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), 790-824.

Preparing Elementary School Teachers to Teach Computing, Coding, and Computational Thinking

Stacie L. Mason
Brigham Young University

Peter J. Rich
Brigham Young University

This literature review synthesized current research on preservice and in-service programs that improve K–6 teachers' attitudes, self-efficacy, or knowledge to teach computing, coding, or computational thinking. A review of current computing training for elementary teachers revealed 21 studies: 12 involving preservice teachers and nine involving in-service teachers. The findings suggest that training that includes active participation can improve teachers' computing self-efficacy, attitudes, and knowledge. Because most of these studies were fairly short-term and content-focused, research is especially needed about long-term outcomes; pedagogical knowledge and beliefs; and relationships among teacher training, contexts, and outcomes.

Learning computer science (CS) skills can benefit students economically and academically. In the United States, job opportunities in computer and information technology are projected to increase 13% in 10 years, compared to 7% overall projected job growth (Bureau of Labor Statistics, 2018a, 2018b). Numerous studies have indicated a host of benefits from learning CS, including improvement in student engagement, motivation, confidence, problem-solving, communication, and science, technology, engineering, and math (STEM) learning and performance (Clements & Gullo, 1984; Kim et al., 2015; Rich, Leatham, & Wright, 2013; Schanzer, Fisler, & Krishnamurti, 2018; Scherer, Siddiq, & Sánchez Viveros, 2018).

Recognizing these benefits, school districts and state governments are increasingly adopting policies that require CS instruction (Rich & Hodges, 2017). According to Stanton et al.'s (2017) report, seven U.S. states had adopted K–12 CS standards, all between 2014 and 2017, and as of 2017, an additional eight states were in the process of doing so. By May 2019, 31 states had adopted CS standards, and six other states were in the process of doing so (Code.org, 2019).

Increased policies have resulted in recent increases and revisions in standards for computational thinking (CT) and CS. For example, in 2013 England established standards for a national computing program of study (Department for Education, 2013). They are broken into four key stages, outlining ways in which pupils should be taught to think logically, understand networking, use technology safely, and design and develop information processing programs in at least two different languages.

Other governments have followed suit. Based on UNESCO's 2011 Information and Communication Technologies (ICT) standards for teachers, Australia has implemented standards for digital literacy that direct teachers to begin teaching computing as early as second grade (New South Wales Education Standards Authority, 2017). In Finland, the standards dictate that teachers must integrate CT into existing curricula beginning in first grade.

In the United States, a CT framework has been developed to help local governments create standards that will suit their own needs (*K–12 Computer Science Framework*, 2016). The framework covers five key concepts: (a) computing systems, (b) networks and the Internet, (c) data and analysis, (d) algorithms and programming, and (e) impacts of computing. These concepts cross all grade bands (K–2, 3–5, 6–8, and 9–12), but include guidance for increasing complexity as students progress from elementary to secondary education.

Most recently, the Computer Science Teachers Association and International Society for Technology in Education (ISTE, 2017) have rewritten their CT standards to indicate how CT should be integrated across a variety of subject areas rather than in CS education alone. These standards indicate how teachers ought to fill the roles of computational learners, leaders, collaborators, designers, and facilitators.

As the demand for CS instruction increases, there is a shortage of K–12 teachers who are trained to teach CS or CT. In a report by Google, Inc., and Gallup, Inc. (2016), 63% of surveyed K–12 principals in schools that did not offer CS instruction said that they lacked qualified teachers. As Kundukulam (2018) noted, CS graduates are more likely to take technology jobs than teaching jobs. Instead of hiring specialists to teach coding and computing, school principals are relying on existing teachers to teach CS (Rich, Browning, Perkins, Shoop, & Yoshikawa, 2018).

Unfortunately, most elementary school teachers have not been trained in the content and pedagogy of CS (Ng, 2017; Rich, Jones, Belikov, Yoshikawa, & Perkins, 2017; Stanton et al., 2017). Lacking training, teachers face emotional and knowledge barriers. To gain the competence and confidence to teach computing, elementary schools need effective preservice and in-service training.

Although extensive research has been published on the topics of teacher professional development (PD) and technology integration, relatively little has been published about preparing elementary teachers to teach CS skills. Several related literature reviews may inform ways to prepare computing teachers effectively. For example, multiple literature reviews have been published about teacher PD (Desimone, Porter, Garet, Yoon, & Birman, 2002; Guskey & Yoon, 2009, Institute for the Advancement of Research in Education at AEL, 2004), technology integration (e.g., Ertmer & Ottenbreit-Leftwich, 2010; Hew & Brush, 2007; Lawless & Pellegrino, 2007), computing education (Crick, 2017; Garneli, Giannakos, & Chorianopoulos, 2015; Kallia, 2017; Rich, Strickland, & Franklin, 2017; Waite, 2017), and CT in education (Grover & Pea, 2013; Heintz, Mannila, & Färngvist, 2016; Ilic, Haseski, & Tugtekin, 2018; Lockwood & Mooney, 2018; Lye & Koh, 2014).

Reviews about teacher training for teaching computing are lacking, especially in elementary (K–6) education. The purpose of this literature review was to examine what research has found about training K–6 teachers to teach CS and helping them to overcome their knowledge and self-efficacy barriers to teach elementary computing.

Definitions and Frameworks

The articles in this review represent the intersection of three concepts whose interplay forms the foundation for successfully preparing elementary computing teachers: (a) CS, (b) barriers to computing instruction, and (c) teacher preparation and development. Before the examination of that interplay, each concept is first defined and discussed separately.

CS

Included on this literature review are studies that relate to various aspects of elementary CS instruction. Certain authors have focused on computing skills, whereas others have focused on coding, robotics, or CT. The *K–12 Computer Science Framework* (2016), which was developed in partnership with states and districts by the Association for Computing Machinery, Code.org, the Computer Science Teachers Association, the Cyber Innovation Center, and the National Math and Science Initiative, used Tucker et al.'s (2003) definition of CS: "The study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society" (p. 6). Accordingly, CS curricula includes all of the following:

> programming, hardware design, networks, graphics, databases and information retrieval, computer security, software design, programming languages, logic, programming paradigms, translation between levels of abstraction, artificial intelligence, the limits of computation (what computers *can't* do), applications in information technology and information systems, and social issues (Internet security, privacy, intellectual property, etc.). (Tucker et al., 2003, p. 6)

CS, then, is the broad discipline that encompasses computing, CT, coding, and other branches dealing with computing connectivity and hardware.

The K–12 Computer Science Framework (n.d.) defined *computing* as "any goal-oriented activity requiring, benefiting from, or creating algorithmic processes" and *code* as "any set of instructions expressed in a programming language." We define *coding* as the act of writing code.

According to Grover and Pea (2013), CT is "viewed as at the core of all STEM disciplines" (p. 38). Put simply, CT is "thinking like a computer scientist" (Wing, 2006, p. 34). Despite the simple definition, CT is a broad term with multiple definitions (Barr & Stephenson, 2011; Jaipal-Jamani & Angeli, 2017; Sadik, Ottenbreit-Leftwich, & Nadiruzzaman, 2017; Shute, Sun, & Asbell-Clarke, 2017). As Wing (2006) further explained, CT is "thinking recursively," "using abstraction and decomposition," and "reformulating a seemingly difficult problem into one we know how to solve," and "involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (p. 33).

The K–12 Computer Science Framework (n.d.) used Lee's (2016) definition: "The human ability to formulate problems so that their solutions can be represented as computational steps or algorithms to be executed by a computer" (p. 3). For Lee, CT is using a computer to solve problems, whereas other definitions allow practitioners to apply principles of CT (e.g., algorithms or pattern-finding) without using computers or solving problems. For example, for the purposes of their literature review, Shute et al. (2017) defined CT as "the conceptual foundation required to solve problems effectively and efficiently (i.e., algorithmically, with or without the assistance of computers) with solutions that are reusable in different contexts" (p. 142).

The articles in this literature review reflect these varied definitions of CT, as well as studies involving other aspects of CS, coding, and programming. Throughout the paper, we treat *CS* as encompassing computing, coding, programming, robotics, and CT.

Barriers

Primary teachers face a variety of barriers to teaching CS. Teachers may face physical barriers, such as a lack of computers or reliable Internet access; institutional barriers in the form of unsupportive administrators or legislators; and emotional barriers, including beliefs, attitudes, and dispositions that hinder technology use. The articles in this review focus largely on intrinsic barriers, including knowledge, attitude, and efficacy barriers, which Ertmer, Ottenbreit-Leftwich, and York (2006) suggested may be more important than external, first-order barriers such as access to technology and resources.

Teacher knowledge significantly affects teacher practice (Borko & Putnam, 1995; Ertmer & Ottenbreit-Leftwich, 2010), and effective technology instruction or integration requires multiple types of knowledge. For teachers to teach with technology, they must understand the content they are teaching, the technology they are using, and pedagogy related to the content, technology, and students (Mishra & Koehler, 2006). A lack of knowledge in any of these areas — content, technology, or pedagogy — could be a knowledge barrier for teachers of coding, computing, or robotics.

Teachers with low technology self-efficacy are less likely to use technology than are confident teachers (Holden & Rada, 2011; Vannatta & Fordham, 2004). Perceived self-efficacy is a judgement of one's ability to perform (Bandura, 1977); technology self-efficacy is a person's belief that "he/she will be successful in using the technology" (Holden & Rada, 2011, p. 347).

Self-efficacy is distinct from what Bandura (1977) called outcomes expectations, the belief that specific behavior will lead to specific outcomes. Both efficacy beliefs and outcome expectations influence a person's likelihood to act. For example, if a teacher thinks students should learn to code, but lacks confidence in her ability to teach coding, she might choose not to teach coding. Similarly, a teacher who is confident in her abilities to use computers and teach effectively might choose not to teach coding if she does not expect that students need to learn coding.

In summary, several barriers may prevent educators from successfully teaching elementary computing. Although first-order barriers such as access and resources are important, the intrinsic, second-order barriers such as low self-efficacy or lack of technological, pedagogical, or content knowledge may enable or prevent them from successfully teaching computing (Ertmer, Ottenbreit-Leftwich, Sadik, Sendurer, & Sendurer, 2012).

Teacher Preparation and Development

This review includes studies involving both preservice and in-service teacher training to prepare elementary school teachers to teach computing, CT, or coding. Although preservice and in-service training may be similar, the needs of preservice teachers differ from those of in-service teachers.

Preparing preservice teachers. Preservice teachers may be more comfortable with technology than are many practicing teachers (Hargreaves & Fullan, 2000), but they lack experience, a particular teaching context, and advanced pedagogical knowledge (Ertmer & Ottenbreit-Leftwich, 2010). Numerous entities have described what preservice teachers should learn. In their *Framework for Understanding Teaching and Learning*, Bransford, Darling-Hammond, and LePage (2005) emphasized three areas of knowledge that preservice teachers must gain: (a) "knowledge of learners," (b) "knowledge of subject matter and curriculum goals," and (c) "knowledge of teaching," including pedagogy, differentiation, assessment, and classroom management (p. 11).

The technology, pedagogy, and content knowledge (TPACK) framework identifies three key, overlapping forms of knowledge used in technology integration: (a) content knowledge, (b) technological knowledge, and (c) pedagogical knowledge (Mishra & Koehler, 2006). The ISTE (2011) Standards for Computer Science Educators stated that teachers must demonstrate content knowledge, effective teaching and learning strategies, and professional knowledge and skills, and create effective learning environments.

Yadav, Stephenson, and Hong (2017) have argued that to prepare to teach CT, preservice K–12 teachers need deep understanding of both their content area and of CT, and that they should "learn to integrate computational thinking into the context of particular subject areas" (p. 61). What these various frameworks share is a concern with both content and pedagogical knowledge.

Ertmer and Ottenbreit-Leftwich's (2010) recommendations, which we apply in this review, suggested that teacher preparation and development programs should promote change in practice by helping teachers gain relevant knowledge, increase self-efficacy, address pedagogical beliefs, and respond to school culture. Teacher educators can do so by providing instruction, models, practice, and opportunities for reflection.

Developing in-service teachers. Practicing teachers tend to have more advanced knowledge and entrenched beliefs than preservice teachers and operate within specific contexts and cultures. To promote teacher change, Ertmer and Ottenbreit-Leftwich (2010) have argued that PD programs should build on teachers' pedagogical content knowledge and "include information about how they can use these tools in very specific ways, within specific content domains, to increase student content learning outcomes" (p. 272). Studies suggest that effective PD is (a) school-based, (b) active, (c) of sufficient duration, (d) coherent, (e) collaborative, and (f) content-focused (Avalos, 2011; Darling-Hammond, Hyler, & Gardner, 2017; Desimone, 2009; Guskey & Yoon, 2009; Odden & Picus, 2014). Desimone's (2009) core conceptual framework suggested that PD studies be evaluated based on the following elements:

1. PD that adheres to principles of effective PD;
2. Changes in teacher knowledge, beliefs, and attitudes;
3. Improved instruction;
4. Improved student learning; and
5. Context, including curriculum, policy, leadership, and teacher and student characteristics.

Kang, Cha, and Ha (2013) provided substantial theoretical and empirical support for Desimone's (2009) framework, which has been widely applied and cited. In this paper we applied Desimone's (2009) core conceptual framework as a lens to interpret in-service teacher studies.

Methods

The purpose of this literature review was to examine what research reveals about training K–6 teachers to teach CS and helping them to overcome their knowledge, attitude, and self-efficacy barriers. We completed a basic literature review in which we "summarize[d] and evaluate[d] the existing knowledge on a particular topic" (Machi & McEvoy, 2016, p. xiv). The lead author completed identification and analysis of studies in consultation with the second author. In studies of interventions to prepare elementary school teachers to teach CS, we asked,

1. How were preservice interventions aligned with elements of effective preservice preparation?
2. Were the in-service PD interventions aligned with elements of effective PD?
3. What changes in knowledge, attitude, and beliefs were reported as a result of preservice and in-service training?

Using ERIC, ProQuest PAIS Index, Scopus, ACM Digital Library, Academic Search Premier, and Google Scholar, we searched for relevant studies using combinations of the following keywords and phrases:

- ("computational thinking" OR "robotics" OR "computing" OR "computer science") AND
- ("teacher training" OR "teacher education" OR "professional development" or "coaching" OR "mentoring") AND
- ("elementary" OR "primary" OR "k-5" OR "k-6" OR "k-3" or "4-6") AND
- ("self-efficacy" OR "attitudes" OR "beliefs").

From the initial 293 results, we included for further analysis empirical studies published in English-language academic journals, books, and conference papers that included an intervention for training elementary school teachers to teach computational concepts or skills, and described changes in teachers' knowledge of, attitudes toward, or self-efficacy for teaching CS.

We also chose to focus on the 10-year period of 2008–2018, as approaches to preparing CS teachers more than 10 years ago may differ significantly from current needs and technologies. We eliminated articles that pertained only to (a) secondary school teachers, (b) administrator attitudes, (c) student perceptions, (d) student outcomes, or (e) teacher attitudes toward something other than teaching computing (e.g., gender).

By scanning titles and abstracts, we narrowed the pool to 46 potential articles. In reading the 46 articles, we identified 15 articles that fit these criteria for further analysis. During the process of writing the paper, we identified six additional newly published studies that matched our inclusion criteria.

After identifying relevant articles, we categorized articles based on whether they involved preservice or in-service teachers, then analyzed each study according to one of two frameworks: studies involving preservice teacher preparation were analyzed using the preservice teacher preparation framework in Figure 1; studies involving PD were analyzed using Desimone's (2009) core conceptual framework (Figure 2).

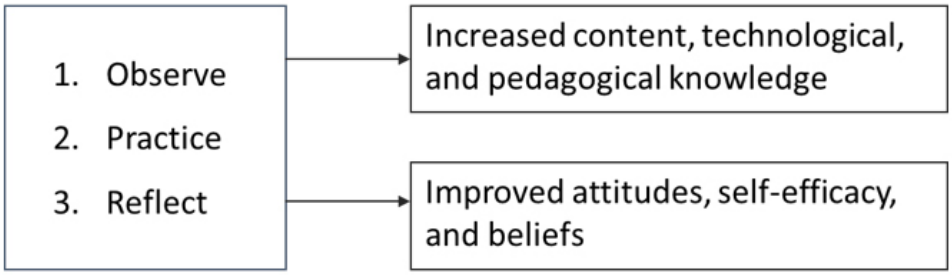

Figure 1. *Preservice teacher preparation framework, based on Ertmer and Ottenbreit-Leftwich's (2010) recommendations for facilitating teacher change.*

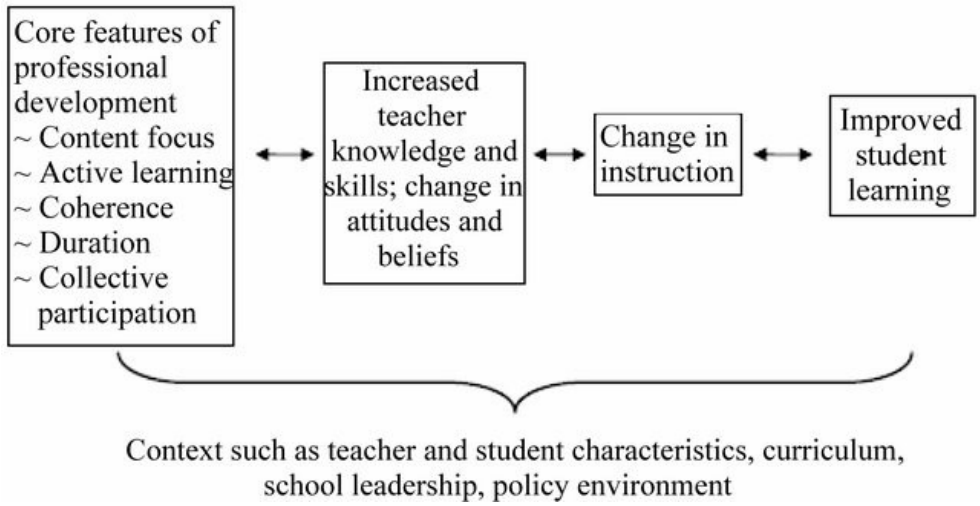

Figure 2. *Core conceptual framework. From "Improving Impact Studies of Teachers' Professional Development: Toward Better Conceptualizations and Measures," by L. M. Desimone, 2009,* Educational Researcher, 38*, p. 185. Copyright 2009 by Sage Publications. Reprinted with permission.*

The two frameworks are similar, but the preservice framework is simpler than Desimone's (2009) framework. Although Ertmer and Ottenbreit-Leftwich's (2010) recommendations may be applied to both preservice and in-service training, Desimone's framework allows for more detailed analysis by including additional elements relevant to PD that may not exist in preservice teacher preparation.

Findings

As noted, our search was narrowed to 21 studies about training K–6 teachers to teach CS and to overcome their knowledge, attitude, and self-efficacy barriers. This section includes summaries of each study in terms of participants, context, focus, learning activities, duration and format, assessment instruments and data, learning outcomes, and attitude/belief outcomes (Appendix A and C), followed by analysis of the studies aligned with the research questions and frameworks described previously (Appendix B and Table 1). We first discuss the 12 preservice teacher studies, followed by the nine in-service teacher studies.

Preservice Elementary CS Teacher Training

A summary of the 12 preservice studies we identified is provided in Appendix A. Following is our analysis of the studies, centered around three main elements:

1. Instruction that includes models, practice, and reflection;
2. Increased content, technological, and pedagogical knowledge; and
3. Improved attitudes, self-efficacy, and beliefs.

Elements of instruction. Although there are numerous elements of effective teacher preparation, Ertmer and Ottenbreit-Leftwich (2010) highlighted the value

of observation, hands-on practice, and reflection. Appendix B summarizes which preservice studies reported each element.

Observation. One way that people learn is through observing examples or models (Bandura, 1977). To understand how to teach any content, preservice teachers need to see examples of effective teaching, and to understand CS it is helpful to see specific skills demonstrated (Ertmer & Ottenbreit-Leftwich, 2010).

As shown in Appendix B, of 12 preservice interventions in this review, only one included demonstrations of both CS concepts and teaching applications (Yadav et al., 2014); three other studies included examples of teaching with technology (Chang & Peterson, 2018; Ma, Lai, Williams, Prejean, & Ford, 2008; Ng, 2017); four included demonstration of coding, robotics, or CT but not examples of teaching children that content (Cetin, 2016; Cetin & Andrews, 2016; Jaipal-Jamani, 2018; Jaipal-Jamani & Angeli, 2017); four studies mentioned neither content nor pedagogical demonstrations (Jeon & Kim, 2017; Kaya, Yesilyurt, Newley, and Deniz, 2018; Kim et al., 2015; Sadik et al., 2017). Possibly, the latter interventions included demonstrations of content or pedagogical skills but did not describe them in those terms. Additionally, one could argue that all the interventions that included instruction in coding, robotics, or CT inherently included models of teaching CS. However, because elementary students are different developmentally from college students, we did not count college-level instruction as a model for primary-level instruction.

Practice. Preservice teachers need practice, not only mastering new technological skills but also teaching skills and concepts to others — ideally to children (e.g., Ertmer & Ottenbreit-Leftwich, 2010; Hammerness, Darling-Hammond, & Bransford, 2005). All preservice studies in this review included some form of practice. As shown in Appendix B, nine interventions included practice with coding, CT, or robotics but no practice teaching the skills and concepts (Cetin, 2016; Cetin & Andrews, 2016; Chang & Peterson, 2018; Jaipal-Jamani, 2018; Jaipal-Jamani & Angeli, 2017; Jeon & Kim, 2017; Kaya et al., 2018; Kim et al., 2015; Yadav et al., 2014).

One study included teaching practice without separate content practice (Sadik et al., 2017). Only two of 12 studies reported providing practice both in mastering CS skills or concepts and teaching those skills or concepts (Ma et al., 2008; Ng, 2017). In Ma et al.'s (2008) study, preservice teachers facilitated an activity with children; in Ng's (2017) study, preservice teachers taught each other. Overall, the 12 studies were much stronger in providing content practice than teaching practice.

Reflection. Reflecting on experiences and new information helps preservice teachers examine their pedagogical beliefs, make sense of their experiences, and assimilate or accommodate new knowledge and beliefs (Smagorinsky, Shelton, & Moore, 2015). Four preservice studies reported including reflection as part of their interventions (Chang & Peterson, 2018; Ma et al., 2008; Sadik et al., 2017; Yadav et al., 2014). In all four cases, written reflections were used as assessment data. Additional studies possibly incorporated reflection but did not describe it as such.

Overall, the preservice studies in this review were moderately aligned with Ertmer and Ottenbreit-Leftwich's (2010) recommended practices for effectively preparing preservice teachers. Although almost all interventions included practice doing CS, only three studies included practice teaching CS. Only a third of interventions

included demonstrations of how to teach such concepts to children, and only a third included reflection. Reported outcomes emphasized content over pedagogy.

Increased knowledge. A prime goal of teacher preparation is to increase teacher knowledge. Teacher knowledge can be categorized in various ways. Here we used the three key types described by Mishra and Koehler (2006): content knowledge, technological knowledge, and pedagogical knowledge. However, in teaching CS content and skills, content and technology are often intertwined.

Content and technological knowledge. CS can be taught as its own content area or integrated into other subjects. The preservice studies in this review include both approaches. As shown in Appendixes A and B, five studies treated CS as subject matter (Cetin, 2016; Cetin & Andrews-Larson, 2016; Jeon & Kim, 2017; Ng, 2017; Sadik et al., 2017). Although these interventions varied in characteristics, we highlight two trends.

First, four of five included both primary and secondary teachers, suggesting a fundamental set of knowledge is essential to understanding CS content, regardless of the level at which it will be taught. Secondary teachers need to have a depth of CS knowledge beyond elementary teachers, but these studies all ensure that both elementary and secondary teachers understand fundamental concepts (e.g., sequences, commands, loops, and conditionals).

Second, four of five studies reported changes in content knowledge, but only one reported changes in pedagogical knowledge, and one reported changes in attitudes or beliefs. Thus, the main focus of many preservice studies appears to be to build teachers' content knowledge first and foremost.

Seven studies focused on integrating computing or CT into other subjects (Chang & Peterson, 2018; Jaipal-Jamani, 2018; Jaipal-Jamani & Angeli, 2017; Kaya et al., 2018; Kim et al., 2015; Ma et al., 2008; Yadav et al., 2014). Four of these interventions were taught within the context of science or STEM methods courses (Jaipal-Jamani, 2018; Jaipal-Jamani & Angeli, 2017; Kaya et al., 2018; Kim et al., 2015). Other contexts included two technology integration courses (Chang & Peterson, 2018; Ma et al., 2008) and one introductory psychology course (Yadav et al., 2014). Although a heavy focus was placed on strengthening teachers' computing content knowledge, all but one integration study reported changes in attitudes or beliefs.

The studies in this review included training in coding, computing, CT, programming, robotics, or a combination thereof. As shown in Figure 3, 11 of 12 preservice interventions included coding or programming, eight included robotics, and six included CT. All six studies involving CT were published in 2017 or 2018, which suggests that studies on training preservice elementary CS teachers is a new and growing area of study. The significant overlaps among coding, CT, and robotics suggest that robots are frequently being used to teach coding or programming, and coding, programming, and robotics can effectively be used to teach CT.

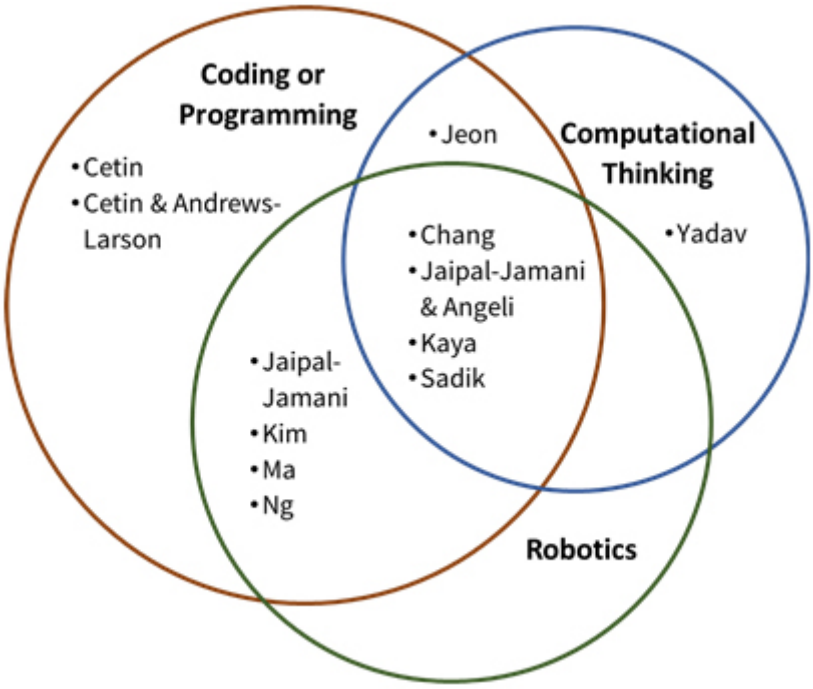

Figure 3. *CS focus in preservice studies, listed by primary author. The relative size of circles reflects the relative number of related studies.*

As shown in Appendix B, nine of 12 preservice studies in this review reported increased knowledge of coding, robotics, or CT. Jeon and Kim (2017) and Kaya et al. (2018) did not assess changes in knowledge. Kim et al. (2015) assessed changes in STEM knowledge and found no significant difference between pre- and posttest scores following a 3-week robotics unit in a STEM instruction course.

Pedagogical knowledge. Pedagogical knowledge is a key objective for preservice teacher development (Bransford et al., 2005; Ertmer & Ottenbreit-Leftwich, 2010; Mishra & Koehler, 2006). Pedagogical knowledge refers to knowledge of how to teach effectively, which incorporates knowledge of human development, learning and teaching theories, and classroom management.

Pedagogical knowledge incorporates pedagogical content knowledge, or understanding how to teach specific subjects, and technological pedagogical knowledge, or knowing how to use technology effectively to teach. As shown in Figure 3, among 12 preservice studies in this review, only four reported increases in pedagogical knowledge following interventions (Chang et al., 2015; Ma et al., 2008; Ng, 2017; Yadav et al., 2014) — the same four studies that provided demonstrations of how to teach computing concepts and skills. This fact demonstrates once again an overall greater focus on developing preservice teachers' CS content knowledge than pedagogical knowledge.

Improved attitudes, self-efficacy, and beliefs. Besides increasing knowledge, preservice teacher preparation should improve self-efficacy and attitudes toward teaching coding, CT, or robotics. As shown in Appendix B, seven studies reported changed attitudes or beliefs, including self-efficacy (Chang &

Peterson, 2018; Jaipal-Jamani, 2018; Jaipal-Jamani & Angeli, 2017; Jeon & Kim, 2017; Kaya et al., 2018; Kim et al., 2015; Yadav et al., 2014). All but one of the integration studies reported changes in attitudes or beliefs, although only one nonintegration study did so.

In all 12 studies, researchers observed improvements in teachers' knowledge, self-efficacy, or attitudes. In addition to the observed changes, several authors noted no significant differences in certain assessed criteria. Cetin (2016) and Cetin and Andrews-Larson (2016) found that their programming training led to no significant change in preservice teachers' attitudes toward computer programming, although it did lead to an increase in knowledge. The lack of change could be explained by the fact that attitudes were positive before the training. The authors also noted that the interventions were relatively brief (6 weeks and 3 weeks).

Similarly, Kaya et al. (2018) reported improved self-efficacy, but no change in outcome expectancy. Kim et al. (2015) found that among preservice teachers, training in robotics did not lead to increased interest in math, technology, and STEM careers, nor to significant increases in understanding of science, technology, engineering, or math. Furthermore, Yadav et al. (2014) found that preservice teachers given training in CT did not have higher comfort or interest in computing, compared with preservice teachers given alternate training in cognition, problem solving, and creativity. The intervention was two 50-minute sessions.

Brief interventions would not be expected to motivate immediate or lasting changes to teachers' career plans nor significantly affect tests in broad STEM subjects. The problem with no observable difference in such studies may be more closely associated with the duration of the training than teachers' interest in CS. In all five of these cases, the authors accounted for the lack of change in an area and observed changes in other areas.

In summary, research on preservice CS teacher training is emergent, with the majority of such studies having been published in the past 2 years. The main focus of most of these studies has been to strengthen teachers' content knowledge and attitudes toward CS. The only studies that focused on developing teachers' pedagogical knowledge were those that followed Ertmer and Ottenbreti-Leftwich's (2010) recommendation of teacher observation. The duration of these studies has varied from as little as two 50-minutes sessions to an entire semester. Despite these differences, results with preservice CS teacher training have been overwhelmingly positive, especially in regard to increasing teachers' knowledge of CS.

In-Service Training

Appendix C and Table 1 summarize the nine studies involving in-service teacher PD. Following the summaries, we analyze the studies using Desimone's (2009) core conceptual framework for evaluating PD studies. The five main elements that we considered were:

1. effective professional development;
2. changes in knowledge, attitudes, and beliefs;
3. changes in instruction;
4. improved student learning; and,
5. context.

Table 1
Reported Inputs and Outcomes in In-Service Studies

Inputs				**Outputs**			
Authors, Year	**CS Practice**	**Lesson Planning**	**Teaching Practice**	**20+ Hours**	**Increased Content Knowledge**	**Increased Peda-gogical Knowledge**	**Changed Attitudes Or Beliefs**
CS as Content							
Bers et al., 2013	X	X		X	X	X	X
Leonard et al., 2018	X			X			X
Marcelino et al., 2018	X			X	X		
Roberts et al., 2018			X		X	X	X
Toikkanen & Leinonen, 2017	X			X	X	X	X
Integrated CS							
P. J. Rich et al., 2017	X			X			X
Carter et al., 2014		X		X			X
Coleman et al., 2016		X	X	X	X	X	X
Hestness et al., 2018	X				X	X	

Principles of effective PD. According to Desimone (2009), the five key features of effective PD are content focus, active learning, coherence, duration, and collective participation.

Content focus. PD with a specific content focus has had better outcomes for teacher learning than PD that lacks subject matter focus (Desimone, 2009). CS can be both taught as its own content area or integrated into other subjects; the studies

in this review include both approaches. As indicated in Appendix C and Table 1, five studies treated CS as subject matter (Bers, Seddighin, & Sullivan, 2013; Leonard et al., 2018; Marcelino, Pessoa, Vierira, Salvador, & Mendes, 2018; Roberts, Prottsman, & Gray, 2018; Toikkanen & Leinonen, 2017); four studies focused on integrating computing or CT into other subjects (Carter et al., 2014; Coleman, Gibson, Cotton, Howell-Moroney, & Stringer, 2016; Hestness, Ketelhut, McGinnis, & Plane, 2018; P. J. Rich et al., 2017). There is considerable variation among the studies in each group. For example, of the two briefest interventions, one approached coding as its own subject (Roberts et al., 2018); the other involved integration of CT (Hestness et al., 2018).

There are also some trends within groups, however. All three of the online interventions approached CS as discrete subject matter (Leonard et al., 2018; Marcelino et al., 2018; Toikkanen & Leinonen, 2017). All three interventions that included embedded PD involved integrating computing or CT into other subjects (Carter et al., 2014; Coleman et al., 2016; P. J. Rich et al., 2017).

The studies in this review included training in coding, computing, CT, programming, robotics, or a combination thereof. As shown in Figure 4, six of the nine in-service interventions included coding or programming. The five studies that included a focus on CT also involved computing, robotics, coding, or programming, which suggests that hands-on experiences can facilitate changes in elementary teacher attitudes or knowledge about CT. Four of the in-service interventions included computing, which was not the focus of any preservice studies.

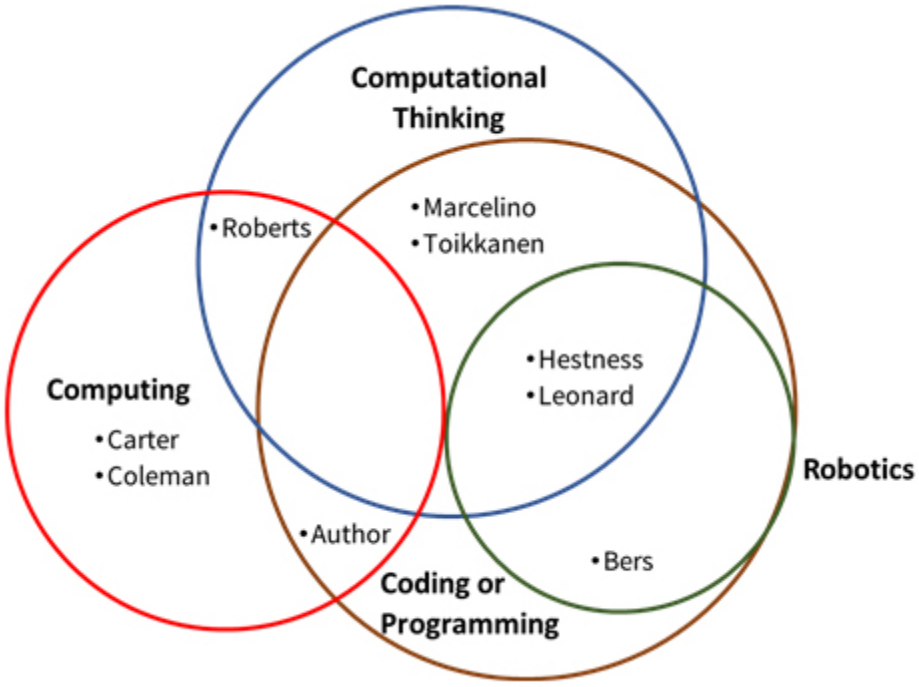

Figure 4. *CS focus in in-service studies, listed by primary author. The relative size of circles reflects the relative number of related studies.*

Active learning. Active learning has had better outcomes for teacher PD than passive learning (Desimone, 2009). Whereas listening to a lecture is passive, discussion, observation, and hands-on practice are active. As shown in Table 1, all nine PD interventions in this review included active learning. Six included programming activities or challenges (Bers et al., 2013; Hestness et al., 2018; Leonard et al., 2018; Marcelino et al., 2018; P. J. Rich et al., 2017; Toikkanen & Leinonen, 2017). Three interventions included lesson planning or curriculum design (Bers et al., 2013; Carter et al., 2014; Coleman et al., 2016). Despite this focus on an active approach, only two included practice teaching (Coleman et al., 2016; Roberts et al., 2018). As with the preservice interventions, the studies in this review focused more on developing teachers' CS content knowledge than providing pedagogical practice.

Coherence. Desimone (2009) suggested two key aspects to coherence: First, learning must cohere with teachers' knowledge and beliefs; second, PD content must be consistent with school, district, and state policy. Among the PD studies in this review, designing and evaluating the PD for coherence with teachers' beliefs and relevant policies did not seem to be a prominent concern; authors did not explain their designs or methods in terms of coherence. Six of the nine studies addressed teacher beliefs of various types and to various extents (Bers et al., 2013; Coleman et al., 2016; Leonard et al., 2018; P. J. Rich et al., 2017; Roberts et al., 2018; Toikkanen & Leinonen, 2017). For example, Leonard et al. (2018) assessed teachers' beliefs regarding outcome expectancy and culturally responsive pedagogy, while Roberts et al. (2018) assessed teachers' self-efficacy and beliefs about CS. Only two studies addressed policy contexts (Hestness et al., 2018; P. J. Rich et al., 2017).

Duration. For PD to be effective, interventions must be of sufficient duration, over an appropriate span of time. Desimone (2009) suggested a general guideline of 20 hours or more of contact spread over a semester. Like the preservice interventions described previously, in-service training programs have varied in length (Appendix C). Three studies included extensive, ongoing PD (Carter et al., 2014; Coleman et al., 2016; P. J. Rich et al., 2017). Three included summer workshops (Bers et al., 2013; Carter et al., 2014; Coleman et al., 2016). Two involved short-term in-service training (Hestness, et al., 2018; Roberts et al., 2018). Three studies involved distance in-service teacher training (Leonard et al., 2018; Marcelino et al., 2018; Toikkanen & Leinonen, 2017). All nine studies, whether brief or extended, reported positive changes in knowledge or attitude, suggesting that brief interventions may be sufficient to promote changes in knowledge, attitudes, or perceived self-efficacy.

Collective participation. Research has shown that PD tends to be more effective when teachers from the same school, grade, or department participate together (Desimone, 2009). Collective participation also suggests collaboration, and most or all interventions in this review incorporated collective participation in terms of collaboration or discussion. Most studies also included participants from the same school, grade, or department (Appendix C). One intervention involved all the teachers in a single school, working in grade-level teams (P. J. Rich et al., 2017). Four interventions targeted teachers from large geographic regions but a narrow band of grade levels (Bers et al., 2013; Carter et al., 2014; Coleman et al., 2016; Hestness et al., 2018). Four interventions, including three online, involved multiple schools and grade levels (Leonard et al., 2018; Marcelino et al., 2018; Roberts et al., 2018; Toikkanen & Leinonen, 2017). In the larger studies, there were enough participants for teachers to find colleagues teaching similar student

populations. However, in the smallest study (Marcelino et al., 2018), only eight participants came from multiple countries and taught K–12, limiting the opportunity for collaborating with teachers from the same school or grade.

The studies in this review suggest that collective participation has been a higher priority when integrating CS than when teaching CS as a subject. All four studies that involved integration included a narrow band of grade levels or a single school, whereas among the five interventions that treated CS as a discrete subject, four spanned several grade levels and large geographic regions.

Based on the features of PD described by Desimone (2009), the PD studies in this review were strong in providing active learning opportunities, but varied in their content focus, coherence, duration, and collective participation. Despite differences, all nine studies reported improvements.

Changes in knowledge, skills, attitudes, and beliefs. According to Desimone's (2009) framework, effective PD should promote "increased teacher knowledge and skills" and/or "changes in attitudes and beliefs" (p. 185). As shown in Table 1, seven studies reported improved attitudes or self-efficacy beliefs, six studies showed increased content knowledge, and five studies reported increased pedagogical knowledge. Only two studies assessed beliefs other than self-efficacy (P. J. Rich et al., 2017; Roberts et al., 2018). According to Pajares (1992), beliefs are harder to change than knowledge. However, to help teachers change their practice, they need either to be taught in ways that are consistent with their beliefs or their beliefs need to be changed (Desimone, 2009; Ertmer & Ottenbreit-Leftwich, 2010; Pajares, 1992). Most of the studies in this review did not explicitly design their intervention to cohere to nor to change teachers' pedagogical beliefs.

In addition to the reported changes, two PD studies reported no significant differences for certain assessed factors. In the study by Coleman et al. (2016), teacher attitude and anxiety did not noticeably affect teacher preparedness to teach computing. Possibly, knowing they would be observed motivated teachers to prepare lessons that they may not have wanted to teach. Leonard et al. (2018) showed small gains in understanding and attitude, but those gains did not reach levels of significance, perhaps because the 45 participants were divided into multiple, small cohorts.

Changes in instruction and student learning. The third element in Desimone's (2009) path model is "change in instruction," followed by "improved student learning" (p. 185). Only one of nine PD studies in this review assessed classroom instruction. Coleman et al. (2016) found that participation in summer institutes positively influenced preparedness (having a plan with clear lesson objectives) and lesson execution (ability to teach lesson without assistance), attitude positively influenced execution, and neither attitude nor anxiety influenced preparation.

Although P. J. Rich et al. (2017) and Toikkanen and Leinonen (2017) did not assess changes in instruction, they did note changes. Toikkanen and Leinonen (2017) observed that "the vast majority [of teachers] had no trouble with the tools and seeing their immediate benefits in their classrooms" (p. 246). Teachers in P. J. Rich et al.'s (2017) study were not required to implement their training, but researchers found that teachers who chose to do so had more positive attitudes toward CS than teachers who did not.

The remaining six studies did not address changes in practice, and none of the nine studies examined student learning. In Guskey's (2002) Model of Teacher Change, changes in teacher practice and student learning precede substantial changes in attitudes and beliefs: "The key element in significant change in teachers' attitudes and beliefs is clear evidence of improvement in the learning outcomes of their students" (pp. 383–384). If the ultimate purpose of PD is to improve student learning, then improving teacher knowledge and attitudes are a necessary, intermediate step on the path.

Context. Underlying the four elements of Desimone's (2009) model, context "operates … as an important mediator and moderator" (p. 185). Context includes "teacher and student characteristics, curriculum, school leadership, [and] policy environment" (p. 185). Numerous theorists and researchers have written on context and of the need for teacher educators to align PD to teachers' contexts (Desimone, 2009). Although each of the studies in this review includes a description of the research context, most interventions were provided to teachers who represented a wide range of contexts.

Only one of the interventions (P. J. Rich et al., 2017) was provided to teachers from a single school. Tailoring training to context is difficult when teachers represent multiple grades, districts, and levels of experience. Policy context was at least a minor concern in two studies (Hestness et al., 2018; P. J. Rich et al., 2017). Leonard et al. (2018) addressed student characteristics by teaching robotics and game design within a framework of culturally responsive pedagogy. Teacher beliefs, primarily self-efficacy beliefs, were addressed in six studies (Bers et al., 2013; Coleman et al., 2016; Leonard et al., 2018; K. Rich et al., 2017; Roberts et al., 2018; Toikkanen & Leinonen, 2017). Most of the studies addressed only one or two aspects of context.

In summary, the few published studies on training in-service elementary CS teachers have thus far yielded positive results. All studies reported increases in either content knowledge, teacher attitudes, or beliefs. Nearly all studies emphasized the development of content knowledge, but fewer than half reported on developing teachers' pedagogical content knowledge. The emphasis in these studies has been firmly fixed on teacher growth, with no studies reporting how students have changed or how changes in student knowledge have affected teacher growth in content knowledge, attitudes, or beliefs about CS.

Study Limitations

In this review we chose to focus on training for elementary school teachers, and therefore excluded sources related only to secondary teachers, because the training and expertise of secondary school teachers differ substantially from the training and expertise of primary school teachers. However, several studies included in this review involved both primary and secondary teachers (Cetin, 2016; Cetin & Andrews-Larson, 2016; Jaipal-Jamani, 2018; Jeon & Kim, 2017; Kaya et al., 2018; Leonard et al., 2018; Marcelino et al., 2018; Sadik et al., 2017; Yadav et al., 2014). Therefore, the results from those studies may not be entirely relevant to elementary school teacher training.

At the same time, we may have missed important contributions from studies involving only secondary teachers. Similarly, the studies in this review included interventions and outcomes related to overcoming barriers to teaching computing

or coding. We are aware of papers related to teacher training for CT, computing, or coding that did not fit our inclusion criteria because they lacked interventions or focused on outcomes other than changes in teacher attitude or content knowledge (e.g., Bower & Falkner, 2015; Francis et al., 2018; Israel et al., 2018; Mannila, Nordén, & Pears, 2018; Rich et al., 2018; Yadav, Gretter, Good, & McLean, 2017). We also chose to focus on the 10-year period of 2008–2018, as approaches to teaching CS more than 10 years ago may not be relevant to current needs. Although a few relevant studies may have been published prior to 2008, only one of the studies in this review was published before 2013, and most were published between 2016 and 2018, demonstrating how nascent our understanding of elementary computing teacher training really is.

As with all research, we are also wary of the publication bias toward studies that show positive results. Because we chose not to search archives of unpublished studies, it is possible that there have been other studies on elementary teacher CS preparation that did not yield positive results. Finally, given the limited number of relevant studies, we did not apply quality exclusion criteria in our selection process; future reviewers may choose to be more selective. In particular, we included recent conference papers to provide the most up-to-date information; however, conference papers often lack the detail, and in some cases the rigor, of peer-reviewed journal articles.

Implications for Practice

At the start of this paper, we argued that elementary school teachers should teach their students CT and coding but lack the knowledge and confidence to do so. The studies we reviewed indicate that training and PD can help elementary school teachers to overcome their knowledge, attitude, and efficacy barriers. They also suggest that effective interventions may vary but should provide opportunities to practice both doing CS and teaching CS.

All 21 studies in this review included active opportunities for practice, thus supporting a link between experience and improved self-efficacy (Ertmer & Ottenbreit-Leftwich, 2010; Mueller, Wood, Willoughby, Ross, & Specht, 2008; Somekh, 2008). However, although most interventions included practice doing coding, robotics, or CT, few interventions included practice teaching.

To prepare to teach CS concepts and skills, both preservice and in-service teachers need practice teaching those concepts in authentic contexts (Ertmer & Ottenbreit-Leftwich, 2010; Hammerness et al., 2005). Interventions that include CS but not teaching practice may be sufficient to influence changes in knowledge, attitudes, and self-efficacy but insufficient to support lasting changes in teacher practice and student learning (Desimone, 2009). Likewise, many of the studies in this review involved relatively brief interventions. Brief interventions may be appropriate for targeted skills training, but we recommend long-term, ongoing, contextualized training for teachers who will be expected to teach coding, CT, computing, or robotics, either as a discrete subject or integrated across the curriculum.

In summary, to help elementary school teachers overcome knowledge and efficacy barriers as they prepare to teach computing, both preservice and in-service teacher trainers should continue to provide elementary school teachers opportunities to practice CS. Teachers appear to appreciate more hands-on, practical coding experiences to develop their own content knowledge.

In addition, preservice training should more often include modeling of CS and teaching skills; practice teaching CS to children; and reflection (Ertmer & Ottenbreit-Leftwich, 2010). In-service teacher training has tended to have a strong CS focus and training, but should pay greater attention to coherence, pedagogical training, and context (Desimone, 2009; Ertmer & Ottenbreit-Leftwich, 2010; Guskey, 2002; P. J. Rich et al., 2017).

In-service training could be improved as teacher trainers address teacher beliefs, include lesson planning and implementation as components of training, and design and deliver instruction specific to school environment, culture, leadership, and teacher and student characteristics (Desimone, 2009; Ertmer & Ottenbreit-Leftwich, 2010).

Implications for Research

All 21 studies in the review focused on short-term outcomes, such as changes in self-efficacy and knowledge. Only three discussed how such changes might translate into improved teaching practice, and none assessed elementary student outcomes. Possibly, measures of change in elementary students are also nascent and may not be widely known among professional developers. For example, only a single validated instrument has been created to measure elementary students' CT (Román-González, Pérez-González, & Jiménez-Fernández, 2017; Román-González, Pérez-González, Moreno-León, & Robles, 2016). Also only a few instruments have been developed to assess elementary student attitudes toward CS (Dorn & Tew, 2015; Gibbons, Hirsch, Kimmel, Rockland, & Bloom, 2004; Hoegh & Moskal, 2009).

Although these instruments existed prior to the majority of studies reviewed in this paper, such instruments are not well known and have not been widely used by researchers to examine the effects of teaching elementary CS. As more researchers become familiar with these tools, more studies may correlate teacher learning of CS with student growth or outcomes.

This review also suggests that recent studies have not sufficiently examined how teacher training affected actual practice. Coleman et al. (2016) examined the influence of PD and attitude on preparedness and execution but did not show the influence of PD on attitude. The next step would be to show the influence of PD on both attitude and practice. Toikkanen and Leinonen (2017) included only a few findings but mentioned that "the vast majority [of teachers] had no trouble with the tools and seeing their immediate benefits in their classrooms" (p. 246).

More data are needed to support their conclusion. P.J. Rich et al. (2017) found that self-efficacy varied widely by teacher background, willingness to experiment, and level of implementation, thus suggesting a relationship among experimentation, implementation, and self-efficacy. However, P. J. Rich et al.'s findings could have been strengthened by including pre- and postsurveys, which might more accurately demonstrate teacher growth over time. More studies that examine the influence of PD on both teacher attitudes and practice are needed.

To understand effective preservice and in-service training for teaching CS, longitudinal studies are needed to show changes in attitudes, beliefs, knowledge, practice, and student outcomes over time. Although longitudinal studies are difficult given word count limits and tenure and promotion requirements, the

field needs such studies to know whether interventions support lasting improvements.

One sign for optimism is that several studies in this review were part of larger studies (Carter et al., 2014; Cetin, 2016; Cetin & Andrews, 2016; Coleman et al., 2016; Jaipal-Jamani, 2018; Jaipal-Jamani & Angeli, 2017), which may include longitudinal data collection and analysis. Thus, reports of the effect of training on practice and student outcomes may increase in the literature over the next several years.

Research that addresses teachers' contexts and pedagogical beliefs is recommended. Interventions should be designed to either cohere to or change teachers' pedagogical beliefs, and for that to happen, researchers must assess teachers' pedagogical beliefs. In addition, the contexts in which teachers work heavily influence teacher change and practice. Intervention studies are needed that examine how policy and culture influence teachers' CS attitudes, beliefs, and practices.

To summarize, the studies in this review show that training can improve elementary teachers' self-efficacy, attitudes, and knowledge for teaching CS. Insofar as this field is relatively new, more studies are needed, especially to understand how preservice and in-service training influences elementary student outcomes; changes in teacher practice, both immediate and long-term; and interactions among context, training, and outcomes.

Conclusion

The purpose of this review was to describe the breadth of research on the training of computing teachers in elementary education in the current era. In this review of literature, we found that (a) few studies have been published about training elementary school teachers to teach computing, coding, and programming, with slightly more studies on preservice teacher training than in-service PD; (b) interventions have focused more on developing elementary CS teachers' content knowledge than their pedagogical knowledge; (c) studies overwhelmingly showed that training can improve teachers' self-efficacy, attitudes, and knowledge, even over relatively short interventions; and (d) the literature has said little about whether or to what extent changes in self-efficacy, attitudes, and knowledge lead to changes in actual practice or improved student learning.

Our research questions focused on examining the current state of training for both preservice and in-service computing teachers. In addition, we asked to what extent these studies reported on changes in teachers' knowledge, attitudes, and beliefs about computing. We analyzed preservice teacher preparation using Ertmer and Ottenbreit-Leftwich's (2010) framework for PD, which proposes that PD that includes modeling, practice, and reflection helps teachers to gain relevant knowledge and increase self-efficacy. We found that most preservice studies included opportunities to practice CS and reported increases in preservice teachers' content knowledge and computing self-efficacy. This result is especially encouraging, considering that most reported interventions were relatively short.

To analyze in-service teacher PD, we used Desimone's (2009) framework for improving the impact of teachers' PD. This framework adds coherence, collective participation, and sufficient duration as core features of effective PD and indicates

that knowledge/skills should increase and positive changes in attitudes and beliefs should occur. As with preservice CS teacher development, we found that even short-term PD can increase teachers' self-efficacy to teach computing. Many efforts included hands-on learning, with teachers creating their own programs and experimenting with robots. Developing teachers' computing content knowledge was the focus of most of these studies. In the few studies where teachers were able to apply their lessons, they reported greater growth in both their knowledge/skills and their attitudes and beliefs about computing education.

Desimone's (2009) framework further suggests that effective PD produces changes in teacher practice that promote student learning. To that end, improving teacher knowledge and self-efficacy is essential but insufficient. Teachers may understand and feel confident to teach CS, but lack motivation to do so. Teachers are under pressure to align their teaching with state standards and assessments; therefore, unless students are being assessed in CS, coherent CS standards are developed, or parents and students are demanding CS instruction, teachers may not take time to teach CS. School districts and state governments are increasingly adopting policies that require CS instruction (Code.org, 2019; Stanton et al., 2017). Such policies are driving teacher change, are needed to support teacher change, and are themselves supported by teacher education in CS.

References

Avalos, B. (2011). Teacher professional development in *Teaching and Teacher Education* over ten years. *Teaching and Teacher Education*, *27*, 10–20. https://doi.org/10.1016/j.tate.2010.08.007

Bandura, A. (1977). Self-efficacy: Toward a unifying theory of behavioral change. *Psychological Review*, *84*, 191–215. https://doi.org/10.1037/0033-295X.84.2.191

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

Bers, M. U., Seddighin, S., & Sullivan, A. (2013). Ready for robotics: Bringing together the T and E of STEM in early childhood teacher education. *Journal of Technology and Teacher Education*, *21*, 355–377.

Borko, H., & Putnam, R. T. (1995). Expanding a teacher's knowledge base: A cognitive psychological perspective on professional development. In T. R. Guskey & M. Huberman (Eds.), *Professional development in education: New paradigms and practices* (pp. 35–65). New York, NY: Teachers College Press.

Bower, M., & Falkner, K. (2015). Computational thinking, the notional machine, pre-service teachers, and research opportunities. In D. D'Souza & K. Falkner (Eds.), *Proceedings of the 17th Australasian Computing Education Conference* (pp. 37–46). Sydney, Australia: Australian Computer Society.

Bransford, J., Darling-Hammond, L., & LePage, P. (2005). Introduction. In L. Darling-Hammond & J. Bransford (Eds.), *Preparing teachers for a changing world: What teachers should learn and be able to do* (pp. 1–39). San Francisco, CA: Jossey-Bass.

Bureau of Labor Statistics. (2018a). *Computer and information technology occupations*. Retrieved from https://www.bls.gov/ooh/computer-and-information-technology/home.htm

Bureau of Labor Statistics. (2018b). *Occupations with the most job growth*. Retrieved from https://www.bls.gov/emp/tables/occupations-most-job-growth.htm

Carter, A., Cotten, S. R., Gibson, P., O'Neal, L. J., Simoni, Z., Stringer, K., & Watkins, L. S. (2014). Integrating computing across the curriculum: Incorporating technology into STEM education. In Z. Yang, H. H. Yang, D. Wu, & S. Liu (Eds.), *Transforming K–12 classrooms with digital technology* (pp. 165–192). Hershey, PA: IGI Global. https://doi.org/10.4018/978-1-4666-4538-7.ch009

Cetin, I. (2016). Preservice teachers' introduction to computing: Exploring utilization of scratch. *Journal of Educational Computing Research*, *54*, 997–1021. https://doi.org/10.1177/0735633116642774

Cetin, I., & Andrews-Larson, C. (2016). Learning sorting algorithms through visualization construction. *Computer Science Education*, *26*, 27–43. https://doi.org/10.1080/08993408.2016.1160664

Chang, Y.-h., & Peterson, L. (2018). Pre-service teachers' perceptions of computational thinking. *Journal of Technology and Teacher Education*, *26*, 353–374.

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology*, *76*, 1051–1058. https://doi.org/10.1037/0022-0663.76.6.1051

Code.org. (2019). *K–12 computer science policy and implementation in states*. Retrieved from Google Docs: https://docs.google.com/document/d/1J3TbEQt3SmIWuha7ooBPvlWpiK-pNVIV5uuQEzNzdkE

Coleman, L. O., Gibson, P., Cotten, S. R., Howell-Moroney, M., & Stringer, K. (2016). Integrating computing across the curriculum: The impact of internal barriers and training intensity on computer integration in the elementary school classroom. *Journal of Educational Computing Research*, *54*, 275–294. https://doi.org/10.1177/0735633115616645

Crick, T. (2017, April). *Computing education: An overview of research in the field*. Retrieved from Swansea University's Research Repository website: https://cronfa.swan.ac.uk/Record/cronfa43589

Darling-Hammond, L., Hyler, M. E., & Gardner, M. (2017, June). *Effective teacher professional development*. Retrieved from Learning Policy Institute website: https://learningpolicyinstitute.org/product/effective-teacher-professional-development-report

Department for Education. (2013, September 11). *Statutory guidance: National curriculum in England: Computing programmes of study*. Retrieved from https://www.gov.uk/government/publications/national-curriculum-in-england-

computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study

Desimone, L. M. (2009). Improving impact studies of teachers' professional development: Toward better conceptualizations and measures. *Educational Researcher*, *38*, 181–199. https://doi.org/10.3102/0013189X08331140

Desimone, L. M., Porter, A. C., Garet, M. S., Yoon, K. S., & Birman, B. F. (2002). Effects of professional development on teachers' instruction: Results from a three-year longitudinal study. *Educational Evaluation and Policy Analysis*, *24*, 81–112. https://doi.org/10.3102/01623737024002081

Dorn, B., & Tew, A. E. (2015). Empirical validation and application of the Computing Attitudes Survey. *Computer Science Education*, *25*, 1–36. https://doi.org/10.1080/08993408.2015.1014142

Enochs, L. G., & Riggs, I. M. (1990). Further development of an elementary science teaching efficacy belief instrument: A preservice elementary scale. *School Science and Mathematics*, *90*, 694–706. https://doi.org/10.1111/j.1949-8594.1990.tb12048.x

Ertmer, P. A., & Ottenbreit-Leftwich, A. T. (2010). Teacher technology change: How knowledge, confidence, beliefs, and culture intersect. *Journal of Research on Technology in Education*, *42*, 255–284. https://doi.org/10.1080/15391523.2010.10782551

Ertmer, P. A., Ottenbreit-Leftwich, A. T., Sadik, O., Sendurur, E., & Sendurur, P. (2012). Teacher beliefs and technology integration practices: A critical relationship. *Computers & Education*, *59*, 423–435. https://doi.org/10.1016/j.compedu.2012.02.001

Ertmer, P. A., Ottenbreit-Leftwich, A. T., & York, C. S. (2012). Exemplary technology-using teachers: Perceptions of factors influencing success. *Journal of Computing in Teacher Education*, *23*, 55–61.

Francis, K., Yáñez, G. A., Chapman, O., Cherkowski, G., Dodsworth, D., Friesen, S., … & Turner, J. (2018). Forming and transforming STEM teacher education: A follow up to Pioneering STEM education. In *Proceedings of 2018 IEEE Global Engineering Education Conference* (pp. 686–693). Piscataway, NJ: IEEE.

Garneli, V., Giannakos, M. N., & Chorianopoulos, K. (2015). Computing education in K–12 schools: A review of the literature. In *Proceedings of 2015 IEEE Global Engineering Education Conference* (pp. 543–551). Piscataway, NJ: IEEE.

Gibbons, S. J., Hirsch, L. S., Kimmel, H., Rockland, R., & Bloom, J. (2004, August). *Middle school students' attitudes to and knowledge about engineering*. Paper presented at the International Conference on Engineering Education, Gainesville, FL.

Google, Inc., & Gallup, Inc. (2016). *Trends in the state of computer science in U.S. K–12 schools*. Retrieved from http://services.google.com/fh/files/misc/trends-in-the-state-of-computer-science-report.pdf

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

Guskey, T. R. (2002). Professional development and teacher change. *Teachers and Teaching: Theory and Practice*, *8*, 381–391. https://doi.org/10.1080/135406002100000512

Guskey, T. R., & Yoon, K. S. (2009). What works in professional development? *Phi Delta Kappan*, *90*, 495–500. https://doi.org/10.1177/003172170909000709

Hammerness, K., Darling-Hammond, L., & Bransford, J. (with Berliner, D., Cochran-Smith, M., McDonald, M., & Zeichner, K.). (2005). How teachers learn and develop. In L. Darling-Hammond & J. Bransford (Eds.), *Preparing teachers for a changing world: What teachers should learn and be able to do* (pp. 358–389). San Francisco, CA: Jossey-Bass.

Hargreaves, A., & Fullan, M. (2000). Mentoring in the new millennium. *Theory Into Practice*, *39*, 50–56. https://doi.org/10.1207/s15430421tip3901_8

Heintz, F., Mannila, L., & Färngvist, T. (2016). A review of models for introducing computational thinking, computer science and computing in K–12 education. In *2016 IEEE Frontiers in Education Conference Proceedings* (pp. 1–9). Piscataway, NJ: IEEE.

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*, 411–435.

Hew, K. F., & Brush, T. (2007). Integrating technology into K–12 teaching and learning: Current knowledge gaps and recommendations for future research. *Educational Technology Research and Development*, *55*, 223–252. https://doi.org/10.1007/s11423-006-9022-5

Hoegh, A., & Moskal, B. M. (2009). Examining science and engineering students' attitudes toward computer science. In *39th Annual IEEE Frontiers in Education Conference Proceedings* (pp. 1–6). Piscataway, NJ: IEEE.

Holden, H., & Rada, R. (2011). Understanding the influence of perceived usability and technology self-efficacy on teachers' technology acceptance. *Journal of Research on Technology in Education*, *43*, 343–367. https://doi.org/10.1080/15391523.2011.10782576

Ilic, U., Haseski, H. I., & Tugtekin, U. (2018). Publication trends over 10 years of computational thinking research. *Contemporary Educational Technology*, *9*, 131–153. https://doi.org/10.30935/cet.414798

The Institute for the Advancement of Research in Education at AEL. (2004, April). *Review of the research: Nine components of effective professional development*. Retrieved from Texas Instruments' Educational and Productivity Solutions

Division website: https://education.ti.com/sites/us/downloads/pdf/research_iare_ael.pdf

International Society for Technology in Education. (2011). *ISTE standards for computer science educators*. Retrieved from https://www.iste.org/standards/for-computer-science-educators

International Society for Technology in Education. (2017). *ISTE standards for educators*. Retrieved from https://www.iste.org/standards/for-educators

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

Jaipal-Jamani, K. (2018). Developing pre-service teachers' self-efficacy and science knowledge through a scaffolded robotics intervention: A longitudinal study. In E. Langran & J. Borup (Eds.), *Proceedings of SITE 2018: Society for Information Technology & Teacher Education International Conference* (pp. 1913–1919). Waynesville, NC: Association for the Advancement of Computing in Education.

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology*, *26*, 175–192. https://doi.org/10.1007/s10956-016-9663-z

Jeon, Y., & Kim, T. (2017). The effects of the computational thinking-based programming class on the computer learning attitude of non-major students in the teacher training college. *Journal of Theoretical and Applied Information Technology*, *95*, 4330–4339.

K–12 Computer Science Framework. (n.d.). Glossary. Retrieved from https://k12cs.org/glossary/

K–12 Computer Science Framework. (2016). Retrieved from https://k12cs.org/wp-content/uploads/2016/09/K%E2%80%9312-Computer-Science-Framework.pdf

Kallia, M. (2017, November). *Assessment in computer science courses: A literature review*. Retrieved from King's College London website: https://royalsociety.org/~/media/policy/projects/computing-education/assessment-literature-review.pdf

Kang, H. S., Cha, J., & Ha, B.-W. (2013). What should we consider in teachers' professional development impact studies? Based on the conceptual framework of Desimone. *Creative Education*, *4*(4A), 11–18. https://doi.org/10.4236/ce.2013.44A003

Kaya, E., Yesilyurt, M. E., Newley, A. D., & Deniz, H. (2018). *Investigating computational thinking self-efficacy beliefs of pre-service elementary teachers*.

Poster session presented at the 2018 ASEE Annual Conference and Exposition, Salt Lake City, UT.

Kim, C., Kim, D., Yuan, J., Hill, R. B., Doshi, P., & Thai, C. N. (2015). Robotics to promote elementary education preservice teachers' STEM engagement, learning, and teaching. *Computers & Education*, *91*, 14–31. https://doi.org/10.1016/j.compedu.2015.08.005

Kundukulam, V. (2018, January 23). 4 ways to truly expand #CSforAll. *EdSurge*. Retrieved from https://www.edsurge.com/news/2018-01-23-4-ways-to-truly-expand-csforall

Lawless, K. A., & Pellegrino, J. W. (2007). Professional development in integrating technology into teaching and learning: Knowns, unknowns, and ways to pursue better questions and answers. *Review of Educational Research*, *77*, 575–614. https://doi.org/10.3102/0034654307309921

Lee, I. (2016). Reclaiming the roots of CT. *CSTA Voice: The Voice of K–12 Computer Science Education and Its Educators*, *12*(1), 3–4.

Lee, J., (2010). *The effects of girl-friendly teaching-learning program on software learning attitude and achievement* (Master's thesis). Korea National University of Education, Cheongju, South Korea.

Leonard, J., Mitchell, M., Barnes-Johnson, J., Unertl, A., Outka-Hill, J., Robinson, R., & Hester-Croff, C. (2018). Preparing teachers to engage rural students in computational thinking through robotics, game design, and culturally responsive teaching. *Journal of Teacher Education*, 69, 386–407. https://doi.org/10.1177/0022487117732317

Lockwood, J., & Mooney, A. (2018). Computational thinking in secondary education: Where does it fit? A systematic literary review. *International Journal of Computer Science Educational in Schools*, *2*(1), 41–60. https://doi.org/10.21585/ijcses.v2i1.26

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K–12? *Computers in Human Behavior*, *41*, 51–61. https://doi.org/10.1016/j.chb.2014.09.012

Ma, Y., Lai, G., Williams, D., Prejean, L., & Ford, M. J. (2008). Exploring the effectiveness of a field experience program in a pedagogical laboratory: The experience of teacher candidates. *Journal of Technology and Teacher Education*, *16*, 411–432.

Machi, L. A., & McEvoy, B. T. (2016). *The literature review: Six steps to success* (3rd ed.). Thousand Oaks, CA: Corwin Press.

Mannila, L., Nordén, L.-Å., & Pears, A. (2018). Digital competence, teacher self-efficacy and training needs. In *Proceedings of the 49th ACM Conference on International Computing Education Research* (pp. 78–85). https://doi.org/10.1145/3230977.3230993

Marcelino, M. J., Pessoa, T., Vieira, C., Salvador, T., & Mendes, A. J. (2018). Learning computational thinking and scratch at distance. *Computers in Human Behavior*, *80*, 470–477. https://doi.org/10.1016/j.chb.2017.09.025

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers College Record*, *108*, 1017–1054.

Mueller, J., Wood, E., Willoughby, T., Ross, C., & Specht, J. (2008). Identifying discriminating variables between teachers who fully integrate computers and teachers with limited integration. *Computers & Education*, *51*, 1523–1537. https://doi.org/10.1016/j.compedu.2008.02.003

New South Wales Education Standards Authority. (2017). *Quality of initial teacher education in NSW: Digital literacy skills and learning report: A report on teaching information and communication technologies in initial teacher education in NSW*. Sydney, Australia: NSW Education Standards Authority.

Ng, W. S. (2017). Coding education for kids: What to learn? How to prepare teachers? In L. Morris & C. Tsolakidis (Eds.), *Proceedings of ICICTE 2017: The International Conference on Information Communication Technologies in Education* (pp. 195–205). Rhodes, Greece: Southampton Solent University.

Odden, A. R., & Picus, L. O. (2014). *School finance: A policy perspective* (5th ed.). New York, NY: McGraw Hill.

Pajares, M. F. (1992). Teachers' beliefs and educational research: Cleaning up a messy construct. *Review of Educational Research*, *62*, 307–332. https://doi.org/10.3102/00346543062003307

Rich, K., Strickland, C., & Franklin, D. (2017). A literature review through the lens of computer science learning goals theorized and explored in research. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 495–500). https://doi.org/10.1145/3017680.3017772

Rich, P. J., Browning, S. F., Perkins, M., Shoop, T., & Yoshikawa, E. (2018). Coding in K-8: International trends in teaching elementary/primary computing. *TechTrends,* 1-19. https://doi.org/10.1007/s11528-018-0295-4

Rich, P. J., & Hodges, C. (Eds) (2017). *Emerging research, practice, and policy on computational thinking*. New York, NY: Springer.

Rich, P. J., Jones, B., Belikov, O., Yoshikawa, E., & Perkins, M. (2017). Computing and engineering in elementary School: The effect of year-long training on elementary teacher self-efficacy and beliefs about teaching computing and engineering. *International Journal of Computer Science Education in Schools*, *1*(1), 1-20.

Rich, P. J., Leatham, K. R., & Wright, G. A. (2013). Convergent cognition. *Instructional Science*, *41*(2), 431-453. doi:10.1007/s11251-012-9240-7?LI=true#page-1

Roberts, M., Prottsman, K., & Gray, J. (2018). Priming the pump: Reflections on training K–5 teachers in computer science. In *Proceedings of the 49th ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 723–728). https://doi.org/10.1145/3159450.3159560

Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*, *72*, 678–691. https://doi.org/10.1016/j.chb.2016.08.047

Román-González, M., Pérez-González, J.-C., Moreno-León, J., & Robles, G. (2016). Does computational thinking correlate with personality? The non-cognitive side of computational thinking. In *Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality* (pp. 51–58). https://doi.org/10.1145/3012430.3012496

Sadik, O., Ottenbreit-Leftwich, A., & Nadiruzzaman, H. (2017). Computational thinking conceptions and misconceptions: Progression of preservice teacher thinking during computer science lesson planning. In P. J. Rich & C. B. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 221–238). https://doi.org/10.1007/978-3-319-52691-1_14

Schanzer, E., Fisler, K., & Krishnamurti, S. (2018). Assessing *Bootstrap:Algebra* students on scaffolded and unscaffolded word problems. In *Proceedings of the 49th ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 8–13). https://doi.org/10.1145/3159450.3159498

Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2018). Technology and the mind: Does learning to code improve cognitive skills? In *Proceedings of the Technology, Mind, & Society 2018 Conference*. https://doi.org/10.1145/3183654.3183658

Shah, A. M., Wylie, C., Gitomer, D., & Noam, G. (2018). Improving STEM program quality in out-of-school-time: Tool development and validation. *Science Education*, *102*, 238–259. https://doi.org/10.1002/sce.21327

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

Smagorinsky, P., Shelton, S. A., & Moore, C. (2015). The role of reflection in developing eupraxis in learning to teach English. *Pedagogies: An International Journal*, *10*, 285–308. https://doi.org/10.1080/1554480X.2015.1067146

Somekh, B. (2008). Factors affecting teachers' pedagogical adoption of ICT. In J. Voogt & G. Knezek (Eds.), *International handbook of information technology in primary and secondary education* (pp. 449–460). https://doi.org/10.1007/978-0-387-73315-9_27

Stanton, J., Goldsmith, L., Adrion, W. R., Dunton, S., Hendrickson, K. A., Peterfreund, A., … Zinth, J. D. (2017). *State of the states landscape report: State-level policies supporting equitable K–12 computer science education*. Retrieved from Education Development Center website: https://www.edc.org/state-states-

landscape-report-state-level-policies-supporting-equitable-k-12-computer-science

STEM Learning and Research Center. (n.d.). *Instruments: Science Teaching Efficacy Belief Instrument (STEBI)*. Retrieved from Education Development Center website: http://stelar.edc.org/instruments/science-teaching-efficacy-belief-instrument-stebi

Toikkanen, T., & Leinonen, T. (2017). The Code ABC MOOC: Experiences from a coding and computational thinking MOOC for Finnish primary school teachers. In P. J. Rich & C. B. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 239–248). New York, NY: Springer https://doi.org/10.1007/978-3-319-52691-1_15

Tucker, A. (Ed.), Deek, F., Jones, J., McCowan, D., Stephenson, C., & Verno, A. (2003, October). *A model curriculum for K–12 computer science: Final report of the ACM K–12 task force curriculum committee*. New York, NY: Association for Computing Machinery.

Vannatta, R. A., & Fordham, N. (2004). Teacher dispositions as predictors of classroom technology use. *Journal of Research on Technology in Education*, *36*, 253–271.

Waite, J. (2017). *Pedagogy in teaching computer science in schools: A literature review*. Retrieved from King's College London website: https://royalsociety.org/~/media/policy/projects/computing-education/literature-review-pedagogy-in-teaching.pdf

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. In P. J. Rich & C. B. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 205–220). https://doi.org/10.1007/978-3-319-52691-1_13

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education*, *14*(1), 1–16. https://doi.org/10.1145/2576872

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. *Communications of the ACM*, *60*(4), 55–62. https://doi.org/10.1145/2994591

Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T. (2011). Introducing computational thinking in education courses. In *Proceedings of the 42nd ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 465–470). https://doi.org/10.1145/1953163.1953297

Appendix A
Summaries of Studies Related to K–6 Preservice Teacher Training for Computing, Coding, and CT

Authors, Year	Participants	Context	Duration, format	Focus	Learning Activities	Assessment/Data	Learning Outcomes	Attitude Outcomes
CS as Content								
Cetin, 2016	56 preservice, K–12 teachers	Programming languages course	30-hour, 6-week unit	Programming	Experimental group used Scratch to create games and animations; control group used C to solve programming problems	Achievement test; practice test; Computer Programming Attitude Scale (Cetin & Ozden, 2015); interviews	Students using Scratch performed significantly higher on parallel achievement and practice posttests than participants who used C	No significant difference between the two groups in attitude toward computer programming
Cetin & Andrews-Larson, 2016	58 preservice, K–12 teachers	Programming languages course	10-hour unit	Computer programming, sorting algorithms	Used Flash to construct visualizations, or animations, of sorting algorithms	Achievement test; Computer Programming Attitude Scale (Cetin & Ozden, 2015)	The experimental group scored significantly higher in achievement than the control group	No significant difference between the two groups in attitude toward computer programming
Jeon & Kim, 2017	110 preservice teachers	CT-based programming course vs. ICT skill-based course	15-week, 45-hour course	Computer programming, CT	Instruction; problem-based learning; constructing a responsive website	Pre/post computer learning attitude assessments (Lee, 2010)		Students in CT course had significantly higher gains in self-efficacy and attitude toward CS education
Ng, 2017	10 preservice, early childhood teachers	Early childhood education program, Hong Kong	Three workshops	Coding, Bee-bots	Lecture, practice, modeling, lesson planning, microteaching, discussion, collaboration	Learning package designed by students	Increased coding skills and ability to design coding activities aligned to learning theory	

Authors, Year	Participants	Context	Duration, format	Focus	Learning Activities	Assessment/Data	Learning Outcomes	Attitude Outcomes
Sadik et al., 2017	12 preservice teachers (11 K–6; 1 secondary)	Advanced computer education course	Several-week course unit	CT, coding, robotics	Collaboratively developed and taught a two-hour instructional activity	Student proposals, blog posts, video discussions, papers, and reflections	Increased understanding of CT, but misconceptions persisted	
Integrated CS								
Chang & Peterson, 2018	59 preservice, K–6 & special ed. teachers	Educational technology course	2-hour activity	CT, robotics, coding	Lecture, exploration, & sharing or reflection	Written reflection	Increased understanding of CT; teaching applications	Improved attitudes, perceptions of relevance
Jaipal-Jamani, 2018	36 preservice, K–8 teachers	Science methods course	Two 3-hour classes	Robotics, programming	Constructing and programming gears and LEGO WeDo robots	Pre/post assessments of interest, self-efficacy, and science content knowledge	Increased understanding of gears	Increased interest in robots & self-efficacy for teaching robotics
Jaipal-Jamani & Angeli, 2017	21 preservice, K–6 teachers	Science methods course	Two 3-hour classes	CT, robotics, programming	Modeling; constructing and programming LEGO WeDo robots	Pre/post assessments of interest, self-efficacy, and science content knowledge; programming activities	Increased knowledge of CT and gears	Increased interest in robots & self-efficacy for teaching robotics
Kaya, Yesilyurt, Newley, & Deniz, 2018	35 preservice teachers	Undergrad science teaching methods course	Six 90-minute classes	CT, robotics, coding	Instruction; robotics challenge with Lego Mindstorms; programmed in code.org; solved Zoombinis video game puzzles	Modified Science Teaching Efficacy Belief Instrument (STEBI-B; Enochs & Riggs, 1990; STEM Learning and Research Center, n.d.)		Self-efficacy increased significantly; outcome expectancy did not
Kim et al., 2015	16 preservice, K–6 teachers	STEM instruction course	3-week course unit	Robot assembly, programming	Assembled and programmed robots using My Robot Time and RoboRobo kits; developed lesson plans	Pre/posttests, surveys, interviews	No significant differences in pre and posttest scores for science, technology, engineering, or	Improved motivation, enjoyment, interest in science, and

Authors, Year	Participants	Context	Duration, format	Focus	Learning Activities	Assessment/Data	Learning Outcomes	Attitude Outcomes
							mathematical knowledge	interest in engineering
Ma, Lai, Williams, Prejean, & Ford, 2008	32 preservice, K–6 teachers	Technology integration course	12-hour training	Robotics	Instruction, programming activities; practice facilitating activities with children; collaborative reflection	Reflective journal entries and interviews	Increased knowledge and skills to facilitate learning	
Yadav, Mayfield, Zhou, Hambrusch, & Korb, 2014	357 preservice, K–12 teachers	Introductory psychology course	Two 50-minute classes	CT	Instruction, problem-solving, examples	CT quiz, Computing Attitude Questionnaire	Greater understanding of CT and CT pedagogy than control group.	No statistically significant difference in comfort or interest.

Appendix B
Reported Inputs and Outcomes in Preservice Studies

Authors, Year	Inputs						Outputs		
	Content Models	Teaching Models	Content Practice	Lesson Planning	Teaching Practice	Reflection	Increased Content Knowledge	Increased Pedagogical Knowledge	Changed Attitudes or Beliefs
CS as Content									
Cetin, 2016	X		X				X		
Cetin & Andrews-Larson, 2016	X		X				X		
Jeon & Kim, 2017			X						X
Ng, 2017		X	X	X	X		X	X	
Sadik et al., 2017				X	X	X	X		
Integrated CS									
Chang & Peterson, 2018		X	X			X	X	X	X
Jaipal-Jamani, 2018	X		X				X		X
Jaipal-Jamani & Angeli, 2017	X		X				X		X
Kaya et al., 2018			X						X
Kim et al., 2015			X	X					X
Ma et al., 2008		X			X	X	X	X	
Yadav et al., 2014	X	X	X	X		X	X	X	X

Appendix C
Articles Related to K–6 In-service Teacher Training for Computing, Coding, and CT

Authors, Year	Participants	Context	Duration, Format	Focus	Learning Activities	Assessment/Data	Learning Outcomes	Attitude Outcomes
CS as Content								
Bers, Seddighin, & Sullivan, 2013	25 early childhood educators	Free institute held in Massachusetts; participants from 7 states	3-day workshop	Robotics & programming	Lecture; discussion; KIWI robotics sets and CHERP programming software; curriculum design	Pre/post questionnaires to assess attitudes, self-efficacy, and knowledge; interviews	Significant increases in technology, pedagogy, and content knowledge	Significant improvement in technology self-efficacy and attitudes toward technology
Leonard et al., 2018	45 K–9 teachers	Wyoming, online graduate course	8-week course	Robotics, game design, & culturally responsive pedagogy (CRP)	Readings & discussion; built & programmed LEGO MindStorms robots; designed games	Pre/post CT attitude survey (Yadav, Zhou, Mayfield, Hambrusch, & Korb, 2011); Dimensions of Success rating tool (Shah, Wylie, Gitomer, & Noam, 2018); games assessed with rating rubric; CRP survey	Small gains in CT understanding—higher improvements for teachers who did game design than for those who did robotics.	Small improvements in CT attitudes—greater effect for game design than robotics.
Marcelino, Pessoa, Vieira, Salvador, & Mendes, 2018	7 K–12 teachers, 1 other participant	University of Coimbra, Portugal; online course	54-hour course	Scratch programming, CT, pedagogy	Individual and collaborative programming activities and project	Activity & project evaluations; Dr. Scratch	Improved CT knowledge and programming skill, but learning depth varied among participants.	
Roberts, Prottsman, & Gray, 2018	3,092 K–5 teachers and staff	University-driven workshops in Alabama and Indiana	1-day workshop	Computing, coding, CT	Teaching & observing using Code.org's Computer Science Fundamentals curriculum and additional CS content	Post-PD surveys provided by Code.org	Increased content and pedagogical knowledge	Improved self-efficacy, content knowledge, beliefs, and attitudes toward CS

Authors, Year	Participants	Context	Duration, Format	Focus	Learning Activities	Assessment/Data	Learning Outcomes	Attitude Outcomes
Toikkanen & Leinonen, 2017	501 K–9 teachers	Finland, online course	2-month MOOC	Teaching programming, CT	Instruction; programming in ScratchJr, Scratch, or Racket; online discussion	Pedagogical ideas shared in Padlet	Increased knowledge and skills to teach programming	Teachers overcame reservations and preconceptions.
Integrated CS								
P. J. Rich et al., 2017	27 K–6 teachers	Title I school, western U.S.	1 year, weekly, embedded PD	Integrating computing and engineering	Engineering challenges, Engineering is Elementary curriculum, computing lessons and activities, Scratch programming	Survey of self-efficacy & beliefs; semi-structured interviews		Significantly more positive technology self-efficacy and beliefs toward computing than comparison group
Carter et al., 2014	53 fourth- and fifth-grade teachers	Southeastern U.S., large, urban school district	5-day workshop; embedded PD	Integrating computing	Instruction, modeling, and lesson plan creation	Survey of computing attitudes and anxiety		Amount of training correlated with decreases in anxiety and improvements in attitude.
Coleman, Gibson, Cotten, Howell-Moroney, & Stringer, 2016	54 fourth- and fifth-grade teachers	Southeastern U.S., large, urban school district	5-day workshop; embedded PD	Integrating computing	Instruction, modeling, lesson plan creation, practice teaching, supported classroom integration	Survey; in-class observation; rating scale for preparedness & execution (ability to teach lesson without assistance)	Summer institute participants scored higher in preparedness and execution than other teachers.	Attitude positively influenced execution. Attitude and anxiety showed no impact on preparation.
Hestness, Ketelhut, McGinnis, & Plane, 2018	13 Grades 3–5 mentor teachers	Mentor teachers from 3 Maryland public school districts	2 half-day workshops, 6 weeks apart	Integrating CT into classroom practice	Learned CT concepts; collaboratively completed robotics challenges with LEGO MindStorms, KIBO, & Think & Learn Code-a-Pillar; discussed integration	Drawings, written reflections, and focus group interviews	New content and pedagogical content knowledge were integrated with previous professional knowledge.	