Coding as Another Language: Computational Thinking, Robotics and Literacy in First and Second Grade

Marina Umaschi Bers, Madhu Govind, and Emily Relkin, *DevTech Research Group, Tufts University*

Corresponding Author: Marina Umaschi Bers, *marina.bers@tufts.edu*

Abstract

This paper explores the integration of coding, CT and literacy by describing a study conducted with first and second grade classrooms in Norfolk, Virginia. A total of 667 students and 57 educators from eight elementary schools, as well as 181 students from two comparison schools participated in a curriculum called Coding as Another Language (CAL) that utilizes KIBO robotics, a developmentally appropriate kit which does not require keyboards or screens. CAL positions the teaching of programming as a symbolic system of representation, a tool for creative expression and communication. Thus, research questions regarding the relationship between students' coding and CT outcomes and their literacy skills were explored, as well as teachers' reactions to the experience, in particular regarding the integrating of teaching computer science and literacy in the early grades. Participation in the CAL-KIBO curriculum was associated with improvement in coding and unplugged CT skills. Baseline literacy skills were related to students' acquisition of CT skills. For example, students who had higher literacy scores at the beginning of the term were more successful in CT tasks. Furthermore, although teachers varied in their perceptions of integrating coding and CT with literacy, our findings suggest that these disciplines may share some cognitive and pedagogical overlap that has yet to be extensively explored in the early computing education field.

Introduction

Educators, researchers, and policymakers are recognizing the need to give children access to computer science (CS) starting at an early age (Barron et al., 2011; Bers, 2018; Bers, 2019; Code.org, 2019; White House, 2016). Recently, the focus has expanded from teaching computer programming to also engaging with a set of underlying cognitive abilities known as computational thinking (CT) (Fayer, Lacey, & Watson, 2017; US Department of Education, Office of Educational Technology, 2017; Wing, 2006, 2011).

In an influential article entitled "Computational Thinking" that appeared in an issue of *Communications of the ACM*, Wing (2006) defined CT as "solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (p. 33). Wing argued that CT should be part of everyone's analytical repertoire. This echoed earlier work by Perlis (1962), who claimed that the "theory of computation" is for everyone, not only computer scientists, and Papert (1980), who proposed that through programming children can form ideas not only about computation but about thinking itself.

CT has received considerable attention over the past several years. There is consensus that CT must be available to thinkers of all disciplines, regardless of their ability to program (Guzdial, 2008; Yadav, 2016). However, there is little agreement on how to define it (Aho, 2012; Allan et al., 2010; Barr & Stephenson, 2011; Cuny, Snyder, & Wing, 2010; Grover & Pea, 2013; Lu & Fletcher, 2009; National Academies of Science, 2010; Relkin, 2018; Relkin & Bers, 2019; Shute, Sun, & Asbell-Clarke, 2017; Yadav, Good, Voogt, & Fisser, 2017). It is widely agreed that CT involves a broad set of analytic and problem-solving skills, dispositions, and habits, rooted in computer science but universally applicable. Examples include thinking recursively, using abstraction to identify salient pieces of a problem, and applying heuristic reasoning to discover a solution and/or identify potential "bugs" or problems (CSTA & ISTE, 2011; Kalelioğlu, Gülbahar, & Kukul, 2016). For definitions that are specifically relevant to young children, CT must also be framed in a developmentally appropriate context (Bers, 2018).

This research builds on these findings and focuses on the creative aspects of computer science for early elementary school children 5-9 years of age. We describe a curriculum called Coding as Another Language (CAL) that focuses on the role of languages, both artificial and natural, for expressive purposes. CAL integrates the teaching of computer programming and literacy by positioning the teaching of CS as another medium for expression (Bers, 2019). In other words, the ultimate goal of mastering a programming language is not only to provide a means of problem-solving but also to allow creation of personally meaningful artifacts that can be shared with others. Ultimately, CAL is informed by the notion that both natural and artificial languages are symbolic systems of representation that can be used for creative expression and communication (Vee, 2017).

Prior research has shown that learning to code can enhance the acquisition of CT and related thinking abilities. Román-González et al. (2018) found improvements in CT in a study of middle school students (ages 12–14) who engaged in the code.org curriculum. Arfé et al. (2019) found improvements on neuropsychological tests of response inhibition and planning in first and second graders who received coding instruction. These studies provide evidence from randomized control trials that learning to code can accelerate the development thinking abilities critical to CT in children. Further studies are needed to evaluate the impact of learning to code with an integrated curriculum such as CAL on young children's CT skills and other aspects of their cognitive development.

We examined three different research questions: 1) How did the CAL curriculum promote students' coding and CT skills? 2) What was the relationship between students' CT skills and their literacy skills? 3) How did teachers react to the experience?

METHODS

Participants

A total of 667 first and second grade students and 57 educators from eight elementary schools (CAL group), as well as 181 students from two comparison schools (No-CAL group), participated in this study. All schools were in the Norfolk Public School district, Norfolk, VA, and participated in the CAL coding curriculum. The curriculum utilizes KIBO robotics, a developmentally appropriate kit designed for children 4 to 7 years old, which does not require keyboards or screens (Bers, 2018; Sullivan, Bers & Mihm, 2017; Sullivan, Elkin & Bers, 2015). The KIBO-21 kit used in this study consists of the KIBO robot, 21 colorful programming wood-based blocks, as well as light, distance, and sound modules and sensors. Children assemble the barcoded blocks, scan them using the robot's embedded barcode scanner, and press the triangle-shaped button on the robot to run the sequence of commands (see Figure 1).

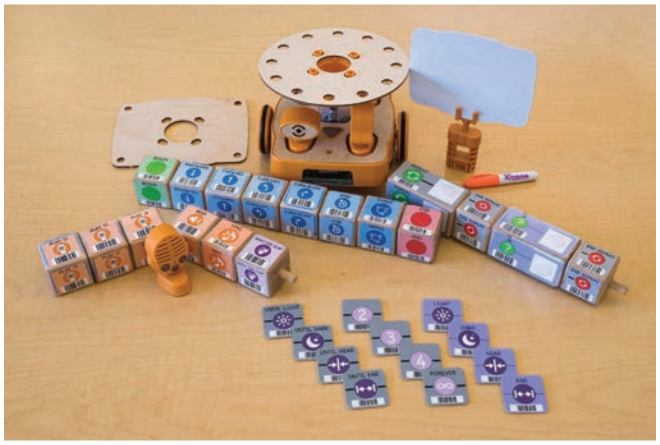


Figure 1. KIBO-21 robotics kit

The CAL-KIBO Curriculum

The CAL-KIBO curriculum consists of 12-15 adaptable lessons administered over a 6-8-week period. Throughout the curriculum, children engage in activities, songs, games, and open-ended projects CAL-KIBO integrates coding and CT with the use of arts and crafts, reading and writing activities that are commonly used in early elementary school. For example, the final lessons involve a project based on a children's book, *Where the Wild Things Are* by Maurice Sendak. Students are invited to write a creative composition about what would happen in their own "Wild Rumpus" and subsequently program their "Wild Rumpus" using KIBOs (see Figure 2). The curriculum is aligned with the Common Core English Language Arts (ELA)/Literacy Framework, as well as Virginia CS Standards of Learning and other nationally recognized CS frameworks (e.g., K-12 CS Framework). Bers (2018) described seven powerful ideas of CT from CS that are developmentally appropriate for early childhood: hardware/software, algorithms, modularity, control structures, representation, debugging, and design process. Each CAL-KIBO lesson engages children in multiple powerful ideas of CT and connects them to foundational literacy and language concepts.

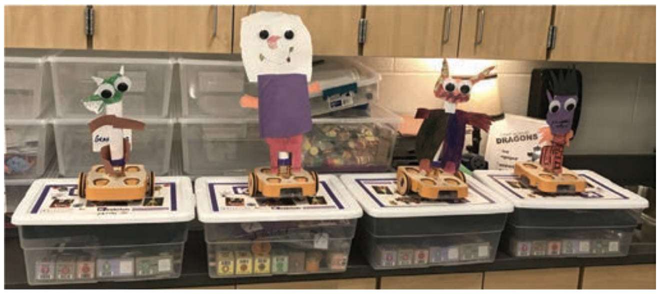


Figure 2. KIBO "Wild Rumpus" final projects created by second grade students

Procedure

Participating teachers received professional development prior to curriculum implementation, as well as ongoing professional learning consisting of virtual coaching and in-person support in the classroom provided by the district's instructional technologists. Teachers implemented the curriculum in their classrooms approximately twice a week (two one-hour lessons). At the end of each lesson, teachers completed a lesson log, a brief online survey that asked questions such as "What were some successes/challenges (if any) during this lesson?" and "Did you modify or adapt the activities in this lesson in any way?". Teachers were observed by the on-site project coordinator or instructional technologist at least two times over the course of the curriculum. Observers used the Positive Technological Development (PTD) Checklist (Bers, 2018) to examine how teachers and students were engaging with KIBO and with the CAL-KIBO lessons.

Measures

Over two years, multiple types of data were collected (see Table 1). Robotics mastery and CT development in both teachers and students were assessed before, during, and after the experience. CT assessments were administered to students who participated in the CAL-KIBO curriculum and to an age and demographically matched comparison group from two other schools in the same district. We collected and analyzed students' standardized literacy scores (DRA and PALS) from the beginning and end of the school year. DRA (Developmental Reading Assessment) is a computerized assessment that evaluates changes in K-8 students' reading level performance. PALS (Phonological Awareness Literacy Screening) is a diagnostic tool that measures children's developing word knowledge, oral reading in context, alphabetic, and phonemic awareness and is used to identify struggling readers and provide additional support.

To understand how teachers reacted to the experience of integrating coding and CT with literacy skills, we conducted surveys, interviews, and focus groups with participating teachers. Surveys were conducted before and after the professional development and consisted of questions related to teachers' self-perceptions of their general coding knowledge (e.g., "I know the definition of an algorithm"), pedagogical content knowledge surrounding how to teach coding (e.g., "I can teach lessons that integrate coding and literacy"), general KIBO robotics knowledge (e.g., "I can recognize common errors with the KIBO programming language and troubleshoot these errors"), knowledge of specific KIBO sensors and modules (e.g., "I know how to use KIBO's Sound Sensor"), attitudes and self-efficacy surrounding the implementation of the CAL-KIBO curriculum (e.g., "I am confident in my ability to implement the CAL-KIBO curriculum in my classroom") and perceptions on literacy (e.g., "What are your priorities in literacy instruction?"). *T*-tests were performed on pre and post-training survey data obtained from $n = 47$ participating first and second grade teachers.

Semi-structured interviews and focus groups with teachers and instructional technologists were conducted at various times (pre-training, during training, pre-curriculum, mid-curriculum, and post-curriculum) and focused on their reactions to KIBO and the CAL-KIBO curriculum. Examples of interview questions included "What has been easy/challenging? How do these activities fit into the rest of your classroom curriculum? If you were to use KIBO again in your classroom, how would you integrate it into your lesson plans?" In each interview, teachers were asked to reflect on their attitudes towards coding and robotics education, their perspectives on teaching literacy and the strengths and challenges of their experiences.

Measuring programming ability and CT skills in young children can be challenging. Two different tools were used to seek proof of learning of specific coding concepts through robotics. First, KIBO Mastery Challenges (KMCs), multiple-choice questions embedded in the curriculum, were administered after specific lessons, and a composite score was calculated from difficulty indices (Hassenfeld et al., 2020). Second, TACTIC-KIBO, a summative assessment of coding and CT skills, was administered after participation in the curriculum. Because children worked in teams throughout the curriculum, KMCs and TACTIC-KIBO provided individualized data regarding learning outcomes that would go unnoticed by just looking at students' final projects. In addition to tool-specific assessments, a validated "unplugged" CT assessment called *TechCheck* was used, which focuses on problem-solving skills of

Table 1. Research Questions and Data Analysis Plan

Research Question	Data Sources	Data Analysis Method
RQ 1: How did the CAL curriculum promote students' coding and CT skills?	TACTIC-KIBO, KIBO Mastery Challenges (KMCs), TechCheck	Descriptive, correlation, t-tests
RQ 2: What was the relationship between students' coding and CT skills and their literacy skills?	KMCs, TechCheck, PALS, DRA	Descriptive and correlation analyses
RQ3: How did teachers react to the experience?	Teacher interviews, surveys, and lesson logs	Thematic analysis, t-tests

the kind required to carry out computer programming without requiring knowledge or experience with coding (Relkin et al., 2020). *TechCheck* requires the transfer of knowledge gained from coding into CT skills useful for solving unplugged challenges that are not explicitly taught in the CAL-KIBO curriculum. *TechCheck* was administered before and after the curriculum in both intervention and control groups (see Table 2). Only results from neurotypical students are included in the analyses that follow since the CT and coding assessment measures have yet to be validated with a neuro-diverse population.

Data Analysis

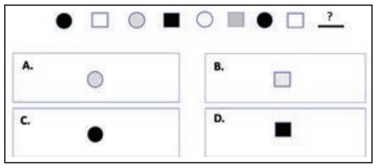
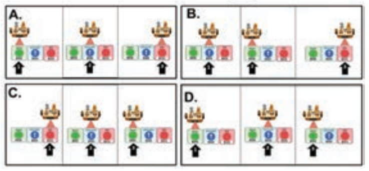
Paired sample *t*-tests and generalized linear mixed modeling were performed on assessment results from students who received CAL-KIBO, as well as students who participated in non-coding classroom activities, to address how the curriculum impacted students' CT skills. To address the relationship between students'CT skills and literacy, correlation, regression, and Bayesian mixed effect modeling were conducted on data from a subgroup of n = 191 students from among the total sample of N = 667 participants representing those who completed pre and post *TechCheck* as well as the DRA and PALS literacy measures.

Teacher interviews and focus groups were transcribed and then analyzed using Braun and Clarke's (2006) six-phase thematic analysis approach. This approach involved reading through the data multiple times, generating initial codes, combining codes into overarching themes, exploring how the themes connected to our initial research question, refining themes with greater detail, and drafting our findings while referring to the data to ensure that our findings provided an accurate representation of teachers' experiences of the curriculum. Common trends derived from this thematic analysis are presented in this paper.

RESULTS

This section organizes the findings based on the three research questions addressed by the study: the CAL curriculum impact on coding and CT skills, the relationship between coding and CT with literacy, and teachers' reactions.

Table 2. Child Study Measures

Measure	*TechCheck*	TACTIC-KIBO	KMCs
Reference	Relkin, de Ruiter, & Bers (2020)	Relkin (2018)	Hassenfeld et al. (2020); Relkin & Bers (2020)
Description	Assessment using "unplugged" (non-coding) tasks to measure CT related problem-solving abilities	Assessment of platform-specific coding and CT abilities in seven sub-domains.	Formative assessment of programming concepts specific to the CAL-KIBO curriculum as "checks of learning". Assesses understanding of semantics and syntax of programming without requiring them to solve problems
Example	What comes next? 	What is the correct order to scan program blocks? 	Which block makes KIBO shake? SHAKE TURN LEFT BEGIN BEEP
Specifications	• 15 multiple-choice questions • Designed for children ages 5-9 • 15 minutes to administer • Score range 0-15 • Validated against expert assessment of CT abilities	• 28 multiple-choice questions • Designed for ages 5-9 • 30-40 minutes to administer • Score range 0-28 • Validated against expert assessment of CT abilities	• 4 assessments, 6 multiple-choice questions each totaling 24 questions • Multiple-choice format • Designed for children ages 5-9 • High interrater reliability • Difficulty index for each question calculated with more weight given to difficult questions. Questions summed into a weighted Difficulty Composite Score

CAL Curriculum Impact on Students' Coding and CT Skills

Descriptive statistics for all coding and CT assessments are shown in Table 3. First and second grade students who participated in the CAL-KIBO curriculum improved significantly on the *TechCheck* assessment, $t(666)= 10.55$, $p < .001$. A grade-matched control group that participated in non-coding classroom activities, the control group, did not significantly improve, $t(180) = 1.81$, $p = .07$. The improvement after 6-8 weeks of CAL-KIBO instruction is consistent with the estimated change in baseline *TechCheck* scores in the absence of coding instruction over approximately six months. A Generalized Linear Mixed Model incorporating taking into account age, grade, classroom, gender, and baseline score revealed that exposure to the CAL-KIBO curriculum was a significant predictor of the *TechCheck* outcome scores, $p < .01$ (Relkin et al., 2021).

We conducted stratified analyses to look for effects by grade. First grade students who received CAL-KIBO improved significantly on *TechCheck*, $t(270) = 9.21$, $p < 0.001$, whereas the control group did not, $t(358.31) = 1.07$, $p = .95$. Second graders in the CAL group also improved significantly, $t(395) = 6.11$, $p < 0.001$ but not as much as first graders possibly due to a ceiling effect on the *TechCheck* assessment in which high baseline scores in second graders reduced the window for observing change (Relkin et al., 2020). A more challenging version of the assessment for second graders has since been created to address this issue (Relkin et al., 2021). A smaller but borderline significant improvement was observed in the second grade non-coding group, $t(109) = 2.34$, $p = .05$ possibly due to a learning effect or chance. Results stratified by grade shows that although first and second graders both improved on *TechCheck* more than their non-coding counterparts, we observed more of a difference in first graders.

Relationships Among Students' Coding, CT, and Literacy Skills

The theoretical framework upon which the CAL curriculum is designed proposes that learning computer programming allows children to gain an alternative form of literacy that permits self-expression in ways that are similar to reading and writing (Bers, 2019; Vee, 2017). Thus, in this study we examined the correlations between coding, CT, and conventional measures of literacy. Specifically, we wanted to know if students who scored higher on state-wide assessments of literacy also performed higher in coding and CT tasks after they completed the CAL-KIBO curriculum. Our measure of coding ability was the student's KMC composite score, and our measure of CT was the student's *TechCheck* score post-curriculum. To estimate students' literacy skills before the intervention, we used Fall standardized literacy scores (DRA and PALS) obtained by the school. There was a moderate positive correlation (Pearson) between baseline *TechCheck* scores and both the DRA ($r = 0.39$, $p < .0001$) and PALS ($r = 0.33$, $p < .0001$). We carried out linear regression to examine whether these two baseline literacy measures predicted the endpoint *TechCheck* scores when baseline *TechCheck* performance was taken into account. A model containing baseline PALS and *TechCheck* scores significantly predicted end point *TechCheck* $F(2, 190) = 38.47$, $p <.0001$. A model containing baseline DRA and *TechCheck* scores also significantly predicted end point *TechCheck* $F(2, 190) =36.18$, $p <.0001$. We conducted Bayesian mixed-effects modeling with *TechCheck* scores as the outcome variable and PALS and DRA literacy measures as well as baseline *TechCheck* scores as predictors. The Bayes factor was >100 indicating "decisive evidence" that baseline literacy measures (PALS and DRA) were predictors of end point *TechCheck* score when baseline *TechCheck* score was included in the model. In summary, these analyses indicate a possible relationship

Table 3. Descriptive Statistics for Coding and CT Variables

	n	Mean (SD)	Min	Max
***TechCheck* Baseline CAL Group**[a]	667	10.09 (2.61)	3	15
***TechCheck* End Point CAL Group**	667	11.03 (2.55)	2	15
***TechCheck* Baseline No-CAL Group**	181	9.50 (2.38)	3	14
***TechCheck* End Point No-CAL Group**	181	9.77 (2.55)	4	14
***TACTIC-KIBO* First Grade**[b]	214	13.10 (3.33)	2	20
***TACTIC-KIBO* Second Grade**	398	18.28 (3.90)	4	26
***KMCs* Second Grade**[c]	217	3.44(1.05)	1.59	6.25

Note.
[a] *TechCheck* assesses children's unplugged problem solving and CT
[b] The Tufts Assessment of Computational Thinking in Children - KIBO version (*TACTIC-KIBO*) assesses children's platform specific coding and CT skills
[c] *KMCs* (KIBO Mastery Challenges) assesses children's KIBO coding proficiency

between the acquisition of CT skills and baseline literacy skills. This could be a consequence of the measures reflecting a child's developmental stage or literacy skills influencing assessment performance on *TechCheck* rather than a specific effect on learning CT.

Teachers' Reactions

Most teachers expressed a high level of engagement when first introduced to KIBO during the training and displayed excitement while working on their own robotic projects. Results of *t*-tests indicated statistically significant increases in each of the 27 survey items assessing $n = 47$ teachers' knowledge of general coding concepts, KIBO skills, and CS pedagogy as well as their attitudes towards coding and robotics education, *t*'s ranging from 5.19-22.40, $p < .001$ across all t-tests. Across all survey items and domains, neither race/ethnicity nor teaching experience impacted participant responses.

We present our qualitative findings from teachers in Table 4, which summarizes teachers' perceived successes and challenges of their curriculum experience, reactions to the coding-literacy integration, and overall factors that impacted curricular implementation. In terms of the CAL curriculum, teachers' interviews and surveys showed that teachers enjoyed KIBO and their students did as well. However, the organization of robotics materials presented its own set of challenges, particularly with shifting materials between classrooms and managing clean-up time. Teachers developed different strategies such as creating a rotational system with fellow teachers, selecting students to be in charge of robotics clean-up, and keeping a set of 3-4 KIBOs in their classrooms at all times. Despite these logistical challenges, during interviews, teachers described being drawn to the hands-on, tangible nature of KIBO, especially in comparison to other screen-based applications felt. Teachers felt KIBO was engaging and developmentally appropiorate for their students.

Table 4. Teachers' Reflections of the CAL-KIBO Curriculum Experience

Topic	Theme	Illustrative Quote
Successes and challenges	Coding beyond the screen	"I learned that coding doesn't just involve sitting in front of a computer and typing things and that it actually involves just using your mind and talking things out and stuff like that"
	KIBO organization	"Because I was sharing with [name omitted] and others that only did it once a week, so [the KIBOs] were in and out, and things got mixed up, so I color-coded mine"
Coding-literacy integration	Resistant to integration	"I almost think they should take the writing components out of it, and just let us focus on the actual straight coding."
	Neither resistant nor receptive	"I'm not reinforcing, 'Oh, capitalization, grammar, this and that', like that's just not happening...I do feel like it hit, definitely, on oral communication... They have to communicate with their buddy, whoever they're working with"
	Receptive to integration	"Each day with each lesson the kids were writing...and reading different things. They had to read [it] over. They had to read other people's instructions."
Factors impacting curricular implementation	Time spent preparing for and implementing lessons	"Most of the lessons are supposed to be an hour, I know, but mine were probably two or more... I was able to tie [the Design Journals] in with the writing more because I spent more time on it."
	Teacher collaboration and utilization of resources	"One of our teachers broke down the KIBO [lesson] for the day and made a PowerPoint, so that we would be able to follow through and...check off the steps as we did them."
	Competing priorities of other lessons and activities	"Teachers are always pressed with a pretty comprehensive curriculum, so adding this in addition was a little overwhelming at times."
	Classroom management	"I felt even groups of four would be too much of a chaotic ruckus. And I felt like the kids would be most successful if they're just working in a partnership."
	Flexibility in adapting lesson activities	"I know everybody adapted and I adapted it by adding extra time. But I stuck to the curriculum. I know some people kinda cut out certain things and whatever, but I wanted to give them the full experience, so I pretty much went by the book."

Teachers varied in how they responded to the integration of literacy in the CAL curriculum; however, there was a distinct trend amongst second grade teachers. During interviews and focus groups, it became clear that teachers who understood literacy instruction as singularly focused on discrete skills (e.g., phonics, punctuation, etc.) were less open to the CAL curriculum and to the overall integration of CT, robotics, and literacy. Conversely, teachers who understood literacy in broader terms and saw meta-cognitive ideas and concepts about reading and writing as essential to the development of robust literacy abilities (e.g., communication, creative expression, awareness of audience and purpose, etc.) were more open to the curriculum.

The analysis of teacher interviews and focus groups revealed several factors that impacted teachers' overall experience: time spent preparing and implementing lessons; collaboration and utilization of resources; competing priorities of other lessons and activities; classroom management; and flexibility in adapting lesson activities. Individual classroom and school contexts played an important role. For instance, teachers who were more successful with the curriculum had manageable classroom sizes, flexible schedules to accommodate CAL lessons, and an adequate number of robotic kits for students to work in small groups. Conversely, teachers who had a large number of students with little floor space, taught in an open-classroom setting, or had rigid grade-level schedules faced more challenges.

Discussion

Participation in the CAL-KIBO curriculum was associated with improvement in coding and unplugged CT skills. It is noteworthy that the measure of unplugged CT (*TechCheck*) showed improvement with exposure to CAL-KIBO even though the curriculum did not explicitly include the types of unplugged activities in *TechCheck*. This finding supports the assertion that the problem-solving improvements were a consequence of knowledge and skills gained while learning to code and not a function of explicit instruction in solving unplugged challenges.

Taken together, our analysis suggests that baseline literacy skills were related to students' acquisition of CT skills. Students who had higher PALS or DRA scores at the beginning of the term were more successful in CT tasks measured by *TechCheck*. These findings may help us develop effective integrated CS curricula and identify core skills that need to be strengthened so that all students can reap the benefits of early childhood computer education.

We encouraged teachers to adapt the curriculum based upon their students' needs and available time. As a consequence, there was variability in the fidelity of implementation of the CAL-KIBO curriculum across schools and classrooms. Other sources of variation included having a different number of robots available in each school. The classroom dynamic differed when KIBO robots were shared by pairs of students as opposed to larger groups of 5-6 students. Classroom size and space were other variables that impacted the classroom experience. The CAL-KIBO curriculum was successful in engaging teachers and generating a high level of enthusiasm. Limitations on time to implement the curriculum and difficulty in organizing materials were common challenges reported by educators.

Implications

The key to successful educational initiatives is to make authentic connections to the teaching that is already going on in the classroom. In this study, we connected coding and CT to literacy. Our findings indicate that first and second grade students improved in their coding and CT skills as a result of participating in the CAL-KIBO curriculum. Although teachers varied in their perceptions of integrating coding and CT with literacy, our findings suggest that these disciplines may share some cognitive and pedagogical overlap that has yet to be extensively explored in the early computing education field. This integration can have a positive impact regarding learning outcomes. In addition, children who participated in the CAL-KIBO curriculum did better on unplugged CT challenges (*TechCheck*) than their counterparts. This improvement occurred despite unplugged CT challenges not being an explicit part of the CAL-KIBO curriculum, suggesting that a transfer of knowledge took place.

Based on our study, we provide the following recommendations for practitioners seeking to integrate CT and coding in their classrooms:

- Time: Allocate enough time in the weekly schedule to prepare for and implement the curriculum. Implementing the curriculum in the winter or spring enables teachers to utilize established classroom routines and behavioral expectations, which are key to maximizing young student's engagement and learning.
- Curricular Alignment: Although the CAL-KIBO curriculum focused primarily on the connections between literacy and CS, and was taught during the literacy block, teachers found ways to connect the activities to other curricular domains such as science and math. We recommend aligning with multiple subject areas, not just literacy, while still framing the teaching of coding as a form of creative expression and communication. The more teachers could see how the lesson aligned with other content instruction, the more they felt comfortable teaching.
- Resources: Teachers benefitted from collaborating with one another, using the virtual and face-to-face support resources, having access to knowledgeable support staff, and co-teaching lessons with instructional technologists. These findings provide insight into the

types of professional learning that early elementary teachers may require to feel comfortable and confident when teaching coding and robotics.

The CAL-KIBO curriculum focused on a single robotics coding platform, KIBO. We have also developed a version of the CAL curriculum that utilizes the free ScratchJr introductory programming language and are currently conducting studies. Future work will address the relative strengths and weaknesses of both curricula regarding different coding platforms and develop collaborations with other school districts in the U.S. and abroad. We will also explore if the CT skills acquired through one programming language transfers to another one.

References

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832–835. https://doi.org/10.1093comjnl/bxs074

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? *Inroads, 2*(1), 48–54. https://doi.org/10.1145/1929887.1929905

Barron, B., Cayton-Hodges, G., Bofferding, L., Copple, C., Darling-Hammond, L., & Levine, M. (2011). Take a giant step: a blueprint for teaching children in a digital age. New York: The Joan Ganz Cooney Center at Sesame Workshop. https://joanganzcooneycenter.org

Bell, T., & Vahrenhold, J. (2018). CS unplugged—how is it used, and does it work?. In *Adventures between lower bounds and higher altitudes* (pp. 497–521). Springer, Cham. https://doi.org/10.1007/ 978-3-319-98355-4_29

Bers, M. U. (2018). *Coding as a Playground: Programming and Computational Thinking in the Early Childhood Classroom*. New York, NY: Routledge Press.

Bers, M. U. (2019). Coding as another language: a pedagogical approach for teaching computer science in early childhood. *Journal of Computers in Education, 6*(4), 499-528.

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

Code.org (2019). https://code.org/

Core Team, R. (2019). *R: a language and environment for statistical computing*. Vienna: R Foundation for Statistical Computing. https://www.R-project.org/

ISTE, CSTA (2011). *Operational definition of computational thinking for K–12 education*. https://id.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf

Cuny, J., Snyder, L., & Wing, J. M. (2010). Demystifying computational thinking for non-computer scientists. Unpublished manuscript in progress, referenced in https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Fayer, S., Lacey, A., & Watson, A. (2017). *BLS spotlight on statistics: STEM occupations-past, present, and future*. Washington, D.C.: U.S. Department of Labor, Bureau of Labor Statistics. https://www.bls.gov

Grover, S., & Pea, R. (2013). Computational thinking in K–12: a review of the state of the field. *Educational Research, 42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

Hassenfeld, Z. R., Govind, M., de Ruiter, L. E., & Bers, M. U. (2020). If You Can Program, You Can Write: Learning Introductory Programming Across Literacy Levels. *Journal of Information Technology Education: Research*, 19, 65-85. https://doi.org/10.28945/4509

Hermans, F., & Aivaloglou, E. (2017). To Scratch or not to Scratch? A controlled experiment comparing plugged first and unplugged first programming lessons. WIPSCE 2017. *Proceedings of the 12th workshop in primary and secondary computing education* (pp. 49–56).

Kalelioğlu, F., Gülbahar, Y., & Kukul, V. (2016). *A framework for computational thinking based on a systematic research review*. https://www.researchgate.net/publication/303943002_A_Framework_for_Computational_Thinking_Based_ on_a_Systematic_Research_Review

Lu, J. J., & Fletcher, G. H. (2009). Thinking about computational thinking. In *ACM SIGCSE Bulletin* (Vol. 41, No. 1, pp. 260-264). ACM. https://doi.org/10.1145/1539024.1508959

Relkin, E. (2018). Assessing young children's computational thinking abilities (Master's thesis). Retrieved from ProQuest Dissertations and Theses database. (UMI No. 10813994).

Relkin, E. & Bers, M. U. (2019). Designing an Assessment of Computational Thinking Abilities for Young Children. In L.E. Cohen & S. Waite-Stupiansky (Eds.), *STEM for Early Childhood Learners: How Science, Technology, Engineering and Mathematics Strengthen Learning* (pp. 85-98). New York, NY: Routledge.

Relkin, E., de Ruiter, L., & Bers, M. U. (2020). TechCheck: Development and Validation of an Unplugged Assessment of Computational Thinking in Early Childhood Education. *Journal of Science Education and Technology*. https://doi.org10.1007/s10956-020-09831-x

Relkin, E., de Ruiter, L., & Bers, M. U. (2021). Learning to Code and the Acquisition of Computational Thinking by Young Children. *Computers & Education*.

Rodriguez, B., Rader, C., & Camp, T. (2016). Using student performance to assess CS unplugged activities in a classroom environment. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 95-100). ACM. https://doi.org/10.1145/2899415.2899465

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

Sullivan, A., Bers, M. U., & Mihm, C. (2017). Imagining, Playing, & Coding with KIBO: Using KIBO Robotics to Foster Computational Thinking in Young Children. *Proceedings of the International Conference on Computational Thinking Education*. Wanchai, Hong Kong.

Sullivan, A., Elkin, M., & Bers, M. U. (2015). KIBO Robot Demo: Engaging young children in programming and engineering: *Proceedings of the 14th International Conference on Interaction Design and Children (IDC '15)*, Medford, MA, June 21-25. New York, NY: ACM

Thies, R., & Vahrenhold, J. (2013). On plugging "unplugged" into CS classes. *Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13)*. Association for Computing Machinery, New York, NY, USA, 365–370. https://doi.org/10.1145/2445196.2445303

U.S. Department of Education, Office of Educational Technology (2017). Reimagining the role of technology in education: 2017 National Education Technology Plan update. https://tech. ed.gov/teacherprep

Vee, A. (2017). *Coding Literacy: How Computer Programming Is Changing Writing*. Cambridge, MA: The MIT Press. https://mitpress.mit.edu/books/coding-literacy

Virginia Department of Education (2021). 2016 Virginia Acts of Assembly Item 138, page 117. https://www.doe.virginia.gov/instruction/computer-science/2016-acts-of-assembly-page117.pdf

White House (2016). *Educate to innovate*. https://www.whitehouse.gov/issues/education/k-12/educate-innovate

Wing, J. M. (2006). Computational thinking. *CACM Viewpoint, 49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Wing, J. (2011). *Research notebook: computational thinking—What and why?* The Link Magazine, Spring. Carnegie Mellon University, Pittsburgh. https://www.cs.cmu.edu/link/researchnotebookcomputational-thinking-what-and-why

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding a 21st century problem solving in K-12 classrooms. *TechTrends 60*, 565-568. DOI: 10.1007/s11528-016-0087-7.

Yadav, A., Good, J., Voogt, J., & Fisser, P. (2017). Computational thinking as an emerging competence domain. In *Technical and vocational education and training* (Vol. 23, pp. 1051–1067). https://doi.org/10.1007/978-3-319-41713-4_49