Computer Science Education

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/ncse20

Empowering Latine elementary school students with disabilities: computer programming through culturally sustaining curriculum

Clare Baek, Dana Saito-Stehberger, Adam Nam, Gerardo Lopez, Sharin Jacob & Mark Warschauer

To cite this article: Clare Baek, Dana Saito-Stehberger, Adam Nam, Gerardo Lopez, Sharin Jacob & Mark Warschauer (02 Sep 2024): Empowering Latine elementary school students with disabilities: computer programming through culturally sustaining curriculum, Computer Science Education, DOI: 10.1080/08993408.2024.2396749

To link to this article: https://doi.org/10.1080/08993408.2024.2396749

Published online: 02 Sep 2024.

COMPUTER SCIENCE EDUCATION
https://doi.org/10.1080/08993408.2024.2396749

Routledge
Taylor & Francis Group

EMPIRICAL STUDIES

OPEN ACCESS Check for updates

Empowering Latine elementary school students with disabilities: computer programming through culturally sustaining curriculum

Clare Baek, Dana Saito-Stehberger, Adam Nam, Gerardo Lopez, Sharin Jacob and Mark Warschauer

University of California, Irvine

ABSTRACT

Background and Context: The study was conducted in a special education classroom in an elementary school with multilingual Latine students, utilizing a computer science curriculum focused on community-based environmental literacy.

Objective: This study explores the experiences of diverse elementary students with disabilities in learning computer programming and identifies instructional strategies that enhance their learning within a culturally sustaining curriculum.

Method: An exploratory case study approach was used to examine students' learning experiences and teachers' instructional strategies during curriculum implementation.

Findings: Students who typically did not engage with peers collaborated effectively, and those with behavioral and performance difficulties exhibited heightened engagement. Instructional strategies included multisensory engagement and connecting environmental and computational concepts to real-life situations.

Implications: The result underscore how a culturally sustaining computer science curriculum can empower diverse students, foster inclusivity, and leverage their strengths through effective teaching practices.

ARTICLE HISTORY
Received 3 October 2023
Accepted 22 August 2024

KEYWORDS
Computer science; disabilities; educational technology; equity

Introduction

In today's rapidly evolving digital era, computer science education has attained paramount importance. Coding ability is now a fundamental component of model education, equipping students with essential skills to navigate an increasingly digital world. These skills extend beyond academic realms, encompassing civic engagement and problem-solving in various facets of life. Computer programming, at its core, fosters computational thinking skills – a set of cognitive processes involving algorithmic and systematic thinking to formulate questions and solve problems (S. R. Jacob & Warschauer, 2018; Song et al., 2021). Computational

CONTACT Clare Baek clareb@uci.edu School of Education, University of California, Irvine, 3200 Education Bldg, Irvine, CA 92697, USA

thinking also pertains to the perspectives that students develop through coding, enabling them to express ideas, collaborate with others, and address real-world problems (Brennan & Resnick, 2012; Shute et al., 2017). Computer science curricula integrated with other subjects can serve as effective tools to enhance students' subject content knowledge. Such curricula foster students' critical thinking, organization, expression, and evaluation of thoughts through problem-solving as they code and utilize innovative media (Fessakis et al., 2013).

Despite significant progress in expanding computer science education in K-12 schools, disparities persist, and certain student populations remain underrepresented. The "Computer Science For All" initiative, announced by President Obama in 2016, led to an increased implementation of computer science courses in K-12 education (Goode et al., 2018; Ladner & Israel, 2016). However, ensuring equitable access to computer science education remains a challenge, particularly for students with disabilities and those from diverse cultural and linguistic backgrounds (Levinson et al., 2021). For example, schools offering computer science courses tend to serve fewer students with disabilities compared to those that do not offer such courses (Fancsali et al., 2018). Furthermore, Latine students are underrepresented in the computer science workforce and in higher-degree computer science programs, often stemming from an early educational pathway with limited opportunities to learn computer science in schools compared to their white peers (Denner et al., 2021).

To address these disparities and promote equity in computer science education, we need to understand how to implement computer science curricula that adequately support diverse students. This entails uncovering unique challenges faced by underrepresented students, recognizing their strengths and potential contributions, understanding the diverse perspectives that enrich the learning environment, and identifying instructional strategies that optimize the learning experience (Bouck et al., 2021). In order to provide meaningful and inclusive content and instruction for culturally and linguistically diverse students in computer science, we need to understand the computer science curricula and instructional strategies that best leverage the cultural assets of these students.

As computer science curricula become more prevalent in K-12 schools, there is a growing need for educators who possess the experience and knowledge required to teach computer science effectively to a diverse student body (Israel et al., 2018). For example, teachers must be equipped to understand and cater to the needs of K-12 students with disabilities and integrate computer science into various subject areas (Bouck et al., 2021). However, the literature is limited in terms of instructional examples of culturally relevant pedagogy that support diverse students in actual computer science settings, thus there is a pressing need for additional research to investigate effective instructional strategies within classroom environments (Madkins et al., 2019). In response to this gap in the literature, we conducted an exploratory case study in a 5th-grade special education class, which comprised multilingual Latine students with disabilities. Our aim was to uncover how students from diverse backgrounds and with disabilities learn computer science through a culturally sustaining curriculum, along with the instructional strategies that facilitate their learning.

Relevant literature

Students with disabilities' experience in coding

The extant literature indicates that students with disabilities often face barriers in accessing and engaging with computer science curricula (Adebayo & Temilola Ayorinde, 2022; Ladner & Stefik, 2017). These barriers include challenges related to the accessibility of the curriculum and coding interfaces (Ladner & Stefik, 2017). Additionally, students with disabilities can experience difficulties with developing and applying problem-solving skills, which are fundamental aspects of computational thinking in coding environments (M. S. Taylor, 2018). Another contributing factor to limited access to computer science for these students is the shortage of teachers equipped to teach computer science to students with disabilities (Ladner & Stefik, 2017).

Nevertheless, existing studies have demonstrated cases of students with disabilities successfully learning computer programming (e.g. Morrison et al., 2020; M. S. Taylor et al., 2017; Thompson, 2016). For example, M. S. Taylor et al. (2017) reported that all three elementary school students with Down Syndrome in their case study successfully learned block-based coding through explicit instruction in computer programming. In Ratcliff and Anderson's (2011) case study, elementary school students with disabilities demonstrated improved attitudes, increased enjoyment, and a greater sense of ownership of their learning as they engaged in programming projects aligned with their interests. Thompson's (2016) study showed that students with learning disabilities could effectively learn writing and coding using a block-based programming language, with high levels of engagement and enthusiasm.

With appropriate support and accommodations, it is possible for students with disabilities to learn the same computer science curriculum as their peers without disabilities and achieve similar positive outcomes. Levinson et al. (2021) found that a student with selective mutism in a general education kindergarten classroom fully participated in a coding curriculum through robotics alongside her peers. This student developed a technical understanding of the computer programming language and began to see computer programming as a language that can be used for communication. Accommodations such as different modes of participation (e.g. contributing non-verbally) were pivotal. Similarly, M. Taylor (2017) found that both kindergarten students with intellectual disabilities and those without could learn coding skills and apply these to maneuver robots, although those with intellectual disabilities needed more sessions. However, all students were capable of identifying errors in their code. Participation in computer science programs is vital for all students, as Plasman et al. (2022) found it influenced the development of STEM attitudes in students both with and without learning disabilities.

A computer programming environment can help students with disabilities develop skills they typically struggle with, such as social skills. The collaborative nature of the computing process, centered around creativity and problem-solving, encourages peer interactions and creates opportunities for students to naturally collaborate (Israel et al., 2015). Indeed, Gribble et al. (2017) found that an elementary school student with Autism was more engaged with their peers in the computer class than in other classes. Munoz et al. (2018) reported that children with Autism with limited social skills actively engaged

with their peers while creating a game in Scratch, demonstrating confidence in interacting with peers. Ratcliff and Anderson (2011) also noted that students collaborated with each other as they shared the projects they had created.

Instructional strategies to support diverse students

The existing literature suggests that effective instructional strategies, such as providing explicit instruction, fostering peer collaboration, and offering multiple means of representation, play a pivotal role in supporting students with diverse needs in computer programming learning (e.g. Bouck & Yadav, 2022; Israel et al., 2015). Snodgrass et al. (2016) found that the instructional strategies that supported students with disabilities in successful computing were not necessarily specific to computing but consisted of fundamental instructional techniques commonly used in other educational settings. These basic instructional strategies encompass providing clear verbal directions for tasks, modeling problem-solving techniques, modeling how to complete given tasks, establishing clear expectations, and giving motivating rewards for task completion (Snodgrass et al., 2016). Similarly, Prado et al. (2022) found that explicitly scaffolded instruction effectively improved the computational thinking skills of students with disabilities. Ratcliff and Anderson (2011) found that students with disabilities produced better learning outcomes with explicit instruction compared to a constructionist approach.

These effective instructional strategies, not limited to the field of computing, suggest that even teachers with limited coding experience can provide effective coding instruction. More research is necessary to identify and ascertain specific instructional practices that are effective in the context of computer programming education for diverse learners. Inclusive coding environments and instructional strategies that support students with disabilities also benefit other diverse learners, such as multilingual students and students from culturally diverse backgrounds (Prado et al., 2022).

Theoretical framework

A culturally sustaining approach is crucial in computer science education for diverse students, including culturally diverse students, linguistically diverse students, and students with disabilities. Similar to students with disabilities being underrepresented, Latine students have also been underrepresented in computer science education, as stated above. The underrepresentation of Latine students in computer science can be attributed to factors including a lack of Latine role models in the field, limited computer science course offerings in Latine-neighborhood schools, and decontextualized computer science instruction that fails to connect with the cultural values of Latine students (Denner & Campe, 2023; S. Jacob et al., 2020).

A culturally sustaining curriculum fosters a supportive and inclusive learning environment that leverages the strengths of Latine students and students with disabilities while sustaining linguistic and cultural pluralism (Paris & Alim, 2014). A culturally sustaining curriculum places value on the diverse cultural backgrounds, identities, and experiences of students (Ladson-Billings, 2014). By recognizing and respecting students' unique abilities and connecting with their real-life experiences, a culturally sustaining curriculum enhances engagement and motivation in learning (Paris, 2012). For example, students can

relate the curriculum to their communities and experiences, thereby stimulating critical thinking and problem-solving skills while drawing on their wealth of community knowledge and cultural values. Integrating computer science curriculum with other subjects through culturally sustaining pedagogy has been shown to have a positive effect on underrepresented students' computer science identity and motivation (Corkin et al., 2020).

This study contributes to the field by investigating the experiences of multilingual Latine elementary school students with disabilities as they learn computer science through a culturally sustaining curriculum integrated with community-based environmental literacy. This unique approach allows students to leverage their cultural values and knowledge to address environmental issues in their own communities. Additionally, this study sheds light on instructional practices to support students with disabilities, as well as culturally and linguistically diverse students. By revealing students' experiences with computer programming and identifying effective instructional strategies, we aim to provide insights to stakeholders in ways to foster inclusive and empowering computer science education for all students.

The computer science and environmental literacy curriculum

The university research team developed, piloted, and implemented two sequential CS curricula for elementary school students in several partner school districts, including District A, which has a high percentage of English Language Learners from Latine backgrounds. To expand the CS curricula by integrating STEM content and addressing real-life issues relevant to diverse learners, the research team partnered with District A. District A has a robust environmental science curriculum for its elementary grades, continually refined through an iterative design and implementation process, guided by the Environment as an Integrating Context model and NGSS (Lieberman, 2013, 2017). Ms. Linda, District A's lead teacher for environmental literacy and a contributor to the 5th-grade environmental literacy curriculum, partnered with the university research team. This collaboration between the university research team and District A aimed to leverage both parties' expertise to create an integrated CS and environmental literacy curriculum, allowing students to engage with environmental science through coding using an innovative computing tool, Scratch. Ms. Linda, who volunteered to co-design and pilot the curriculum, was chosen through convenience sampling.

The curriculum was co-designed during the weekly co-design meetings with the classroom teacher, Ms. Linda, who taught special education for 12 years. Ms. Linda did not have any prior coding experience or computer science teaching experience before implementing this computer science curriculum. Ms. Linda provided the environmental science resources and content direction. Excluding two lessons, all 5th-grade environmental literacy lessons were incorporated into the new curriculum. One lesson, traditionally focusing on creating a Venn Diagram on paper, was transformed into an interactive Scratch project to visualize ecosystem relationships. The gardening lesson was maintained outside the integrated curriculum time. The CS components, designed by the university team, included support for students with disabilities such as step-by-step coding instructions with visual cues and example Scratch projects for student reference. To ensure ease of implementation for educators with varying levels of experience, the

curriculum was structured with slides, comprehensive lesson plans, and workbooks with substantial examples and step-by-step guidance, following the models of the previous two CS curricula of the university research team.

Feedback was integral to the development process, with Ms. Linda reviewing and commenting on all materials during the weekly one-hour co-design sessions. The research team, particularly the first and second authors, responded by adapting materials to better suit the students' needs, such as adding lessons on specific coding concepts necessary for their final projects and providing additional scaffolding.

During the implementation, Ms. Linda led the instructions of the lessons and projects with minimal support from the researchers. The class paraeducator provided additional support to students. The lessons conducted throughout the academic year involved students creating projects related to the environmental science concepts covered in each unit (e.g. natural systems) using Scratch. These lessons were integrated with literacy resources, including vocabulary word banks and sentence structures (see Figure 1), designed to support students in their English language acquisition. The curriculum's sequence and lesson descriptions are detailed in Table 1. The pacing was adaptable, with lesson durations varying according to student needs and Ms. Linda's discretion.

All students used the personal Chromebooks provided by the school district. The culminating classroom project, titled "Imagining Change", tasked students with identifying environmental issues within their school, capturing them through pictures, brainstorming innovative solutions, and subsequently creating projects that showcased the transformation from "before" to "after". These projects also featured persuasive messages advocating for environmental solutions, presented through written or spoken text (i.e. audio recording), with the aim of sharing their proposals with the school administrator.

Positionality statement

The first author is a first-generation Korean immigrant who began her education in a beginning English as a Second Language (ESL) class during middle school. Prior to becoming a researcher, she was a computer science teacher for students with disabilities. Drawing from her personal and professional experiences, her research centers on designing and implementing computer science curricula for diverse learners, including English language learners and students with disabilities. The second author taught English to culturally diverse learners in Costa Rica and Southern California for 20 years during which she developed the understanding of linguistic, cultural, and social struggles that young multilingual learners face. Using her experiences, her research focuses on promoting access to computing for diverse learners through culturally responsive pedagogies. The third author grew up in a household with immigrant parents, speaking both Korean and English. He was introduced to coding in fifth grade at his elementary school where he was challenged to think computationally while learning Scratch. He has continued to take computer science courses while pursuing higher levels of education. The fourth author is a Latine first-generation college student and a second-generation immigrant. He has experience teaching computer programming to college students and AI literacy to middle school students. The fifth author is an Egyptian American who grew up in a high-poverty household with parents with disabilities. Her understanding of the challenges faced by students in embracing diversity in language, ethnicity, ability, and income serves as the

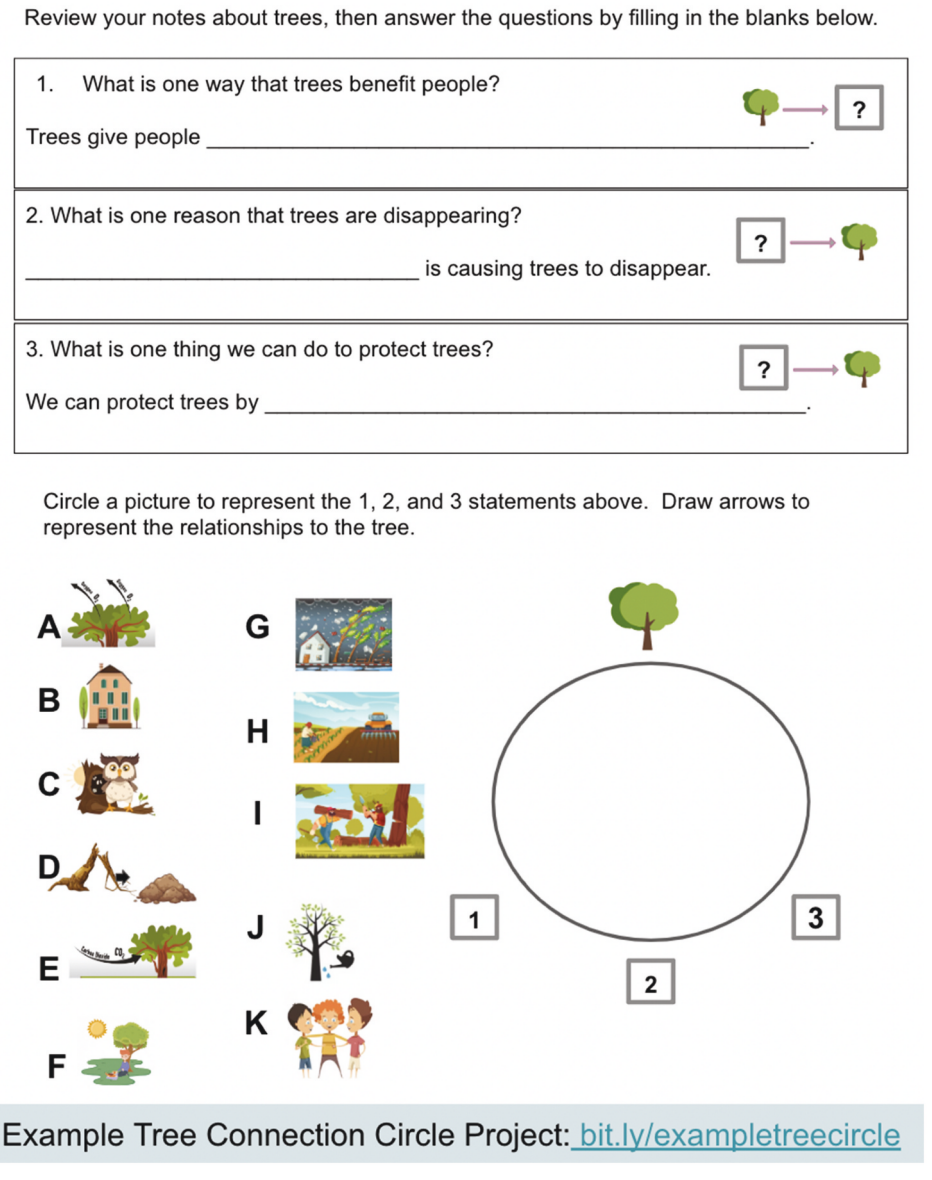

Figure 1. Support provided for English language learners.

driving force behind her work as a teacher and a researcher. The last author is a professor of education and a former Spanish-bilingual math teacher in California public schools. He is the father of a child with cognitive disabilities.

Research questions

We aim to answer the following questions:

Table 1. Sequence of the curriculum.

Lesson	Topic	Description
1	Setting up Scratch accounts	Logging into Scratch and joining the teacher's Scratch class
2	Introduction to Scratch	Learning how to code in Scratch environment
3	Scratch charades	Playing a Scratch version of charades to learn how block-based programming works
4	Coding concepts – sequence, algorithm, program	Learning the concepts of sequence, algorithm, program, and debugging
5	Coding concepts – events, loops, animation	Learning the concepts of events, loops, animation, and debugging
6	Broadcasting and receiving	Learning the concepts of broadcasting and receiving
7	Systems	Learning about a natural system and a human system
8	Connection circle – Kelp forest system	Learning how parts in a system all work together Remixing a kelp forest system in a connection circle in Scratch to model how the components of a kelp forest interact with each other using arrows and visualizations
9	About trees	Learning about trees and its relationship to nature and human systems Creating a program in Scratch to communicate about trees
10	Building a connection circle	Learning about the cause-and-effect relationship of trees in a system through a connection circle Creating students' own tree connection circle in Scratch
11	Reflection and share	Reflecting and sharing about the cause-and-effect relationship of nature systems and human systems
12	Environment in my community	Exploring and identifying environmental problems in students' community Collect data (pictures)
13	Imagining change	Creating a project in Scratch that communicates current environmental problems, solutions to mitigate these issues, and the ideal environment after the problems have been resolved using data collected (e.g. pictures of the environmental problems)
14	Role model	Learning and discussing about computer scientist from diverse background who overcame challenges to pursue their dreams

RQ1: How do multilingual Latine elementary school students with disabilities learn computer programming through a culturally sustaining computer science curriculum in a special education classroom?

RQ2: What instructional strategies did the teacher employ to teach the students to learn computer programming through this culturally sustaining computer science curriculum in a special education classroom?

Methodology

This study employs an embedded single-case study design (Yin, 2009) to investigate the computer programming learning processes and experiences of multilingual Latine students with disabilities within the context of a culturally sustaining computer science curriculum. Additionally, this study seeks to shed light on the instructional strategies employed by the teacher to support students' learning. The case study methodology is appropriate for this research as we aim to explore research questions without specific propositions. Our aim is to gain insights into the phenomena as they unfold during the implementation of the computer science curriculum in this particular classroom, offering guidance for future research directions (Yin, 2009, 2011). The embedded design allows multiple nested units of analysis. The overall single case is a classroom that is

implementing a computer science curriculum using a block-based programming language, Scratch, with integrated lessons on community-based environmental literacy. Embedded within that are the teacher and individual students who we also consider as additional units of analysis. We thus analyze both the collective classroom dynamics and the distinct experiences of individual participants.

Setting

This case study was conducted in a 5th-grade special education classroom in the United States, consisting of 12 Latine students with disabilities, all of whom were also designated as English Language Learners. The class comprised seven students with Specific Learning Disabilities, two students with Other Health Impairments, one student with Autism, and two students with Intellectual Disabilities. This research is part of an ongoing partnership between university researchers and a school district that serves over 95% of Latine students, the majority of whom come from low-income families, and half of whom are designated as English Language Learners. The primary objective of this partnership is to develop an integrated curriculum for district-wide implementation across elementary schools. All names mentioned in this paper are pseudonyms.

Data collection and analysis

The data collected for this study encompassed various sources, including co-design meeting notes, audio recordings of classroom observations, audio recordings of informal interviews with students, audio recordings of semi-structured teacher interviews, field notes, and student artifacts. This comprehensive collection facilitated an in-depth analysis of both student learning experiences and outcomes, as well as teacher instructional strategies (Yin, 2009). The student artifacts comprised literacy scaffolding sheets, data gathered by students (such as pictures taken by students and a campus map indicating green spaces), as well as the Scratch projects they developed. Throughout the 2022–2023 school year, Ms. Linda implemented the co-designed curriculum once a week. Ms. Linda and the research team regularly reviewed the co-design meeting notes to ensure the curriculum's alignment with the team's goals and student progress.

The first and second authors of this study visited the classroom weekly as participant observers, providing assistance in facilitating the curriculum's implementation. A central recording device captured each class session in its entirety, while handheld recorders were used by the authors for impromptu student interviews during independent work time. All audio recordings of classroom observations and interviews were transcribed verbatim using transcription software. Prior to conducting the study, we sent parent permission slips to parents and provided students with study information sheets in both English and Spanish, allowing them the option to opt out of the study.

Transcripts from classroom observations, student interviews, and teacher interviews were analyzed to address the research questions. Collective engagement and collaboration in learning were examined through classroom transcripts. The learning processes for coding, computational thinking, and environmental science were evaluated using both classroom and student interview transcripts, focusing on how students incorporated their cultural backgrounds and creativity into their projects. The teacher interviews provided

insights into the pedagogical strategies employed by the teacher and further illuminated students' learning experiences.

We employed thematic analysis with an inductive approach to analyze the transcription of the audio-recorded data of classroom observations, student informal interviews, and teacher interviews. The first and the third authors together followed the thematic analysis steps (Braun & Clarke, 2006) by first iteratively reading the transcription and noting initial ideas in multiple cycles. Then, the initial codes were generated, and excerpts from the transcripts relevant to each code were collated. The collated codes were assigned to potential themes, and the research team reviewed the themes together to check if the coded extracts assigned to the themes were relevant. Then, the first, third, and fourth authors worked in collaboration to define the themes after iteratively refining the themes. The excerpts were re-examined to identify any additional codes that align with the established themes. The coding scheme for student learning experiences and teacher instructional strategies are presented in Tables 2 and 3, respectively.

Triangulation was employed by the authors to validate the emergent themes, cross-reference data across different sources, and capture additional details for more comprehensive findings. The field notes provided additional context and captured non-verbal classroom dynamics, such as students moving seats to sit next to each other for collaboration. Noteworthy observations, like a student coming up with innovative solutions, were included in the field notes in detail and later cross-checked with audio transcripts for comprehensive analysis. Artifacts, including Scratch projects and worksheets, were analyzed to further examine student learning outcomes both at the class level and for individual students. To assess the Scratch projects, a checklist developed by the authors and Ms. Linda was used, outlining essential skills related to coding, computational thinking, environmental science, and communicating environmental problems and solutions. Student worksheets were examined to check that the students were able to complete the required assignments using accurate language and literacy, environmental science concepts, and coding knowledge.

Results

Student learning experiences

Throughout the curriculum, students acquired coding knowledge, refined scientific communication skills, and gained environmental literacy while actively participating in various Scratch projects. All students completed the final project (see Figure 2 for an example), "Imagining Change", and presented their work to both the class and the school vice principal. However, two students did not master all the coding skills outlined in the curriculum's checklist; specifically, they encountered difficulties with the correct application of a loop – a coding block in Scratch for repeating actions. Despite this challenge, their final projects still demonstrated the integration of the coding skills and environmental literacy, acquired over the course of the year. Our analysis revealed several key themes, shedding light on students' experience when learning computer programming through the culturally sustaining computer science curriculum.

Table 2. Coding scheme for student learning experiences.

Theme	Description of the theme	Data Source	Example
Making progress (lower performing students or students with behavioral issues)	Struggling students noticeably improving through learning CS -including their academic or behavioral improvement	Teacher interview, student artifacts	[Ms. Linda's interview excerpt]: "She is my lowest student. . . I just really think she didn't find it interesting in the beginning. Um, but then something clicked and she was like, oh yeah, I could do this".
Making connection to students' lives	Students connect lessons/ projects to their community, daily lives, culture, interest, knowledge, and experiences.	Class observation transcript, student informal interview, teacher interview	[Ms. Linda's interview excerpt]: "They are still like saying . . . there's too much trash here, or, oh, the trees are dying . . . they still talk about all of that stuff even now that we're done. So I feel like it's been ingrained in their minds now and it was kind of like a double dose pairing it with computer science".
Producing novel ideas/ discoveries	Students come up with new or creative ideas to problem-solve	Fieldnotes, observation transcript, student informal interview, Scratch artifacts	[From fieldnotes]: John came up with an innovative solution to place a dot on each arrow to activate the coded message in audio or text. The teacher shared the solution with the whole class. All other students in the classroom used John's idea and placed a dot on their arrow.
Experiencing challenges	Challenges that students experience with learning coding.	Field notes, Scratch artifacts	Some students were not able to incorporate the "loop" block into their script in Scratch correctly.
Collaborating with peers	Students help each other to solve problems or share ideas	Observation transcript, teacher interview	[Ms. Linda's interview excerpt] "Um, and I can't imagine how awesome it must feel. . ..to be able to be good at something and you can actually help your peers with it. Like kids that are not usually . . . chatting together, are working together. They do it when it comes to computer science . . . to anything that has to do with, um, technology really".

Problem-solving using novel ideas

John, a student with Autism, creatively resolved an issue with the Scratch interface. The task involved creating an arrow representing a causal relationship in an environmental phenomenon. When clicked, the arrow should display a text or audio message explaining the relationship. Despite correct coding, the messages remained inactive upon clicking the arrows, leading to student frustration and attempts at solutions. John, who is typically off-task, took the initiative to investigate the problem by clicking various parts of the arrows and ultimately discovered a specific section for each arrow that activated the message. He solved the issue by drawing a large dot on each arrow using the Scratch paint feature (see Figure 3). Users instinctively clicked the dot, consistently activating the message. Other students

Table 3. Coding scheme for instructional strategies.

Theme	Description of the theme	Data Source	Example
Activating previous knowledge	Connecting to what was taught or discussed in previous lessons in the CS class or in other subjects	Class observation transcript	"So remember last week when we were having a couple of problems clicking on the arrows, and we couldn't find where to click it . . . what was one of the things that we did in order to, to fix it?. . ..Guess what? You were debugging . . . "
Step-by-step instructions, breaking down complex-tasks	Explaining tasks to be completed or coding concepts step-by-step using words such as ("and then"",next"", after this"). Teacher breaks down one task into multiple subcomponents.	Class observation transcript	"So last week we learned that we, if we wanted to keep looping, it needs to stay forever. And then you're gonna insert the next costume, and then you're gonna wait however seconds you need to wait to make it look".
Modeling	Demonstrating a specific skill, behavior, or thought process to learners. Teacher serves as a model, showcasing the desired actions or thinking strategies that learners should emulate or should not emulate.	Class observation transcript	"So I'm gonna click the flag. What, what is this telling the tree? Go to X . . . negative 130 and y . . . negative nine . . . And, and then, um, we wanted to, we wanted to grow to 20%".
Connecting to students' lives	Connecting the content to students' community, daily lives, culture, interest, student knowledge, and student experiences.	Class observation transcript	"So maybe sharing makes people feel warm and comfortable and grateful and they become friends. So that could be, um, a cause right? Um, a friendship (is the effect) . . . So the tree causes, the tree does something (and effect happens)".
Giving choices	Giving students choices of modality (e.g. recording voices or typing text, ways they create the projects) or other choices (e.g. the way they complete the project)	Class observation transcript	"So there's more than one answer. So you can copy mine or you can make, you know, do your own". "I'll help you. Do you wanna record your voice or do you want to text?"

adopted John's solution by drawing dots on their arrows to indicate the correct spot for activating the message.

Challenges students experienced

Some students encountered challenges with the Scratch interface, particularly in distinguishing the various colors of Scratch blocks, each representing different functions. Field notes and student informal interviews revealed some students struggled to differentiate between the "Looks" and "Sound" blocks, which both appeared in different shades of purple. Additionally, the "Events" and "Control" blocks posed difficulties in discernment for some students. Ms. Linda, during an interview, expressed concerns about potential difficulties for students with vision impairments, as certain colors may appear similar to them.

As discussed above, two students had difficulty applying the loop block to animate their Sprites (characters in Scratch) in the final project, which was a requirement listed on their checklist. Figure 4 shows an example of how a student could not use the "forever loop" block to make actions repeat. Instead, this student inserted the "forever loop" block into the coding environment without connecting it to other blocks. However, these students were able to fulfill other requirements, including creating a character that presented an environmental solution using persuasive language. Figure 4 shows that the student was able to use the initialization ("when this sprite

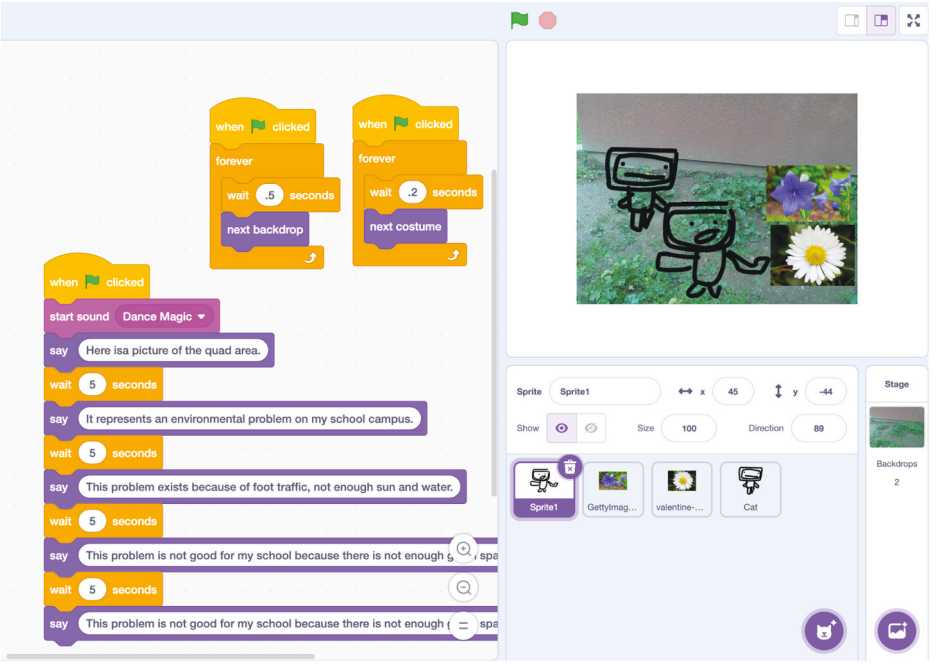

Figure 2. An example of the imagining change project.

clicked"), text blocks, and control blocks ("wait blocks") successfully to display the message. Our findings suggest that comprehending the "loop" concept – particularly the integration of action blocks within a loop – poses a significant challenge for some students.

Culturally sustaining learning experience

The curriculum incorporated lessons that foster a computer science identity among Latine students. For example, students engaged in a "role model" lesson, which included a discussion about a Latina computer scientist. Students watched a video wherein the Latina computer scientist shared her academic and career trajectory, detailing how she grew up in a Spanish-speaking household and overcame language barriers in school. Moreover, she emphasized the applicability of technology in various fields of interest to the students, reinforcing the curriculum's focus on using coding to address and propose solutions for environmental issues. Following this lesson, students expressed a heightened interest in computer science.

Furthermore, students connected their personal experiences to classroom discussions and projects. John introduced the idea of planting mango trees, inspired by a family member's garden. This led to the discussion about the viability of growing mango trees in their local environment, including factors that might hinder their growth, and prompted questions on climatic conditions at the location of John's family's garden. Such personal expressions and connections to projects spurred motivation to learn coding. For example, Jenny, who has an intellectual disability and faces challenges in other subjects, enthusiastically

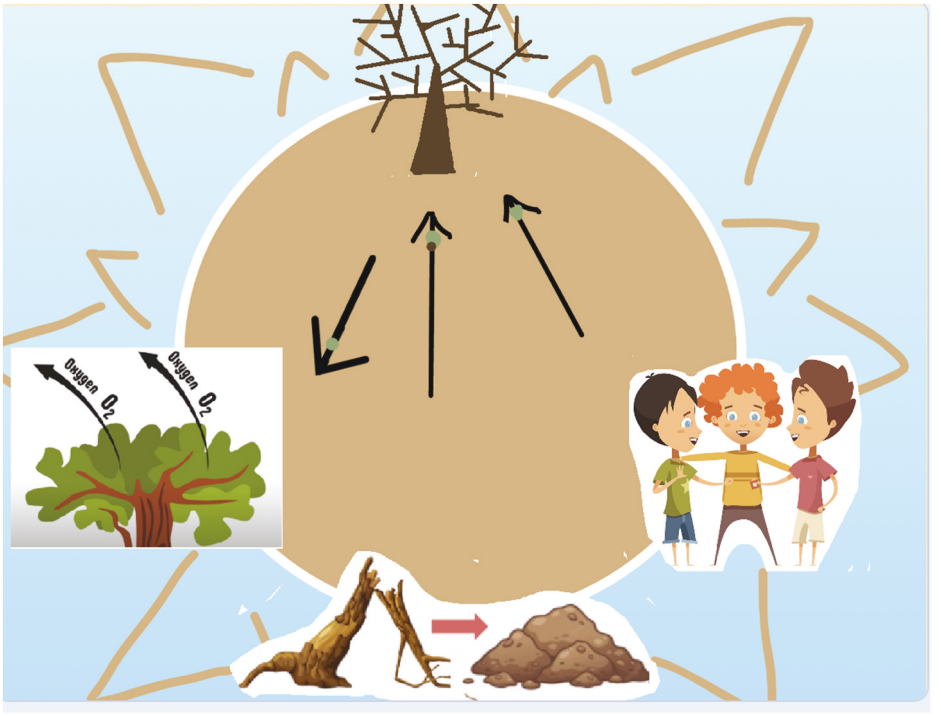

Figure 3. Students' innovative solution to solve interface issues.

participated in the curriculum and completed the projects. Ms. Linda attributed Jenny's success in this curriculum as being able to express herself and her creativity (see Figure 5).

> For her, she likes to color, she likes to draw. Um, so she likes to...do little crafts...I think once she saw like everybody's projects and ... what you, they were able to do with that and that it was actually pretty fun. I think she kind of...came around because towards the end she got really really good. And I think it was just her opening up and giving herself the opportunity to enjoy.

Students also enriched classroom discussions by addressing issues important to them, sharing insights and perspectives on these community issues. During the lesson on the community's environmental issues, several students highlighted the recurring problem of their peers littering in the school cafeteria by presenting photographs they had taken as evidence of this concern within their school community. One student noted, "Some kids eat like very messy or they just like to throw food on the floor", leading to another student's suggestion that they should educate their peers on appropriate cafeteria conduct. The class recognized the significance of maintaining cleanliness in shared spaces, linking it to communal responsibility-one student pointed out the undue burden on the janitor, stating, "It also takes forever to the [sic] janitor to clean all of it".

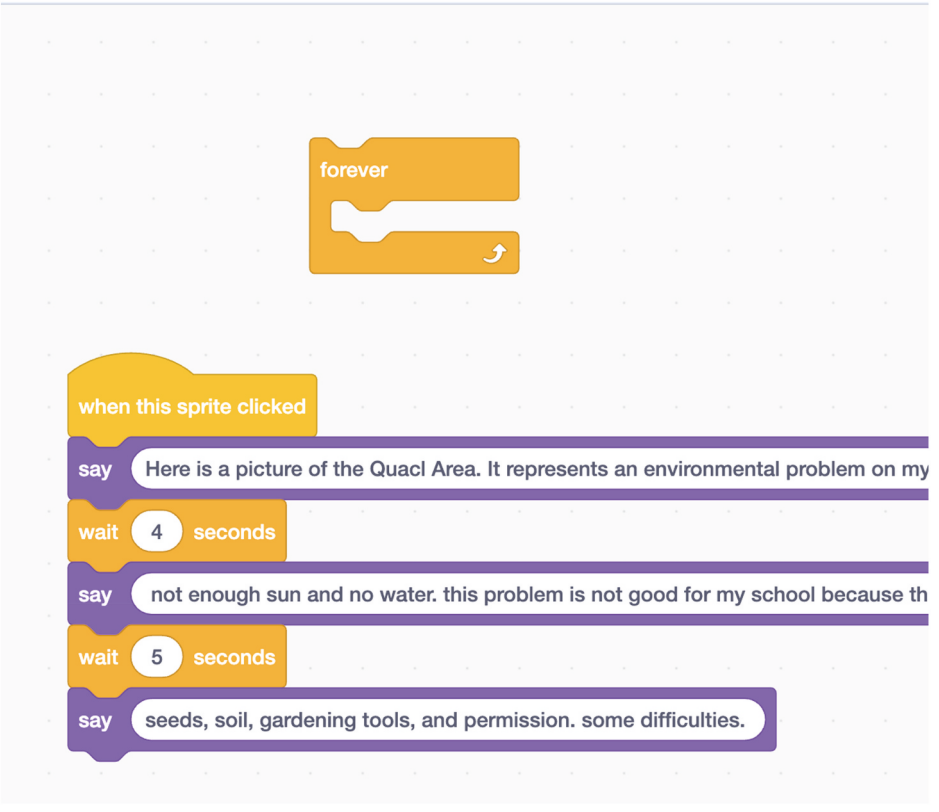

Figure 4. Students' challenges with understanding loop.

Increased collaboration with peers

Students in this classroom, who usually have limited engagement with peer collaboration, collaborated with each other frequently during the computer science class. Ms. Linda shared during an interview that she had previously faced challenges in fostering collaboration among her students. She noted that students who had been in special education programs throughout their academic lives often faced difficulties when it came to collaborative tasks. As a result, Ms. Linda generally avoided assigning group projects due to her students' reluctance to work collaboratively. However, when reflecting on the computer science class, Ms. Linda remarked, "Collaboration doesn't work for them … but this [the computer science curriculum], this worked and it was amazing". Ms. Linda attributed the students' increased confidence to these positive social interactions, stating, "It must feel good to be good at something and you can actually help your peers with it".

As the year progressed, students naturally began to move around the classroom to collaborate, asking questions and exchanging ideas. For example, John, who had notably advanced coding skills, became a go-to person for students to get help. Additionally, some students sat together to learn from each other and share coding ideas. During the "Imagining Change" project, some students paired up to assist with recording messages and providing feedback on each other's content as well as code.

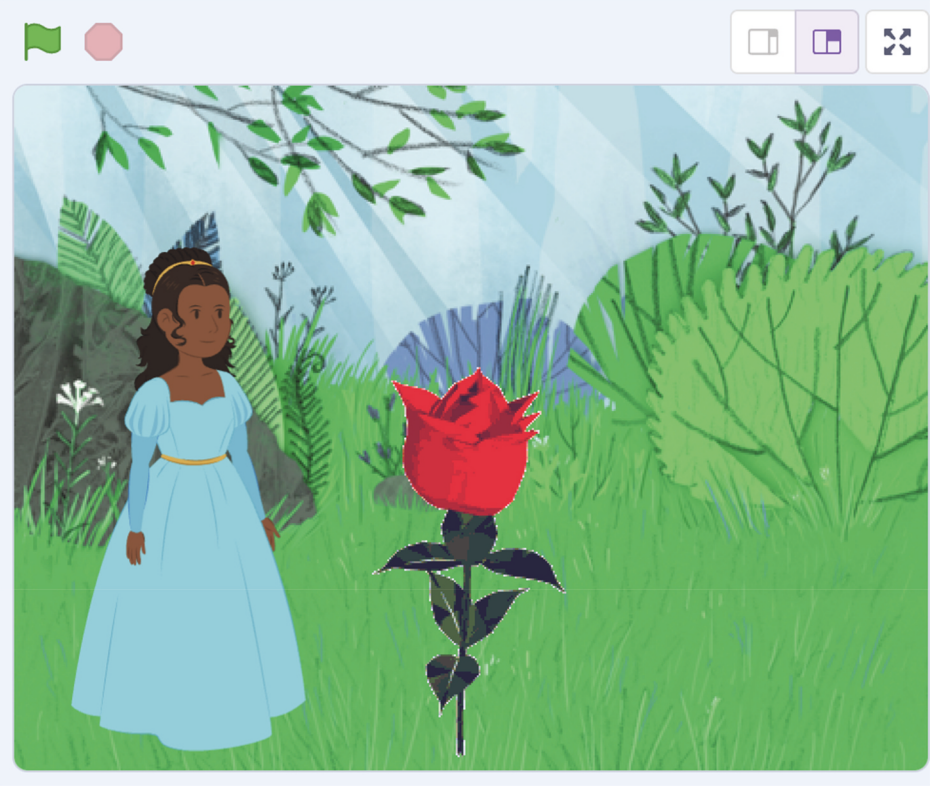

Figure 5. A student expressing their creativity.

Improvement of struggling students

Students who had behavioral and academic struggles showed notable improvements as they engaged with the curriculum throughout the school year. For example, Jenny, a student with Intellectual Disability, initially displayed a lack of engagement and fell behind her peers in learning coding concepts. However, as the year progressed, Jenny demonstrated increased interest in the curriculum, grew more confident in learning the material, and became more actively involved in creating projects. Ms. Linda shared:

> She is my lowest student . . . I didn't really think it had to do with her academic abilities. I just really think she didn't find it interesting in the beginning . . . but towards the end she got really, really good.

Ms. Linda credited this notable improvement to three factors. First, Jenny was able to incorporate her interests into her projects (e.g. picking colors she likes). Second, Jenny saw what her peers created and realized that creating projects in Scratch is "actually fun". Third, persistent encouragement and praise from the teacher helped her to feel more confident. Jenny often exclaimed, "This is fun" during class while working on her project. Also, during the connection circle lesson where Jenny made interactive arrows to represent cause-and-effect relationships, she fully grasped the concept and was able to explain the relationships correctly during an informal interview, sharing that all the relationships were clear to her.

Another student, Charlie, who had initially exhibited defiant attitudes towards the lessons, began actively engaging in the lessons and displaying enthusiasm towards the end of the school year. Charlie, a student with Learning Disability and behavioral challenges, had initially refused to participate in the lessons and projects, making negative comments about the tasks. However, towards the end of the year, Charlie's participation notably increased. For example, Charlie contributed to class discussions by sharing thoughtful ideas and suggestions. Charlie successfully completed the "Imagining Change" project, following all the requirements of the project of coding multiple loops, animating two Sprites, presenting clear messages, and inserting changing backdrop blocks to demonstrate "before" and "after" changes. Further, Charlie was the only student who utilized the touch screen feature of the Chromebook to draw his own Sprites (Figure 2).

Instructional strategies

We examined the instructional strategies employed by Ms. Linda to scaffold students' learning of coding concepts and environmental literacy by analyzing the class observation transcript and teacher interviews. These strategies encompassed providing linguistic resources, guiding students in understanding instructions, and fostering their active engagement. We identified a set of instructional strategies that effectively supported multilingual students with disabilities in learning computer programming through a culturally sustaining curriculum. The identified instructional strategies link to students' positive learning experiences and outcomes.

Engaging multiple senses

Ms. Linda employed multi-sensory approaches to guide students during the lessons. For example, before the activity where students observed environmental issues around their school, Ms. Linda emphasized the importance of paying attention to what they see, smell, and hear. Students took pictures of the environmental problems they identified, engaging multiple senses in the process. After taking the pictures, students selected the pictures of the environmental problem that best represented what they saw, smelled, and heard. Using the selected pictures, students created an interactive program with graphics, texts, and audio that highlight the environmental problem, and, further, solutions.

In communicating the solutions, Ms. Linda encouraged students to communicate from their own perspectives on what they saw in their own words. Students were repeatedly engaged in the problem they saw and the solutions they brainstormed through different senses. This process enabled students to express themselves through computing. During the interview, Ms. Linda emphasized how using computer science to learn environmental literacy was effective for students as it involves multiple senses.

> I feel like it's been ingrained in their minds and it was kind of like a double dose pairing it with computer science . . . they were not only able to see it, [but] feel it, touch it, hear it like it was all modalities.

Giving choices

Students were given choices in all the assignments. For example, in communicating a persuasive message to the school administrator, students were given a choice of either typing the text to be displayed or recording their voice to read the text. This flexibility allowed students to work according to their preferences. Ms. Linda frequently reminded students that there was more than one way to solve problems, encouraging creativity and personalization in their projects.

Encouraging student experts

Students were encouraged to become experts and assist with problem-solving coding questions or errors. Some students developed creative solutions to coding challenges, and Ms. Linda actively sought their assistance when she couldn't answer a coding question. If there was an error in the code, the expert students were consulted to help with the debugging process. The process of involving students to help other students naturally led to peer collaboration, in which students started asking each other for help. Ms. Linda shared that this process of helping each other promoted students' confidence, particularly, for those students who were the lowest performing students.

Connecting to students' lives

Throughout her teaching, Ms. Linda consistently connected lessons to the students' own community and prior knowledge. For instance, when explaining the final project centered around the school environment, she referred back to what students had previously learned on the environmental systems and how this all led to this project concerning their own surroundings. For example, Ms. Linda stated:

> We are going to take all that knowledge that you guys have about systems and how they work, and we are going to look at our environment in my [our] community.

During class instruction, Ms. Linda actively engaged students' prior knowledge by relating each lesson to what they had learned in the previous weeks, such as coding concepts they had learned. Prior to beginning a new lesson each day, Ms. Linda reviewed the vocabulary introduced in the previous lesson, reminding the students of topics like "debugging", and providing examples of debugging activities they had completed in the previous week. Moreover, she rephrased words using familiar vocabulary, for instance, likening "habitat" to a "house".

Further, in explaining a new task, Ms. Linda related the task with real-life examples that interested the students. For example, when Ms. Lidna explained that the students had to write out the script of what the Sprites should say about the environmental issues before typing or recording it in the Scratch environment, a student asked whether this process was similar to movies in which characters speak. Ms. Linda validated the student's analogy, stating:

> Yeah . . . it's what you're gonna say. So, you know, before a movie, they have to read what they are gonna say beforehand. They don't just do it spontaneously. So that's what you are gonna spend some time here writing out what you're gonna say.

Providing explicit instructions and modeling

Ms. Linda provided explicit instruction by breaking down a task into subtasks step-by-step and modeling the process. In explaining what students need to create in Scratch, Ms. Linda systematically broke down the steps. Ms. Linda divided tasks into multiple subtasks, clearly indicating each step with phrases like "first" and "second". Also, Ms. Linda used "and then" after each sentence to explain the tasks step-by-step.

> So you have two tasks. The first one you are coding is to record the relationship or write it down in the *Say* block. And then after that . . . you can animate a Sprite.

Ms. Linda introduced new coding concepts and features of Scratch by modeling them on the Smart Board in front of the class. Ms. Linda walked through the steps herself while engaging the students, such as how to use the blocks to initiate an action and how to change numbers to move the characters. For example, she demonstrated how she would put the Scratch blocks together one by one to create the desired outcome. In doing so, Ms. Linda narrated her thoughts out loud so students could follow her thinking process. Also, Ms. Linda demonstrated a trial and error process by putting in a number, testing out how the Sprite looks with the number, and changing the number to produce the desired size, location, or color while narrating her process out loud.

Discussion

The findings of this study demonstrate that students with disabilities can effectively engage in computer programming and use it as a medium to address real-world issues within their community. Also, the findings highlight how culturally diverse students can successfully engage in computer programming by expressing their creativity and leveraging their cultural assets and diverse perspectives. All students were English language learners, and the embedded literacy support throughout the curriculum, along with the teachers' instructional strategies, yielded successful outcomes. The students were able to create projects that communicated environmental problems and proposed potential solutions while utilizing scientific language to convey their ideas. The success of these students accentuates the importance of providing students with disabilities with the same opportunities as their peers, how students with disabilities can benefit from using coding as a tool to learn other content areas, and that the computing field can benefit from students with disabilities' creativity and talent (Bouck & Yadav, 2022; Wille et al., 2017). For example, one student with Autism used an innovative method to problem-solve an issue with the Scratch interface that the class struggled with, which shows how students with disabilities can learn differently and can generate novel solutions to solve problems (Wille et al., 2017).

The success of the students in learning computer programming and producing meaningful projects can be attributed, in part, to the instructional strategies employed by the teacher. Using multiple senses provided various channels for students to process and communicate information, which supported students' understanding and retention of coding concepts. Involving multiple senses is particularly beneficial for students with disabilities as it provides alternative channels for students to access and process information if they have limitations in one or more senses (Algrni, 2020). Technology serves as

a valuable tool in providing multiple senses and modalities, which can enhance the comprehension of content for students with disabilities (Anderson & Putman, 2020). Involving multiple senses is effective in promoting computational thinking skills, and technology can be utilized to create this multi-sensory learning environment (Katai et al., 2008, 2014).

Guiding students to become coding experts was another effective strategy that promoted students' confidence and encouraged peer collaboration. When a student figured out a solution to a problem for the whole class using innovative ideas, the teacher shared it with the whole class to encourage other students to use the solution. Also, the teacher actively asked the students for help with coding concepts that she could not figure out. This approach of involving students as experts empowered students to be the agents of their own learning by contributing to the instruction and leading to improved learning experiences and outcomes (Reeve & Shin, 2020). In this learning environment, the students and teachers learn from each other instead of the teacher being the sole knowledge distributor (Knowlton, 2000).

Encouraging students to see themselves as experts facilitated peer collaboration as the students sought help from one another, and shared their own code to help their peers. The project-sharing component of the class further promoted peer collaboration, as each student presented their completed projects in front of the class. This opportunity allowed students to observe and appreciate the different creative approaches taken by their peers, leading to an environment where they felt comfortable asking each other for assistance and exchanging ideas on how to create specific features in their projects.

Students exhibited heightened social engagement with their peers in the computer science class. Our finding aligns with the previous literature indicating that those students who do not typically interact with their peers actively interacted with peers during coding class (Gribble et al., 2017; Munoz et al., 2018). The increased collaboration can be attributed to a couple of factors. First, students with disabilities and English language learners who typically lack confidence in other subjects could feel confident in a computer science class as they are creating projects integrating their own ideas and preferences, such as colors they like, characters they want to put in, and the sound effects they want to integrate. Also, in the curriculum we implemented, the community-based environmental literacy component bolstered students' confidence as students bring in what they see in their own environment, a familiar environment that the students know best. Second, students and their peers shared similar interests while making a game in Scratch. In turn, the coding environment became a welcoming environment for children to foster their social skills as they shared ideas with each other.

Connecting coding lessons to students' lives and experiences and giving multiple means of engagement was crucial for engagement and relevance for students (Israel et al., 2019). Making projects that are relevant to students, including environmental pictures they took, the colors they chose for the characters, and the music they chose to input, promoted engagement and confidence. Giving choices on how students present the information (e.g. text, audio) and presenting an environmental scene of the school they chose pertain to Universal Design for Learning (Israel et al., 2019).

Relating coding exercises to real-world issues, such as environmental problems in students' own communities, is powerful in improving the motivation and confidence of underrepresented students as they apply their knowledge and skills in meaningful local

contexts (Crooks et al., 2015). Addressing students' own community's problems and coming up with solutions using their knowledge elicited a sense of purpose and social responsibility of students to address local challenges. This approach is particularly helpful for Latine students as they value their community and culture, utilizing the affordances of Latine students who bring in a wealth of community cultural knowledge (Luna & Martinez, 2013). As the teacher shared, the students took ownership of their learning as they were discussing environmental problems in their own communities with their own pictures, presented through the coding projects they created using their ideas. The students were able to engage in environmental discussion beyond the classroom as they noticed environmental issues outside of the classroom and identified them by communicating with their peers and teachers. Students discussed and examined their own school environment where they could make tangible changes, rather than discussing the ice cap melting at the North Pole (Lieberman & Hoody, 1998). Thus, the environmental phenomena were relatable and mutable, which increased their motivation.

The strength of this culturally sustaining curriculum centers on its ability to motivate students to learn environmental science within the context of their own environments and stories. Students shared stories related to environmental science, such as stories about mango trees in their relative's backyard and the weather in the relative's city. The role of *Familismo* has a significant influence on Latine students' academic engagement and outcomes (Azpeitia & Bacio, 2022). The culturally sustaining curriculum led students to learn through storytelling, such as describing the environmental scene in their community and in their family's community, which nurtured Latine students' strengths of learning through storytelling (Jacob et al., 2022). Additionally, the role model lesson featuring a Latine computer scientist can positively influence Latine students' attitudes towards computer science (Tukachinsky et al., 2017). The role model's emphasis on how coding can be applied across various domains can reinforce the practical application of coding skills to tackle community environmental issues, thereby strengthening the students' identity as coders.

Explicit instruction has been reported to be an effective approach to teaching English language learners, culturally diverse students, and students with disabilities (Piazza et al., 2015). We found that explicit instruction was pivotal in supporting the needs of the students. The teacher's clear explanation and modeling helped demystify complex coding concepts, making them more accessible and comprehensible for students with disabilities. Also, breaking down a task or a concept step-by-step supported students in understanding the directions clearly. The teacher defined the vocabulary by rephrasing the words using vocabulary that is familiar to the students as she went through the lessons, including vocabulary related to computer programming, environmental literacy, and any words that the students might not understand; this is an effective strategy to support English language learners and students with disabilities to attain academic language (Doran, 2015).

Our finding on the effectiveness of explicit instruction for students with disabilities aligns with the existing literature that students with disabilities learn better with explicit instruction (Ratcliff & Anderson, 2011). The students in our case study responded well to explicit instructions and tasks dictated to them clearly. Having a constructive approach with open lessons could have been frustrating for the students as they might not know how to manage tasks and time on their own. Also, the teacher used modeling to simulate

the tasks and computational thinking steps that the students need to conduct for both environmental literacy content and coding concepts. The strategy of modeling the coding process (e.g. how to make Sprites move in certain directions) while verbally narrating the steps in front of the students is helpful in supporting the understanding of students with disabilities and English language learners (Bouck & Yadav, 2022). Explicit instruction with visual modeling is effective for students with Autism in teaching science concepts (Knight et al., 2011).

Furthermore, the study sheds light on the power of technology as an empowering tool to learn STEM content while also promoting language and literacy through innovative modes for diverse students. By providing an accessible and creative platform like a block-based computer programming language, students were able to express their thoughts and ideas in ways that may have been previously difficult through conventional means. The students fostered their computational thinking perspectives by expressing their ideas through creating a product using computational thinking concepts and collaborating with peers to share ideas and get help (Brennan & Resnick, 2012).

However, there were some challenges that students faced in using the Scratch interface. Some students expressed that the Scratch block colors were hard to distinguish, which is a challenge as each color block represents different types of functions in Scratch. The indistinguishable colors could be particularly difficult for students with vision impairment or students prone to cognitive overload, and therefore, it is vital to involve the experiences of students and teachers with disabilities in designing and refining the technology interfaces (Baek & Aguilar, 2022). For example, Morrison et al. (2020) designed an inclusive physical programming language for children with visual impairments by involving students with visual disabilities in their iterative design process to get feedback to improve the design. Further, the technology interfaces should be continually assessed and modified to concern multifaceted issues such as culturally relevant challenges (Baek & Doleck, 2024). One feature in the interface that is seemingly easy to use for a group of students may be difficult to use for other students.

Conclusion

The findings of this study emphasize that computer programming can be a transformative tool in learning environments for Latine elementary school students with disabilities. The demonstrated ability of the students to create projects addressing environmental problems and proposing solutions in scientific language speaks to the power of integrating technology and computer science education in inclusive classrooms to promote STEM knowledge. Also, this study demonstrates effective institutional strategies that accommodate students with diverse needs, and culturally relevant approaches that nurture engagement, empowerment, and success.

Future research should examine involving students with disabilities to evaluate the coding interface. As one student in our case study figured out a solution to deal with a Scratch interface issue, students with disabilities can provide innovative feedback to improve the design and accessibility of the coding interface. Through the feedback of students with disabilities and observation of students with disabilities using a coding interface, stakeholders can find usability difficulties and ways to improve them. Also, future studies should examine effective ways to teach

more difficult coding concepts such as "loops" to students, as a few students in our case study were not able to grasp this concept. More research and discussions on improving the curriculum, instructional strategies, and coding interface for culturally diverse students, English language learners, and students with disabilities are necessary to ensure that "For All" in "Computer Science For All" truly includes all students (Ladner & Israel, 2016). This case study aims to move the discussions one step forward in the direction of achieving "Computer Science For All" for all students.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the National Science Foundation under Grant Number 1923136 and Grant Number 2317832.

Ethics declaration

The conduct of this research has been approved by the University of California, Irvine Institutional Review Board #20173675.

References

Adebayo, E., & Temilola Ayorinde, I. (2022). Efficacy of assistive technology for improved teaching and learning in computer science. *International Journal of Education and Management Engineering*, *12*(5), 9–17. https://doi.org/10.5815/ijeme.2022.05.02

Algrni, N. S. (2020). The effectiveness of using multisensory approach in enhancing achievement and retention of English vocabulary amongst intermediate female students with EFL learning disabilities. *Journal of Education and Practice*, *11*(9), 148–159.

Anderson, S. E., & Putman, R. S. (2020). Special education teachers' experience, confidence, beliefs, and knowledge about integrating technology. *Journal of Special Education Technology: A Publication of Utah State University, the Association for Special Education Technology, and the Technology and Media Division of the Council for Exceptional Children*, *35*(1), 37–50. https://doi.org/10.1177/0162643419836409

Azpeitia, J., & Bacio, G. A. (2022). "Dedicado a mi familia": The role of Familismo on academic outcomes among latinx college students. *Emerging Adulthood*, *10*(4), 923–937. https://doi.org/10.1177/21676968221099259

Baek, C., & Aguilar, S. J. (2022). Past, present, and future directions of learning analytics research for students with disabilities. *Water Science & Technology: A Journal of the International Association on Water Pollution Research*, *55*(6), 931–946. https://doi.org/10.1080/15391523.2022.2067796

Baek, C., & Doleck, T. (2024). Learning analytics: A comparison of Western, educated, industrialized, Rich, and democratic (WEIRD) and non-weird research. *Knowledge Management & E-Learning: An International Journal*, *16*(2). https://doi.org/10.34105/j.kmel.2024.16.011

Bouck, E. C., Sands, P., Long, H., & Yadav, A. (2021). Preparing special education preservice teachers to teach computational thinking and computer science in mathematics. *Teacher Education and Special Education: The Journal of the Teacher Education Division of the Council for Exceptional Children*, *44*(3), 221–238. https://doi.org/10.1177/0888406421992376

Bouck, E. C., & Yadav, A. (2022). Providing access and opportunity for computational thinking and computer science to support mathematics for students with disabilities. *Journal of Special Education Technology*, *37*(1), 151–160. https://doi.org/10.1177/0162643420978564

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, *3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

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

Corkin, D. M., Ekmekci, A., & Fisher, A. (2020). Integrating culture, art, geometry, and coding to enhance computer science motivation among underrepresented minoritized high school students. *The Urban Review*, *52*(5), 950–969. https://doi.org/10.1007/s11256-020-00586-8

Crooks, C. V., Burleigh, D., Snowshoe, A., Lapp, A., Hughes, R., & Sisco, A. (2015). A case study of culturally relevant school-based programming for first nations youth: Improved relationships, confidence and leadership, and school success. *Advances in School Mental Health Promotion*, *8*(4), 216–230. https://doi.org/10.1080/1754730X.2015.1064775

Denner, J., & Campe, S. (2023). Equity and inclusion in computer science education: Research on challenges and opportunities. *Computer Science Education: Perspectives on Teaching and Learning in School*, 85.

Denner, J., Martinez, J., & Thiry, H. (2021). Strategies for engaging Hispanic/Latino youth in the US in computer science. In Y. A. Rankin & J. O. Thomas (Eds.), *Research anthology on instilling social justice in the classroom* (pp. 203–221). IGI Global.

Doran, P. R. (2015). Language accessibility in the classroom: How UDL can promote success for linguistically diverse learners. *Exceptionality Education International*, *25*(3). https://doi.org/10.5206/eei.v25i3.7728

Fancsali, C., Tigani, L., Toro Isaza, P., & Cole, R. (2018). A landscape study of computer science education in NYC: Early findings and implications for policy and practice. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, Baltimore, MD, USA (pp. 44–49).

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education*, *63*, 87–97. https://doi.org/10.1016/j.compedu.2012.11.016

Goode, J., Flapan, J., & Margolis, J. (2018). Computer science for all: A school reform framework for broadening participation in computing. In W. G. Tierney, Z. B. Corwin, & A. Ochsner (Eds.), *Diversifying digital learning: Online literacy and educational opportunity* (pp. 45–65). Johns Hopkins University Press.

Gribble, J., Hansen, A., Harlow, D., & Franklin, D. (2017). Cracking the code: The impact of computer coding on the interactions of a child with autism. *Proceedings of the 2017 Conference on Interaction Design and Children*, Stanford, California, USA (pp. 445–450).

Israel, M., Lash, T., Ray, M., & Marsland, B. (2019). *Universal design for learning within computer science education*. Creative Technology Research Lab. University of Florida.

Israel, M., Ray, M. J., Maa, W. C., Jeong, G. K., Eun Lee, C., Lash, T., & Do, V. (2018). School-embedded and district-wide instructional coaching in K-8 computer science: Implications for including students with disabilities. *Journal of Technology & Teacher Education*, *26*(3), 471–501.

Israel, M., Wherfel, Q. M., Pearson, J., Shehab, S., & Tapia, T. (2015). Empowering K–12 students with disabilities to learn computational thinking and computer programming. *Teaching Exceptional Children*, *48*(1), 45–53. https://doi.org/10.1177/0040059915594790

Jacob, S., Nguyen, H., Garcia, L., Richardson, D., & Warschauer, M. (2020). Teaching computational thinking to multilingual students through inquiry-based learning. *2020 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, *1*, 1–8. https://doi.org/10.1109/RESPECT49803.2020.9272487

Jacob, S., Parker, M. C., & Warschauer, M. (2022). Integration of computational thinking into English language arts. Computational thinking in PreK-5. *Empirical Evidence for Integration and Future Directions*, 55–63. https://doi.org/10.1145/3507951.3519288

Jacob, S. R., & Warschauer, M. (2018). Computational thinking and literacy. *Journal of Computer Science Integration*, *1*(1), 1. https://doi.org/10.26716/jcsi.2018.01.1.1

Katai, Z., Juhasz, K., & Adorjani, A. (2008). On the role of senses in education. *Computers & Education*, *51*(4), 1707–1717. https://doi.org/10.1016/j.compedu.2008.05.002

Katai, Z., Toth, L., & Adorjani, A. K. (2014). Multi-sensory informatics education. *Informatics in Education - an International Journal*, *13*(2), 225–240. https://doi.org/10.15388/infedu.2014.13

Knight, V. F., Smith, B. R., Spooner, F., & Browder, D. (2011). Using explicit instruction to teach science descriptors to students with autism spectrum disorder. *Journal of Autism & Developmental Disorders*, *42*(3), 378–389. https://doi.org/10.1007/s10803-011-1258-1

Knowlton, D. S. (2000). A theoretical framework for the online classroom: A defense and delineation of a student-centered pedagogy. *New Directions for Teaching and Learning*, *2000*(84), 5–14. https://doi.org/10.1002/tl.841

Ladner, R. E., & Israel, M. (2016). For all" in "computer science for all. *Communications of the Acm*, *59* (9), 26–28. https://doi.org/10.1145/2971329

Ladner, R. E., & Stefik, A. (2017). AccessCSforall: Making computer science accessible to K-12 students in the United States. *SIGACCESS Accessibility and Computing*, *118*(118), 3–8. https://doi.org/10.1145/3124144.3124145

Ladson-Billings, G. (2014). Culturally relevant pedagogy 2.0: A.K.A. The remix. *Harvard Educational Review*, *84*(1), 74–84. https://doi.org/10.17763/haer.84.1.p2rj131485484751

Levinson, T., Hunt, L., & Hassenfeld, Z. R. (2021). Including students with disabilities in the coding classroom. In *Teaching computational thinking and coding to young children* (pp. 236–248). IGI Global.

Lieberman, G. A. (2013). *Education and the environment: Creating standards-based programs in schools and districts*. Harvard Education Press.

Lieberman, G. A. (2017). Environmental principles for everyday life: Yesterday, today, and tomorrow. *Social Studies Review*, *56*, 35–38.

Lieberman, G. A., & Hoody, L. L. (1998). Closing the achievement gap: Using the environment as an integrating context for learning. *Results of a nationwide study*.

Luna, N. A., & Martinez, M. (2013). A qualitative study using community cultural wealth to understand the educational experiences of latino college students. *Journal of Praxis in Multicultural Education*, *7*(1), 2. https://doi.org/10.9741/2161-2978.1045

Madkins, T. C., Martin, A., Ryoo, J., Scott, K. A., Goode, J., Scott, A., & McAlear, F. (2019). Culturally relevant computer science pedagogy: From theory to practice. *Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, Minneapolis, MN, USA (pp. 1–4). https://doi.org/10.1109/RESPECT46404.2019.8985773

Morrison, C., Villar, N., Thieme, A., Ashktorab, Z., Taysom, E., Salandin, O., Cletheroe, D., Saul, G., Blackwell, A. F., Edge, D., Grayson, M., & Zhang, H. (2020). Torino: A tangible programming language inclusive of children with visual disabilities. *Human–Computer Interaction*, *35*(3), 191–239. https://doi.org/10.1080/07370024.2018.1512413

Munoz, R., Villarroel, R., Barcelos, T. S., Riquelme, F., Quezada, A., & Bustos-Valenzuela, P. (2018). Developing computational thinking skills in adolescents with autism spectrum disorder through digital game programming. *IEEE Access*, *6*, 63880–63889. https://doi.org/10.1109/ACCESS.2018.2877417

Paris, D. (2012). Culturally sustaining pedagogy: A needed change in stance, terminology, and practice. *Educational Researcher*, *41*(3), 93–97. https://doi.org/10.3102/0013189X12441244

Paris, D., & Alim, H. S. (2014). What are we seeking to sustain through culturally sustaining pedagogy? A loving critique forward. *Harvard Educational Review*, *84*(1), 85–100. https://doi.org/10.17763/haer.84.1.982l873k2ht16m77

Piazza, S. V., Rao, S., & Protacio, M. S. (2015). Converging recommendations for culturally responsive literacy practices: Students with learning disabilities, English language learners, and socioculturally diverse learners. *International Journal of Multicultural Education*, *17*(3), 1–20. https://doi.org/10.18251/ijme.v17i3.1023

Plasman, J. S., Gottfried, M., Freeman, J., & Dougherty, S. (2022). Promoting persistence: Can computer science career and technical education courses support educational advancement for students with learning disabilities? *Policy Futures in Education*, 147821032110499. https://doi.org/10.1177/14782103211049913

Prado, Y., Jacob, S., & Warschauer, M. (2022). Teaching computational thinking to exceptional learners: Lessons from two inclusive classrooms. *Computer Science Education*, *32*(2), 188–212. https://doi.org/10.1080/08993408.2021.1914459

Ratcliff, C. C., & Anderson, S. E. (2011). Reviving the turtle: Exploring the use of logo with students with mild disabilities. *Computers in the Schools*, *28*(3), 241–255. https://doi.org/10.1080/07380569.2011.594987

Reeve, J., & Shin, S. H. (2020). How teachers can support students' agentic engagement. *Theory into Practice*, *59*(2), 150–161. https://doi.org/10.1080/00405841.2019.1702451

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158. https://doi.org/10.1016/j.edurev.2017.09.003

Snodgrass, M. R., Israel, M., & Reese, G. C. (2016). Instructional supports for students with disabilities in K-5 computing: Findings from a cross-case analysis. *Computers & Education*, *100*, 1–17. https://doi.org/10.1016/j.compedu.2016.04.011

Song, D., Hong, H., & Oh, E. Y. (2021). Applying computational analysis of novice learners' computer programming patterns to reveal self-regulated learning, computational thinking, and learning performance. *Computers in Human Behavior*, *120*, 106746. https://doi.org/10.1016/j.chb.2021.106746

Taylor, M. (2017). *Computer programming with early elementary students with and without intellectual disabilities* [Doctoral dissertation]. University of Central Florida. http://purl.fcla.edu/fcla/etd/CFE0006807

Taylor, M. S. (2018). Computer programming with pre-K through first-grade students with intellectual disabilities. *The Journal of Special Education*, *52*(2), 78–88. https://doi.org/10.1177/0022466918761120

Taylor, M. S., Vasquez, E., & Donehower, C. (2017). Computer programming with early elementary students with down syndrome. *Journal of Special Education Technology*, *32*(3), 149–159. https://doi.org/10.1177/0162643417704439

Thompson, R. (2016). Teaching coding to learning-disabled children with kokopelli's world. *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, Cambridge, United Kingdom.

Tukachinsky, R., Mastro, D., & Yarchi, M. (2017). The effect of prime time television ethnic/racial stereotypes on latino and Black Americans: A longitudinal national level study. *Journal of Broadcasting & Electronic Media*, *61*(3), 538–556. https://doi.org/10.1080/08838151.2017.1344669

Wille, S., Century, J., & Pike, M. (2017). Exploratory research to expand opportunities in computer science for students with learning differences. *Computing in Science & Engineering*, *19*(3), 40–50. https://doi.org/10.1109/MCSE.2017.43

Yin, R. K. (2009). Case study research: Design and methods. In *Applied social research methods series* (4th ed., pp. 240). SAGE Publications.

Yin, R. K. (2011). Applications of case study research. In *Applied social research methods* (pp. 192). SAGE Publications.