essential skill that students should be equipped with, stating that learning programming would help students prepare for more advanced learning of computer technology:

> When they [students] started middle school, they'll get a lot more computer classes and tech classes … so I like them to have that, those wealth of programming experiences, before I send them off to middle school, and they're gonna' start making choices that will kind of start affecting where their education might take them later on.

Recognizing the necessity of integrating programming, Mr. Miller expressed willingness and interest in integrating programming, stating, "I feel really comfortable. I could handle it. You know, I mean, like it wouldn't be that hard to incorporate."

Additionally, he pointed out the alignment between programming and science learning practices, further reinforcing the value and synergy between programming and science instruction, noting,

> It [programming] definitely parallels, like, when we're doing experiments or lab, there are similar design processes. … We had students doing similar things, like, with their computer

> programming things, like that problem-solving, tweaking, going back to all those things.

Multidimensional Value of Scratch

This theme specifically relates to Mr. Miller's perceptions about the value of Scratch as an age-appropriate instructional tool to support integration of programming at the elementary level. Mr. Miller perceived Scratch as a programming tool that provides students with a new way for knowledge representation, reinforcement, and assessment. He perceived that Scratch allows students to demonstrate knowledge and express ideas creatively. Additionally, he pointed out the value of Scratch in fostering student interaction and collaboration.

He described Scratch as an "age-appropriate" tool that provides students an efficient and meaningful way to represent science concepts:

> If it's a unit, that's harder to make a project for. So, like, layers of the earth, we could go and make a 3D model, we could find how to do that. But here, I could do a quick thing, and students can manipulate. It could just be manipulated that model a lot more and show more information with Scratch.

Mr. Miller believed that by integrating a science concept in Scratch programs, students could reinforce their understanding of the concept. As he stated, "I can kind of get them to have to repeatedly think about the concept. I want them to put it in the [Scratch] program, that you have to keep thinking about it, to program it."

He also commented on the value of Scratch in assessing students' mastery of science knowledge. For example, when he taught the concept of sound with Scratch, he could see how students articulated their understanding in Scratch programs: "I'm trying to figure out where they got this from, how they would tell me about sound." He saw Scratch as a new way for students to demonstrate learning and creatively express ideas. He valued the open-ended features of Scratch and its potential for fostering student creativity, noting,

> Scratch offers a broad range of creative ways for the kids to tell me what they learned. … There are no real limits on it, except for their imagination, whatever they can kind of picture and do with it. I really liked that aspect of it.

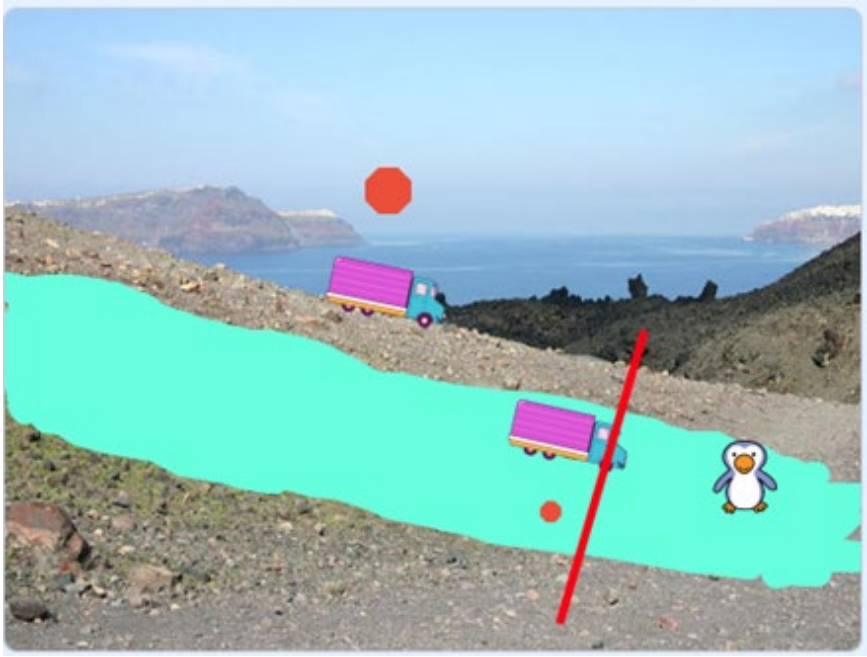
He also appreciated the value of Scratch in fostering student collaboration. He shared that Scratch offers an efficient platform for students to review their peers' work and help each other with problem-solving: "They're engaged with their friends that are trying to help them out. So there's an aspect of that I like a lot, too."

Development of TCK and TPK

The Scratch project and the lesson plan Mr. Miller developed during the summer PD workshop demonstrated that he developed both technological and pedagogical knowledge about Scratch integration.

TCK. TCK involves understanding how to use technology to teach and enhance student understanding of specific content knowledge (Koehler & Mishra, 2009). Mr. Miller created a Scratch animation to illustrate the concept of friction (Figure 2). To demonstrate the impact of different surface types on the amount of friction and its subsequent influence on velocity, he designed an illustrative animation scenario: two identical trucks race on two different types of surfaces (rough vs. smooth). To demonstrate the impact of friction on velocity, he programmed the truck moving on the smooth surface to reach the finish line earlier than the truck moving on the rough surface. This animation allowed students to observe how friction varies across surface types and how it ultimately influenced velocity.

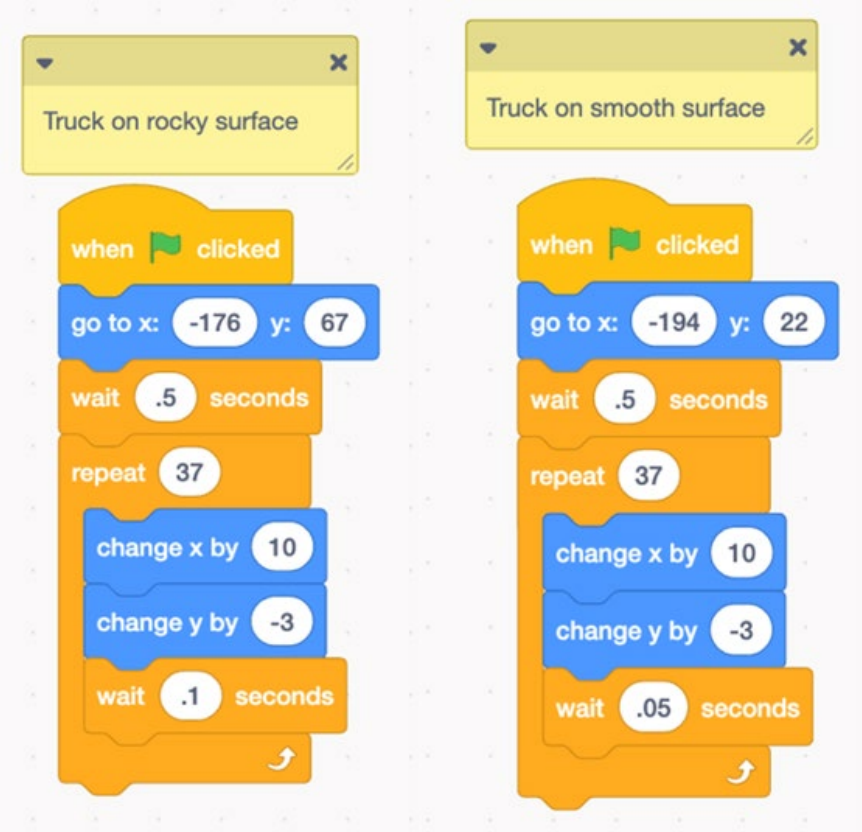
Figure 2
Scratch Animation Scenario



In his Scratch codes (Figure 3), he used an *event* block (i.e., when the green flag clicked) to trigger the program, a *motion* block with specified variable values to ensure the trucks move to a desired position, and *control* blocks (i.e., wait and repetition loop) to ensure that a sequence of instructions are repeated a specific number of times to achieve a desired outcome (i.e., configure the different wait time for the two trucks to arrive at the finish line in different time). His Scratch project indicated the development of a fundamental understanding of the programming concepts (i.e., events,

repetition loop, variable, and algorithm) introduced in the PD, as well as his ability to use Scratch to demonstrate a specific science concept.

Figure 3
Scratch Codes



TPK. TPK requires the understanding of pedagogical strategies to use technology to teach specific subject content (Koehler & Mishra, 2009). Mr. Miller successfully built and implemented a Scratch-integrated lesson plan that addressed both science and CS learning standards. His lesson plan outlined learning objectives and activities that centered around representing and demonstrating science knowledge with Scratch. This indicated that Mr. Miller developed pedagogical knowledge about using Scratch to support science instruction.

The Scratch lesson plan (Figure 4 and 5) was intended to introduce students to continents and oceans. He included specific science learning standards, CS learning standards, and learning objectives in his lesson plan (Figure 4). The stated learning objective indicated that he integrated Scratch primarily as a knowledge demonstration tool; he expected students to demonstrate science knowledge through their Scratch program.

Figure 4
Lesson Plan Part A: Learning Standards and Instructional Objectives

Lesson Segment & Time Est.	Materials	Instructional Sequence	Teacher/Student Actions
Introduction	Scratch Programming, Chrome books. Parts of a Plant - Review - Hook World Map Picture to Upload	1. *Sample program of a character labeling….* 2. *Teacher will examine basic introduction to programming with scratch for students*	Teacher will demonstrate a scratch program that labels a Plant…students will listen! Teacher will demonstrate basic scratch programming
Body		1. Teacher will pair students with partners – work independently, but can share knowledge collectively with their partner. 2. Students will be encouraged to examine and explore the scratch program as they have their character: a. Move to each continent i. Identify Each continent correctly ii. Label Each Continent correctly b. Move to Each Ocean i. Identify each ocean correctly ii. Label each ocean correctly. 3. When finished each student will have an animation that correctly labels each ocean and continent.	Students will begin working on their presentation. When they encounter problems, they will first explore, before consulting partner and exploring options with their partner, before they may consult one other group, before they approach teacher. Extension: IF students get done early, they can work on developing sound, and animations on their presentation.
Closure Total Time =		Students will share their animation with two other members of the class. They will check for errors, and share their programs with each other.	Students will share their end result with 2 other students Students will check each other for correct labeling Students will share their program with the other students.
Assessment Plan:		Primary Objective: Students will correctly identify and Label Continents of the world. – students final animation will correctly identify all the oceans and all the continents. – for those that struggle, they can demonstrate knowledge by verbally naming the correct answers while looking at a map.	

Lesson Segment & Time Est.	Materials	Instructional Sequence	Teacher/Student Actions
		Secondary Goal: Students will show improvement in using scratch depending upon starting ability. – they will be able to verbalize	

Figure 5
Lesson Plan Part B: Instructional Activities and Assessment Plan

SOL(s): USI.2 The student will interpret maps, globes, photographs, pictures, or tables to
a) locate the seven continents and five oceans;

CS. 5.4 The student will analyze, correct, and improve (debug) an algorithm that includes sequencing, events, loops, conditionals, and variables.

5.5 The student will create a plan as part of the iterative design process, both independently and collaboratively using different strategies (e.g., pair programming, storyboard, flowchart, pseudocode, story map).

5.6 The student will break down (decompose) a larger problem into smaller sub-problems, both independently and collaboratively.
5.9 The student will evaluate and solve problems that relate to inappropriate use of computing devices and networks.

5.10 The student will determine whether passwords are strong, explain why strong passwords should be used, and demonstrate proper use and protection of personal passwords.

Instructional Objectives: Student will create a scratch program that correctly Identifies the continents and oceans of the world, starting with a blank map.

NOTE: The sections below expand as you type in electronic version.

Although Mr. Miller included CS learning standards in the lesson plan (Figure 4), the instructional activities and assessment plan reflect a primary focus on teaching science concepts, with minimal emphasis on programming knowledge. He planned to provide sample Scratch programs and encourage students to collaborate in building their Scratch projects. He provided clear guidelines concerning what should be incorporated into the final Scratch projects (Figure 5). In terms of assessment, he focused primarily on assessing science knowledge demonstrated in their Scratch projects rather than programming/CS skills.

RQ2. Implementation and Challenges of Integrating Scratch

Implementation

We identified three themes regarding Mr. Miller's Scratch implementation. He (a) employed Scratch to reinforce and demonstrate students' science understanding, (b) implemented concrete strategies to engage and scaffold students in Scratch exploration, and (c) fostered student growth mindset and creativity with Scratch.

Scratch to Reinforce and Demonstrate Student Science Understanding. The lesson plan and lesson recordings captured Mr. Miller's design and implementation of a five-lesson unit on the topic of sound. The initial two lessons were unplugged and focused on project clarification and concept instruction. In Lesson 1, he clarified the requirements and goals of the final project with video demonstration. To teach the concept of sound, he provided two videos to introduce the components and mechanisms necessary to make sounds, then led whole-class discussions about what students learned. To reinforce student understanding, he showed a video of a musical instrument and fostered reflective thinking by posing questions like, "What's vibrating on the instrument? How do you know it's vibrating? How it makes different sound?"

Lesson 2 focused on students' drawing a prototype of an instrument. He started the lesson by reviewing the mechanism of sound and then reemphasized the final project requirements: "Show me what's vibrating or how it vibrates inside your instrument." He provided example diagrams and pictures to illustrate the components students needed to include in their drawings then encouraged students to share their initial design ideas. In the remaining class time, students worked independently on their drawings.

Scratch was implemented in Lessons 3, 4, and 5 of the unit. Mr. Miller used Scratch for two purposes: to reinforce students' science knowledge and to allow them to demonstrate understanding. Regarding conceptual reinforcement, he explained that "I can kind of get them to repeatedly think about the concept, I want to put it in the program, that you have to keep thinking about it, to program it" (interview). Mr. Miller encouraged students to articulate their conceptualization of soundthrough Scratch programming. This was exemplified in Lesson 3 when the following directions were provided:

> So your program needs to demonstrate what you've learned about sound. … How can you make a change? And then maybe other properties of sound that you could do again? What do we mean by that? How can you make sound change? What are some ways we can make sound change? Pitch? How do we be more technical than just say pitch? Right? I kind of wonder how you change pitch? What's going on when you change [pitch]?

Before students started creating their Scratch projects, he explained sample codes (i.e., eventblocks) to introduce how to run a program and showed students how to navigate the Scratch website to explore existing

programs. Then, he let students work on their Scratch programs and asked them to bring their programs to him for individual feedback. In Lessons 4 and 5, students continued to work on their Scratch projects, and Mr. Miller reviewed students' projects and provided each student with feedback on possible improvements.

Concrete Strategies to Engage and Scaffold Students in Exploring Scratch. Mr. Miller used several strategies, including peer mentoring and collaboration, reviewing and commenting on students' Scratch work, and modeling to engage students and scaffold students' work with Scratch. At the beginning of Lesson 3, he encouraged students to share their Scratch programs with peers and seek help from each other if they ran into any programs. He stated, "I love having the kids mentor each other. … They're engaged with their friends that are trying to help them out. So there's an aspect of Scratch that I like a lot, too" (interview).

Mr. Miller reviewed each student's Scratch program and provided individual feedback in class. He offered generous praise for student accomplishments, commenting, "That's spectacular. Okay, cool. I can't wait to see what you add." When providing feedback to students, he also prompted students to think about solutions when their programs did not work, and in some cases, he modeled problem-solving strategies.

For instance, when reviewing one student's program, he asked, "Can we figure out a way to make those characters always come back to where they're supposed to be?" The student indicated that he did not know how to fix the issue. Mr. Miller then modeled the strategy and then encouraged the student to apply the same strategy to fix a similar issue in their program, saying, "You're going to go and do that on your own."

Fostering Growth Mindset and Creativity. Lesson plan, lesson recordings, and interview data indicated that Mr. Miller promoted student attributes such as self-challenge, self-expression, and creativity during Scratch activities. In his lesson plan, he noted questions to ask students when reviewing their projects including, "What can be improved. How can you improve function? How can you improve visual appeal?" In observed lessons, he always prompted students to improve their work with questions like, "How can you make it better?"

In the interview, he explained that he perceived it important to help students focus more on the thinking and design process, not just completing the project: "I just as a science teacher, I kind of wish they had that thinking process more."

To foster creativity, Mr. Miller encouraged students to communicate science concepts and demonstrate creativity using Scratch. In the observed lessons, he mandated only that students demonstrate their science knowledge, without imposing any specific format or design requirements on their Scratch projects. He explained, "I don't want to give them so much. I want them to show their imagination" (interview).

Challenges in Implementation

Interview data revealed three major challenges encountered during Mr. Miller's integration of Scratch: insufficient programming content and pedagogical knowledge, difficulty with time management, and student frustration with programming.

Insufficient Programming Content and Pedagogical Knowledge. Mr. Miller did not specifically address programming concepts in Scratch-integrated lessons. In the interview, he indicated a limited understanding of programming content and pedagogical knowledge and uncertainty about how to incorporate programming instruction in a substantial, impactful manner. When asked about how he felt about integrating programming in his lessons, he acknowledged, "I don't have the concepts memorized. … I'd have to relook the standards on that…" He expressed a need for a clear guideline about implementing CS standards, stating,

> Probably just more explicit understanding from the state, or whoever. This is what this [CS standards] means. This is what we're looking for at a fifth-grade level. That would probably be the biggest thing, so I have an idea of what to kind of shoot for.

Time Management. Mr. Miller indicated that he planned to continue incorporating Scratch programming in his lessons; however, time management was a major challenge. He noted that predicting the time required for Scratch activities was difficult: "It's kind of hard sometimes to always predict how long will this take?" This resulted in him having to allocate more time than initially planned. Mr. Miller set high expectations for his students' Scratch work, urging them to constantly improve and push themselves. However, accommodating various students' needs and difficulties during programming added to the time management issue. As he put it,

> My initial goals are big, with this idea of pushing to something better down the road, and sooner or later you run out of time. You could because you can continue doing those programs forever and always making them better.

Student Frustration. Mr. Miller said that some of his students were frustrated and struggling with programming, likely due to their limited programming experience and fixed mindset. He observed that some students were having difficulty understanding the fundamental concepts of programming and were unable to build a functional program. As a result, when required to develop a Scratch program, some students expressed their frustration by saying things like, "I don't understand," or "I don't get it."

Another factor related to student frustration, as noted by Mr. Miller, was their mindset when facing challenges. Some students got frustrated because they struggled to learn from failure. For example, when something went wrong in their code, they would see it as their fault rather than an opportunity to find solutions. Miller stated,

> They don't have the mindset of 'if I hit play, and the program doesn't do what I want, that's their fault.' … And that they don't always, like, take ownership of that fifth grade. They're like, I didn't do that. Yeah, you did. It's okay.

RQ3. Students' Use Scratch to Express Science Understanding

As elaborated in the section of data collection and analysis, we explored students' project types, their expressed ideas about sound in the projects, and project creativity to understand how they used Scratch to articulate understanding about sound.

Project Types

In terms of types of Scratch projects created, 12 of the 13 students built storytelling projects, either in the form of a conversation between multiple sprites or narration of a single sprite to speak about their understanding of sound (Figure 6-7). One student created an animation to demonstrate the varying pitches of sound (Figure 8).

Figure 6
Storytelling Project Example A

Figure 7
Storytelling Project Example B

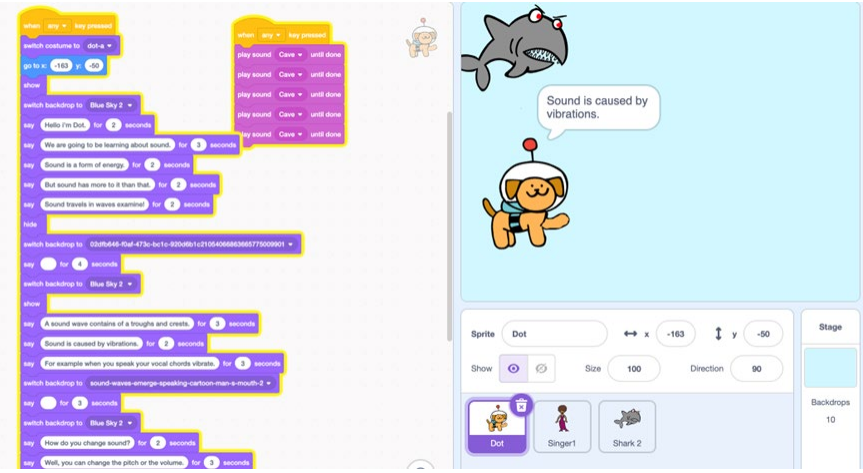
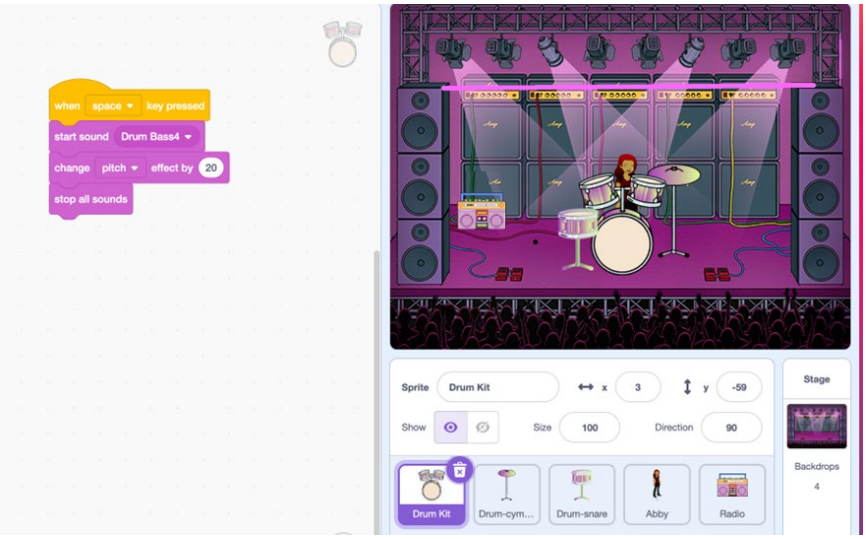


Figure 8
Animation Project



Expressed Ideas About Sound

Most of the student programs indicated that students accurately understood that sound is made by vibrations (11/13) and that the pitch of a sound can change (8/13; appendix). Three students used their programs to communicate that the transmission of sound requires a medium. One student's program suggested that they believed sound is made by molecules, an erroneous conception.

Creativity

Students demonstrated creativity in their Scratch projects in multiple ways, including the design of storytelling scenarios, the design of visual appearance, and the manipulation of music and sound. While most students built storytelling projects, students' storytelling scenarios varied. Some programmed an instructional tutorial, in which a virtual character presented a lecture about sound (Figure 9). Others programmed dialogs structured in a question-and-answer format between sprites (Figure 10).

Student creativity was also demonstrated in visual appearance. Some students programmed backdrops to change according to the content displayed. Some added dynamic visuals (e.g., turning letters) or images to decorate their backdrops. A few students added audio in the background for the sprites to talk about sound (Figure 11). Other students added musical components or sound effects in their programs to demonstrate pitch changes (Figure 12).

Figure 9
Example of Scratch Scenario – Instructional Tutorial

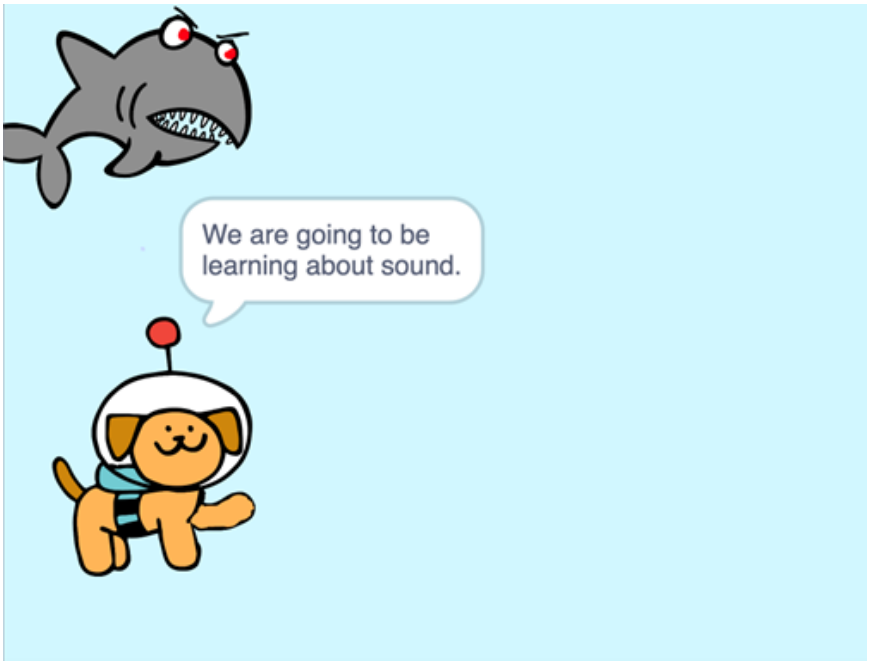


Figure 10
Example of Scratch Scenario – Q & A Conversation

Figure 11
A Scratch Project With Audio Narrations

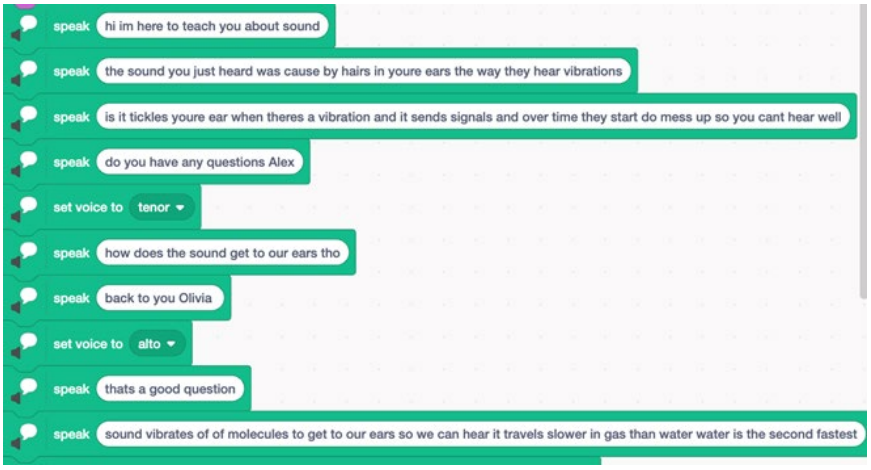
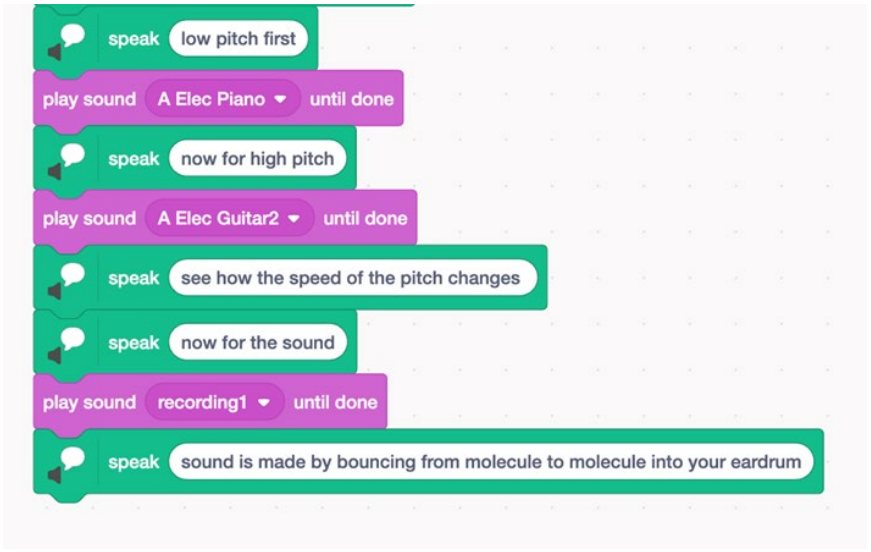


Figure 12
Example of a Student's Scratch Project With Embedded Music and Sounds



Discussion

This study provides an empirical example of an elementary teacher's knowledge development, teaching practices, challenges, and student outcomes of integrating Scratch in science teaching after participating in a STEM-focused PD. It substantiates the prior literature, which found that Scratch is more than just a programming tool. It is a valuable resource for enhancing teaching and learning in noncomputer science subjects (Rodríguez-Martínez et al., 2020).

The participant intentionally implemented Scratch as a medium for students to articulate and demonstrate their understanding of science

concepts, rather than as a stand-alone programming activity. As a result, students were able to create functional Scratch programs to express their understanding of a science concept. The findings highlight the value of Scratch in providing new opportunities to reinforce and assess student science understanding, foster collaboration and growth mindset, and represent and express science knowledge creatively. However,despite the instructional focus on integrating Scratch, explicit instruction in programming concepts was rarely observed during implementation. This lack of explicit programming support may have contributed to student frustration and constrained the range and complexity of the Scratch projects students were able to develop.

Building on the existing literature, this study identified specific challenges an elementary teacher faced when integrating Scratch into science instruction. It underscores the need for targeted support in developing TCK and PCK specific to Scratch, as well as programming-related content and pedagogical knowledge, for successful integration of Scratch in content teaching. The findings also shed new light on the link between teacher learning and student learning, indicating that the teacher's use of Scratch in content teaching influences the way students use Scratch to express their content understanding. Furthermore, the challenges the teacher faced with programming were reflected in the students' Scratch artifacts.

Teachers Change Pathways Through the Lens of IMPG

Examined through the lens of TPACK and IMPG, the findings presented here suggest that the PD supported the development of Mr. Miller's beliefs, knowledge, and practice of Scratch integration. In the personal domain, Mr. Miller developed positive beliefs and built technological and pedagogical knowledge regarding Scratch integration in science. In this study, while Mr. Miller recognized the importance of teaching programming, he did not explicitly address programming knowledge in his teaching, which may have been due to his own lack of programming knowledge. Insufficient programming knowledge is a common challenge teachers face when integrating programming technology into elementary instruction (Greifenstein, 2021). The insufficient support in programming may have led to student frustration when using Scratch, a challenge identified by Mr. Miller.

In the practice domain, Mr. Miller implemented concrete strategies to support student science learning with Scratch. He leveraged the "open-ended" affordance of Scratch to foster student growth mindset in problem-solving, a key attribute for successful STEM learning (Stolhmann, 2022). Scratch does not prescribe a definitive end goal; instead, it supports continuous exploration, experimentation, and iterative refinement of one's program (Resnick et al., 2009). This unique affordance of Scratch provided ample opportunity for students to encounter challenges and errors. Teachers can harness these moments to instill growth mindset attributes, such as perseverance and resilience.

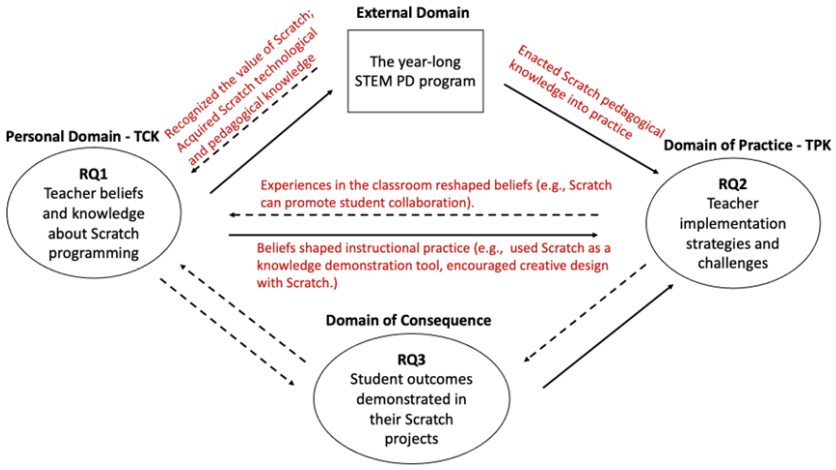
In the domain of consequence, the students, after learning with Scratch, were able to use Scratch to creatively articulate their understanding of sound. However, Mr. Miller indicated that a lack of programming content

and pedagogical knowledge was a significant challenge, which may have resulted in student frustration in programming — another challenge acknowledged by Mr. Miller.

Students' limited programming knowledge and skills were reflected by limitations in their Scratch projects, including lack of diversity and inefficiency. For example, most students built storytelling programs to directly "speak out" the science ideas through static speech bubbles. Students' Scratch programs were lengthy and inefficient. Rather than using control blocks (e.g., repetition loops and conditional logic), they built a long stack of action blocks, indicating a lack of familiarity with efficient structures of programming and limited understanding of abstract programming concepts (Robins et al., 2003). Their limited programming knowledge and skills may constrain their ability to build diverse and efficient programs. To fully leverage the power of Scratch to enhance disciplinary learning, it is essential that students have ample support in developing programming knowledge and skills.

These findings reflect the interplay between domains (Figure 13). For example, believing Scratch to be an effective tool to demonstrate knowledge and foster creativity, Mr. Miller used it primarily to help students articulate understanding of science concepts, and he particularly focused on creative design with Scratch.

Figure 13
The Interplay Between IMPG Domains



Meaningful Integration of Scratch Programming in Elementary Curricula

The existing research highlights several components to support the effective integration of programming in non-CS curricula: fostering interdisciplinary connections, encouraging active learning, expression, and creativity with programming, introducing programming concepts, and fostering computational thinking skills (e.g., Israel et al., 2015; Popat & Starkey, 2019; Resnick et al., 2009).

Fostering Interdisciplinary Connections

An essential goal of integrating programming in non-CS instruction is helping students understand the connections between programming and other subjects and how these different skills can inform each other (Hsu et al., 2022). Scratch is a valuable tool to introduce programming and engage students in disciplinary learning (e.g., Sáez-López et al., 2016). In this study, Mr. Miller used Scratch to engage students in science learning; however, without adequate programming knowledge support, students may not have understood why programming is valuable in learning science.

A meaningful interdisciplinary activity should help students develop a deeper understanding of both the subject matter and programming (Israel & Lash, 2020). While this study demonstrated the value of Scratch in supporting disciplinary learning, its interdisciplinary potential was not fully leveraged. This finding suggests that PD, beyond building teachers' programming knowledge and skills, should place greater emphasis on modeling and demonstrating ways Scratch can be used to naturally communicate computational concepts while simultaneously enriching students' learning experiences across disciplinary contexts.

Encouraging Active Learning, Expression, and Creativity

The goal of introducing programming to children is not necessarily to develop computer scientists but to "nurture a new generation of creative, systematic thinkers comfortable using programming to express their ideas" (Resnick et al., 2009, p. 60). Echoing this view, Mr. Miller emphasized students' growth mindset in programming. In particular, he promoted active exploration, self-expression, and creativity in Scratch exploration.

Developing a growth mindset is pivotal in learning programming as it is often perceived as a challenging skill to acquire (Bain & Barnes, 2014). Students with a growth mindset tend to persevere in learning programming and perceive challenges as valuable learning opportunities (Blackwell et al., 2007). This study highlighted the potential of Scratch in fostering students' growth mindset, another affordance of Scratch that should be emphasized in future PD.

Providing Explicit Support With Computational Knowledge and Problem-Solving Skills

This study indicated that including programming technology in disciplinary instruction does not guarantee student development of programming knowledge and skills. As evidenced in the present study, inadequate programming knowledge and a lack of scaffolding may lead to student frustration, which may ultimately constrain creativity and engagement in programming (Kim et al., 2015).

To sustain student interest and participation in programming, providing explicit instruction of computational knowledge and scaffolding in their computing problem-solving is essential (Israel et al., 2015; Rich et al.,

2020). When integrating Scratch, explaining the computational concepts underlying the coding blocks, illustrating programming logic with flowcharts, and modeling problem-solving processes can help expand students' computational knowledge and enable them to approach a programming problem more efficiently and methodologically (e.g., Liu et al., 2023; Rodríguez-Martínez et al., 2020).

Ultimately, to achieve meaningful integration of Scratch, teachers need to address both content knowledge and programming knowledge in their lessons. This requires prolonged support to develop content and pedagogical knowledge in both subject matter and programming. Further, teachers need support in building the interdisciplinary connections between these domains through PD (Liu et al., 2025).

Limitations and Implications

A case study focuses on constructing an in-depth understanding of the particular case; the findings cannot be generalized to other cases or populations (Creswell & Poth, 2016). Additionally, as with any qualitative research, there is a risk of researcher bias affecting the findings (Noble & Smith, 2015). We employed strategies such as peer examination and member checking to enhance the rigor and trustworthiness of our data analysis, but the potential for researcher bias cannot be entirely eliminated.

Drawing on the insights from this case study, we propose that future research focus on investigating multiple cases to gain a more comprehensive understanding of ways teacher beliefs, attitudes, instructional practices, and challenges related to Scratch and programming are shaped by their PD experiences and instructional contexts. An exploration of contextual factors such as grade level, subject area, and student needs is crucial for gaining a deeper understanding of what influences teacher beliefs and practices of Scratch integration. Ultimately, this knowledge will inform ways to tailor PD programs to meet the needs of different teachers.

In terms of PD design, this study highlights that building teachers' understanding of programming and its relevance to their existing curricula is an essential step for meaningful integration of programming (Jocius et al., 2021). Developing teachers' programming content and pedagogical knowledge can help them fully leverage the power of Scratch to promote students' interdisciplinary learning.

The process of changing teacher beliefs, knowledge, and instructional practices is long and complicated, requiring prolonged support, continuous reflection and practice, and evidence of positive outcomes (Clarke & Hollingsworth, 2002). Therefore, to promote meaningful integration of Scratch programming, PD needs to include sustained and targeted support in programming content and pedagogical knowledge, opportunities for teachers to reflect and experiment with integration strategies, and possibly, feedback about student progress to help teachers make informed decisions on how to improve their instructional practice. Additionally, the potential affordance of Scratch in building students'

growth mindsets in problem-solving should be highlighted to teachers in future PD activities.

Conclusion

The findings presented here corroborate the affordances of Scratch as a valuable instructional tool for supporting elementary teacher integration of programming into content instruction and highlight both the progress and challenges encountered by the teacher when integrating programming into science instruction. By providing insights into Mr. Miller's beliefs, knowledge, experiences, and practice, as well as the ways students used Scratch, the study contributes to the broader conversation about effective strategies for integrating programming into elementary education and PD to support teachers in doing so.

Declaration of Interest Statement

The authors have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

References

Aksit, O., & Wiebe, E. N. (2020). Exploring force and motion concepts in middle grades using computational modeling: A classroom intervention study. *Journal of Science Education and Technology*, *29*(1), 65–82. https://doi.org/10.1007/s10956-019-09800-z

Bain, G., & Barnes, I. (2014, June). Why is programming so hard to learn? *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education* (p. 356). Association for Computing Machinery. https://doi.org/10.1145/2591708.2602675

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K–12: What is involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48–54. https://doi.org/10.1145/1929887.1929905

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2010). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *International Journal of Technology and Design Education*, *20*(3), 305–317.

Blackwell, L. S., Trzesniewski, K. H., & Dweck, C. S. (2007). Implicit theories of intelligence predict achievement across an adolescent transition: A longitudinal study and an intervention. *Child Development*, *78* (1), 246–263.

Brennan, K., & Resnick, M. (2012, April). *New frameworks for studying and assessing the development of computational thinking* [Conference presentation]. Annual meeting of the American Educational Research Association, Vancouver, Canada.

Broza, O., Biberman-Shalev, L., & Chamo, N. (2023). "Start from scratch": Integrating computational thinking skills in teacher education program. *Thinking Skills and Creativity*, 48, 101285.

Burke, Q., & Kafai, Y. B. (2010). Programming and storytelling: Opportunities for learning about coding and composition**.** In *Proceedings of the 9th International Conference on Interaction Design and Children* (pp. 348–351).

Burke, Q., & Kafai, Y. B. (2012). The writers' workshop for youth programmers: Digital storytelling with scratch in middle school classrooms. classrooms. *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (SIGCSE 2012; pp. 433–438). Association for Computing Machinery. https://doi.org/10.1145/2157136.2157264

Clarke, D., & Hollingsworth, H. (2002). Elaborating a model of teacher professional growth. *Teaching and Teacher Education*, *18*(8), 947–967.

Creswell, J. W., & Poth, C. N. (2016). *Qualitative inquiry and research design: Choosing among five approaches*. Sage Publications.

Dasgupta, S., & Hill, B. M. (2018, April). How "wide walls" can increase engagement: evidence from a natural experiment in Scratch. Scratch. *Proceedings of the2018 CHI Conference on Human Factors in Computing Systems* (CHI '18; Paper 361, pp. 1–11). Association for Computing Machinery. https://doi.org/10.1145/ 3173574.3173935

Febrian, A., & Lawanto, O. (2018). Do computer science students understand their programming Task?—A case study of solving the Josephus variant problem. *International Education Studies*, 11(12), 26. https://doi.org/10.5539/ies.v11n12p26

Greifenstein, L. (2021, August 16–19). Effective feedback on elementary school Scratch programs. In *Proceedings of the 17th ACM Conference on International Computing Education Research*, Virtual Event (pp. 417-418). https://doi.org/10.1145/3446871.3469779

Hsu, T. C., Chang, C., Wu, L. K., & Looi, C. K. (2022). Effects of a pair programming educational robot-based approach on students' interdisciplinary learning of computational thinking and language learning. *Frontiers in Psychology*, *13,* 888215.

Israel, M., & Lash, T. (2020). From classroom lessons to exploratory learning progressions: Mathematics+ computational thinking. *Interactive Learning Environments*, *28*(3), 362–382.

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, *82,* 263–279.

Jang, S. J., & Tsai, M. F. (2012). Exploring the TPACK of Taiwanese elementary mathematics and science teachers with respect to use of interactive whiteboards. *Computers & Education*, *59*(2), 327–338.

Jocius, R., O'Byrne, W. I., Albert, J., Joshi, D., Robinson, R., & Andrews, A. (2021). Infusing computational thinking into STEM teaching. *Educational Technology & Society*, *24*(4), 166–179.

Kobsiripat, W. (2015). Effects of the media to promote the scratch programming capabilities creativity of elementary school students. *Procedia-Social and Behavioral Sciences*, 174, 227–232.

Koehler, M., & Mishra, P. (2009). What is technological pedagogical content knowledge (TPACK)? *Contemporary Issues in Technology and Teacher Education*, *9*(1), 60–70. https://citejournal.org/volume-9/issue-1-09/general/what-is-technological-pedagogicalcontent-knowledge/

Kafai, Y. B., & Burke, Q. (2014). *Connected code: Why children need to learn programming*. MIT Press.

Kim, C., Kim, D., Yuan, J., Hill, R. B., Doshi, P., & Thai, C. N. (2015). Robotics to promote elementary education pre-service teachers' STEM engagement, learning, and teaching. *Computers & Education, 91,* 14–31.

Liu, R., Luo, F., & Israel, M. (2023). Technology-integrated computing education in early childhood: A systematic literature review. *Journal of Educational Computing Research*, *61*(6), 1275–1311.

Liu, R., Maeng, J., Moots, S. & Garner, J. (2025). Building elementary teachers' capacity for computer science instruction through professional development: A randomized control trial. Contemporary Issues in Technology and Teacher Education, 25(1), 87–126. https://citejournal.org/volume-25/issue-1-25/general/building-elementary-teachers-capacity-for-computer-science-instruction-through-professional-development-a-randomized-control-trial

Lopez, V., & Hernandez, M. I. (2015). Scratch as a computational modelling tool for teaching physics. *Physics Education*, *50*(3), 310.

Mladenović, M., Rosić, M., & Mladenović, S. (2016). Comparing elementary students' programming success based on programming environment. *International Journal of Modern Education and Computer Science, 8*(8), 1–10. https://doi.org/10.5815/ijmecs.2016.08.01

Montiel, H., & Gomez-Zermeño, M. G. (2021). Educational challenges for computational thinking in K–12 education: A systematic literature review of "Scratch" as an innovative programming tool. *Computers*, *10*(6), 69.

Noble, H., & Smith, J. (2015). Issues of validity and reliability in qualitative research. *Evidence-based Nursing*, *18*(2), 34–35.

Ogegbo, A. A., & Ramnarain, U. (2021). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, *58*(2), 203–230. https://doi.org/10.1080/03057267.2021.1963580

Pektas, E., & Sullivan, F. (2021, July). Storytelling through programming in Scratch: Interdisciplinary integration in the elementary English

language arts classroom. classroom. *Proceedings of the Fifth Asia-Pacific Society for Computers in Education International Conference on Computational Thinking and STEM Education*. https://par.nsf.gov/biblio/10355622

Parsazadeh, N., Cheng, P. Y., Wu, T. T., & Huang, Y. M. (2021). Integrating computational thinking concept into digital storytelling to improve learners' motivation and performance. *Journal of Educational Computing Research*, *59*(3), 470–495.

Popat, S., & Starkey, L. (2019). Learning to code or coding to learn? A systematic review. *Computers & Education*, *128*, 365-376.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM*, *52*(11), 60–67. https://doi.org/10.1145/1592761.1592779

Rich, P. J., Browning, S. F., Perkins, M. L., Shoop, T., Yoshikawa, E., & Belikov, O. (2018). Coding in K-8: International trends in teaching elementary/primary computing. *TechTrends*, *63*(3), 311–329. https://doi.org/10.1007/s11528-018-0295-4

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

Rich, K. M., Yadav, A., & Larimore, R. (2020). Teacher implementation profiles for integrating computational thinking into elementary mathematics and science instruction. *Education and Information Technologies*, *25*(4), 3161–3188. https://doi.org/10.1007/s10639-020-10115-5

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, *13*(2), 137–172. https://doi.org/10.1076/csed.13.2.137.14200

Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020). Computational thinking and mathematics using Scratch: An experiment with sixth-grade students. *Interactive Learning Environments*, *28*(3), 316–327. https://doi.org/10.1080/10494820.2019.1612448

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two-year case study using "Scratch" in five schools. *Computers & Education*, *97*, 129–141.

Sentance, S., & Csizmadia, A. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies*, *22*(2), 469–495.

Smith, S., & Burrow, L. E. (2016). Programming multimedia stories in Scratch to integrate computational thinking and writing with elementary students. *Journal of Mathematics Education*, *9*(2), 119–131.

Stake, R. E. (1995). *The art of case study research*. Sage Publications.

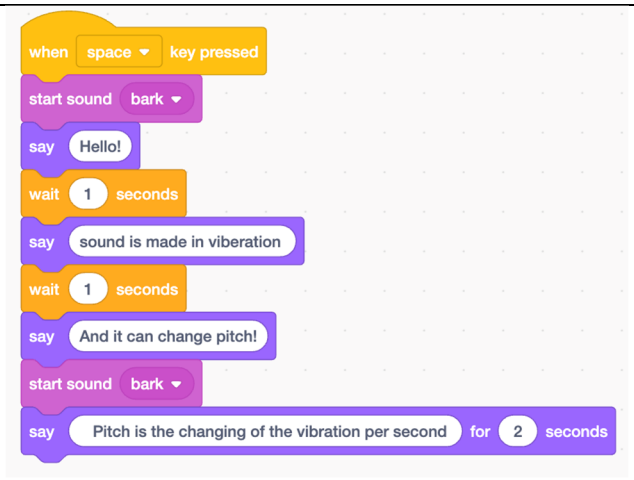
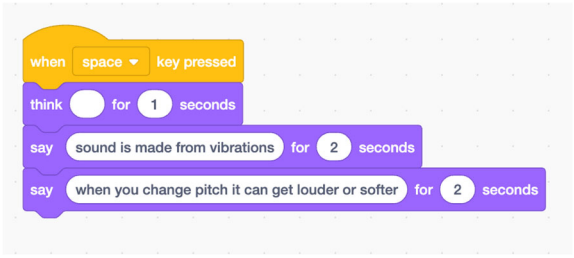
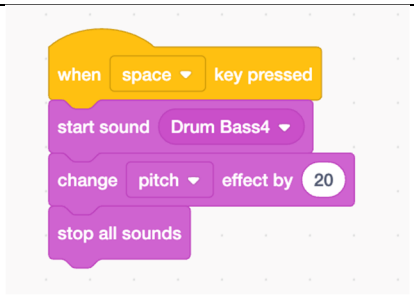
Stohlmann, M. (2022). Growth mindset in K-8 STEM education: A review of the literature since 2007. *Journal of Pedagogical Research*, *6*(2), 149–163.

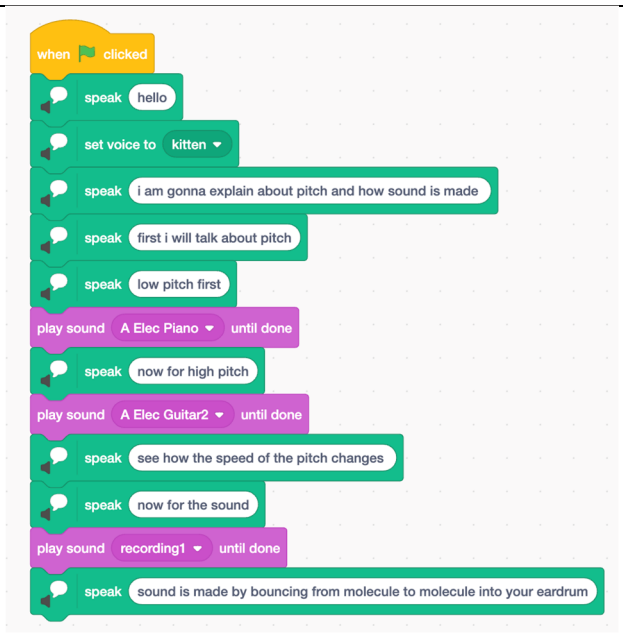
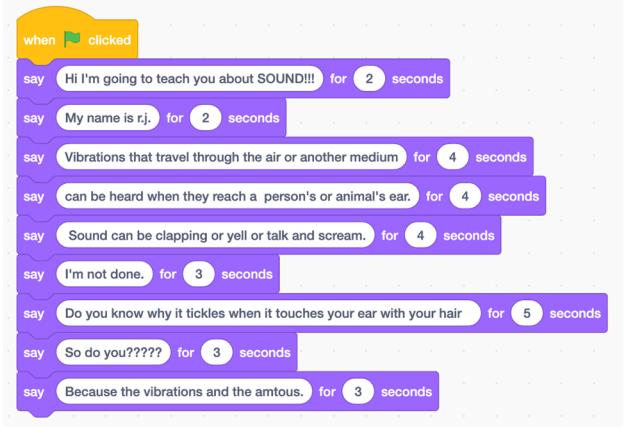
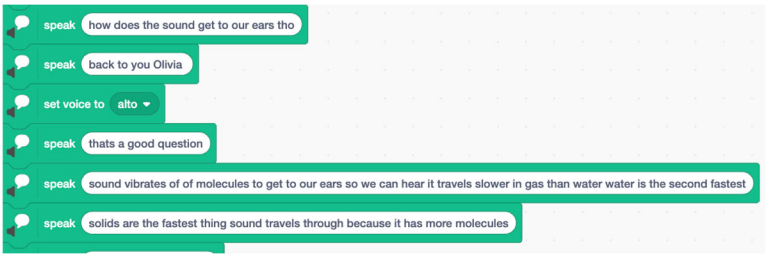
Thomas, D. R. (2006). A general inductive approach for analyzing qualitative evaluation data. *American Journal of Evaluation*, *27*(2), 237–246. https://doi.org/10.1177/1098214005283748

Tucker-Raymond, E., Puttick, G., Cassidy, M. J., Harteveld, C., & Troiano, G. M. (2019). "I broke your game!": Critique among middle schoolers designing computer games about climate change. *International Journal of STEM Education*, *6*(1). https://doi.org/10.1186/s40594-019-0194-z

Appendix

Examples of Student Articulation of Science Understanding in Scratch

Articulated Ideas of Sound	Example of Students' Scratch Codes
Sound is made by vibration (n = 11)	 
The pitch of sounds can change (n = 8)	

	
Mediums are needed to transmit sound (n = 3)	 
Sound is made by molecules: (n = 1)