Teachers' Goals Predict Computational Thinking Gains in Robotics

Eben B. Witherspoon[1*], Christian D. Schunn[1]

eben.witherspoon@pitt.edu; schunn@pitt.edu

[1]*Learning Research and Development Center, University of Pittsburgh*

[*] Corresponding author: Eben B. Witherspoon. Address: Learning Research and Development Cetner, 3939 O'Hara St., Pittsburgh, PA 15260. Email: eben.witherspoon@pitt.edu

Abstract:

Purpose: Computational thinking (CT) is widely considered to be an important component of teaching generalizable computer science skills to all students in a range of learning environments, including robotics. However, despite advances in the design of robotics curricula that can teach CT, actual enactment in classrooms may often fail to reach this target. Understanding the various instructional goals teachers' hold when using these curricula may offer one potential explanation for disparities in outcomes.

Design: In this study, we examine results from *N*=206 middle school students' pre- and post-tests of computational thinking, attitudinal surveys, and surveys of their teacher's instructional goals, to determine if student attitudes and learning gains in computational thinking are related to the instructional goals their teachers endorsed while implementing a shared robotics programming curriculum.

Results: Our findings provide evidence that despite using the same curriculum, students showed differential learning gains on the computational thinking assessment when in classrooms with teachers who rated computational thinking as a more important instructional goal; these effects were particularly strong for women. Students in classroom with teachers who rated computational thinking more highly also showed greater maintenance of positive attitudes towards programming.

Originality/Value: While there is a growing body of literature regarding curricular interventions that provide computational thinking learning opportunities, this study provides a critical insight into the role that teachers may play as a potential support or barrier to the success of these curricula. Implications for the design of professional development and teacher educative materials that attend to teachers' instructional goals are discussed.

Keywords: computational thinking, robotics, programming, goals, teacher learning

Introduction

Computer science education is now widely considered to be an integral part of a well-rounded K-12 science, technology, engineering and mathematics (STEM) education. In the United States, the Computer Science for All initiative urges that computer science (CS) learning opportunities be provided not just within specialized elective classes or after-school clubs, but also in general education classes that offer these experiences to every student (Smith, 2016). In part, this policy shift is driven by a growing need for some base level of competence in computing for students to remain competitive in a job market that increasingly requires computational knowledge and skills, regardless of career trajectory. The U.S. Bureau of Labor Statistics (2017) predicts that the fastest growing careers in the coming decade are likely those that will require some degree of computational literacy, and the ability to use computers and programming logic to solve problems in a variety of applications. Educational researchers have sometimes used the term Computational Thinking (CT) to describe this particular 21st century skill. A canonical and complete definition of CT remains unsettled in the literature, leading some to advocate for the pragmatic approach of identifying core and peripheral concepts of CT; core aspects typically include decomposing problems, designing algorithmic solutions, and abstracting those solutions to multiple contexts (Voogt et al., 2015). Therefore, while many definitions of CT exist, most emphasize the importance of drawing on heuristics from the field of computer science to solve problems, and applying the knowledge and skills of computer science to problem solve across a variety of contexts and subjects (Barr and Stephenson, 2011; Wing, 2006).

Educational psychologists have studied the possible cognitive benefits of using computer science in K-12 to develop generalizable problem solving skills like computational thinking for

decades (Klahr and Carver, 1988; Pea and Kurland, 1984). In particular, specific computational thinking concepts from computer science such as "commands execute in sequence", "conditional statements determine if and when to pass control of the program to a new set of commands" and "programs repeat the commands a set number of times or until a condition is met" may be generalizable across programming languages and contexts . However, still relatively little is known about particular pedagogical practices that might be linked to effective instruction in this class of generalizable computational skills.

Robotics is one field that has been studied by educational psychologists as a learning environment that could potentially provide authentic opportunities to learn generalizable computer programming skills in an applied setting (Grover and Pea, 2013). Relatively recent advances in the design of educational technologies, informed by research in the learning sciences, have shown promise in providing students with generative learning experiences that may help develop the generalizable programming knowledge and skills prioritized by initiatives like CS for All (Lye and Koh, 2014). For example, block-based graphical programming languages can reduce syntax errors, allowing novice programmers to focus on the logic of their programs control structure (Kelleher and Pausch, 2005; Robins et al., 2010). Specific to robotics educational curricula, virtual simulations such as those used in the current study can reduce the mechanical errors often introduced by physical robots, thereby reducing the cognitive load of beginning programmers. Such simulated virtual curricula have been proven to teach programming as well as physical robotics, but more efficiently (Liu, Schunn, et al., 2013). Additionally, there is emerging evidence that certain features of these virtual robotics learning environments may be associated with measurable gains in generalizable Computational Thinking knowledge and skills (Witherspoon et al., 2017, 2018).

In the context of CS for All, educational robotics programs present themselves as a convenient option for school districts aiming to take up this initiative. In the last few decades, robotics programs have become almost ubiquitous in middle schools and high schools, both in elective after-school programs and more recently in compulsory education as the required technology becomes more broadly affordable (Melchior et al., 2005). However, in many K-12 settings, technology-rich programs like robotics are implemented within Technology Education ("Tech Ed") departments, which have historically focused on vocational training in specific and often localized industrial technologies, and are taught by teachers with varied training and experience in computer programming (Shields and Harris, 2007). Teachers in these classrooms often hold a broad range of teaching certifications, from Business, Computer and Information Technology to Career, Technical and Agricultural Education, and most teachers who are tasked with teaching robotics are unlikely to have received specific professional development targeted towards teaching either CS or CT (Ericson et al., 2008; Stephenson and Gal-Ezer, 2010). As use of robotics for teaching CT expands, limitations in teacher expertise may act as a bottleneck on positive learning outcomes.

The Critical Role of Teachers

It is well established that teachers play a critical role in student learning and attitudes; however, a variety of mechanisms may mediate these effects in technology-rich environments. Generally speaking, teacher beliefs about pedagogy and content interact with the written curriculum to determine ways that instructional materials are implemented, often creating disparities between curriculum as designed and curriculum as enacted (Remillard, 2005). Particularly in technology-rich environments, external barriers such as lack of training and hardware or software resources, and internal barriers such as confidence with the material,

valuation of technology, and beliefs about how students learn, could inform how teachers interpret and enact curriculum (Ertmer et al., 1999). Teachers are known to vary greatly in their understanding of CT and their attitudes towards integrating it into their classrooms, but CT educational opportunities are often limited to preservice computer science teachers (Yadav et al., 2014). Further, inquiry and project-based STEM reform curricula like those often found in robotics, which aim for students to construct knowledge through largely self-directed exploration, require substantial shifts in teaching practice from traditional, direct instruction methods (Schneider and Krajcik, 2002). Therefore, it is likely that large variance exists in the particular curricular focus and pedagogical approach to CT instruction across robotics programs, as well as in learning outcomes for students.

In addition to influencing achievement, variation in the way curricular materials are presented in robotics classrooms may also influence another important outcome of CS for All: students attitudes towards programming (Witherspoon et al., 2018). Maintaining students motivation to engage in programming activities may be particularly difficult in non-elective classrooms (e.g., in middle schools that require all students to take a course in technology education); research suggests that overall student valuation of STEM subjects tends to decline beginning in the middle school years (Wigfield and Eccles, 2000). However, it is possible for well-supported activities in middle school to maintain individual interest levels, which can predict long term, self-generated engagement through college (Harackiewicz and Hulleman, 2010; Hidi and Renninger, 2006). Other attitudinal interventions that can be linked to pedagogy, such as identity development through engagement in authentic tasks of the discipline, and fostering students beliefs about their ability to do programming, can also predict students achievement and continued participation in CS majors and careers (Collins, 2006; Engle, 2006;

Lent et al., 2016). Therefore, examining students' attitudinal responses to different pedagogical approaches while using a robotics programming curriculum could also offer important insights into effects on both students' achievement and persistence.

Teacher Goals

Understanding teachers' instructional goal setting could provide one useful framework for predicting how teachers activate resources in ways that differ from the designed curriculum. By "instructional goal", we mean a specific statement that expresses what students should learn in the language of a particular discipline, and is situated within a student-driven model of how learning progresses (Stein and Meikle, 2017). Teachers' goals that are explicitly stated and refined into sub-goals at the lesson planning stage may improve the design of instructional activities that increase student achievement (Hiebert et al., 2017). Research has also suggested that instructional goal setting may be an emergent process that is responsive to a particular context (Aguirre and Speer, 1999).

In learning environments like Tech Ed classrooms, where a relatively recent shift in focus to computing technology has led to the acceptance of a broader variety of teacher certifications, teacher rotation between multiple topical units, and a range of new tools and curricula, departmental goals can often be complex and ill-defined. It is likely that Tech Ed teachers hold multiple instructional goals simultaneously, and that those may at times conflict with the written curriculum, determining which goals are implemented in the classroom (Davis, Janssen, & Van Driel, 2016). Therefore, rather than circumventing these challenges with "teacher-proof" curricular materials, it is necessary for curricular designers to consider curricular enactment as a "local phenomenon that arises as a result of a number of factors, including…teachers' goals, local constraints, and teachers' pedagogical values" (Drake and Sherin, 2006). Curriculum

developers aiming to teach CT may benefit from understanding the goals endorsed by Tech Ed robotics teachers implementing their curriculum, to better provide strategies to deal with potentially competing goals. Additionally, understanding Tech Ed robotics teachers' instructional goals could aid in the design of professional development that ensures all teachers have the knowledge and skills needed to align their instructional activities with higher level curricular goals.

Therefore, while robotics curricular materials may be designed with intent to provide opportunities to learn Computational Thinking, these goals are often altered by teachers on the ground during moment-to-moment interactions with students. Particularly, in-service Tech Ed robotics teachers may hold alternate goals for their classrooms based on past experiences (i.e., general goals about problem solving vs. specific goals about computational thinking), and under the pressure of a complex and novel learning environment may be more likely to revert to prior pedagogical practices that are more familiar (i.e., focusing on performance outcomes like building the physical robot vs. learning outcomes like understanding computational concepts). This variation in goals can lead to variation in student learning by classroom, even when teachers have relatively similar experience, teach in similar learning contexts, and are using the exact same curricular materials.

A better understanding of the importance teachers place on the different goals they have in these classrooms may help predict when and how these differences in enactment may manifest, and the effect that they have on student learning. Importantly, this information will be useful for curriculum designers to account for in development of teacher instructional materials and professional development. In this study, we examine how teachers' ratings of the importance of instructional goals around CT in middle school robotics classrooms are related to student

learning of CT. Specifically, we were interested if we would find differences based on CT instructional goals for Tech Ed teachers using the same virtual robotics programming curriculum, suggesting that these goals may be contributing factors to discrepancies in enactment that produce variation in students Computational Thinking learning opportunities.

Methods

Sample

We examined the development of computational thinking in robotics classes in which all student in the school were enrolled, within schools across multiple regions of the United States. All human subjects research received Institutional Review Board (IRB) approval prior to the commencement of the study. The analyses presented here examine a sample of N=206 middle-school aged students (M_{age}=12.3, SD_{age}=1.1) within classrooms in four school districts, focusing on teachers with clearly differentiated instructional goals (described below). Students in this sample predominately identified as White (72%), with multi-racial (18%) and Asian (6%) making up the next two largest groups; the rest of the students either answered "Other", "I don't know" or were from a variety of groups (e.g., Indian/Middle Eastern, Native American/Pacific Islander) that each made up less than 1% of the data. Unlike elective robotics classes which are often predominately male, robotics classrooms in which all students in the school were enrolled consisted of a relatively evenly split by self-identified gender (51% female). Many of the students in these courses (69%) had some prior experience with robotics before, but the majority of students (77%) were engaging with this particular virtual robotics curriculum for the first time.

In addition to student assessments, we also distributed multiple rounds of weekly surveys to N=10 teachers across the US, which asked them to rate their instructional goals for their classes on a weekly basis. Overall, our response rate from the teacher surveys was about 47%. All of the responding teachers had earned a Master's degree, were certified in a range of specialties closely related to Technology Education (e.g., Business, Computers & Information Technology; Career and Technical Education, Technology Education), and had a relatively high number of years of teaching experience overall (M_{years}=12.6, SD_{years}=6.0). Additional details on the four teachers selected for further analysis are presented in a later section.

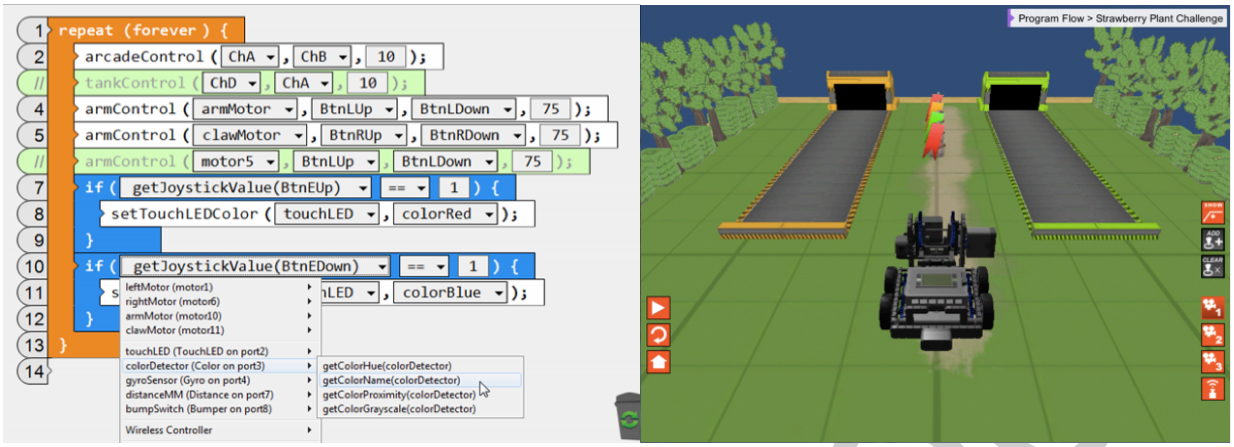
Curricular Materials

The robotics curriculum used here, developed by Carnegie Mellon University and Robomatter, involves a sequence of lessons in robotics programming utilizing a visual programming language, ROBOTC Graphical (see Figure 1A). On average, instruction with the curriculum ran for about 10 weeks, and included 24 mini-lessons across 4 units including topics both specific to robotics (i.e., basic movement, sensors, repeated decisions) as well as core computational thinking concepts (i.e., abstraction, decomposition, systems thinking). Earlier versions of a similar virtual robotics curriculum have been reported on in previous studies (see Witherspoon et al., 2017, 2018). The curricular materials incorporate elements which were designed to support efficient learning and transfer of generalizable computational skills: procedural scaffolds (worked examples, guided videos), dynamic mini-challenges, visual programming language, and Robot Virtual Worlds (RVW), a virtual robotics programming environment designed to emphasize the programming aspects of robotics, while maintaining student interest and engagement (see Figure 1B). These features reflect a constructionist

approach to instruction, in which learners build increasingly complex programmed solutions and construct an understanding of the requisite programming principles (Papert, 1980).

First, to provide a shared context for each unit, students are provided with a short introductory video to frame the subsequent lesson activities. These videos are learner-paced and present visual support together with a conversational narrative around the key concepts, to reduce extraneous processing and foster generative processing (Mayer, 2008). Partial scaffolding (Puntambekar and Hubscher, 2005) is introduced by way of questions to check students understanding, step-by-step instruction on a conceptually related robotics programming activity, and a brief post activity quiz to assess understanding, followed by the open-ended application of these skills within a game-like challenge in the virtual programming environment, allowing students to apply their knowledge more independently.

Students can iteratively test modular programmed solutions with simulated VEX IQ robots in a three-dimensional virtual platform. Finally, these solutions are "remixed and reused" (Brennan and Resnick, 2012) to complete more complex virtual challenges, in which learners must apply their previous programming knowledge to problem solving tasks that foreground computational thinking principles like abstraction, decomposition, and systems thinking. To solve these challenges, students used a programming language called ROBOTC Graphical (see Figure 1A) to develop programmed solutions. ROBOTC Graphical has a visual programming language interface, intended to allow students to focus on the broader logic of programming while deemphasizing the particular syntactic requirements of more traditional programming languages.



(A) (B)

Figure 1. Samples of the ROBOTC graphical programming language (A), and a RVW virtual robotics programming task (B). In this task, using if/else statements, loops, and sensors, students program the robot to sort flags onto the left or right conveyor belt based on the color, which is dynamically assigned.

By representing robotics challenges in a virtual environment, this curriculum offers affordances over physical robotics programs by reducing the potential frustration and distractors of mechanical error, enabling students to focus on higher-level computational principles of programming. While physical robots may have some advantages, a study by Liu et. al (2013) found that students using an earlier version of this technology achieved learning gains in programming content equivalent to students using physical robots, but in significantly less time. Further, simulating robot movement reflects an authentic engineering practice (see Michel, 2004), and virtual robots are also less expensive than physical ones, allowing the benefits of the curriculum to reach a broader population where the costs of physical robotics curricula can be prohibitive.

Measures & Procedures

Teacher instructional goals. In order to understand which instructional goals teachers were emphasizing in these classrooms, we distributed a weekly online survey to teachers throughout the semester in which they were using the curriculum. These surveys were developed through pilot studies consisting of pre-lesson goal setting activity conducted with a small group of local robotics teachers using the same virtual curriculum. From these pilot studies, we noted that only some teachers were setting goals related to core CT concepts that were included in the curriculum. These teachers included goals such as "students learn that in a conditional loop, the condition determines when/how long the commands repeat," while other teachers identified goals such as "students will complete lesson activities 1-3". The resulting surveys used in the current study asked teachers to rate the importance of a set of goals focused on specific computational thinking learning outcomes (e.g. "During class this week, my goal was that students would learn…that programs execute commands in sequence") on a 3-point Likert scale from (1) Least Important to (3) Most Important (see Appendix A for sample teacher goals measures). Additionally, teachers were asked to provide demographic information such as level of teaching experience, teaching certification, and prior exposure to the curriculum. These surveys were purposefully kept relatively brief to promote survey completion.

Overall, teachers were given nine opportunities to respond to the survey over the course of a semester. From the total group of ten teachers who received the survey, four teachers provided a sufficient number of responses ($n \geq 5$) across all items to generate a reasonably robust measure of their average rating of each goal, and so these four teachers were purposively selected for additional analysis. The four teachers selected for final analysis were all white, male, and had a similar level of teaching experience (M_{years}=13.8, SD_{years} = 3.0). Overall, teachers

tended to rate most goals as at least moderately important; based on the distribution of teachers' responses, we used a median split to group them into two categories: "Low CT", consisting of two teachers who had an average overall rating of computational thinking goals of 2.5 or below (M_{rating}=2.1, SD_{rating}=0.3) across 11 combined ratings, and "High CT", consisting of two teachers who had an average overall rating of computational thinking goals of 2.5 or higher (M_{rating}=2.8, SD_{rating}=0.4) across 12 combined ratings. In other words, teachers who typically rated the goals as only moderately important versus teachers who typically rated the goals as most important; this difference in ratings was a large effect size (Cohen's d= 2.2). Both High CT teachers held Technology Education certifications, while one Low CT teacher held a Business, Computers and Information Technology certification, and the other held both Career & Technical Education and Biology certifications. In each group, one teacher reported having approximately 4 years of experience with the curriculum, while the second teacher in each group was using the curriculum for the first time.

Computational thinking assessments. After grouping the four teachers based on their rating of CT goals, we then examined the pre- and post-test scores of students in each of these teachers' classrooms, to see if there were significant differences in learning as measured by the assessments of computational thinking for students in Low CT teachers' classrooms (n=57) and students in a High CT teachers' classrooms (n=149; see Table 1).

The primary outcome measure was an externally-created computational thinking assessment used as a post-test. It consisted of five multiple choice items that were adapted for a robotics context from the Exploring Computer Science - Principled Assessment of Computational Thinking (PACT) (Goode and Margolis, 2011). These assessments were

specifically created using evidence-centered design to assess knowledge, skills and attributes associated with computational thinking practices[1].

An alternative assessment was needed that could be used to verify equivalence of both general programming skills and computational thinking skills across classes before instruction, as well as avoid test-retest effects. We had previously developed such an assessment that contained programming and computational thinking items (see Appendix B for sample assessment items). These items were developed to target three core programming concepts common across a range of accepted frameworks of programming and computational thinking; sequences, conditions and iteration (see College Board, 2016; Computer Science Teachers Association, 2016; Bienkowski et al., 2015) and have been shown in prior work to be a reliable measure of students programming and computational thinking knowledge (see).

Our school district partners requested that we reduce class time required for an assessment that is only establishing equivalence at the class level. Therefore students received one of four randomly assigned sections of the computational thinking assessment at pre-test; each of the four sections of the pre-test consisted of five multiple choice items each. The overall average Armor's θ was $\theta = .44$ for the pre-test, and $\theta = .74$ for the post-test[2]. Relatively low theta values are common for relatively short assessments that are intended to cover a range of concepts. When corrected to account for possible attenuation of correlation caused by the measurement error, the pre-post correlation was $\rho = .60$ (see Fan, 2003).

Attitudes towards programming. Additionally, students also completed a short attitudinal survey prior to the pre and post-test exams, with 12 items that asked students about their interest,

[1] Sample items can be found at https://pact.sri.com/resources.html

[2] Armor's θ and polychoric correlations are similar to Cronbach's α and Pearson's correlations, respectively, but are more appropriate for binary data (item correct vs. item incorrect; see Panter et al., 1997)

competency beliefs, and development of identity in computer programming (see Appendix C for sample survey items). Different scales were used across these items as a strategy for slowing respondents down and getting them to read each item more closely, and to allow for different measures of intensity (i.e., frequency of experiences vs. strength of endorsement). Interest was gauged through four items (e.g., "I wonder about how computer programs work", Cronbach's α=.87), rated along a four-point Likert scale (e.g. "Never" to "Every Day"). Four items gauged level of identity as a programmer (e.g., "My family thinks of me as a programming person", α=.88), rated along a four-point Likert scale (e.g. NO! to YES!). Competency beliefs were gauged through four items (e.g., "I am sure I could do advanced work in programming", α=.83) rated along a six-point Likert scale (e.g. "Strongly Disagree" to "Strongly Agree"). Based on a prior pilot survey, which suggested that students struggled to accurately rate their competency prior to obtaining some knowledge of the content, only these items were measured using retrospective pre- items (i.e., students were asked at post to rate both their competency at the beginning of the curriculum and their competency now; see Pratt et al., 2000). Attitudinal measures at both pre and post were significantly correlated with each other, but not so high as to be redundant measures. For ease of interpretation across these different scales, prior to analyses all attitudinal measures were converted to a proportion, with the lowest rating as 0 and the highest rating as 1.

Table 1.

Descriptive statistics of student and teacher background characteristics and pre-test levels for High CT and Low CT classroom groups (SD in parentheses), along with the t-test contrast of the pre-test group means and the 95% CI of the difference in means for each measure.

Measures	**Teacher Goal Groups**		*t*	95% CI
	CT Low (N=57)	CT High (N=149)		
Teacher characteristics†				
Teacher rating of CT	2.1 (.30)	2.8 (.38)	-	-
Teacher exp. (years)	14.0 (1.4)	13.5 (4.9)	-	-
Student characteristics				
Student robotic exp.	33%	30%	-0.5	-18%, 10%
Student CS2N exp.	9%	28%	-2.9**	-31%, -6%
Student age (years)	11.7 (1.2)	12.5 (1.0)	-5.2**	-1.2, -0.6
Student assessments				
Pre-test	2.2 (1.2)	2.3 (1.1)	0.7	-0.21, 0.46
Student surveys				
Competency Beliefs	.53 (.20)	.51 (.20)	0.6	-0.04, 0.08
Identity	.59 (.13)	.52 (.19)	1.5	-0.02, 0.17
Interest	.66 (.15)	.60 (.19)	1.2	-0.04, 0.15

*Note. **p<.01*

†*For teacher data, a dash (-) is shown in place of t-statistics and 95%CI because of low teacher sample size*

Analyses

Average pre-test scores of students in the two High CT teachers' classrooms were compared against the average pre-test scores of students in the two Low CT teachers' classrooms using a simple t-test, as well as other teacher and student characteristics to establish that the High CT and Low CT groups were comparable. Post-test scores were analyzed using ANCOVA, comparing differences in average post-test scores in the two groups while controlling for the pre-test score, age, and curricular experience, to increase statistical power by accounting for individual differences in students' pre-tests, and to account for slight differences in pre-group

composition on those variables. Finally, motivation variables were also measured using an ANCOVA of post-survey scores, controlling for pre-survey scores, age and curriculum experience.

Results

We first examined whether or not initially the two groups of students in the Low CT and High CT classrooms were relatively comparable on the assessment of computational thinking. A Levene's robust test for homogeneity of variance showed that there were no significant differences in variance between the two Low CT and High CT groups at pre-test, $F(1, 204)<1$, $p=.79$. Further, no significant differences were found in pre-test scores ($t=-0.75$, $p=.45$, $d=0.11$), between students in a classroom taught by a teacher with a Low CT rating ($M_{score}=2.2$, $SD_{score}=1.2$) or students in a classroom taught by a teacher with a High CT rating ($M_{score}=2.3$, $SD_{score}=1.1$; see Figure 1).

Critically, on the post-test, being in a classroom taught by a High CT rating teacher was associated with significantly higher scores ($M_{score}=2.6$, $SD_{score}=1.3$) than being in a classroom with a teacher who gave a Low CT rating ($M_{score}=2.1$, $SD_{score}=1.4$; $t= -2.67$, $p<.01$, $d=0.41$; see Figure 2). Thus, we have evidence of differential gains by teacher goals even when the same curriculum is being used.

However, the two groups were not fully equivalent by background. To account for small differences found in age and prior experience with the curriculum between the Low CT and High CT groups, an ANCOVA was conducted on post-test scores, controlling for pre-test scores, age, and prior experience with the curriculum. Even with these controls, students in the High CT group showed higher mean post-test scores than students in the Low CT group, $F(4,198)=2.90$,

p=.06, although these differences were no longer statistically significant, and the effect size was reduced to d=0.29, meaning the two distributions overlap approximately 88% (see Figure 2).

Importantly, aligning with the priority of CS for All in a robotics classroom in which all student in the school are enrolled, our results also show that only in High CT classrooms, girls had a significantly higher score [$F(4,141)=3.95$, $p<.05$, d=0.30] on the post-test (M_{score}=2.8, SE_{score}=0.1) than boys (M_{score}=2.4, SE_{score}=0.2), even when controlling for pre-test scores, age and curriculum experience. Thus, while in our sample significant differential gains in CT between the two groups were not found overall when including these additional controls, having a robotics teacher that endorsed CT goals shows significantly higher learning gains for women relative to men.

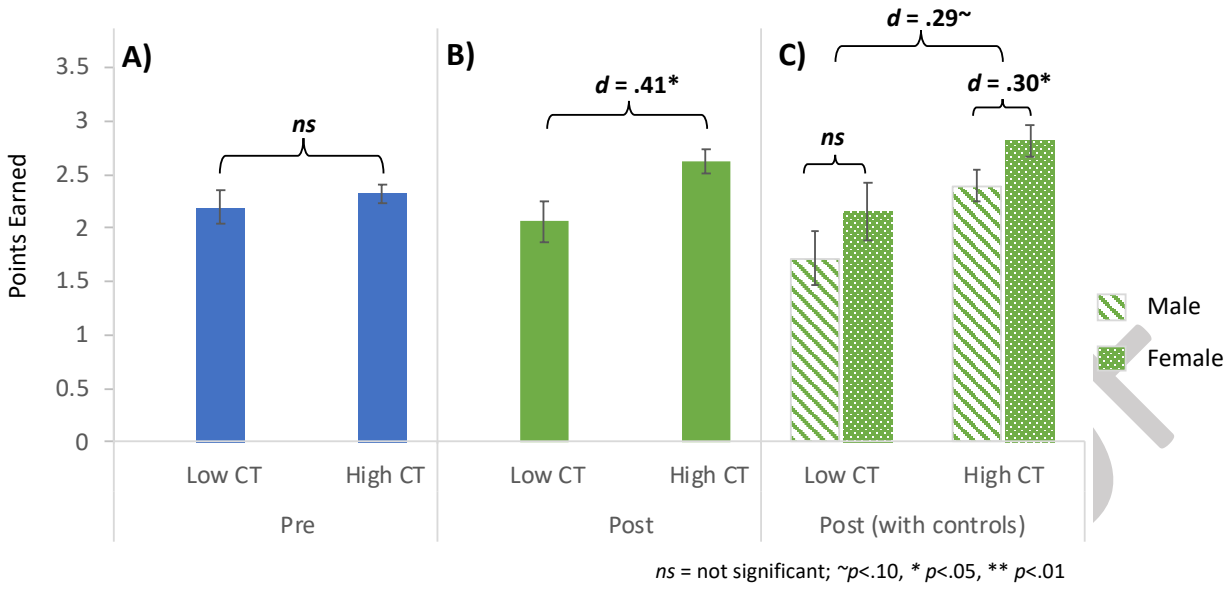


Figure 2. Differences between student scores by teacher rating of computational thinking goals (with SE bars), on A) pre-test and B) post-test, and C) post-test with controls for pre-test, age and prior experience.

For the final set of analyses, we examined differences in attitudinal measures for students in classroom with Low or High CT teachers. Overall, at pre- there were no significant differences between the two groups in Competency Beliefs (t = 0.62, p=.54), Identity (t = 1.2, p =.23) or Interest (t = 0.70, p = .48). At post, an ANCOVA revealed that while there were no significant differences between the two groups in Competency Beliefs [F(4,192)=1.22, p=.27, d = .16], the High CT group had significantly higher post-survey scores in both Identity [F(4,108)=6.73, p<.05, d=0.59] and Interest [F(4,108)=10.88, p<.01, d=0.71], when controlling for pre-survey, age and curriculum experience (see Figure 3). Importantly, these higher scores represent a relative maintenance of programming Identity and Interest from pre-test scores for those students in the High CT group, while students in the Low CT group largely experienced

significant declines in both programming Identity (t=-2.81, p<.05, d=.41) and Interest (t=-3.74, p<.01, d=.46).

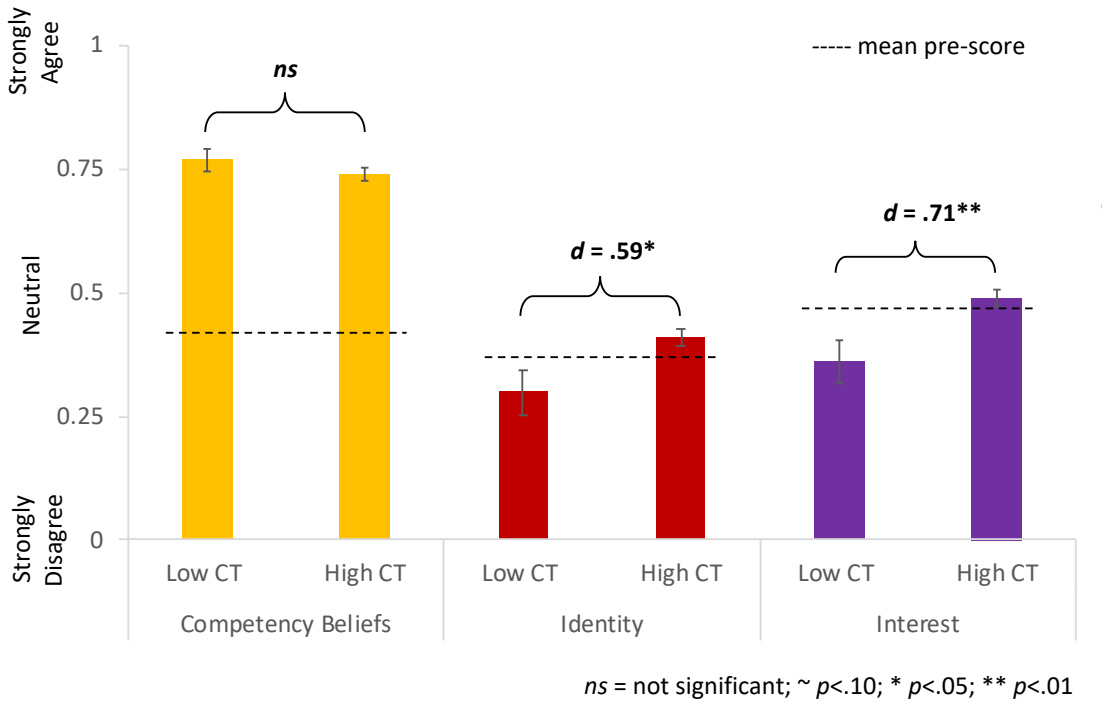


Figure 3. Post-motivation by teacher rating of computational thinking goals (with SE bars), controlling for students' pre-motivation scores (mean shown as dotted line), age, and curriculum experience.

Discussion

Overall, our results show that when teachers endorsed computational thinking as a critical instructional goal, their students had gains in computational thinking and also had greater maintenance of positive attitudes towards programming. Particularly, girls in mixed-gender robotics classrooms with a teacher who endorsed computational thinking instructional goals

outperformed boys, suggesting that this re-framing of the instructional focus of these traditionally male-dominated learning environments may help support achievement in programming for young women. Importantly, these differences in outcomes were found across teachers with similar experience and who were implementing the same virtual robotics curriculum. These findings suggest the key role instructional goals play in the development of Computational Thinking, similar to the mathematics education literature which propose that teachers goals act as a "north star", guiding a variety of instructional decisions (Stein and Meikle, 2017). Further, this study lays the foundation for future work examining how the diverse goals held by educators who teach Computational Thinking, in a broad range of learning environments, may determine how designed curriculum materials are adapted during curricular enactment. While the current study did not examine the specific curricular adaptations that were made, this study makes clear that an understanding of the instructional goals endorsed by the teacher is a significant contributor to student learning outcomes, and one that must be accounted for in the design of curriculum. Including teacher educative materials that provide insight into the design of lesson activities, identify aspects of enactment that are critical to the learning goal, and develop own teachers' capacity to design CT lessons, may help teachers avoid adaptations that reduce opportunities to learn computational thinking (Brown and Edelson, 2003; Davis and Krajcik, 2005).

Additionally, these findings suggest a need for ongoing professional development support for teachers that not only provides instruction on the use of the materials, but also explicitly attends to the goals of the curriculum, and potential areas where these goals may come into conflict with goals held by the teacher. Prior research has suggested that goal-coherent professional development can play a key role in reducing ambiguity and uncertainty and

supporting teachers sensemaking around new curricular initiatives where conflicting goals may exist (Allen and Penuel, 2015). This may be particularly important in the expanding range of educational programs like robotics in Technology Education environments, which are often tasked with incorporating computer science and Computational Thinking into their ongoing curriculum. The demands of these new initiatives often represent a large shift from the prior pedagogical approaches and instructional goals teachers in these spaces are familiar with (Schneider and Krajcik, 2002). Without adequate attention paid to the ways in which these goals may diverge from those already in place, initiatives like CS for All and innovative curricular reforms may experience a bottleneck in their ability to see the desired gains in student learning.

Limitations

The inferences that can be drawn from this study are limited by a number of factors. First, the analyses conducted are correlational in nature and there was no random assignment to experimental condition. Therefore, we cannot be certain whether the combination of exposure to the curriculum and the teacher instructional goals are in fact causing the observed differences in scores, or if other unobserved factors may be contributing to the larger gains for students in the High CT group. In a related way, due to our limited student and teacher sample sizes, analyses were unable to account for nesting within schools or classrooms. We are therefore uncertain that there are not school-level differences that may be contributing to differences in student gains. Another limitation with the current data was the relatively low correlation between our pre- and post-tests due to the practical necessity of a limited number of assessment items. The error introduced when these imperfectly correlated pre-tests were included as a control resulted in our statistical tests for knowledge gains being underpowered.

Second, due to the distributed nature of the classrooms around the US, we did not have observational measures of instruction, and therefore we do not know what teachers did to produce the changes in outcome. We also were unable to interview participating teachers to uncover how prior teacher certification programs and professional development opportunities may have influences their instructional goals. However, pilot interviews with local robotics teachers using this virtual curriculum indicate that although all teachers identified "problem solving and learning how to think" as core instructional goals, only those who selected CT goals similar to those in our surveys would enact lessons in ways that emphasized CT relevant features of the curricular activities. For example, while enacting an activity in which students program a robot to move a row of boxes, a teacher who selected a student learning goal that "conditional statements determine when to pass control of the program to a new set of commands" directed students' to think about using conditions to generate a program that allows the robot to account for different distances between the boxes. Future research should explore the ways in which teachers introduce activities, guide class discussion, and respond to student questions/struggles as possible vehicles of the effects of teacher goals on student learning outcomes (Stein et al., 2008; Stein and Meikle, 2017).

Conclusion

While prior studies using a similar virtual robotics curriculum have demonstrated that students may gain generalizable programming knowledge and skills from these learning experiences, here we show that even within similar classrooms using the exact same curriculum, differences may appear, and that teachers' instructional goals may be a significant contributor to these differences. Future work would benefit from gathering a larger teacher sample which would allow us to statistically account for different contextual factors, such as nesting effects

within different schools and how certification may play a role in robotics teachers' conceptualization of their instructional goals. Further, additional development of the survey of instructional goal setting, and qualitative interviews and classroom observations with teachers, could provide additional insight into the mechanisms through which these goals manifest in classroom activities, how teachers conceptualize goals around computational thinking in the classroom, and what framing of these goals may be most productive for teaching students generalizable programming knowledge and skills.

Acknowledgements

This work was supported by a grant from the National Science Foundation, Division of Research on Learning in Formal and Informal Settings (DRL 1418199). The opinions are those of the authors and do not represent the policies of the funding agency. We would also like to acknowledge Ross Higashi and Josh Jarvis for their consultation and support with the development and design of the curricular materials, and Mary Kay Stein for her insights and comments on an earlier draft. This research has been approved by the Human Research Protection Office at the University of Pittsburgh.

References

Aguirre, J. and Speer, N.M. (1999), "Examining the Relationship Between Beliefs and Goals in Teacher Practice", *The Journal of Mathematical Behavior*, Vol. 18 No. 3, pp. 327–356.

Allen, C.D. and Penuel, W.R. (2015), "Studying Teachers' Sensemaking to Investigate Teachers' Responses to Professional Development Focused on New Standards", *Journal of Teacher Education*, Vol. 66 No. 2, pp. 136–149.

Barr, V. and 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*, Vol. 2 No. 1, pp. 48–54.

Bienkowski, M., Snow, E., Rutstein, D. and Grover, S. (2015), *Assessment Design Patterns for Computational Thinking Practices in Secondary Computer Science : A First Look*, Menlo Park, CA, available at: http://pact.sri.com/resources.html

Brennan, K. and Resnick, M. (2012), "New frameworks for studying and assessing the development of computational thinking", *Annual American Educational Research Association Meeting, Vancouver, BC, Canada*, pp. 1–25.

Brown, M. and Edelson, D.C. (2003), *Teaching as Design: Can We Better Understand the Ways in Which Teachers Use Materials so We Can Better Design Materials to Support Their Changes in Practice?*, *Evanston, IL: The Center for Learning Technologies in Urban Schools*, available at: http://www.inquirium.net/people/matt/teaching_as_design-Final.pdf

College Board. (2016), *AP Computer Science Principles*, New York, NY, available at: https://apstudent.collegeboard.org/apcourse/ap-computer-science-principles/about-the-exam

Collins, A. (2006), "Cognitive Apprenticeship", in Sawyer, R.K. (Ed.), *The Cambridge Handbook of the Learning Sciences*, Cambridge University Press, Cambridge, UK, pp. 47–

60.

Computer Science Teachers Association. (2016), *Interim CSTA K-12 Computer Science Standards*, New York, NY, available at: http://www.csteachers.org/page/standards

Davis, E.A., Janssen, F.J.J.M. and Van Driel, J.H. (2016), "Teachers and science curriculum materials: where we are and where we need to go", *Studies in Science Education*, Routledge, Vol. 7267 No. May, pp. 1–34.

Davis, E.A. and Krajcik, J.S. (2005), "Designing Educative Curriculum Materials to Promote Teacher Learning", *Source: Educational Researcher*, Vol. 34 No. 3, pp. 3–14.

Drake, C. and Sherin, M.G. (2006), "Practicing Change: Curriculum Adaptation and Teacher Narrative in the Context of Mathematics Reform", *Curriculum Inquiry*, Vol. 36 No. 2, pp. 154–187.

Engle, R.A. (2006), "Framing Interactions to Foster Generative Learning: A Situative Explanation of Transfer in a Community of Learners Classroom", *The Journal of the Learning Sciences*, Vol. 15 No. 4, pp. 451–498.

Ericson, B., Armoni, M., Gal-Ezer, J., Seehorn, D., Stephenson, C. and Trees, F. (2008), *Ensuring Exemplary Teaching in an Essential Discipline: Addressing the Crisis in Computer Science Teacher Certification*, New York, NY.

Ertmer, P.A., Paul, A., Molly, L., Eva, R. and Denise, W. (1999), "Examining Teachers ' Beliefs About the Role of Technology in the Elementary Classroom", *Journal of Research on Computing in Education*, Vol. 32 No. 1, pp. 54–72.

Fan, X. (2003), "Two approaches for correcting correlation attenuation caused by measurement error: Implications for research practice", *Educational and Psychological Measurement*, Vol. 63 No. 6, pp. 915–930.

Goode, J. and Margolis, J. (2011), "Exploring Computer Science", *ACM Transactions on Computing Education*, Vol. 11 No. 2, pp. 1–16.

Grover, S. and Pea, R. (2013), "Computational Thinking in K-12: A Review of the State of the Field", *Educational Researcher*, Vol. 42 No. 1, pp. 38–43.

Harackiewicz, J.M. and Hulleman, C.S. (2010), "The Importance of Interest: The Role of Achievement Goals and Task Values in Promoting the Development of Interest", *Social and Personality Psychology Compass*, Vol. 4 No. 1, pp. 42–52.

Hidi, S. and Renninger, K.A. (2006), "The Four-Phase Model of Interest Development", *Educational Psychologist*, Vol. 41 No. 2, pp. 111–127.

Hiebert, J., Morris, A.K. and Spitzer, S.M. (2017), "Diagnosing Learning Goals: An Often-Overlooked Teaching Competency", in Lueders, T., Philipp, K. and Lueders, J. (Eds.), *Diagnostic Competence of Mathematics Teachers*, 11th ed., Springer International, Cham, Switzerland, pp. 193–206.

Kelleher, C. and Pausch, R. (2005), "Lowering the Barriers to Programming : a survey of programming environments and languages for novice programmers", *Science*, Vol. 37 No. 2, pp. 83–137.

Klahr, D. and Carver, S.M. (1988), "Cognitive objectives in a LOGO debugging curriculum: Instruction, learning, and transfer", *Cognitive Psychology*, Vol. 20 No. 3, pp. 362–404.

Lent, R.W., Miller, M.J., Smith, P.E., Watford, B.A., Lim, R.H. and Hui, K. (2016), "Social cognitive predictors of academic persistence and performance in engineering: Applicability across gender and race/ethnicity", *Journal of Vocational Behavior*, Elsevier Inc., Vol. 94 No. 0827470, pp. 79–88.

Liu, A., Newsom, J., Schunn, C. and Shoop, R. (2013), "Students Learn Programming Faster

through Robotic Simulation", *Tech Directions*, Vol. 72 No. march, pp. 16–19.

Liu, A., Schunn, C.D., Flot, J. and Shoop, R. (2013), "The role of physicality in rich programming environments", *Computer Science Education*, Routledge, Vol. 23 No. 4, pp. 315–331.

Lye, S.Y. and 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*, Elsevier Ltd, Vol. 41, pp. 51–61.

Mayer, R.E. (2008), "Applying the science of learning: evidence-based principles for the design of multimedia instruction.", *The American Psychologist*, Vol. 63 No. 8, pp. 760–769.

Melchior, A., Cohen, F., Cutter, T. and Leavitt, T. (2005), *More than Robots: An Evaluation of the FIRST Robotics Competition Participant and Institutional Impacts*, Waltham, MA.

Michel, O. (2004), "Webots TM : Professional Mobile Robot Simulation", *International Journal of Advanced Robotic Systems*, Vol. 1 No. 1, pp. 39–42.

Panter, A.T., Swygert, K.A., Dahlstrom, W.G. and Tanaka, J.S. (1997), "Factor Analytic Approaches to Personality Item-Level Data", *Journal of Personality Assessment*, Vol. 68 No. 3, pp. 561–589.

Papert, S. (1980), *Mindstorms*, Basic Books, Inc., New York, NY, available at: http://dl.acm.org/citation.cfm?id=1095592

Pea, R.D. and Kurland, D.M. (1984), "On the cognitive effects of learning computer programming", *New Ideas in Psychology*, Vol. 2 No. 2, pp. 137–168.

Pratt, C.C., McGuigan, W.M. and Katzev, A.R. (2000), "Measuring Program Outcomes : Using", *American Journal of Evaluation*, Vol. 21 No. 3, pp. 341–349.

Puntambekar, S. and Hubscher, R. (2005), "Tools for scaffolding students in a complex learning

environment: What have we gained and what have we missed?", *Educational Psychologist*, Vol. 40 No. 1, pp. 1–12.

Remillard, J.T. (2005), "Examining Key Concepts in Research on Teachers' Use of Mathematics Curricula", *Review of Educational Research*, Vol. 75 No. 2, pp. 211–246.

Robins, A., Rountree, J. and Rountree, N. (2010), "Learning and Teaching Programming : A Review and Discussion", *Computer Science Education*, Vol. 13 No. 2, pp. 137–172.

Schneider, R.M. and Krajcik, J. (2002), "Supporting Science Teacher Learning : The Role of Educative Curriculum Materials", *Journal of Science Teacher Education*, Vol. 13 No. 3, pp. 221–245.

Shields, C.J. and Harris, K. (2007), "Technology Education : Three Reasons Stereotypes Persist", *Journal of STEM Teacher Education*, Vol. 44 No. 2, pp. 60–72.

Smith, M. (2016), "Computer Science For All", *The White House Blog*, pp. 1–13.

Stein, M.K., Engle, R. a., Smith, M.S. and Hughes, E.K. (2008), "Orchestrating Productive Mathematical Discussions: Five Practices for Helping Teachers Move Beyond Show and Tell", *Mathematical Thinking and Learning*, Vol. 10 No. 4, pp. 313–340.

Stein, M.K. and Meikle, E. (2017), "The nature and role of goals in mathematics education", in Spangler, D. and Wanko, J. (Eds.), *Research Companion to Principles to Action*, National Council of Teachers of Mathematics, Reston, VA, pp. 1–11.

Stephenson, C. and Gal-Ezer, J. (2010), "Computer Science Teacher Preparation is Critical", *ACM Inroads*, Vol. 1 No. 1, pp. 61–66.

U.S. Bureau of Labor Statistics. (2017), *Employment Projections: Occupational Projections and Worker Characteristics*, available at: https://www.bls.gov/emp/ep_table_107.htm

Voogt, J., Fisser, P., Good, J., Mishra, P. and Yadav, A. (2015), "Computational thinking in

compulsory education: Towards an agenda for research and practice", *Education and Information Technologies*, Vol. 20 No. 4, pp. 715–728.

Wigfield, A. and Eccles, J.S. (2000), "Expectancy–Value Theory of Achievement Motivation", *Contemporary Educational Psychology*, Vol. 25 No. 1, pp. 68–81.

Wing, J.M. (2006), "Computational thinking", *Communications of the ACM*, Vol. 49 No. 3, p. 33.

Witherspoon, E.B., Higashi, R.M., Schunn, C.D., Baehr, E.C. and Shoop, R. (2017), "Developing computational thinking through a virtual robotics programming curriculum", *ACM Transactions on Computing Education*, Vol. 18 No. 1, available at:https://doi.org/10.1145/3104982

Witherspoon, E.B., Schunn, C.D., Higashi, R.M. and Shoop, R. (2018), "Attending to structural programming features predicts differences in learning and motivation", *Journal of Computer Assisted Learning*, Vol. 34, pp. 115–128.

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S. and Korb, J.T. (2014), "Computational Thinking in Elementary and Secondary Teacher Education", *ACM Transactions on Computing Education*, Vol. 14 No. 1, pp. 1–16.

Appendix A. Sample teacher goal items

Rate the following goals as Least Important to Most Important to your class this week. Then, indicate how often you spent time during class on the following topics.

You do not need to marks goals that are not applicable. If there are other goals you had that are not listed, use an "Other" box and briefly describe them.

"During class this week, my goal was that students would learn..."

	Least Important	Important	Most Important
That programs execute each command in order from top to bottom, unless otherwise directed.	O	O	O
That programs use conditional statements to determine if and when to pass control of the program to a new set of commands.	O	O	O
That programs will repeat the commands inside a looping structure either a set number of times, or until a condition is met.	O	O	O

Appendix B. Sample assessment items

Table A1. An overview of the dimensions and programming concepts in the bank of items used for the pre-assessment.

Dimensions	**Items**	**Concepts**	**Items**	**Example content**
Robot Programming	10	Sequences	*3*	*What sequence of movements will get the robot to the end of the maze?*
		Conditions	*4*	*At what distance sensor value will the robot stop moving?*
		Loops	*3*	*Which actions will the robot repeat if the bumper sensor is pushed in?*
Computational Thinking	10	Sequences	*3*	*Will the removal of this line of the program change the display on a heart monitor?*
		Conditions	*4*	*At what combination of blood pressure readings will this heart monitor emit an alarm?*
		Loops	*3*	*Which of these two programs will identify the correct blood pressure in the least number of iterations?*

Sample robotics programming item

Take a look at the program plan below. How will each individual line of code be run once it is programmed?

Line 1: Move forward for 5 seconds, at 100% speed
Line 2: Turn left 1 rotation, at 50% speed
Line 3: Move forward for 5 seconds, at 50% speed
Line 4: Turn right 1 rotation, at 50% speed

Select one:

- O Only the first command runs
- O The commands are run in order according to their line numbers
- O All commands run at once
- O The commands are run in a random order

Which of the following is true about conditions?

Select one:

- O They must always end up either true or false
- O They represent decision-making logic in a program
- O You can write a condition that is always true or always false
- O All of the above

Sample computational thinking item

Scenario: Personal Fitness Devices
Personal fitness devices use electronic sensors to continuously monitor and track data about a user's heath such as steps taken, calories burned, and heart rate.

The BP-Sure company is developing a new feature for their fitness device that also measures the user's blood pressure, using sensors that detect a user's heartbeat. When the heart pushes blood through the arteries, the device records "Pressure 1", and when the heart is resting, the device records "Pressure 2".

The device can determine if a user's blood pressure is in the Normal, Medium or High range, by comparing blood pressure readings to the chart below.

Use the chart below to answer questions #19, #20 and #21.

Blood Pressure	Pressure 1 (p1)		Pressure 2 (p2)
Normal BP	p1 <= 120	**AND**	p2 <= 80
Medium BP	121 <= p1 <= 139	**AND**	81 <= p2 <= 89
High BP	p1 >= 140	**OR**	p2 >= 90

A new programmer on the team writes the following series of steps to determine the display when a user is in the "Normal BP" range:

```
(Line 1)    IF (p1 <= 120 AND
(Line 2)       p1 <= 121 AND
(Line 3)       p2 <= 80  AND
(Line 4)       p2 <= 81)
(Line 5)    THEN set display = "Normal BP"
```

Which lines can be removed to make the code more efficient, while not changing the code output?

Select one:
- O Line 1 and Line 4
- O Line 2 and Line 3
- O Line 2 and Line 4
- O Line 1 and Line 3

Appendix C. Sample survey items

Sample competency belief items

"I am sure that I can learn programming."

	Strongly Agree	Somewhat Agree	Agree	Disagree	Somewhat Disagree	Strongly Disagree
With what I knew on the FIRST DAY of the course…	O	O	O	O	O	O
With what I know TODAY…	O	O	O	O	O	O

"I could get an A on a programming assignment in class."

	Strongly Agree	Somewhat Agree	Agree	Disagree	Somewhat Disagree	Strongly Disagree
With what I knew on the FIRST DAY of the course…	O	O	O	O	O	O
With what I know TODAY…	O	O	O	O	O	O

"I am sure that I could do advanced work in programming."

	Strongly Agree	Somewhat Agree	Agree	Disagree	Somewhat Disagree	Strongly Disagree
With what I knew on the FIRST DAY of the course…	O	O	O	O	O	O
With what I know TODAY…	O	O	O	O	O	O