Liu, R., Maeng, J. L., Moots, S. C., & Garner, J. K. (2025). Building elementary teachers' capacity for computer science instruction through professional development: A randomized control trial. *Contemporary Issues in Technology and Teacher Education, 25*(1), 87-126.

Building Elementary Teachers' Capacity for Computer Science Instruction Through Professional Development: A Randomized Control Trial

Ruohan Liu
Seattle University

Jennifer L. Maeng
University of Virginia

Shanan C. Moots & Joanna K. Garner
Old Dominion University

This randomized control trial evaluated the impact of a year-long professional development (PD) program on elementary teachers' CS content knowledge, self-efficacy, and CS implementation. It also investigated teachers' PD experiences and expectations. The findings suggest that treatment teachers' self-efficacy for teaching CS significantly improved as a result of the PD, and a statistically significant difference was found between treatment and control teachers' self-efficacy by the end of the PD. A higher percentage of treatment teachers than control teachers reported implementing CS-integrated lessons in their classrooms, and treatment teachers identified effective activities to engage students in learning CS and several barriers they experienced in CS integration. Treatment teachers' CS content knowledge remained the same from before to the end of the PD year, but a statistically significant decrease was observed for control teachers' CS content knowledge from pre- to end of year. Between group year-end differences were not statistically significant. At the end of the PD year, more treatment teachers demonstrated greater CS knowledge on 3 of 5 items than control teachers. Findings highlight the potential need for continual, sustained, and relevant CS PD over the academic year. Further study is underway with additional cohorts of the PD model.

To fully participate in a world where computing technology is ubiquitous, children need to begin learning computer science (CS) at an early age (Febrian et al., 2018; Kale et al., 2018; Saad, 2020). Elementary school is an ideal place to start introducing CS, as many children this age already have experience interacting with computers; however, they may not have a clear understanding of how computers work and need teachers' guidance (Israel et al., 2015; Kafai & Burke, 2014; Palfrey & Gasser, 2008). Introducing CS to elementary students helps develop their technological fluency, problem-solving, and critical thinking skills (Israel et al., 2015; Mladenović et al., 2017) and fosters positive perceptions of CS, which can motivate students to pursue CS-related pathways in the future (e.g., Tran, 2018).

Over the past few years, efforts to promote CS instruction in K-12 schools have increased. In the United States, initiatives such as "CS for All" and "Hours of Code" have promoted CS learning among K-12 students and teachers. Many states have adopted legislation requiring public schools to offer CS instruction in a phased manner (e.g., Code.org, CSTA, & ECEP Alliance, 2022).

At the elementary level, teachers are typically expected to introduce CS by integrating CS concepts and practices into core content areas (e.g., math, science, English language arts; Century et al., 2020; Ozturk et al., 2018). Despite these efforts, broadening elementary student access to CS instruction is particularly challenging because many elementary teachers lack pre- or in-service preparation in this area (Mason & Rich, 2019; Rich & Hodges, 2017; Stanton et al, 2017). Acknowledging this issue, the present study implemented a year-long PD program designed to support elementary teachers' CS integration and examined changes in teacher CS content knowledge, CS self-efficacy, and integration of CS curriculum standards following the PD. This study aimed to expand the knowledge of how to develop elementary teachers' capacity for CS integration.

Literature Review

U.S. K-12 Computer Science Instruction

The K-12 Computer Science Teacher Association (CSTA, 2017) has outlined five core concepts (i.e., computing systems; networks and the internet; data and analysis; algorithms and programming; and impacts of computing) and seven core practices (i.e., fostering an inclusive computing culture; collaborating around computing; recognizing and defining computing problems; developing and using abstractions; creating computational artifacts; testing and refining computing artifacts; and communicating about computing) that should be included in K-12 CS instruction. In the U.S., the CSTA Standards have been broadly adopted or modified to guide CS instruction. Still, CS is not often a top priority in elementary grades as it is not directly connected with standardized tests (Google & Gallup, 2015). As recently as 2023, only 11.3% of elementary students had access to foundational CS instruction (Code.org, CSTA, & ECEP Alliance, 2023).

CS instruction in elementary grades is implemented by integrating CS within other core curricula, creating opportunities to engage all students

in CS learning. The interconnection of CS concepts and skills with fundamental elementary subjects paves the way for interdisciplinary learning experiences (Mason & Rich et al., 2019; Rich et al., 2019; Waterman et al., 2019). For example, in CS, *decomposition* means breaking down a complex problem or system into smaller, manageable components (K-12 Computer Science Framework, 2016). In elementary mathematics, *decomposition* helps students solve math problems strategically (i.e., breaking down a complex arithmetic problem into multiple steps and doing multiplication and division before addition and subtraction; Gane et al., 2021). Understanding core CS concepts can help students apply them to solve problems in different disciplinary areas and support their learning in other disciplines (Kwon et al., 2021).

Opportunities for early engagement in CS integration play a crucial role in fostering the interest, curiosity, and enthusiasm necessary for sustained involvement in CS and related domains (Armoni & Gal-Ezer, 2014). However, a significant disparity exists regarding K-12 students' access to CS. For example, the 2023 State of Computer Science Education Report points out that female students are enrolled in foundational CS instruction at a rate two times lower than male students. Hispanic and Black students are also markedly underrepresented; their participation in foundational CS is 1.5 times lower than their White and Asian peers (Code.org, CSTA, & ECEP Alliance, 2023). These disparities may be related to the limited offering of CS instruction in schools with larger percentages of minority students (Warner et al., 2021), resulting in the lack of opportunities for minority students to engage in CS at an early age.

Teacher CS Content Knowledge and Self-efficacy

Teacher knowledge and self-efficacy influence instructional practice and student outcomes (Shulman, 1986; Yadav, Gretter et al., 2016; 2018). Implementing effective CS instruction requires substantial knowledge of CS content, pedagogical strategies, and appropriate technology (Basu & Tate, 2021). Lack of knowledge in any of these areas may hinder a teacher's implementation of CS instruction (Mason & Rich, 2019). In terms of CS content knowledge, the CSTA Standards specify subconcepts that teachers need to understand to teach CS in elementary classrooms, such as devices, hardware and software, variables, algorithms (e.g., 1B-CS-01, 1B-CS-02, 1B-AP-08, etc.; CSTA, 2017). By developing a deep understanding of these foundational CS concepts, teachers can create engaging and meaningful learning experiences that empower students in CS learning (Yadav, Berges et al., 2016).

A teacher's self-efficacy in teaching CS also influences their instructional practice (Rich et al., 2021; Zhou et al., 2020). Self-efficacy refers to a person's belief in their capacity to perform certain tasks successfully or achieve specific outcomes and is thought to be linked to students' educational success (Bandura, 2006; Tschannen-Moran & Hoy, 2001). Teachers with high self-efficacy are more likely to put more effort into learning and teaching CS, whereas teachers with low self-efficacy may be reluctant to teach or choose not to teach CS (Mason & Rich, 2019; Rich et al., 2021). Low self-efficacy is a major barrier many novice teachers experience when implementing CS instruction, and it is often related to limited CS content knowledge, unfamiliarity with CS pedagogies, lack of

instructional support, and feelings of isolation (Buss & Gamboa, 2017; Israel et al., 2022; Yadav et al., 2016).

PD to Support Elementary Teachers

Recent efforts have been made to support the development of K-12 teachers' CS content and pedagogical knowledge through preservice teacher preparation and PD programs (e.g., Hubbard & D'Silva, 2018; Jocius et al., 2020; Milliken et al., 2019). A systematic review of K-12 CS PD programs over a 10-year period indicated that most programs lacked a mechanism to provide sustained support to teachers after the initial workshop (Luo & Liu, 2022).

Sustained PD with a clear content focus and ongoing support is essential to reinforce teacher knowledge, develop teacher self-efficacy, and address challenges in CS implementation (Darling-Hammond et al., 2017). Additionally, as elementary teachers are expected to integrate CS into other subjects, continued and targeted PD support to help teachers identify the connection between CS and other subjects is crucial for effective CS integration (Bers et al., 2022, Liu, 2022).

Studies focusing on elementary teacher outcomes following CS PD have investigated teacher experience, engagement, and understanding of CS concepts (e.g., Hestness et al., 2018; Ketelhut et al., 2019; Yadav et al., 2018;). Two studies, both observational, reported improvement in elementary teacher CS content knowledge and self-efficacy after PD (e.g., Ketelhut et al., 2019; Rich et al., 2021). For example, Ketelhut et al. implemented a series of PD workshops to support elementary teacher integration of CS into elementary science curricula. After analyzing teacher reflection journals, Ketelhut et al. concluded that PD participation contributed to changes in teacher CS perceptions and practice. Rich et al. explored the development of elementary teacher CS self-efficacy after participating in a 5-day PD workshop. Results of post-PD survey responses indicated improved teacher self-efficacy following PD.

While these studies offer important insights into teacher changes and outcomes following PD participation, none employed an experimental design, making it challenging to ascertain how and to what extent changes would be attributed to PD. Randomized control trials (RCT) provide a rigorous method to evaluate the effectiveness of a particular treatment or intervention (Hariton & Locascio, 2018). RCTs have been used to evaluate the impact of PD programs on teacher outcomes across content areas, including science, physical education, language education, and social science (Barr et al., 2015; Escriva-Boulley et al., 2018; Tong et al., 2015).

For example, Maeng et al. (2020) implemented an RCT to evaluate changes in elementary science teachers' conceptual understanding, confidence, and classroom implementation of problem-based learning (PBL) after a year-long PD program focusing on science teaching methods. Results indicated significant improvement in treatment teachers' understanding and confidence in implementing the targeted teaching strategies compared to control teachers, while also identifying PD components that contributed to teacher improvements.

Similarly, Schoen et al. (2019) employed an RCT to evaluate the impact of a 2-week summer institute on secondary teacher content and pedagogical knowledge in statistics and probability. Results indicated a significant positive impact of the program on teacher content knowledge and indicated that the program had a greater impact on more experienced teachers.

Despite their use in other content areas, the lack of application of RCT designs to evaluate CS PD may raise the question of whether the observed teacher outcomes can be attributed to PD participation. Teachers who volunteer for CS PD may possess an inherent interest and motivation to learn CS, potentially influencing positive post-PD outcomes. An experimental design, such as a randomized control trial (RCT), can provide evidence for the extent to which a PD program contributes to changes in teacher CS knowledge, self-efficacy, and implementation of CS instruction.

Present Study

Using an RCT design, we investigated how elementary teacher understanding of five fundamental CS concepts emphasized in both national and state CS standards (i.e., computer science, computer programmer, computing device, algorithm, and variable) changed after participation in PD. We also examined changes in teacher self-efficacy in terms of their confidence in implementing computing technology, teaching programming concepts, and building CS-integrated lessons as a result of participation. This study investigated the following research questions:

1. How did treatment teacher CS content knowledge change as a result of participation in the PD, and how did their CS content knowledge compare to that of control group teachers?

2. How did treatment teacher self-efficacy in teaching CS change as a result of participation in the PD, and how did their self-efficacy compare to that of control group teachers?

3. How did treatment teachers' self-reported implementation of each CS curriculum standard compare with control teachers?

4. How did treatment teachers perceive their PD experience, and what additional support did they request?

We hypothesized that teachers assigned to the treatment condition would have greater CS content knowledge and CS self-efficacy compared to teachers assigned to the control condition and that a greater proportion of treatment teachers would implement CS lessons during the school year compared to control teachers.

Theoretical Framework

Situated learning theory (SLT) served as the theoretical framework for this study. SLT assumes that understanding is constantly under construction,

knowledge must be learned in an authentic context of how it will be used, and knowledge is developed as a result of interactions between individuals (Orgill & Bodner, 2007). SLT also stresses the role of social interactions during learning, emphasizing learning through authentic activities and in an authentic context. Informed by SLT, teachers will more successfully integrate CS into instruction when they learn CS in a context like that in which they will be teaching it. Cognitive apprenticeship, coaching (expert support), practice, collaboration, and reflection are key features of PD developed from a situated learning perspective (McLellan, 1996).

The PD program that served as the context for the present study was aligned with SLT, supporting teachers to apply their CS knowledge and skills in their own classrooms and providing teachers with sustained support across an entire academic year, including both a summer PD institute and follow-up academic-year support. Additionally, the PD program in the present study incorporated other effective PD practices: clear content focus, active learning activities, collaboration, coaching and expert support, modeling, and offering feedback and opportunities for reflection (as recommended by Darling-Hammond et al., 2017). Activities to help teachers build self-efficacy through the development of their CS knowledge, instructional expertise, and a sense of community in CS learning were implemented into the PD. Specific PD content and activities are detailed in the method section.

Methods

This study used an RCT design to investigate changes in CS content knowledge, self-efficacy, and CS implementation for teachers participating in PD (treatment condition) compared to teachers in a control condition (i.e., business as usual). Data sources include pre and post summer institute, mid-year, and year-end Likert scale and open-ended survey responses, and PD observations. Approval to conduct human subjects research was secured prior to the commencement of the study activity.

Randomization and Sample

This study was implemented in one of the first states that mandated K-12 CS education. K-5 teachers (n = 77) from 23 schools applied and met the criteria to participate. Randomization was conducted at the school level: half of the schools that applied were assigned to the treatment condition (11 schools, 33 teachers), in which teachers participated in the PD program immediately. The other half of schools were assigned to the delayed treatment condition (i.e., referred to as control condition – business as usual – in this study; 12 schools, 44 teachers), in which teachers were provided the opportunity to participate in the PD program in the year following data collection (Figure 1).

By the end of the year-long study, 29 participants from 10 schools in the treatment group and 39 participants from 10 schools in the control group were retained. After excluding invalid and missing data, data of 16 treatment teachers from 10 schools and 21 control teachers from 10 schools were included for analysis (Table 1). Participants reported teaching experience ranging from 2 to 26 years; most participants (n = 27)

had over 10 years of teaching experience. School composition is presented in Table 2.

Figure 1
RCT Design

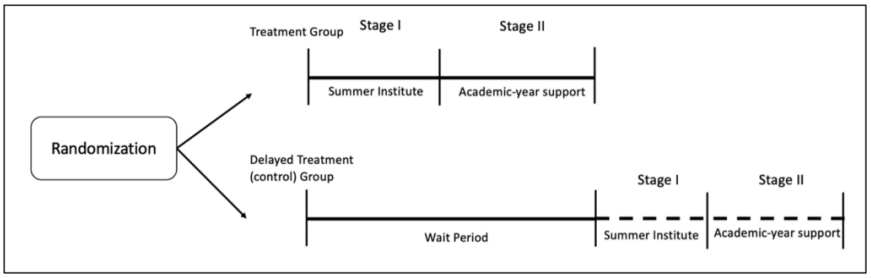


Table 1
Participant Demographic and Educational Background

Category	Treatment (*n* = 16)	Control (*n* = 21)
Gender		
Male	0 (0%)	2 (9.5%)
Female	16 (100%)	19 (90.5%)
Race/Ethnicity		
White	14 (87.5%)	18 (85.7%)
Black	1 (6.3%)	2 (9.5%)
Asian	1 (6.3%)	0 (0%)
American Indian/Alaska Native	0 (0%)	1 (4.8%)
Hispanic	0 (0%)	0 (0%)
Educational Background [a]		
Has Education degree	15 (93.8%)	19 (90.5%)
Has STEM degree	0 (0%)	0 (0%)
Elementary emphasis	9 (56.3%)	17 (81.0%)
Secondary emphasis	0 (0%)	0 (0%)
SPED emphasis	1 (6.3%)	0 (0%)
Ed tech emphasis	2 (12.5%)	0 (0%)
Other emphasis [b]	3 (18.8%)	2 (9.5%)

Category	Treatment (*n* = 16)	Control (*n* = 21)
Grade Level		
K-2	5 (31.3%)	3 (14.3%)
3-5th grade	3 (18.8%)	9 (42.9%)
K-5	7 (43.8%)	7 (33.3%)
Not reported	1 (6.3%)	2 (9.5%)
[a] Educational background can be more than 100% because participants could select multiple options. [b] Other emphasis includes childhood education, music education, education leadership, ESOL, and library science.		

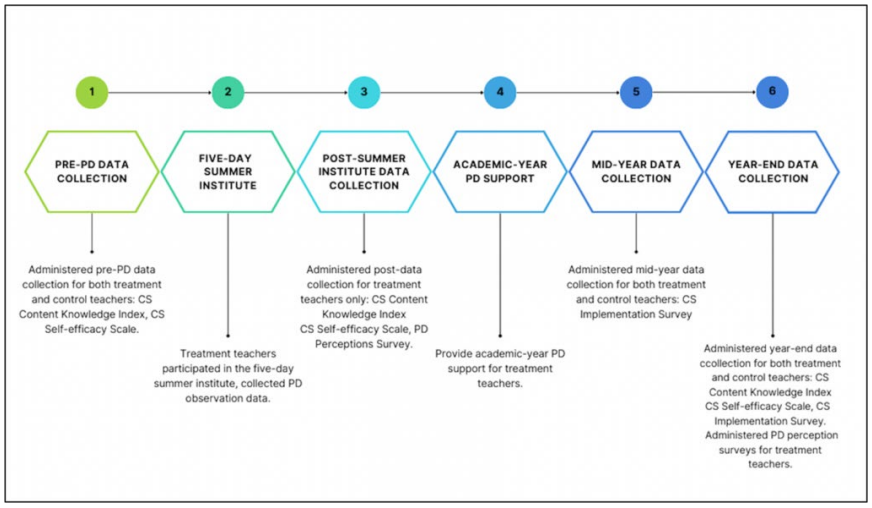
Table 2
School Demographics

Group	Enrollment Mean (*SD*)	URP Mean % (*SD*) [a]	FRPM Mean % (*SD*) [b]
Treatment (*n* = 10)	555 (225)	45.2% (.30)	47.5% (.22)
Control (*n* = 10)	527 (230)	46.0% (.26)	48.1% (.20)
[a] Underrepresented population (i.e., Black, Hispanic, American Indian/Alaska Native, Native Hawaiian/Pacific Islander). [b] Receive free and reduced-price meals			

PD Context and Intervention

The PD program that served as the context for the present study was designed to develop K-5 teachers' CS content knowledge, self-efficacy, and expertise to integrate CS into their existing curricula and align with the previously described principles of SLT. The PD also aimed to prepare teachers to support other teachers at their school in CS instruction. The intervention included two components: a 5-day summer institute and the academic-year PD support (Figure 2).

Figure 2
The Timeline of the PD Implementation and Data Collection



Summer Institute

The 5-day (25 hours) online summer institute took place during summer 2021. It was facilitated by CS experts from a statewide CS advocacy and teacher training organization. The daily schedule of the summer institute consisted of 2.5-hour synchronous sessions in the morning with 1.5-hour asynchronous meetings individually or in groups and a 1-hour "office hours" session in the afternoon. Informed by the effective PD design elements (Darling-Hammond et al., 2017), this PD program incorporated clear CS content focus and specific learning objectives that teachers would achieve:

- Foundational understanding of CS and CS elements/concepts.
- Knowledge of CS curriculum standards.
- Coding skills using Scratch programming language.
- Understanding of how to design and teach CS-integrated lessons.
- Ability to plan and implement local CS professional learning activities.
- Awareness of resources and tools to support teacher and student learning in in-person and online classrooms.
- Confidence in coaching others in CS education.

Additionally, active learning, peer collaboration, and expert coaching and modeling were featured in this program. During the summer institute, teachers engaged in a variety of activities and strategies, such as unplugged activities and programming to learn to integrate CS Standards into elementary subject such as reading, writing, science, mathematics, and social studies. Each day of the institute focused on a specific CS content; participants were provided with explicit lectures and collaboration opportunities to learn and practice CS pedagogies and coaching skills (Table 3).

Table 3
Daily Schedule of Summer Institute

Day	Content Focus	Examples of Activities
1	- Introduction to CS. - Dive deep into state standards of CS instruction. - Explore unplugged lessons, curriculum frameworks, and lesson design.	Lectures on CS definitions, reflection and discussion, group work in unplugged lessons, develop CS-integrated lessons, debrief and share.
2	- Introduction to CS integration. - Identify connections between core content and CS standards. - Explore tools of integration and how to plan for integrated lessons.	Lectures on CS integration, reflection and discussion, group work to identify connections between CS concepts and concepts in other subjects, group work to develop CS-integrated lessons, debrief and share.
3	- Build CS coaching identity. - Identify CS coaching strategies. - Build a PD plan to introduce CS to team members.	Lectures on CS coaching, reflection and discussion, role-play activities, group work to create a CS coaching plan, debrief and share.
4	- Explore issues of equity and inclusion in CS education. - Learn about equitable CS pedagogy and teaching practices. - Explore accessibility toolkits for CS instruction.	Lectures on equity, inclusion, and accessibility in CS education, reflection and discussion, group work to revise CS coaching plans, discussion on TED talks and articles about equity issues in CS education.
5	- Become a CS advocate. - Explore resources for CS instruction. - Practice unplugged CS instruction. - Practice CS coaching.	Discussion and reflection, demonstration of CS instructional resources, practice CS coaching, group work on CS lesson plan analysis.

Academic-Year PD Support

To ensure a sustained duration of professional support, the PD program extended into the academic year through a Networked Improvement Community (NIC; McKay, 2017). The present study implemented the NIC by (a) encouraging teachers to work collaboratively to address identified problems in CS integration; (b) connecting teachers with CS coaches regularly through the academic year for resource sharing and knowledge support; (c) providing access to a suite of webinars and field-tested instructional materials to support teachers' CS integration; and (d) offering four 2-hour themed online PD sessions (i.e., Using Unplugged to Boost Plugged Integration; Google CS First; AI Basics; Cybersecurity, Ciphers, and Puzzles) to reinforce teacher understanding of CS concepts and support them to integrate CS Standards into the core curriculum.

Measures

The measures consisted of a CS content knowledge index, CS self-efficacy scale, CS implementation survey, PD perception survey, and PD

observations. Table 4 depicts the alignment between the research questions, measures, and analytic strategies.

Table 4
Content Knowledge Index

Research Question	Measure	Analytic Strategy
1. How did treatment teachers' CS content knowledge change as a result of participation in the PD and how did their CS content knowledge compare to that of control group teachers?	CS Content Knowledge Index	Paired sample *t*-tests Analysis of covariance (ANCOVA)
2. How did treatment teachers' self-efficacy in teaching CS change as a result of participation in the PD and how did their self-efficacy compare to that of control group teachers?	CS Self-efficacy Scale	Paired sample *t*-tests Analysis of covariance (ANCOVA)
3. How did treatment teachers implement each CS curriculum standard compare to that of control teachers?	CS Implementation Survey	Descriptive statistics Inductive thematic analysis (Braun & Clarke, 2006)
4. How did treatment teachers perceive their PD experience and what additional support did they request?	PD Perceptions Survey, PD observation field notes	Descriptive statistics Inductive thematic analysis (Braun & Clarke, 2006)

This measure consisted of five open-ended response items developed by the research team and was informed by the participants' state's CS Standards (Table 5, Appendix A). Beyond the four core CS Standards, the state's CS Standards also include "Cybersecurity" and "Networking and the Internet." While these topics were covered during the PD sessions, they were not specifically incorporated into the content knowledge index. The index was designed to prioritize foundational concepts that teachers are typically less familiar with and that require targeted emphasis.

Additionally, we intended to ensure that this measure remained concise and manageable. Support for face and content validity was established through review by a panel with expertise in CS education, elementary education, and program evaluation. Their feedback on the items was incorporated into the final version of the instrument used in the study. The instrument was also pilot tested with a group of teachers (n = 30) who completed the PD prior to the RCT.

Table 5
Alignment Between State CS Standards and CS Knowledge Index Items

CS Standards	Expected Student Outcomes	Content Knowledge Index Item
Impacts of Computing	**CSF.23** The student will evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.	What is computer science?
Algorithms and Programming	**CSF.21**. The student will create programs demonstrating an understanding that program development is an ongoing process that requires adjusting and debugging along the way.	Describe what a computer programmer does. What is an algorithm?
Computing Systems	**CSF.1** The student will a. compare the structures, functions, and interactions between application software, system software, and hardware; and b. explore the relationship between hardware and software using the Internet of Things.	What makes a device a computer?
Data and Analysis	**CSF.9** The student will evaluate the tradeoffs in how data elements are organized and where data is stored.	In what ways is the term "variable" used differently in computer science than in math and science?

Self-Efficacy Scale

This measure consisted of nine Likert-scale items adapted from the Teachers' Self-efficacy in Computational Thinking instrument (Bean et al., 2015). Modifications that were pilot-tested included using a six-point scale instead of a five-point scale and replacing Items 9 and 10, which relate to the Common Core and *Next Generation Science Standards* (n.d.), with a single item about the state CS Standards (Appendix B). Cronbach's α for the revised instrument was calculated and determined to be .92 at the pretest and .92 at the posttest, indicating good reliability.

CS Implementation Survey Items

This 8-item survey (Appendix C) was administered at the end of the first semester and at the end of the school year to measure the frequency with which participants self-reported implementing CS-integrated instruction. Teachers were asked to indicate whether they had implemented CS-integrated lessons and the number of lessons taught that included learning objectives related to each CS Standard. If they indicated they did not implement any CS-related lessons, they were prompted to explain why.

PD Perception Survey Items

Likert scale items (Appendix D) administered post summer institute and at the year-end of PD elicited participant perceptions of their experience (12 items) and areas where they need additional PD support (14 items). Two additional open-ended items elicited teacher perceptions about useful PD components and additional PD support needed.

PD Observation Field Notes

PD was recorded, and observation fieldnotes were taken to identify the nature of teacher/teacher and teachers/facilitator interactions, signs of engagement, fatigue, understanding, discontent, questions among participants, implementation as planned (e.g., administrative, structural issues), and the nature of instruction.

Data Analysis

For Likert scale items (e.g., self-efficacy scale, PD perception items), the frequency of teacher endorsement for each item and descriptive statistics (*M*, *SD*) were calculated. Teacher pre, post, and year-end CS content knowledge responses were analyzed using systematic data analysis (Miles & Huberman, 1994) with a scoring rubric validated by the expert panel (Appendix A). Each item was scored from 0 to 3 (0 = *I don't know*, 1 = *did not meet expectations*, 2 = *partially met expectations*, 3 = *met expectations*) following the rubric. Five items were included in the CS content knowledge index (total scores ranged from 0 to 15).

Each response was rated by at least two raters to ensure interrater reliability of the scores. The raters scored 30% of the dataset independently; the initial scores were then compared and agreed upon through discussion. Cohen's k = .75 for the 30% subset, indicated strong interrater reliability (Richard & Koch, 1977). The raters then proceeded to independently score the remaining 70% of the data, ensuring consistency through peer debriefing to address any discrepancies that arose.

Paired sample *t*-tests were conducted to compare changes in treatment teacher PRE(i.e., before PD starts) to POST (i.e., immediately after the summer institute), and PRE to YEAR-END(i.e., at the end of the following school year) mean self-efficacy and content knowledge scores. The data satisfied the assumptions of paired sample *t*-tests as (a) the dependent variables (i.e., pre-CS knowledge score, post-CS knowledge score, year-end CS knowledge score, pre-self-efficacy scale, post-self-efficacy scale, year-end self-efficacy scale) were continuous; (b) the subjects were independent of one another; (c) each pre, post, and year-end measurement originated from the same subject; (d) results from the Shapiro-Wilk test and analysis of Q-Q plots support the assumption that the measured differences were normally distributed. Univariate analysis of covariance (ANCOVA), in which pre-PD scores served as a covariate, was used to compare treatment and control group year-end outcomes in CS content knowledge and self-efficacy.

Inductive thematic analysis, as described by Braun and Clarke (2006), was used to analyze open-ended survey responses (e.g., perceptions of the useful PD components and additional support needed) and PD observations. This approach involved "identifying, analyzing, and reporting patterns within data" (p.79). The first author analyzed the open-ended survey responses and identified initial categories within the data. Through an iterative analysis process and peer debriefing with the second author, we expanded and refined the coding categories to construct themes aligned with the research questions. In the results that follow, teacher responses are identified using a teacher ID (TID) number to maintain confidentiality.

Results

RQ1

Treatment teacher CS content knowledge did not change as a result of PD participation. Paired-sample *t*-tests indicated no significant change in treatment teacher CS content knowledge scores from pre to post, post to year-end, and pre to year-end (for all, $p > .05$; Table 6). Control group CS content knowledge scores declined significantly from pre to year-end ($p = .002$).

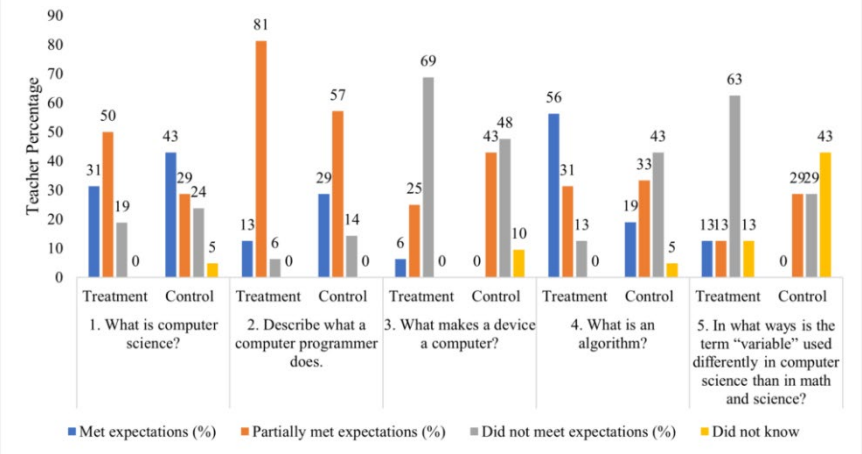
Table 6
Pre, Post, and Year-End CS Content Knowledge Scores

Group	Pre Content Knowledge Score *M* (*SD*)	Post Content Knowledge Score *M* (*SD*)	Year-End Content Knowledge Score *M* (*SD*)
Treatment (n = 16)	9.2 (1.5)	9.3 (1.4)	9.2 (2.0)
Control (n = 21)	9.8 (2.2)	Not administered	8.0 (2.4)

ANCOVA results indicated no statistically significant difference between the treatment and control group year-end CS content knowledge scores after controlling for pre scores, $F(1, 34) = 3.51$, $p > .05$. PD participation (i.e., condition) explained approximately 10.7% of the variance in year-end CS knowledge.

The year-end percentage of treatment and control group teachers whose responses met, partially met, or did not meet expectations varied by concept (Figure 3). A greater proportion of treatment than control teachers met or partially met expectations in their understanding of what CS is (81% treatment vs. 72% control), what a programmer does (94% treatment vs. 86% control), and what an algorithm is (87% treatment vs. 52% control). For the item assessing understanding of what makes a device a computer, 31% of treatment group teachers and 43% of control teachers met or partially met expectations. For the item assessing understanding of a variable, 26% of treatment teachers and 29% of control teachers met or partially met expectations.

Figure 3
Teacher Year-End CS Content Knowledge Outcomes



Item 1. What Is CS?

At the end of the year, 31% of treatment teachers met expectations for the item, "What is CS?" These teachers accurately described CS as the study of computers and/or computing technology and identified some essential components (e.g., algorithms and programming, computational systems, networks, etc.) and applications (e.g., solving real-life problems, improving the efficiency of work, etc.) of CS. For example, one teacher described CS as "Study of algorithmic language to make the computer design a program that responds to our needs. Computer knowledge includes hardware, software, principles, and their application" (TID: 2033).

About 50% of treatment teachers partially met expectations. These teachers were able to connect CS to the study of computers but did not elaborate on what it involves or overemphasize the role of programming in this field. For example, one teacher defined CS as "The study of computers and programming" (TID: 2015). Another teacher similarly wrote, "It's the study of computers as it relates to programming" (TID: 2065). The remaining 19% of treatment teachers did not meet expectations in their understanding of CS. They conflated CS with using technology. For example, these teachers described CS simply as "using computers" or "using technology." In the control group, 24% of teachers did not meet expectations in their understanding of CS, and 6% indicated they did not know what CS is about.

Item 2. Describe What a Computer Programmer Does

Following the PD, 13% of treatment teachers met expectations; they were able to identify that a computer programmer writes and tests code to ensure computer applications and software programs function properly (e.g., "Computer programmer writes and modifies computer codes that design software and applications and make them function properly," TID 2033). However, a majority of treatment teachers (81%) only mentioned that a programmer writes code without including the crucial aspects of

testing and modifying codes (e.g., "A person who writes programs," ID 2060). About 6% of the treatment teachers were not able to specify what a computer programmer does compared to 14% of the control teachers.

Item 3. What Makes a Device A Computer?

Following the 1-year PD, most treatment teachers did not meet expectations in their responses to this item. Only one teacher was able to accurately identify all four key components of a computer: input, output, processor, and memory. Approximately 25% of treatment teachers were able to identify at least two of these key components, while around 69% could only identify one or none. In comparison, 48% of control teachers could only identify one or none, and 10% of control teachers indicated that they did not know what makes a device a computer.

Item 4. What Is an Algorithm?

Approximately 56% of treatment teachers met expectations in their understanding of an algorithm. Their responses indicated that they knew an algorithm is step-by-step instructions that produce a result (e.g., "Step-by-step instructions that allow a computer to carry out a series of directions or actions"; TID: 2021). About 31% of treatment teachers partially met expectations (e.g., describing an algorithm as a mathematical formula without elaboration or not specifying that an algorithm is sequential). For example, one treatment teacher defined it as "A set of rules used to create a computer program" (TID: 2033). About 13% of treatment teachers did not meet expectations, compared with 43% of control teachers; they were not able to identify the stepwise nature of an algorithm. For example, one of these treatment teachers described an algorithm as "the pattern/code the computer follows" (TID: 2004).

Item 5. How Is a Variable Used Differently in CS Than in Math and Science?

Only 13% of treatment teachers were able to distinguish correctly how a variable is used differently in CS and math and science. For example, one responded, "A variable in CS is a location where information is stored, in math a variable is part of an equation and is relative to the equation's data" (TID: 2023). Another 13% of treatment teachers partially met expectations. They were able to interpret what a variable means in CS (e.g., "changes based on the information in the program," and "represents a storage location"), but these teachers did not describe how it is used differently in math or science.

Most of the treatment teachers (76%) did not meet expectations as they could not correctly specify what variable means in different subjects (e.g., describing the variable as "an anomaly") or indicated that they did not know what variable means. In the control group, similarly, most of the teachers (72%) did not meet expectations or expressed uncertainty regarding the meaning of a variable.

RQ2

Results of paired-sample *t*-tests of pre, post, and year-end self-efficacy scores indicated that treatment teacher CS self-efficacy significantly improved following the summer institute (pre to post, $p < .001$, Cohen's $d = 1.3$) and that this improvement was retained at the end of the year (pre to year-end, $p < .001$, Cohen's $d = 1.2$; Table 7). These differences were both statistically and practically meaningful, with a large effect size (defined in Fan, 2001).

Table 7
Participants' Pre, Post, and Year-End CS Self-Efficacy Scores

Group	Pre-Self-Efficacy *M* (*SD*)	Post-Self-Efficacy *M* (*SD*)	Year-End Self-Efficacy *M* (*SD*)
Treatment ($n = 16$)	32.6 (8.0)	42.1 (5.0)	41.0 (6.2)
Control ($n = 21$)	34.1 (8.1)	Not administered	33.2 (10.4)

Results of ANCOVA indicate a significant difference between treatment and control teacher year-end self-efficacy scores after controlling for pre-self-efficacy score, $F(1, 34) = 20.36$, $p < .001$, Cohen's $d = 2.5$. Condition (i.e., participation in the PD) explained approximately 60.5% of the variance in year-end self-efficacy. This represents a statistically and practically meaningful difference between treatment and control group year-end self-efficacy (Fan, 2001).

RQ3

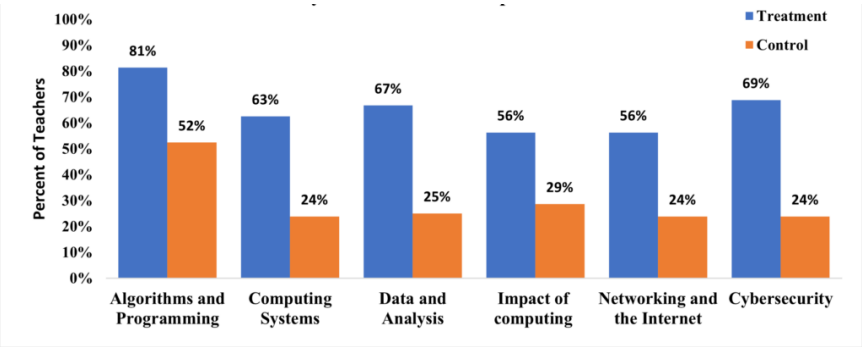
During the academic year, 88% of treatment teachers reported implementing lessons that integrated CS curriculum standards, compared to 62% of control teachers. A greater proportion of treatment than control teachers reported teaching lessons related to each of the six CS Standards (Figure 4).

Teachers who indicated they implemented CS-integrated lessons were asked to describe the learning activities that they perceived as effective in engaging students. Teachers' responses indicated that students exhibited high engagement in hands-on exploration with computing tools ($n = 14$), unplugged activities ($n = 14$), and programming ($n = 11$). In terms of exploring with computing tools, one teacher commented, "We used Ozobots and Spheros to teach algorithms in order to retell a story with simple to numerous details. Children enjoyed the retell and were able to describe the steps and debug the issues" (TID: 2021; treatment, midyear survey). Regarding unplugged activities, one teacher described:

> Creating anchor charts modeling how to complete certain tasks and troubleshoot different tasks has been helpful and effective. Students enjoyed sequencing activities as it connected to writing by making it very hands on by writing algorithms, or steps, of how

to do things like eat an oreo while actually having a chance to eat oreos. (TID: 2029; treatment, year-end survey)

Figure 4
Teacher Implementation of CS Curriculum Standards Over the School Year



Some teachers indicated that students were highly engaged in programming, making comments such as "Students enjoyed when they were given an opportunity to code using algorithms or block coding in order to show what they had learned in a lesson for a curriculum such a science reading math" (TID: 2021, treatment, year-end survey).

Teachers who indicated they did not teach any lessons that explicitly targeted CS Standards were asked to explain why. Responses suggested that competing priorities (n = 8) and time constraints (n = 8) were the major barriers to implementing CS-integrated lessons. In terms of competing priorities, one treatment teacher commented, "I work more with small group EL elementary students. My lesson is more targeted towards reading, math, and science" (TID: 2033, treatment, year-end survey). Another one similarly commented, "As a Special Education teacher, I have been focused on social skills, behavior management, and reading skills" (TID: 2015, treatment, midyear survey). In terms of time constraints, teachers commented that they did not have time to explicitly integrate CS into their instruction. For example, one treatment teacher wrote "no time outside curriculum" (TID: 2019, treatment, year-end survey).

Control teachers mentioned a lack of knowledge of integrating CS (n = 7) and low confidence in teaching CS (n = 2). Many control teachers indicated that they had never participated in CS professional learning and, thus, did not have the necessary knowledge and expertise to integrate CS. For example, one teacher wrote, "I am unfamiliar with computer science standards and how to teach anything related to computer science" (TID: 2072, control, year-end survey).

RQ4

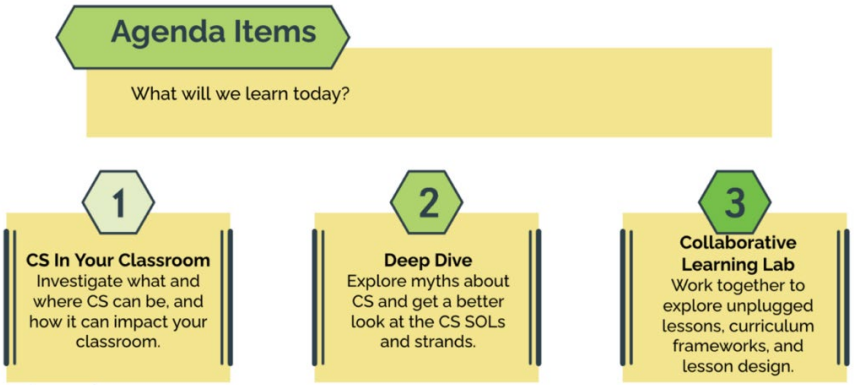
Overall, treatment teachers indicated positive perceptions of their PD experience, and they identified the content-focused nature of the PD,

opportunities for active learning and collaboration and expert support to be effective.

Content-Focused

Eighty-seven percent of teachers found the learning objectives to be clear, and 93% believed that the sample lesson plans provided aligned with state standards. Each day facilitators explicitly introduced the CS theme and goals to make sure teachers had a clear understanding of the content focus and learning objectives (e.g., Figure 5; PD observations). The facilitators also encouraged teachers to share ideas about CS, CS instruction, and CS standards to activate teachers' initial content knowledge and then built on their responses to help them develop a more formal understanding of CS based on the CS standards (Day 1 Observation).

Figure 5
Summer Institute Day 1 Content Focus PD Slide



Note. PowerPoint slide artifact, Observation Day 1.

Teachers reported learning about CS standards and CS integration strategies to be helpful. For example, one teacher commented, "Teaching computer science standards doesn't always have to involve technology. You can use unplugged activities and lessons and be equally as effective." Another teacher commented, "Coding and sol [CS Standards of Learning] objectives can be used together to create interesting lessons" (TID: 2025, treatment, year-end survey). Another teacher similarly commented that the most useful thing they learned in PD is "becoming familiar with the CS standards and how to put them into simpler terms" (TID: 2070, treatment, year-end survey).

Active Learning

Of teachers, 83% were satisfied with the learning activities they engaged in during the PD. PD facilitators supported teachers in CS learning through multiple ways, including reflective discussions, collaboration, and unplugged activities (PD observations). For example, facilitators encouraged teachers to discuss and share ideas with peers using a Jamboard activity when introducing CS concepts and CS standards (Day 1

Observation). During the CS coaching session, facilitators engaged teachers in a role-play activity where they practiced how to address content-based and interpersonal challenges in CS teaching (Day 3 Observation). Post-summer institute survey responses indicated that engaging in unplugged activities made them feel less overwhelmed about integrating CS. One teacher said,

> One thing I found most useful and not intimidating were the unplugged ideas. I don't have to learn new technologies for them, and we explored multiple ways we are already using some of these ideas or how we can easily implement them. (ID: 2024, treatment, postsurvey).

Collaboration

Of teachers, 74% felt a part of a learning community during PD, and 93% indicated that facilitators fostered an atmosphere of trust and communication. Teachers collaborated in several ways during PD activities. Each day, teachers engaged in a 2-hour "collaborative learning lab" session where they worked with peers to brainstorm lesson plan ideas, develop CS-integrated lesson plans, and create a CS coaching plan (Day 1-5 Observation).

Expert Support

About 93% of teachers thought that the facilitators met their needs as teacher-learners, and a similar proportion (93%) agreed that the PD facilitators provided timely knowledge and instructional support. Expert support was provided in multiple ways, including offering explicit instructions on CS concepts and standards (Day 1 Observation), guiding teacher discussions and reflections (Day 1-5 Observation), and providing scaffolding and feedback on CS lesson plans and coaching plans (Day 2-5 Observation). Survey responses also highlighted the value of the instructional resources provided. For example, one teacher commented, "I learned that there are many resources 'out there' that I was unaware of to help me integrate the CS standards into my lessons" (ID: 2040, postsurvey). Another teacher similarly indicated that the most useful component of PD was "the different lesson plans that were provided throughout the course" (ID: 2047, postsurvey).

Additional Support

Teachers indicated areas for which they desired additional support for learning to integrate CS. These included continued PD learning opportunities focused on programming, integration of CS standards, and CS curriculum development; more resources for CS instruction; more opportunities for collaboration; and additional time to learn and implement CS. Regarding continued PD learning opportunities, teachers mentioned more "training on coding" and "have in-person support and opportunities." For additional CS resources, teachers commented that "continued reminders and simple lesson plan ideas" and "ideas for ways to incorporate CS into daily lessons" would be helpful. About the collaboration, one teacher commented that "working with my district's

faculty members who took the PD with me during the summertime to incorporate the CS strands across the K-5 SOL strands" would be useful. A few teachers mentioned that "I need more time" and "additional time in my schedule" to learn how to integrate CS.

Discussion

This RCT explored the effectiveness of a year-long PD program aligned with SLT on elementary teacher CS content knowledge, self-efficacy, and implementation of CS instruction. This study contributes to the literature in several ways. First, unique to previous studies, the RCT design allowed us to measure whether and to what extent teacher changes in CS content knowledge, self-efficacy, and implementation could be attributed to PD participation. Second, participants identified several components of the PD they perceived as useful and areas for additional support in CS instruction. These findings may inform the future development of CS PD that is specifically designed to meet elementary teachers' unique needs.

CS Content Knowledge

Contrary to our hypothesis, the present study found no significant change in treatment teacher CS content knowledge scores from pre- to post-PD and no significant difference between treatment and control teacher CS content knowledge scores. The proportion of treatment teachers who met or partially met expectations was greater than that of control teachers for understanding what CS is, what a programmer does, and what an algorithm is. However, the proportion of control teachers who met or partially met expectations was higher than that of treatment teachers for understanding what makes a device a computer and what a variable is.

These mixed results contradict previous research, which reported improvements in teacher CS content knowledge after PD (e.g., Mouza et al., 2016; Yadav et al., 2018). These conflicting outcomes may be due to differences in the PD approaches or the knowledge measure, as described next.

First, it takes prolonged support to evoke changes in teacher CS understanding (e.g., Liu, 2022; Ketelhut et al., 2019). Although the PD program in the present study extended support to teachers by including academic-year PD support, explicit CS content instruction was provided most intensively during the five-day summer institute, and it was only one of the foci of the summer institute (others were developing skills related to coaching, understanding equity in CS instruction, and lesson planning for CS integration). As a result, teachers might not have developed a solid understanding of the targeted CS concepts.

Additionally, teacher reflection and enactment of new knowledge, as well as knowledge exchange with peers, is crucial to fostering knowledge acquisition (Clarke & Hollingsworth, 2002; & Zhao & Fan, 2022). However, teachers in the present study struggled with competing priorities and time constraints during the school year. Thus, they may not have been able to fully engage in these activities to reinforce their CS knowledge.

Second, the existing studies that reported improvement in teacher CS knowledge often relied on self-reported survey responses or qualitative feedback from teachers (e.g., Mouza et al., 2016; Yadav et al., 2018). These methods may help identify subtle improvements in teacher knowledge but may not be an accurate measure of changes in specific CS concepts. In contrast, the present study used a criterion-based measure to evaluate changes in teacher understanding of specific CS concepts, providing a more objective and targeted way to evaluate teacher knowledge change after PD.

Teacher Self-Efficacy for Teaching CS

Consistent with our hypothesis, treatment teacher self-efficacy improved significantly from pretreatment to year-end, and their year-end self-efficacy was greater than their peers in the control group after controlling for pre-self-efficacy. This finding supports prior results that PD participation can build teacher CS self-efficacy and confidence (e.g., Ravitz et al., 2017). Developing teacher self-efficacy in CS instruction is particularly important because CS is commonly considered challenging and requires much effort (Zhou et al., 2020). Teachers with high self-efficacy are more likely to be persistent in the face of challenges (Bandura, 1994), and they can potentially influence students' motivation and shape their self-efficacy when learning a challenging subject (Ross, 1992).

The research emphasizes that mastery experience (i.e., successful experiences of accomplishing specific tasks) and vicarious experience (i.e., the experience of observing others perform a task successfully) are significant sources of self-efficacy (Klassen et al., 2010). In this study, PD facilitators explicitly modeled CS instruction and provided opportunities for teachers to engage in hands-on design and practice CS lessons. These activities may have provided teachers with sufficient mastery and vicarious experience to increase their self-efficacy. Additionally, teachers were provided CS instructional resources to use, which may also have helped build their self-efficacy. These results extend those of prior studies (e.g., Liu, 2022; Ni et al., 2021) by highlighting the particular PD components that may empower elementary teachers to feel more confident in their ability to teach CS.

Teacher Implementation of CS Instruction

Teacher implementation of CS instruction directly influences student access to CS learning (Rich et al., 2021). More than half of treatment teachers (88%) and control teachers (62%) reported implementing CS lessons throughout the academic year. This result may be due to the state mandate to integrate CS into instruction. A greater proportion of treatment than control teachers reported teaching lessons related to each of the six CS standards, which may relate to the instructional support teachers received during PD.

Treatment teachers expressed the value of gaining knowledge about CS standards and instructional strategies to facilitate CS integration throughout the PD, which may have increased their willingness and ability to integrate CS standards into instruction. These findings support prior

work indicating that PD participation helps teachers feel better prepared for CS instruction (Milliken et al., 2019; Ozturk et al., 2018).

The results of the present study are consistent with those of previous studies indicating that teachers face various obstacles when implementing CS instruction in their classrooms (Israel et al., 2022; Bers et al., 2022; Ketelhut et al., 2019). For instance, similar to Israel et al.'s finding that competing priorities, limited time, insufficient content and pedagogical knowledge, and low teacher buy-in are common challenges faced by elementary teachers in CS integration, teachers in our study also identified challenges related to competing priorities, time constraints, limited knowledge and expertise, and low confidence. Teachers indicated that they viewed CS as a separate skill to be added to instruction rather than something that could be integrated into existing curricula. Although PD facilitators provided explicit instruction on CS concepts and Standards, they may not have adequately emphasized the connection between CS concepts and other content taught. For elementary teachers, sustained CS exposure within their instructional contexts and emphasizing the connection between CS concepts and the concepts taught in their current lessons may be important in supporting teacher CS integration.

Our findings also support Bers et al.'s (2022) claim that managing competing priorities and instructional time is a challenge for elementary school teachers implementing CS instruction. In addition to these common challenges, it is important to recognize and anticipate other obstacles elementary teachers may have, such as addressing students' diverse needs in CS learning (Liu, 2022; Ketelhut et al, 2019). Recognizing and acknowledging the challenges elementary teachers face is critical to providing effective support as they strive to integrate CS instruction.

Implications for CS PD Design and Implementation

The present study identified several essential PD components that are aligned with SLT and prior literature (e.g., Darling-Hammond et al., 2017), including the need for ongoing PD support, access to additional CS resources like sample lesson plans, and opportunities for collaborative learning. It is crucial to include these features in future CS PD designs.

According to SLT, effective learning occurs in a context that mirrors the authentic context where the learned knowledge and skills are used (Orgill & Bodner, 2007). Elementary teachers are expected to integrate CS into other subjects. Thus, it is essential that PD offers explicit support to help teachers establish the connection between CS and their existing subject content (Reding & Dorn, 2017). By aligning CS concepts and practices within elementary subjects, PD can help teachers develop a more nuanced understanding of how to meaningfully incorporate CS into their curricula (Liu, 2022). The research also emphasizes opportunities for participation in model lessons, teaching practice, collaboration, and coaching reflection as essential factors for teacher knowledge development in PD (Ingvarson et al., 2005).

To develop and maintain teacher CS self-efficacy and support classroom implementation, it is crucial that PD promotes teachers' mastery and vicarious experiences. Providing opportunities for teachers to create their

own CS instructional artifacts, practice teaching strategies through microteaching sessions, and participate in peer and coteaching can foster mastery experiences in CS instruction (Luo & Liu., 2022; Rich et al., 2017). Explicitly modeling CS pedagogies may build teachers' vicarious experience (Reding & Dorn, 2017). A valuable approach to introducing CS pedagogies to teachers is the Teacher-Learner-Observer model, in which teachers play the roles of the different stakeholders in the classroom (e.g., learners, observers, etc.) to build CS content and pedagogical knowledge (Nakajima & Goode, 2019; Zhou et al., 2020).

To address the challenges teachers face implementing CS, it is important to build a school culture of CS integration (Israel et al., 2022). This may include addressing time constraints through common planning periods that teachers can use to develop lesson plans, share instructional strategies, and reflect on their teaching experience (Israel et al., 2015). To address teacher concerns about competing priorities, PD should focus on curricular alignment, helping teachers recognize how CS intersects with other content and how students can use programming as an expression and communication tool (Bers et al., 2022). The more elementary teachers can envision how CS aligns with other content instruction, the more they may feel comfortable including it in instruction (Liu, 2022; Bers et al., 2022).

Taken together, our findings and prior literature related to CS PD suggest the following actionable strategies for future CS PD to support the development of elementary teacher CS content knowledge and self-efficacy and address challenges to CS implementation (Table 8).

Table 8
Suggestion for Future CS PD for Elementary Teachers

Goals	Suggestion for Future PD
Developing Content Knowledge	- Emphasize the **alignment** between CS standards and elementary curricula (Bers et al., 2022). - Provide **follow-up support** post formal PD sessions (Luo & Liu, 2022). - Structure CS **content focus** following the framework of technological pedagogical and content knowledge (TPACK; Sun et al., 2023).
Developing self-efficacy and pedagogical practice	- Engage teachers in **model lessons** - the activities they are designing for students. - Create built-in time for teachers to **collaborate** on lesson planning and encourage **peer teaching.** - Provide **one-on-one coaching** in the context of teachers' classrooms (Campbell & Malkus, 2011). - Provide **constructive feedback** on teachers' lesson plans and classroom implementation. - Encourage teachers to **reflect** on new knowledge and pedagogies on regular basis to reinforce their CS understanding (Liu, 2022).
Addressing teacher challenges (e.g., time constraints, resource access)	- Provide sample lesson plans and **resources**. - Foster school-wide collaborative PD (Israel et al., 2021). - Build teacher **professional learning communities** (PLCs) within and between PD learning sessions (Ni et al., 2021).

Limitations and Future Research

As with any study, potential threats to the validity of the findings should be acknowledged. First, due to participant attrition, the study sample is relatively small (n = 37), which may reduce the level of significance of the findings (Fowler & Lapp, 2019). Despite several strategies designed to reduce teacher attrition (e.g., conducting regular follow-ups with participants, providing incentives, etc.), attrition was unavoidable. The existence of attrition bias should be noted; treatment teachers may be more likely to drop out due to the additional demand for their time and effort (Spieth et al., 2016).

Second, insufficient adherence to the assigned treatment by the participants (implementation fidelity) may have impacted the results (Spieth et al., 2016). In addition, the study was conducted during the 2021-2022 school year, when schools were still operating in a remote or hybrid environment, and this may have affected teachers' ability to participate in school year PD, implement lessons, or otherwise remain engaged. For example, treatment teachers may not have fully participated in the academic-year PD, which may diminish the effectiveness of the PD program resulting in insignificant findings.

Third, measures that rely on participant self-report may have introduced response bias (Rosenman et al., 2011). Future research should consider incorporating additional data sources (e.g., interviews, teacher reflection journals, instructional artifacts, etc.) to triangulate the findings reported here.

Beyond understanding whether changes in teacher CS content knowledge, self-efficacy, and CS implementation occurred, future research should examine how these changes occur. Theoretical frameworks such as the Interconnected Model of Professional Growth provide a lens to understand domains of teacher change and change pathways in professional learning (e.g., Clark & Hollingsworth, 2002) and may be useful in guiding future investigations of what and how teacher changes are evoked by different factors in CS PD. Exploring the quality of teacher implementation of CS standards (e.g., successful alignment, misalignment, students' engagement, etc.) through classroom observations and artifact analysis is also important. Finally, to comprehensively evaluate PD efficacy, future studies should include student data to evaluate whether teacher learning translates into student learning (Darling-Hammond et al., 2017).

Conclusion

Overall, the results of this RCT provide evidence that elementary teachers need continual, sustained, and content-relevant CS PD support, especially ongoing support during the academic year, to build and reinforce their understanding of CS concepts. The PD program in the present study effectively improved elementary teachers' CS self-efficacy and motivated teachers to implement CS Standards in instruction. Further research is underway to examine the impact of the PD on additional cohorts.

Our results suggest that to effectively support elementary teachers in CS integration, it is essential to explicitly support teachers in envisioning the connections between CS and their existing content instruction and provide ready-to-use resources to facilitate CS implementation. Teachers also need ample time for reflection, practice, and peer collaboration to reinforce their CS understanding. Ultimately, developing elementary teachers' CS content and pedagogical knowledge and supporting their classroom CS implementation is the first step toward broadening elementary student participation in CS education.

References

Armoni, M., & Gal-Ezer, J. (2014). Early computing education: Why? what? when? who? *ACM Inroads*, *5*(4), 5459.

Bandura, A. (1994). Self-efficacy. In V.S. Ramachaudran (Ed.), *Encyclopedia of human behavior* (Vol.4, pp.71–81). Academic Press.

Bandura, A. (2006). Guide for constructing self-efficacy scales. In F. Pajares & T. Urdan (Eds.), *Self-efficacy beliefs of adolescents* (Vol. 5, pp. 307–337). Information Age Publishing.

Barr, D. J., Boulay, B., Selman, R. L., McCormick, R., Lowenstein, E., Gamse, B., Fine, M., & Leonard, M. B. (2015). A randomized controlled trial of professional development for interdisciplinary civic education: Impacts on humanities teachers and their students. *Teachers College Record*, *117*(2), 1–52.

Basu, S., Rutstein, D., & Tate, C. (2021). *Building teacher capacity in K–12 computer science by promoting formative assessment literacy* [White paper]. National Comprehensive Center. https://files.eric.ed.gov/fulltext/ED615714.pdf

Bean, N. H., Weese, J. L., Feldhausen, R., & Bell, R. S. (2015). Starting from scratch: Developing a pre-service teacher training program in computational thinking. *Proceedings of Frontiers in Education Conference* (pp. 1–8). IEEE. https://doi.org/10.1109/fie.2015.7344237

Bers, M. U., Govind, M., & Relkin, E. (2022). Coding as another language. *Computational Thinking in PreK-5*, 30–38. https://doi.org/10.1145/3507951.3519285

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101. https://doi.org/10.1191/1478088706qp0630a

Buss, A., & Gamboa, R. (2017). Teacher transformations in developing computational thinking: Gaming and robotics use in after-school settings. In P. Rich & C. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 189–203). Springer, Cham. https://doi.org/10.1007/978-3-319-52691-1_12

Campbell, P. F., & Malkus, N. N. (2011). The impact of elementary mathematics coaches on student achievement. *The Elementary School Journal*, *111*(3), 430–454. https://doi.org/10.1086/657654

Century, J., Ferris, K. A., & Zuo, H. (2020). Finding time for computer science in the elementary school day: A quasi-experimental study of a transdisciplinary problem-based learning approach. *International Journal of STEM Education*, *7*(1). https://doi.org/10.1186/s40594-020-00218-3

Clarke, D., & Hollingsworth, H. (2002). Elaborating a model of teacher professional growth. *Teaching and Teacher Education*, *18*(8), 947–967. https://doi.org/10.1016/s0742-051x(02)00053-7

Computer Science Teachers Association. (2017). CSTA K-12 computer science standards, Revised 2017, http://www.csteachers.org/standards

Computer Science Teachers Association (2020). *Standards for Computer Science Teachers*. https://csteachers.org/teacherstandards.

Code.org, CSTA, & ECEP Alliance (2022). *2022 State of computer science education: Understanding our national imperative.* https://code.org/assets/advocacy/stateofcs/2022_state_of_cs.pdf

Code.org, CSTA, ECEP Alliance. (2023). *2023 State of computer science education*. https://code.org/assets/advocacy/stateofcs/2024_state_of_cs.pdf

Darling-Hammond, L., Hyler, M. E., & Gardner, M. (2017*). Effective teacher professional development*. Learning Policy Institute. https://learningpolicyinstitute.org/product/teacher-prof-dev

Escriva-Boulley, G., Tessier, D., Ntoumanis, N., & Sarrazin, P. (2018). Need-supportive professional development in elementary school physical education: Effects of a cluster-randomized control trial on teachers' motivating style and student physical activity. *Sport, Exercise, and Performance Psychology*, *7*(2), 218.

Fan, X. (2001). Statistical significance and effect size in education research: Two sides of a coin. *The Journal of Educational Research*, 94(5), 275–282. https://doi.org/10.1080/00220670109598763

Febrian, A., Lawanto, O., Peterson-Rucker, K., Melvin, A., & Guymon, S. E. (2018, June), *Does everyone use computational thinking? A case study of art and computer science majors* [Paper presentation]. American Society for Engineering Education Annual Conference & Exposition, Salt Lake City, Utah. https://peer.asee.org/30344

Fowler, S. B., & Lapp, V. (2019). Sample size in quantitative research: Sample size will affect the significance of your research. *American Nurse Today*, *14*(5), 61-63.

Gane, B. D., Israel, M., Elagha, N., Yan, W., Luo, F., & Pellegrino, J. W. (2021). Design and validation of learning trajectory-based assessments for

computational thinking in upper elementary grades. *Computer Science Education*, *31*(2), 141–168. https://doi.org/10.1080/08993408.2021.1874221

Google Inc. & Gallup Inc. (2016). Diversity gaps in computer science: Exploring the underrepresentation of girls, Blacks and Hispanics. http://goo.gl/PG34aH

Hariton, E., & Locascio, J. J. (2018). Randomised controlled trials – the gold standard for effectiveness research. *BJOG: An International Journal of Obstetrics & Gynaecology*, *125*(13), 1716. https://doi.org/10.1111/1471-0528.15199

Hestness, E., Ketelhut, D. J., McGinnis, J. R., & Plane, J. (2018). Professional knowledge building within an elementary teacher professional development experience on computational thinking in science education. *Journal of Technology and Teacher Education*, *26*(3), 411–435.

Hubbard, A., & D'Silva, K. (2018). Professional learning in the midst of teaching computer science. *Proceedings of the 2018 ACM Conference on International Computing Education Research* (pp. 86–94). https://doi.org/10.1111/1471-0528.15199

Ingvarson, L., Meiers, M., & Beavis, A. (2005). Factors affecting the impact of professional development programs on teachers' knowledge, practice, student outcomes & efficacy. *Education Policy Analysis Archives*, *13*(10). https://doi.org/10.14507/epaa.v13n10.2005

Israel, M., Liu, R., Yan, W., Sherwood, H., Martin, W., Fancsali, C., Rivera-Cash, E., & Adair, A. (2022). Understanding barriers to school-wide computational thinking integration at the elementary grades: Lessons from three schools. In A. Ottenbreit-Leftwich, & A. Yadav (Eds.), *Computational thinking in prek-5: Empirical evidence for integration and future directions* (pp. 64–71). Association for Computing Machinery. https://doi.org/10.1145/3507951

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, *82*, 263–279. https://doi.org/10.1016/j.compedu.2014.11.022

Jocius, R., Joshi, D., Dong, Y., Robinson, R., Cateté, V., Barnes, T., Albert, J., Andrews, A., & Lytle, N. (2020). Code, connect, create: The 3c professional development model to support computational thinking infusion. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. https://doi.org/10.1145/3328778.3366797

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

K–12 Computer Science Framework. (2016). https://k12cs.org/

Kale, U., Akcaoglu, M., Cullen, T., Goh, D., Devine, L., Calvert, N., & Grise, K. (2018). Computational what? relating computational thinking to teaching. *TechTrends*, *62*(6), 574–584. https://doi.org/10.1007/s11528-018-0290-9

Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2019). Teacher change following a professional development experience in integrating computational thinking into elementary science. *Journal of Science Education and Technology*, *29*(1), 174–188. https://doi.org/10.1007/s10956-019-09798-4

Klassen, R. M., Tze, V. M. C., Betts, S. M., & Gordon, K. A. (2010). Teacher efficacy research 1998–2009: Signs of progress or unfulfilled promise? *Educational Psychology Review*, *23*(1), 21–43. https://doi.org/10.1007/s10648-010-9141-8

Kwon, K., Ottenbreit-Leftwich, A. T., Brush, T. A., Jeon, M., & Yan, G. (2021). Integration of problem-based learning in elementary computer science education: Effects on computational thinking and attitudes. *Educational Technology Research and Development*, *69*, 2761–2787. https://doi.org/10.1007/s11423-021-10034-3

Liu, R. (2022). *Elementary school teachers' changes in computational thinking understanding and instruction following professional development : A qualitative multi-case study*. [Doctoral dissertation, University of Florida]. UF Digital Library. https://original-ufdc.uflib.ufl.edu/UFE0059045/00001

Luo, F., & Liu, R., (2022, April). *A systematic literature review of K-12 computer science professional development* [Paper presentation]. Annual Conference of the American Educational Research Association, San Diego, CA, USA.

Maeng, J. L., Whitworth, B. A., Bell, R. L., & Sterling, D. R. (2020). The effect of professional development on elementary science teachers' understanding, confidence, and classroom implementation of reform-based science instruction. *Science Education*, *104*(2), 326–353.

Mason, S. L., & Rich, P. J. (2019). Preparing elementary school teachers to teach computing, coding, and computational thinking. *Contemporary Issues in Technology and Teacher Education, 19*(4). https://citejournal.org/volume-19/issue-4-19/general/preparing-elementary-school-teachers-to-teach-computing-coding-and-computational-thinking.

McKay, S. (2017). *Five essential building blocks for a successful networked improvement community*. www.carnegiefoundation.org/blog/five-essential-building-blocks-for-asuccessful-networked-improvement-community/

McLellan, H. (1996). *Situated learning perspectives*. Educational Technology Publications.

Miles, M. B., & Huberman, M. A. (1994). *Qualitative data analysis: An expanded sourcebook* (2nd ed.). Sage Publications.

Milliken, A., Cody, C., Catete, V., & Barnes, T. (2019). Effective computer science teacher professional development. *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*. https://doi.org/10.1145/3304221.3319779

Mladenović, M., Boljat, I., & Žanko, Ž. (2017). Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level. *Education and Information Technologies*, *23*(4), 1483–1500. https://doi.org/10.1007/s10639-017-9673-3

Mouza, C., Pollock, L., Pusecker, K., Guidry, K., Yeh, C. Y., Atlas, J., & Harvey, T. (2016). Implementation and outcome of a three-pronged approach to professional development for CS principles. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. https://doi.org/10.1145/2839509.2844585

Nakajima, T. M., & Goode, J. (2019). Transformative learning for computer science teachers: Examining how educators learn e-textiles in professional development. *Teaching and Teacher Education*, *85*, 148–159. https://doi.org/10.1016/j.tate.2019.05.004

Ni, L., Bausch, G., & Benjamin, R. (2021). Computer science teacher professional development and professional learning communities: A review of the research literature. *Computer Science Education*, *33*(1), 29-60. https://doi.org/10.1080/08993408.2021.1993666

Orgill, M., & Bodner, G. M. (2007). *Theoretical frameworks for research in chemistry/science education*. Prentice Hall.

Ozturk, Z., Dooley, C. M., & Welch, M. (2018). Finding the hook: Computer science education in elementary contexts. *Journal of Research on Technology in Education*, *50*(2), 149–163. https://doi.org/10.1080/15391523.2018.1431573

Palfrey, J., & Gasser, U. (2008). Opening universities in a digital era. *New England Journal of Higher Education*, *23*(1), 22–24. http://files.eric.ed.gov/fulltext/EJ850701.pdf

Ravitz, J., Stephenson, C., Parker, K., & Blazevski, J. (2017). Early lessons from evaluation of computer science teacher professional development in Google's CS4HS program. *ACM Transactions on Computing Education*, *17*(4), 1–16. https://doi.org/10.1145/3077617

Reding, T. E., & Dorn, B. (2017). Understanding the "teacher experience" in primary and secondary CS professional development. *Proceedings of the 2017 ACM Conference on International Computing Education Research*. https://doi.org/10.1145/3105726.3106185

Rich, K. M., Yadav, A., & Schwarz, C. (2019). Computational thinking, mathematics, and science: Elementary teachers' perspectives on

integration. *The Journal of Technology and Teacher Education*, *27*(2), 165–205. https://eric.ed.gov/?id=EJ1224462

Rich, P. J., & Hodges, C. B. (2017). *Emerging research, practice, and policy on computational thinking*. Springer International Publishing.

Rich, P. J., Mason, S. L., & O'Leary, J. (2021). Measuring the effect of continuous professional development on elementary teachers' self-efficacy to teach coding and computational thinking. *Computers & Education*, *168*, 104196. https://doi.org/10.1016/j.compedu.2021.104196

Richard, L. J., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, *33*(1), 159–174. https://doi.org/10.2307/2529310

Rosenman, R., Tennekoon, V., & Hill, L. G. (2011). Measuring bias in self-reported data. *International Journal of Behavioural and Healthcare Research*, *2*(4), 320-332. doi: 10.1504/IJBHR.2011.043414

Ross, J. A. (1992). Teacher efficacy and the effects of coaching on student achievement. *Canadian Journal of Education / Revue Canadienne de L'éducation*, *17*(1), 51–65. https://doi.org/10.2307/1495395

Saad, A. (2020). Students' computational thinking skill through cooperative learning based on hands-on, inquiry-based, and student-centric learning approaches. *Universal Journal of Educational Research*, *8*(1), 290–296. https://doi.org/10.13189/ujer.2020.080135

Schoen, R. C., LaVenia, M., Chicken, E., Razzouk, R., Kisa, Z., & Boylan, M. (2019). Increasing secondary-level teachers' knowledge in statistics and probability: Results from a randomized controlled trial of a professional development program. *Cogent Education*, *6*(1). https://doi.org/10.1080/2331186X.2019.1613799

Shulman, L. P. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher*, *15*(2), 4–14. https://doi.org/10.3102/0013189x015002004

Spieth, P. M., Kubasch, A. S., Penzlin, A. I., Illigens, B. M., Barlinn, K., & Siepmann, T. (2016). Randomized controlled trials – a matter of design. *Neuropsychiatric Disease and Treatment*, *12*, 1341–1349. https://doi.org/10.2147/NDT.S101938

Stanton, J., Goldsmith, L., Adrion, W. R., Dunton, S., Hendrickson, K. A., Peterfreund, A., Yongpradit , P., Zarch, R., & Zinth, J. D. (2017). *State of the states landscape report: State-level policies supporting equitable K–12 computer science education*. https://www.edc.org/state-states-landscape-report-state-level-policies-supporting-equitable-k-12-computer-science

Sun, J., Ma, H., Zeng, Y., Dong Soo Han, & Jin, Y. (2022). Promoting the AI teaching competency of K-12 computer science teachers: A TPACK-based professional development approach. *Education and Information*

Technologies, *28*(2), 1509–1533. https://doi.org/10.1007/s10639-022-11256-5

Tong, F., Irby, B. J., & Lara-Alecio, R. (2015). Teachers' perception of virtual professional development in a randomized control trial. *International Journal of New Technology and Research*, *1*(7), 58–61.

Tran, Y. (2018). Computer programming effects in elementary: perceptions and career aspirations in STEM. *Technology, Knowledge and Learning*, *23*(2), 273–299. https://doi.org/10.1007/s10758-018-9358-z

Tschannen-Moran, M., & Hoy, A. W. (2001). Teacher efficacy: Capturing an elusive construct. *Teaching and Teacher Education*, *17*(7), 783–805. https://doi.org/10.1016/s0742-051x(01)00036-1

Waterman, K. P., Goldsmith, L., & Pasquale, M. (2019). Integrating computational thinking into elementary science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning. *Journal of Science Education and Technology*, *29*(1), 53–64. https://doi.org/10.1007/s10956-019-09801-y

Warner, J. R., Childs, J., Fletcher, C. L., Martin, N. D., & Kennedy, M. (2021, March). Quantifying disparities in computing education: Access, participation, and intersectionality. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (pp. 619–625). https://doi.org/10.1145/3408877.3432392

Yadav, A., Berges, M., Sands, P., & Good, J. (2016). Measuring computer science pedagogical content knowledge: An exploratory analysis of teaching vignettes to measure teacher knowledge. *Proceedings of the 11th workshop in primary and secondary computing education* (pp. 92–95). https://doi.org/10.1145/2978249.2978264

Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education*, *26*(4), 235–254. https://doi.org/10.1080/08993408.2016.1257418

Yadav, A., Krist, C., Good, J., & Caeli, E. N. (2018). Computational thinking in elementary classrooms: measuring teacher understanding of computational ideas for teaching science. *Computer Science Education*, *28*(4), 371–400. https://doi.org/10.1080/08993408.2018.1560550

Zhao, D., & Fan, L. (2022). What is the most important source for teachers' knowledge development? A meta-analysis of 27 empirical studies on the sources of teachers' knowledge. *Best Evidence in Chinese Education, 10*(2), 1375–1393. https://doi.org/10.15354/bece.22.ar036

Zhou, N., Nguyen, H., Fischer, C., Richardson, D., & Warschauer, M. (2020). High school teachers' self-efficacy in teaching computer science. *ACM Transactions on Computing Education*, *20*(3), 1–18. https://doi.org/10.1145/3410631

Appendix A
Content Knowledge Items and Rubric

What Is Computer Science?

Met expectations (3)	Partially met expectations (2)	Did not meet expectations (1)	Did not know (0)
Description accurately describes computer science as the study of computers, computational systems, algorithmic processes, including their principles, design, implementation, and impact on society. Responses may identify programming, artificial intelligence, computer systems and networks, security, database systems, human computer interaction, vision and graphics, numerical analysis, software engineering, bioinformatics, and theory of computing as key components of the field. Responses may indicate that computer scientists design and analyze algorithms to solve programs and study the performance of computer hardware and software.	Description accurately describes computer science as the study of computers and computational systems but may overemphasize the role of programming in the field or deemphasize the importance of understanding how computers are used to solve problems.	Description identifies CS as *only* related to programming.	Response indicates participant doesn't know.
Note: Adapted from https://undergrad.cs.umd.edu/what-computer-science and https://teacherslounge.codevirginia.org/portal/en/kb/articles/what-is-computer-science			

Describe what a computer programmer does.

Met expectations (3)	Partially met expectations (2)	Did not meet expectations (1)	Did not know (0)
Response indicates that computer programmers write and test code that allows computer applications and software programs to function properly. They turn the program designs created by software developers and engineers into instructions that a computer can follow. They may translate designs from software developers and engineers into workable code. They may also update or expand the code of existing programs or test programs for errors, finding and resolving faulty lines of code.	Response indicates that computer programmers write OR test code, but not both.	Response indicates participant doesn't know.	Response indicates participant doesn't know.
Note: Adapted from https://www.bls.gov/ooh/computer-and-information-technology/computer-programmers.htm and https://www.computerscience.org/careers/computer-programmer/			

What makes a device a computer?

Met expectations (3)	Partially met expectations (2)	Did not meet expectations (1)	Did not know (0)
Response identifies the 4 key components of a computer: input, output, processor, and memory and any description or elaboration of these components accurately describes them and their relationship to each other. Input: a way of translating information into a digital format that the computer can process. Output: a way of translating the digital information computers process and store into a format humans can understand. Processor: the part of the machine that controls storing digital information and caries out the instructions. It is the control center for everything the computer does. Memory: computers need things to process, this is stored in memory.	Response accurately identifies 2 of the key components of a computer, but may also include non-components. Any description or elaboration of the accurately-identified components accurately describes them and/or their relationship to each other.	Response accurately identifies fewer than two key components of a computer, and may also include non-components. Any description or elaboration of the accurately-identified components may not accurately describe them and/or their relationship to each other. Or Response indicates participant doesn't know.	Response indicates participant doesn't know.
Note: From https://teacherslounge.codevirginia.org/portal/en/kb/articles/overview-computing-systems			

What is an algorithm?

Met expectations (3)	Partially met expectations (2)	Did not meet expectations (1)	Did not know (0)
Describes algorithms as step-by-step instructions that produce a result. Response may indicate that humans use algorithms to decompose processes into step-by-step instructions, and often algorithms are used to create processes that can be automated. Algorithms have the following characteristics: (1) Use a common set of instructions that are clearly defined and produce consistent results, (2) The instructions are carried out in the correct order to produce the desired result, and (3) Produce a result and eventually end.	Describes an algorithm as a mathematical formula without elaboration or indication of the stepwise nature of algorithms.	Response indicates participant doesn't know.	Response indicates participant doesn't know.
Note: From https://teacherslounge.codevirginia.org/portal/en/kb/articles/overview-algorithms-and-programming			

In what ways is the term “variable” used differently in computer science than in math and science?

Met expectations (3)	Partially met expectations (2)	Did not meet expectations (1)	Did not know (0)
Response accurately describes how the term variable is used in both computer science and math or science. In computer science, **a variable is a name that represents data stored in memory.** While the program is running the variable’s value can change. When the program is done running the values entered are lost unless they are moved to a more permanent type of memory like a text file. Variable names can contain letters and numbers and should describe the data the variable holds. [a] In math, a variable is a symbol which functions as a placeholder for varying expression or quantities, and is often used to represent an arbitrary element of a set. In addition to numbers, variables are commonly used to represent vectors, matrices, and functions.[b] In science, a variable is an object, event, idea, feeling, time period, or any other type of category you are trying to measure; anything that can change or be changed (i.e., any factor that can be manipulated, controlled for, or measured in an experiment).[c]	Response accurately describes how the term variable is used in computer science but does not include a description of how a variable is used in either math or science.	Response conflates how the term variable is used in computer science and math or science or Response indicates participant doesn’t know.	Response indicates participant doesn’t know.
[a] https://teacherslounge.codevirginia.org/portal/en/kb/articles/input-and-variables. [b] https://en.wikipedia.org/wiki/Variable_(mathematics) [c] https://nces.ed.gov/nceskids/help/user_guide/graph/variables.asp			

Appendix B
Self-Efficacy Scale

How strongly do you agree or disagree with the following statements?	Strongly Disagree	Disagree	Somewhat Disagree	Somewhat Agree	Agree	Strongly Agree
I feel confident using computer technology.						
I know how to teach programming concepts effectively.						
I feel confident writing simple programs for the computer.						
I can promote a positive attitude toward programming in my students.						
I can guide students in using programming as a tool while we explore other topics.						
I feel confident using programming as an instructional tool within my classroom.						
I can adapt lesson plans incorporating programming as an instructional tool.						
I can create original lesson plans incorporating programming as an instructional tool.						
I can identify how programming concepts relate to the state CS curriculum standards.						

Appendix C
CS Implementation Survey

1. Did you teach any lessons that explicitly targeted CS standards of learning (SOLs) between the beginning of the school year and the end of January?

Yes
No

2. If yes, approximately how many lessons related to each of the following CS SOL strands did you teach between the beginning of the year and the end of January? If a lesson was designed to target multiple CS SOL strands, count it for each strand.

	None	1-2 lessons	3-4 lessons	5 or more lessons
Algorithms and Programming				
Computing Systems				
Cybersecurity				
Data and Analysis				
Impacts of Computing				
Networking and the Internet				

3. If not, then why?

Appendix D
PD Perception Survey

Part 1. PD perception survey items

How strongly do you agree or disagree with the following statements?	Time	Strongly Disagree (%)	Disagree (%)	Somewhat Disagree (%)	Somewhat Agree (%)	Agree (%)	Strongly Agree (%)	Mean (SD)
1. Communications regarding the PD support were received in a timely manner.	Post							
	Year end							
2. The PD learning objectives were clear to me.	Post							
	Year end							
3. I was provided with sample lesson plans that fit state standards.	Post							
	Year end							
4. The facilitators had adequate knowledge of the subject.	Post							
	Year end							
5. The facilitators created an atmosphere of trust and open communication.	Post							
	Year end							
6. I am satisfied with my interactions with the facilitators.	Post							
	Year end							
7. As needed, the facilitators were available to answer questions and provide direction.	Post							
	Year end							
8. I felt a rapport with other participants.	Post							
	Year end							
9. I am satisfied with my interaction with my peers.	Post							
	Year end							
10. I felt part of a learning community.	Post							
	Year end							
11. I found the online format of this PD as effective as previous in-person PD I've attended.	Post							
	Year end							

How strongly do you agree or disagree with the following statements?	Time	Strongly Disagree (%)	Disagree (%)	Somewhat Disagree (%)	Somewhat Agree (%)	Agree (%)	Strongly Agree (%)	Mean (SD)
12. The PD coaches and facilitators met my needs as a teacher-learner.	Post							
	Year end							
13. I would recommend this PD program to other colleagues.	Post							
	Year end							
14. I will integrate what I learned in the PD in my teaching.	Post							
	Year end							

Part 2. Survey items prompting teacher request for future support through PD

What additional support do you need to implement what you learned during the PD program into your instruction? I would benefit from additional PD in (select all that apply):
- Integrating the Virginia Computer Science Standards into instruction
- Choice Integrating algorithms and programming into instruction
- Integrating information about computing systems into instruction
- Integrating data and analysis into instruction
- Teaching about the impacts of computing
- Teaching about networking and the Internet
- Programming (any language)
- Coding in a block language (e.g., Scratch)
- Coding in a text-based language (e.g., Python)
- Choice Participating in curriculum writing (related to CS)
- Integrating CS instruction into remote teaching
I would benefit from additional PD in (select all that apply): - Other (Write in) - Text