BootUp Professional Development Annual Report 2018-2019

Evaluation Performed by

Peter J. Rich, PhD

Table of Contents

1. Purpose

BootUp is a non-profit (501(c)(3)) organization whose mission is to prepare elementary educators to teach coding to children. This document reports on the evaluation of BootUp's professional development efforts across 8 sites throughout the 2018-2019 academic year.

1.1. Stakeholders

Primary stakeholders are those directly affected by participation or investment in BootUp professional development and support. During the 2018-2019 academic cycle, primary stakeholders in this report consisted of BootUp administration and staff, as well as the administrators, 8 district coaches and 283 teachers in the following school districts[1]:

Table 1.1
Participating school districts and demographics

School District	State	Year of BootUp Participation	# of participating schools	# of Teachers Trained	# of students taught*
		1.25	130	291	105,309
Suburban_1	Utah	1	30	56	41,850
Suburban_2	Utah	2	62	62	45,000
Urban_1	Utah	1	12	34	7,925
Rural_1	Utah	1	4	8	1,200
Rural_2	Iowa	1	3	40	1,027
Urban_2	Utah	1	8	31	4,746
Rural_3	Wyoming	1	5	25	450
Rural_4	Washington	2	6	35	3,111

* estimates based on enrolled elementary students at participating schools in each district.

Secondary stakeholders are indirectly affected by BootUp's training. Key secondary stakeholders in this experience were students of these teachers. About 105,000 K-6 students across 4 states were indirectly affected by the teacher training that BootUp provided during 2018-2019. Secondary data was collected from students in one district (Suburban_2) to measure gains in computational thinking ability and to measure student attitudes toward coding.

[1] Pseudonyms used to protect school district identities

1.2. The BootUp Professional Development Model

BootUp works with school districts to provide continual professional development over the course of 3 years. BootUp's professional development model consists of several components. Namely:

(a) working with district personnel to secure funding,
(b) training and working with a district coach,
(d) training teachers through 5-8 full-day professional development meetings in the summer and throughout the school year,
(e) providing open teaching resources to teach elementary coding (see https://bootuppd.org/curriculum), and
(f) on-site visits for demonstrations and co-teaching opportunities for teachers.

Ideally, this model is executed over a 3-year period in a scaffolded manner, described by year below:

Year 1: BootUp takes responsibility for the majority of training, including multi-day and monthly professional development (PD) meetings. The district coach is invited to participate in the planning and execution of these meetings, but is not expected to take charge and train the teachers. The district coach is a person who is typically hired by the local education agency (i.e., "district") to oversee the management and training of computing teachers.

Year 2: The district coach begins to share the responsibility for co-teaching professional development meetings, carrying a heavier load of this teaching as the school year progresses.

Year 3: The district coach assumes primary responsibility for delivering continuous professional development, with BootUp personnel playing a support role. The rationale behind this method is that it prepares the district coach to train new coding teachers as they come along, and to know how to support teachers in their learning. The intent is to provide a model that is both scalable and sustainable. By the end of the third year of professional development, BootUp hopes to have sufficiently developed the capacity within the district to train coding teachers without its help. In this way, BootUp's Professional Development model hopes to create sustainable and scalable capacity in a school district over the course of 3 years.

2. Methods

The primary method of data collection was through surveys administered during the first and during the final professional development for each district, and formative surveys administered at the end of each PD experience.

2.1. Primary questions

There were two primary evaluation questions:

1. How does participation in BootUp affect elementary teachers' confidence and competence to teach computing?
2. What aspects of BootUp trainings are most/least effective?

The first survey examined teachers' beliefs about coding in four primary ways: *values*, *computational efficacy*, *coding efficacy*, and *teaching efficacy*. Teachers were asked to rate their agreement with statements about their beliefs in each of these three areas on a scale of 1 (strongly disagree) to 6 (strongly agree). At the final training, teachers responded to the same questions. This provided a comparison for growth in teacher confidence to teach coding over the course of the training.

The final survey also included items about teachers' confidence in several other areas, but were specific to coding and required teachers to have completed the BootUp professional development in order to understand the coding-specific jargon.

An additional source of data was provided in the form of a follow-up survey given at the end of each professional development. Teachers were asked a few questions about what they thought went well and what could be improved. BootUp professional development staff used these results formatively throughout the year to improve upon their trainings.

2.2. Secondary questions

In one school district (Suburban_2), secondary data was collected from students. In meeting with the district, it was decided to measure students' computational thinking and their attitudes toward coding. Secondary research questions were:

1. What is the effect of learning coding on students' computational thinking?
2. What is the effect of learning coding on students' attitudes toward coding?

To answer question 1, we used the Computational Thinking test (CTt). The Computational Thinking test (CTt) (Román-González, Pérez-González, & Jiménez-Fernández, 2017) is one of the only existing, validated measures for computational thinking that is appropriate for younger students. The CTt is a 28-item exam that takes 15-45 minutes to complete.

Suburban_2 School District stakeholders were also interested in knowing more about students' attitudes toward coding. In order to measure attitudes, we researched over a dozen existing STEM or coding attitude scales. Unfortunately, none of these adequately addressed attitudes toward computational thinking or coding at a level suitable for elementary-aged children. As such, we developed the Elementary Student Coding Attitudes Survey (ESCAS), based on items from existing scales, as well as our own experience and insights with coding. This was then administered to students during their last month of school. The scale was validated with ~6000 Suburban_2 School District students in grades 4-6. It will be used in other districts in future years to measure changes in student attitudes toward coding over time. Specifically, the ESCAS measures: Coding Confidence, Coding Interest, Social Value, Perceptions of Coders, and Coding Utility. Section 5 presents an overview of the findings of each of these secondary questions.

3. Executive Summary

1. By all measures, BootUp-trained teachers deemed their professional development training a success. Nearly all teachers indicated they increased their confidence to teach coding due to their participation in BootUp professional development.

2. Teachers increased most in their confidence to teach coding, followed closely by their confidence in their own coding ability. They also increased in their computational thinking ability (albeit to a lesser degree) and in their valuation of the importance of coding (which started at a relatively high point).

3. Teachers found BootUp's hands-on learning opportunities to be the most effective component of the BootUp PD model. Other highly effective components were (in order): coder resources, videos, sharing projects/ideas, and model teaching. Teachers were less positive about topic discussions, site visits and peer coaching.

4. Regarding knowledge of coding, teachers demonstrated most confidence with sequences, algorithms and loops. They were less secure in their knowledge of conditionals, variables, and functions.

5. Regarding teachers' computational thinking knowledge, they were most confident with their ability to identify patterns, think algorithmically, understanding of logic, and evaluation. They were less secure with decomposition and abstraction.

6. Teachers remain less confident in their ability to foster several computational perspectives. This may be an area of emphasis for future training.

7. Elementary students demonstrated computational thinking abilities slightly above their international counterparts at the same grade levels. They increased their abilities about 1 grade level on the Computational Thinking test.

8. Students demonstrated positive attitudes toward coding. These attitudes are strongly influenced by their peers and parents, which influences their confidence and utility value for coding. Older and female students appear to value coding less.

4. Results

The school districts that participated in BootUp trainings in 2018-2019 varied in size greatly. Five participating districts have fewer than 10 elementary schools each, whereas urban districts like Suburban_2 and Suburban_1 consist of over 60 schools each (though only half of Suburban_1 schools participated in this first year of training). Owing to this difference, smaller districts may get lost when reporting the overall effect of BootUp PD on teachers. Therefore, I present the data both in the aggregate, as well as broken down by district. This may help to interpret both overall and local effects of PD and to highlight localized findings that may be hidden otherwise.

4.1. Teaching Demographics Patterns

This section provides a description of participants as well as a brief overview on the actual teaching of coding that BootUp-trained teachers engaged in throughout the year, as reported on the year-end survey.

4.1.1. Teacher Demographics

The vast majority of teachers were either computer lab or library media specialists (see Figure 1). These teachers tend to teach students at every level of elementary school, usually seeing the same group 1x/week. While library media specialists tend to be full-time teachers, technology/lab specialists tend to be part-time. BootUp participants in all the Utah schools tended to fulfill these roles.

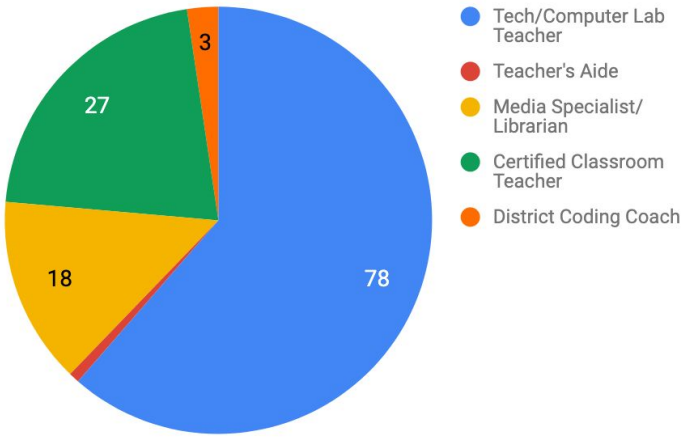


Figure 1. Teachers' professional role

There was also a distinct second group of classroom teachers. Classroom teachers spend the majority of their time with the same group of students throughout the week. They are licensed teachers and are highly trained (bachelors or masters). Teachers in Rural_4, Rural_3 and Rural_2 tended to be classroom teachers (see Table 4.1.1.a).

On average, teachers reported participating in 5.15 BootUp trainings throughout 2018-2019. However, this participation seemed to also vary according to geography and teacher role. Teachers in Rural_4 (3.75), Rural_3 (2.5), and Rural_2 (3.13) participated in fewer BootUp trainings than those in Suburban_1 (5.04), Urban_1 (4.56), Suburban_2 (6.75), Rural_1 (6.00), and Urban_2 (5.6).

Table 4.1.1.a

Professional teaching role

	Tech/Computer Lab Teacher	Teacher's Aide	Media Specialist/ Librarian	Certified Classroom Teacher	District Coding Coach
Overall	78	1	18	27	2
Suburban_1	26	0	0	0	0
Urban_1	3	0	9	0	0
Suburban_2	36	0	6	0	0
Rural_1	6	0	0	0	1
Rural_2	1	0	0	14	0
Urban_2	2	1	1	1	0
Rural_3	1	0	0	5	1
Rural_4	3	0	2	7	1

BootUp-trained teachers also tended to have 9.82 years of average teaching experience, with 6.75 of those dedicated to teaching students at the current grade level. However, teaching coding was relatively new, as teachers only reported having 1.25 years of coding teaching experience by the end of the 2019 school year (which included their 2019-2020 year of training). Further analysis revealed that certified (N = 46) and non-certified teachers (N = 76) did not differ significantly in the number of years teaching coding or even their experience teaching the same grade levels they currently teach. However, there was a statistically significant difference in their overall teaching experience ($p = .022$), with certified teachers averaging nearly 4.5 more years teaching overall. Table 4.1.1.b breaks down median teaching experience by district (median was used to account for outliers).

Table 4.1.1.b

Median Years of Teaching Experience

District	#	overall	this grade	coding
Overall	127	9.82	6.75	1.25
Suburban_1	26	9.0	6.5	2.0
Urban_1	12	11.0	10.0	1.0
Suburban_2	42	2.5	2.5	1.0
Rural_1	7	2.0	2.0	1.0
Rural_2	15	22.0	11.0	1.0
Urban_2	5	12.0	8.0	1.0
Rural_3	7	9.0	8.0	1.0
Rural_4	13	11.0	6.0	2.0

4.1.2. Teaching Patterns

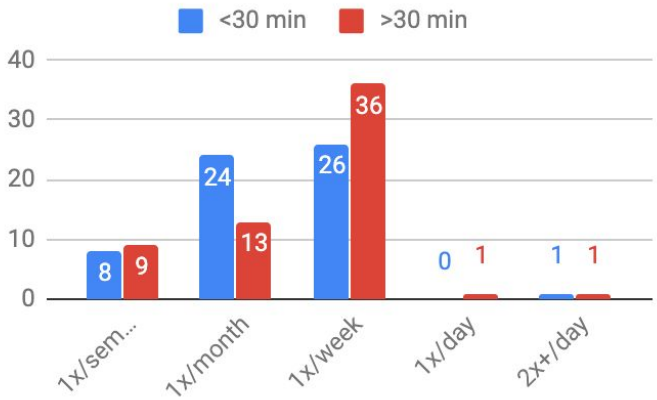
Teachers reported how often they taught coding, as well as how long these sessions lasted. While the majority of teachers taught coding to the same group of students once a week or more often (N = 65), teachers in half of the districts would only see the students once a month or less frequently (N = 54). The frequency with which a teacher taught the same students tended to be tied to their district (see Table 4.1.2). For example, 96% of teachers in Suburban_2, Urban_2, Rural_1, and Rural_3 districts taught coding to the same group of students once a week or more often, whereas only 16.1% of teachers in other districts taught the same group of students that often. Thus, it might be deduced that teaching role or district policy regarding how often a specific teacher teaches coding has a strong effect on a student's opportunity to learn to code.

Table 4.1.2

Frequency teaching coding to the same group of students

District	#	1x/semester	1x/month	1x/week	1x/day	2x+/day
Overall	127	17	37	62	1	2
Suburban_1	26	7	14	3	0	2
Urban_1	12	0	8	1	0	0
Suburban_2	42	0	0	39	1	0
Rural_1	7	0	1	5	0	0
Rural_2	15	3	10	2	0	0
Urban_2	5	0	0	5	0	0
Rural_3	7	0	1	5	0	0
Rural_4	13	7	3	2	0	0

Teachers also reported the average length of their coding classes. There was almost an even split between those teaching classes lasting longer than 30 minutes (N = 60) and those lasting less than 30 minutes (N = 59).



Comparing these differences by controlling for frequency revealed that teachers who teach coding for 30+ minutes are more likely to teach once a week (63.3%) than those who teach shorter classes (45.8%) (χ^2 = 13.2 (4), p = .010).

4.2. The Effect of BootUp Training on Teachers

In this section, I present the findings on the primary questions, starting with three open-ended questions. Open-ended questions allow teachers to respond using their own words rather than a predetermined set of choices. While open-ended questions tend to result in a greater diversity of answers, they give voice to teachers. Following the findings of these open-ended questions, I present the results of teachers' experience as measured on different scales (which allows a more consistent comparison over time).

4.2.1. Change in Confidence to teach coding to kids

Perhaps the most direct source of measuring teachers' changes in their confidence is to ask them directly how their confidence has changed. Teachers provided this response to the question "How has your confidence to teaching computing changed over the course of this year? What do you believe has led to this change (or lack thereof)?"

Using an open-ended coding scheme (Glaser & Strauss, 1967), I coded teachers responses to indicate the direction and cause(s) of their changes in their confidence to teach computing. 100 of the 105 teachers who responded to this question directly indicated that their confidence to teach coding increased. Figure 4.2.1 reveals the different causes that teachers attributed this growth to. BootUp's training was mentioned 61% of the time, while actual experience teaching coding or practicing it on their own time was mentioned 39% of the time. Teachers also mentioned specific BootUp resources, such as videos and hands-on training as the cause for this increase.

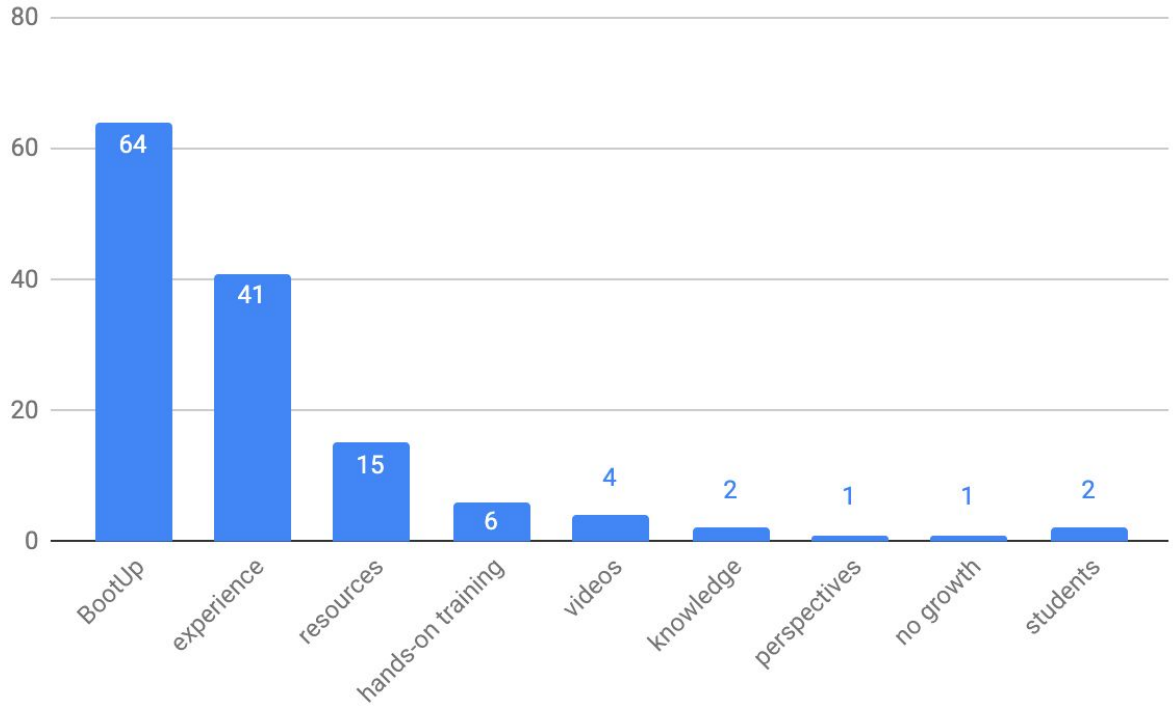


Figure 4.2.1. Analysis of teachers' comments about what caused their growth in confidence to teach coding.

While there is not space in this report to include all teacher comments, the following quotes represent many of the ways in which teachers responded to this question about their change in confidence.

> I feel **more prepared and better equipped** to teach coding because of the projects that are already prepared and because of Mckay (the instructor) demonstrating a lesson in my class.

I am much more confident teaching computing to students because I learn with them. I started teaching computing with small things I was comfortable with, and as I learned, I taught the children too. And many times, those students teach me a few things about it too! **The experience of actually doing computing has increased my knowledge so much more and helped me to become more confident.**

BootUp has GREATLY increased my confidence in teaching coding. I have gone from teaching hour of code only to teaching full units. I have gone from having coding lessons be only my capstone projects to having them be my flexible activities (no internet, double classes, etc.). I have gone from struggling to come up with coding lessons to being able to create lessons that integrate with other curriculum. I have gone from direct teaching all concepts to encouraging collaboration, communication and critical thinking in my coding lessons.

I have become more confident in being a coding facilitator instead of knowing how to do everything perfectly. So I am more confident going into a coding situation where I may not know all the answers but I can ask students questions to help them find or figure out the answers themselves.

These BootUp trainings have been my saving grace! **I would not have been able to do as much as I have done without McKay and these trainings!! At first they were so hard.** I didn't even know that this new coding program had been put in place. So I was completely lost and frustrated at the first. The first facilitator wanted us to learn all she knew in a few hours. I know she knew a lot and wanted us to learn but it was so 'in your face.' McKay was able to come in and gently take us older minds and help us learn slowly and help our knowledge grow. I still don't know or understand all that was taught. But as time goes, I will be able to learn with my students with this great start.

When I was hired last year my focus was mostly on keyboarding mixed with Office 365 and a tiny bit of code.org etc. **The Boot Up trainings this year have completely changed with way I teach and understand coding.**

It has improved dramatically. I still struggle with whether or not I am good enough or smart enough. But I love the material and recognize its importance. I think training has helped but also having a classroom of students gasp and cheer as they are understanding it has been the biggest confidence builder. **Having students in SPED who have never felt like they belonged at school tell that coding is the most important thing to them has been amazing!**

> I was so very nervous to teach coding in my class. But with **the lessons taught through BootUp has made it much easier.**

4.2.2. Successes

Teachers were asked, "what successes have you had in teaching coding?" Asking the question in this way allows teachers to focus on any aspect of their experience that they feel successful with. 97 teachers provided responses to this question. While there were a few comments about increased ability to teach computing, teachers across the board focused comments about success on their students. Student interest, increased knowledge of coding, and successes with coding dominated teachers' success stories.

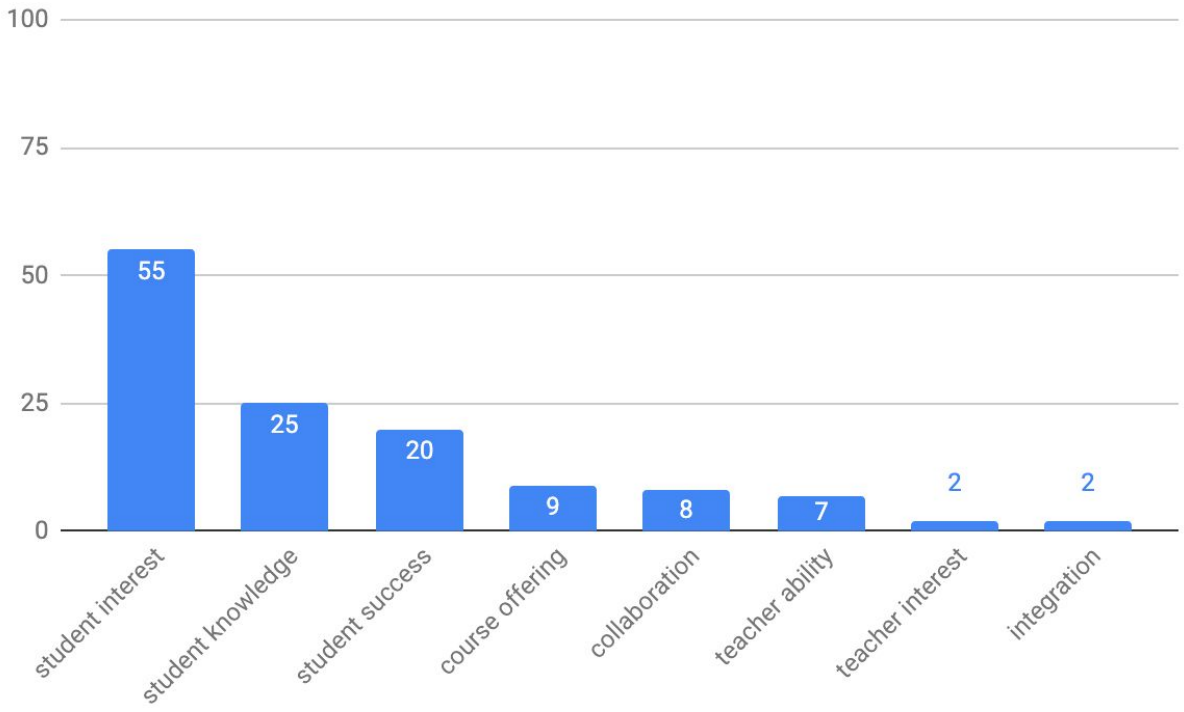


Figure 4.2.2. Analysis of teachers' comments about their successes with coding in 2018-2019.

Fully 55 of 97 comments indicated that students have a high interest in computing, using words such as "excited," "love," and "engaged" to describe student interest. Nearly 20% mentioned student interest leading to students extending their learning outside of the classroom. Many teachers indicated that students had knowledge and abilities that enabled them to solve problems or be creative due to their participation in coding. Teachers discussing student success mentioned success directly, as in students overcoming a challenge or, in nearly half of the success stories, students who normally struggle in other aspects of school succeeding with coding.

The following page provides a few representative comments of the successes BootUp teachers reported having with teaching coding in 2018-2019.

The first example that came to mind is **an autistic 5th grader. He is most content on the days when we are coding.** I believe he likes the logic of writing code.

I have been able to watch my students go from hating computing because they don't understand it to loving it because they are able to grasp the concepts and feel successful at it.

I have seen students who struggle in other areas catch the Spark and love this. I have seen kids learn to work together to create. I have seen kids get excited in my class.

Students come in to class excited to code and some of them work on the projects at home.

The greatest part of coding has been the interaction with my students, they absolutely love it. **My students have become so engaged with coding and look forward to it every week.**

I have seen students who struggle in other subjects excel in coding. They get so engaged and excited! **They finally get to be the ones who can help other students.** Also, I have kids asking when we're going to do Scratch next and creating projects during their free time, at home, and whenever they can.

Seeing kids problem solve and try new things to create what they wanted a sprite to do was my favorite part. They were so proud, and often did it on their own. I've also loved the teamwork I've seen and the support they give their partner.

My students love coding class. They are so creative and excited about their products. **They don't want to stop!**

I feel like the students are less intimidated by coding and enjoy it more than they did at the beginning. Many were hesitant to start something so different and new but now seem quite comfortable and willing to do new things.

This year I have found that the **students are participating in group discussions and have opinions and ideas on how to improve projects.** Their interest in coding and their excitement is a huge success.

The students are excited, **they want to be able to stay and finish their projects** and they want to keep learning.

My favorite successes are when one of my students finds a solution to a problem that I couldn't find. I praise them in front of the class and their face lights up.

I have only taught 1st graders and they have truly surprised me with some of the programs they come up with. They love to create, I believe it is second nature to them to do so.

Having those who are not "computer nerds" like computers. I did a survey at the beginning of the year about who my students believed were the typical coders, unanimously it was "fat, white, man, living in parent's basement." Now I have multiple girls who are loving it. I have a fourth grade girl who doesn't like Scratch so much but from what she learned by doing Scratch was able to try JavaScript and she LOVES JavaScript. I have multiple SPED students who are trying their best to learn more math and reading to make better projects. **A fifth grader who started the year throwing chairs at me and calling me every four letter word possible, and at least one five letter word, is now my coding expert and teaches the new students how to code.** It has been an amazing experience.

4.2.3. Challenges

Teaching a new content comes with many challenges. In reporting their own challenges to teaching coding, elementary teachers primarily indicated that time was their greatest challenge, followed by their own lack of understanding. Students' knowledge and interest were also mentioned by nearly 1 out of every 5 teachers.

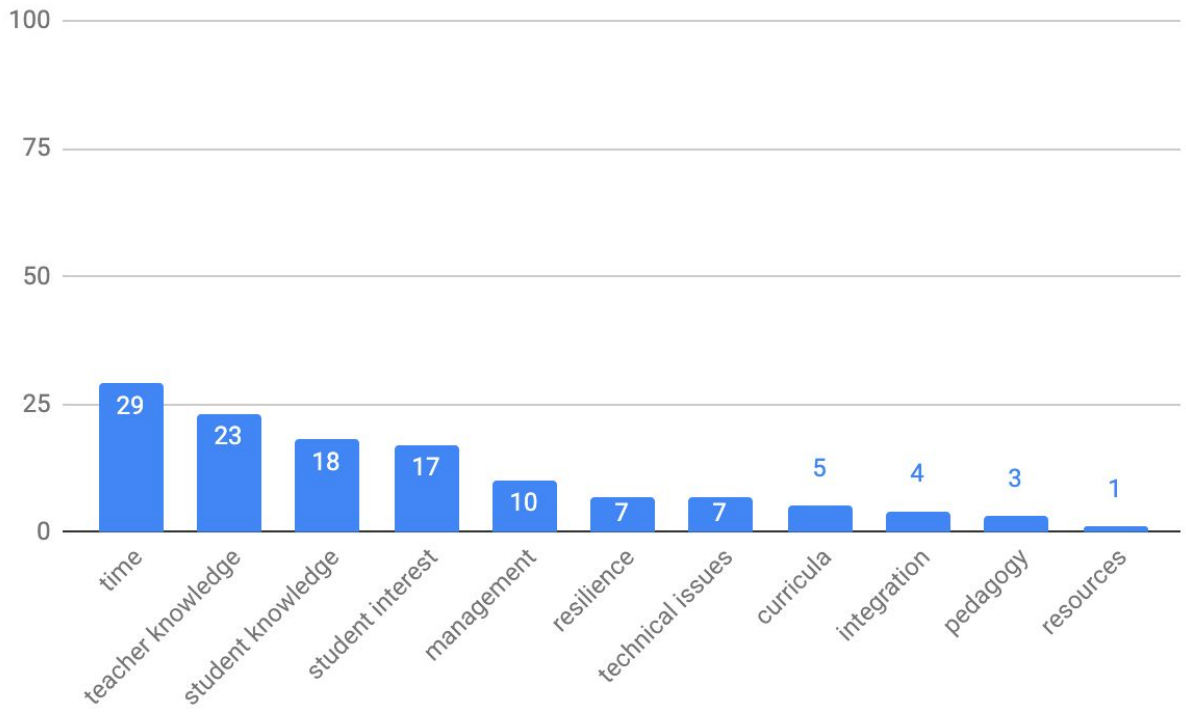


Figure 4.2.3. Analysis of teachers' comments about their challenges with coding in 2018-2019.

Comments about ***time*** tended to emphasize the lack thereof. Most time-based challenges mentioned that a teacher only had 30 minutes to teach and that logging students into the computers would often take 10-15 minutes of that time.

Comments about ***teacher knowledge*** emphasized teachers' feelings of inadequacy in understanding content and "staying ahead" of their students.

Student knowledge challenges tended to deal with two issues: (a) the difficulty that some students experienced when learning to code, or (b) differentiating instruction for students with varying ability (or a different grade levels) to code.

Student interest challenges indicated that not all students really liked coding. This challenge stands in stark contrast to the many positive comments in the "success" question about high student interest in coding. Many comments about a lack of interest were accompanied by resilience challenges, indicating that students who were not interested in coding often gave up easily when presented with more difficult problems.

The following quotes provide a snapshot of teachers' challenges in their own words.

Not enough time to really get started. Only have a 30 minute class and getting logged in, instructions, deciding only left about 10 minutes to really work.

Finding time within my curriculum within my limited **time frame of 30 minute classes** weekly.

Getting teachers to see the value of taking a half hour to an hour out of their teaching schedule to code each week. Finding time in the week to teach coding.

Not having enough time to learn enough to be "ahead" of my kids. I started teaching after the school year started and learning the school lingo and requirements took some getting used to. There were also no benchmarks or guidelines - just teach scratch, scratch jr, some keyboarding, and STEM activities once a month. I have 15 min of prep time. That's enough time to log in and get the room set up and that's it. If the district/school values this sort of education, they will have to expend more resources to training.

Feeling like the kids know more than I do. *But getting over that*.

Not knowing everything that I believe is required to know when teaching this subject.

3rd grade has been my biggest challenge! Class has to move so slowly and **concepts need to be repeated in order for students to understand them.**

Understanding the terminology myself! Also, I've noticed that kids who do well in the classroom, **"the text book perfect kids" struggle with coding because they're not used to failure**. My students who are in resource and are slower love coding because they don't give up and keep trying.

The same challenge as teaching other areas. **The variety of learner ability** and the inconsistency of students attending/being pulled during specialty time.

ADHD kids and teacher have a hard time going line by line.

The biggest challenge is **keeping up with all the kids at different levels.** Making sure the kids who are more advanced stay challenged and then helping the kids who are falling behind catch up and just making sure they understand the concept. When we work on a scratch project, I feel like I'm just running around the room the whole time answering questions.

Helping students, and **getting around to everyone.**

It's mostly dealing with behaviors, from throwing equipment to crawling on the tables. **My challenge has all been classroom management.**

My challenges have been with **students that are defiant and shout out in class making a scene.** It gets the other students off task and then you lose control over the whole class.

4.2.4. Teachers' confidence and competence to teach coding

Pre/Post Data were available from all districts regarding their beliefs about coding. Validation of the self-efficacy survey revealed that four different beliefs were measured:

- **Values**: Teachers' impressions of the importance of learning coding, especially as it applies to elementary-aged children.
- **Computational Thinking self-efficacy**: Teachers' belief in their ability to think computationally (i.e., to break a larger problem into smaller problems, to think algorithmically, and the solve problems). Computational thinking has received several different definitions since Jeannette Wing's renewal of the term in 2006. In this report, I use a simplified version of the definition provided by ISTE, the International Society for Technology in Education, and the Computer Science Teachers Association. To wit, Computational Thinking is a process of solving problems logically in a way that they can be solved by an information processing agent (see https://www.iste.org/standards/computational-thinking for a full definition). CT has been correlated with existing problem-solving scales (Román-González, Pérez-González, & Jiménez-Fernández, 2017).
- **Coding self-efficacy:** Teachers' beliefs about their own ability to code, including their ability to apply fundamental concepts such as loops, conditional logic, variables, etc.
- **Coding Teaching Efficacy:** Teachers' belief in their ability to teach elementary coding to young children.

A 6-point Likert-type scale was used to measure teachers' changes in their CT. An even-numbered scale encourages teachers to indicate either a positive or negative degree of confidence, avoiding neutral answers (which might occur with an odd-number scale). The reason I chose to use a 6-point scale is because a 6-point scale forces respondents to show which way they lean (i.e., more or less confident). Research has demonstrated that a 6-point Likert-type scale is a reliable method for measuring self-efficacy (Reeve, Kitchen, Sudweeks, Bell, & Bradshaw, 2011). See Appendix A for the specific questions asked.

To analyze scores, questions were normalized to the same scale (because some questions were stated negatively). Average scores were then calculated for all teachers who responded to the surveys across all 8 districts. Scores were aggregated for each of the subscales (values, CT self-efficacy, Coding Self-Efficacy and CT teaching efficacy). Figures 4.2.4a-d demonstrate the change experienced by teachers in each participating district between their first (N=245) and final (N=128) BootUp PDs in 2018-2019 using these composite scores (see Appendix B for a breakdown of scores within each district).

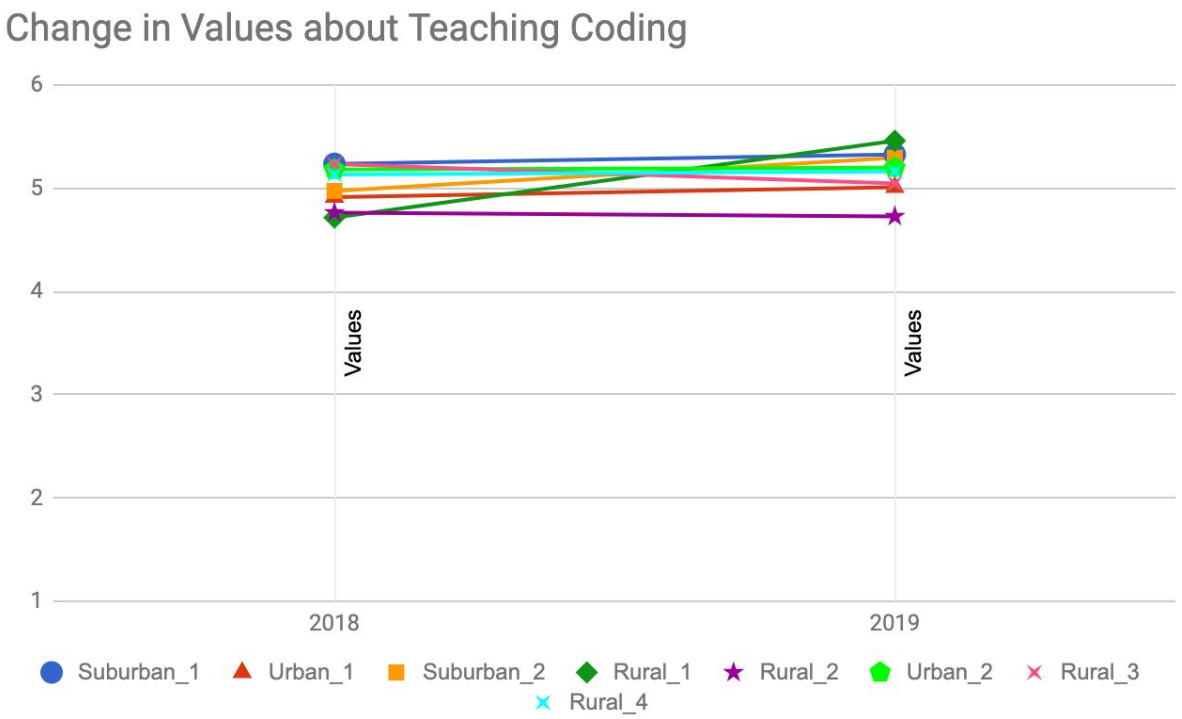


Figure 4.2.4.a. Teachers' changes in values for teaching coding from first to last professional development

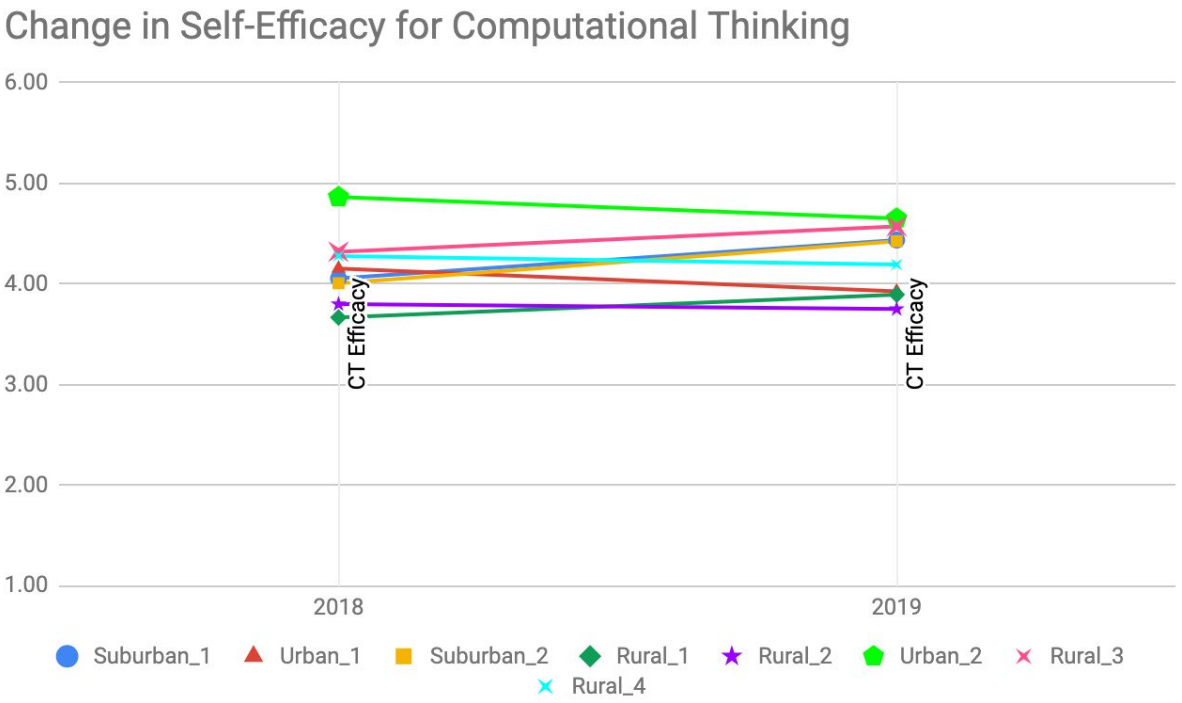


Figure 4.2.4.b. Teachers' changes in self-efficacy for computational thinking from first to last professional development

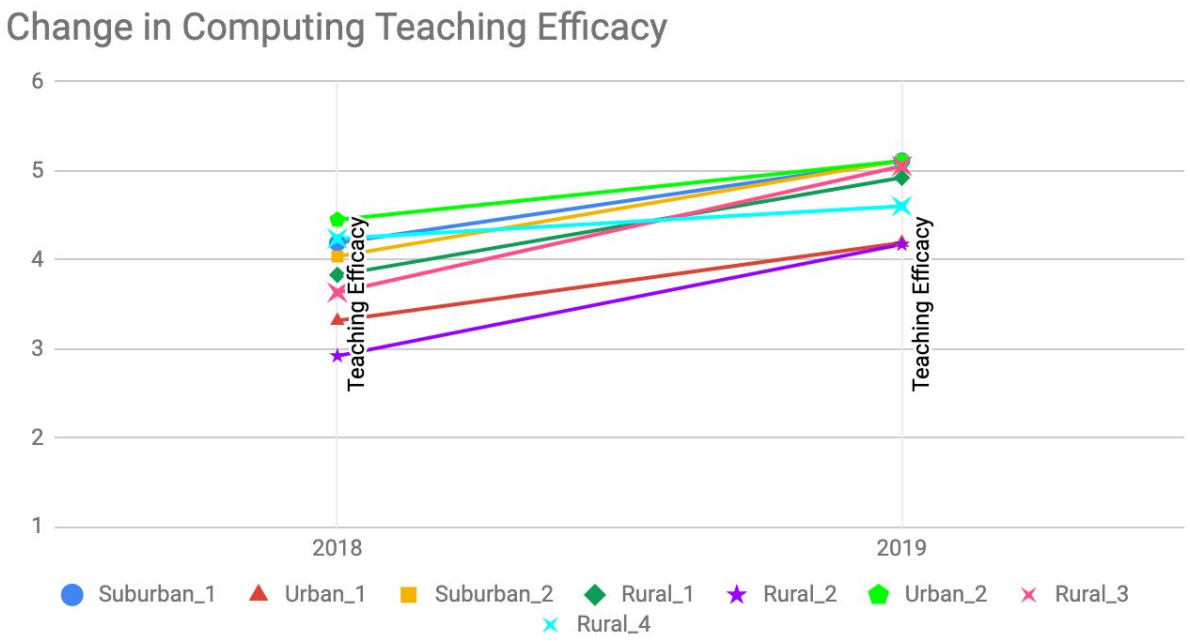


Figure 4.2.4.c. Teachers' changes in values for coding teaching efficacy from first to last professional development

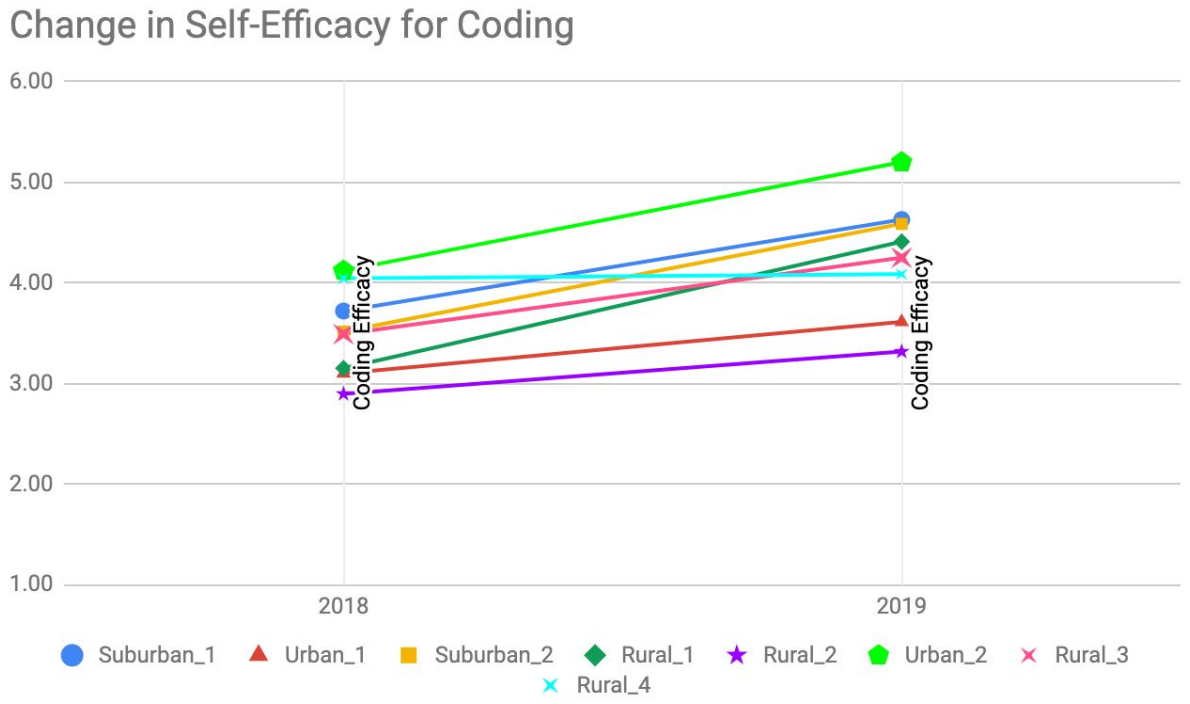


Figure 4.2.4.d. Teachers' changes in self-efficacy for coding from first to last professional development

4.2.5. Value Beliefs for Coding

In all districts, teachers' value beliefs began relatively high (near or above a 5 on a 6-point scale). This indicates that they already began the training believing that it is important for young children to learn how to code. Statistical analyses demonstrated a statistically significant gain with a small effect ($p = .027$, Cohen's $d = .26$). The reason to measure teachers' beliefs about the value of teaching coding is because a teacher's valuation of a subject is the lens through which they both teach and communicate about its importance to students. If a teacher does not believe it is important to teach a subject, this is also apparent to students and can severely limit their own excitement for that subject. Fortunately, it appears the teachers who participated in BootUp trainings in 2018-2019 had a high valuation for coding and slightly (but significantly) increased this valuation over the course of their training.

4.2.6. Computational Thinking

There was a clear upward trend in most districts regarding teachers' self-efficacy for coding and computational thinking (i.e., competence), and their efficacy for teaching coding (i.e., confidence). Teachers' average beginning self-efficacy score for Computational Thinking began higher at 4.06 out of 6 points. By the last training of 2019, this score rose to 4.27 ($p = .004$; Cohen's $d = .517$). While there is still room for improvement (hopefully teachers will score themselves ≥5 by the end of their third year), this is clearly a good upward movement in their confidence to think computationally.

4.2.7. Coding Confidence

Teachers were initially the least secure in their confidence to code, scoring themselves at a 3.46 on a 6-point confidence scale. By the end of the training, this number rose to 4.30. This gain proved to be statistically significant ($p < .000$), which indicates that the change in teachers' scores was not likely due to chance. What's more the gain appears to have been practically significant, with a strong effect size (Cohen's $d = .894$), demonstrating a change of nearly 1 standard deviation in growth.

4.2.8. Teaching Confidence

Teachers began the professional development slightly more confident about their ability to *teach* coding at the beginning of the year (3.84/6.00). Fortunately, teachers' confidence in their ability to teach coding increased even more than their confidence in their coding and computational thinking, rising a full point to 4.84. This is nearing the 5.00 threshold that I would expect to see teachers' confidence reach as they increase in experience in teaching coding to young children. This change was both statistically significant ($p = .007$) and had a large effect (Cohen's $d = .934$).

By these measures alone, it would appear that BootUp's year-long professional development was successful in increasing elementary teachers' competence and confidence to teach elementary computing.

4.2.9. Specialists vs. Classroom Teachers?

As I analyzed the data, there is an important caveat to the evaluative statement about success. The results above are aggregated summaries across all locations. A quick perusal of figures 1-4 shows that teachers' growth in their confidence looks different for the various locations. While teachers' coding and teaching efficacy appear to have risen across all sites, their computational thinking efficacy remained flat at several sites. Further investigation revealed that this difference could be explained by whether or not a teacher was a specialist or a classroom teacher. For some reasons, districts where the participants were classroom teachers appear to not have grown in their computational thinking efficacy or, in some cases, decreased their CT efficacy.

The starkest case of this difference occurred at Rural_4. Rural_4 teachers did not appear to have gained as much confidence as with other teachers. Upon inquiring why this might be, I learned from Rural_4 district leaders that teachers were not required to teach computing regularly. Furthermore, actual implementation of computing in the classroom was less frequent for Rural_4 teachers than for teachers in other districts, which was evident in their self-report numbers for teaching frequency. It is likely that their lack of implementation provided teachers with less opportunity for growth, which is reflected in their efficacy scores.

4.3. Other measures of confidence and competence

The final survey also asked teachers about several other components of their computational thinking that BootUp taught throughout the year. These measures come directly from the principles, practices, and perspectives promoted by CSK12.org, an organization that provides a framework for computational thinking standards in K-12 education. These scales are divided into teachers' personal knowledge and their teaching knowledge. I first address the former and then the latter.

4.3.1. Personal Computing Knowledge

Teachers were asked to rate their ability to understand six core computing concepts: algorithms, conditionals, functions, loops, sequence, and variables. Table 5 shows that, by the end of the first year, teachers are most comfortable with sequence, algorithms, and loops, scoring themselves near or above a 5 on each of these areas. Functions, conditionals, and variables all appear to be more difficult for teachers. Though the order changed slightly, this follows the same pattern observed in 2017-2018. There are likely several reasons for this. First, while fundamental to programming, these are more advanced computing concepts. As a professor who has taught programming myself, these are precisely the concepts that beginning coding students find most difficult (in my experience). Second, In some cases, these concepts were not introduced in the BootUp trainings until the last training, or even until the second year of training. As such, it is reasonable to expect teachers to be less comfortable with these specific concepts until they develop greater experience with coding themselves. This will be an important area to track over time to see if teachers increase in their conceptual understanding of coding.

Table 4.3.1.a

Teachers' confidence for coding concepts

	Sequence	Algorithms	Loops	Functions	Conditionals	Variables
Overall	5.03	4.94	4.89	4.26	4.05	3.93
Suburban_1	5.19	5.04	4.96	4.46	4.23	4.31
Urban_1	3.89	4.00	4.11	3.33	3.33	3.22
Suburban_2	5.45	5.73	5.43	4.68	4.38	4.20
Rural_1	4.50	4.00	4.67	4.50	4.00	3.83
Rural_2	4.47	3.47	3.67	3.47	2.93	2.87
Urban_2	5.00	5.20	5.40	5.40	5.20	5.00
Rural_3	5.00	5.17	4.67	4.17	3.67	4.00
Rural_4	5.17	4.92	5.08	3.58	4.25	3.67
2017-2018*	8.23	8.09	8.45	6.91	6.68	6.73

* the 2017-2018 scale for this measure was 1-10

Validation of the scale used to measure teachers' coding knowledge revealed that computational thinking (CT) and coding are two related, but separate, constructs. Accordingly, we asked teachers to rate their confidence with the different components of computational thinking: decomposition, pattern recognition, algorithms, abstraction, logic, and evaluation. Table 6 reveals that, on average, teachers rated their CT ability positively, though they did not reach the 5-6 point threshold for higher confidence in any single area. I expected teachers to rate their pattern recognition skills highly (after all, it's a skill many of them teach in kindergarten), and to rate abstraction as a more difficult skill. Both of these hypotheses bore out. However, I was somewhat surprised to find that teachers rated decomposition as the second-lowest area of confidence and algorithms as the second-highest (I would have expected these two to be flipped).

Table 4.3.1.b

Teachers' confidence with own computational thinking knowledge

	Pattern Recognition	Algorithms	Logic	Evaluation	Decomposition	Abstraction
Overall	4.69	4.64	4.33	4.19	3.97	3.78
Suburban_1	5.00	4.88	4.58	4.50	4.19	3.96
Urban_1	3.67	3.56	3.22	3.44	3.00	2.89
Suburban_2	5.20	5.35	4.90	4.48	4.35	4.15
Rural_1	3.67	3.33	3.67	3.33	3.67	3.50
Rural_2	4.00	3.27	3.73	3.67	3.00	3.13
Urban_2	5.40	5.60	5.20	4.80	5.40	4.80
Rural_3	4.00	4.00	3.50	3.83	3.50	3.50
Rural_4	4.55	4.82	3.82	4.09	4.00	3.45
2017-2018	4.77	4.68	4.18	3.95	3.95	3.68

4.3.2. Teaching Knowledge

Throughout their BootUp training, teachers learned about several different aspects of teaching computing that extend beyond coding or computational thinking. In this evaluation, we look at how teachers were able to foster computational practices and perspectives (Brennan & Resnick, 2012). Brennan and Resnick define the former as "the practices designers develop as they program" and the latter as "perspectives designers form about the world around them and about themselves" (p. 3).

Table 4.3.2.a demonstrates that teachers are most confident with the practice of teaching persistence. Research has found that elementary coding teachers report the development of resilience in the face of failure as an important aspect of teaching coding (Rich at al., 2018). Teachers rated all practices in the 4-5 range, indicating that they are beginning to feel comfortable with these practices, but have yet to reach a high level of comfort across the board. Teachers' ratings for teaching these practices in 2018-2019 closely mirrored those of 2017-2018, with tinkering/remixing and debugging switching places. This appears to have been primarily due to a much larger jump in their confidence to tinker/remix. This jump may possibly be attributed to BootUp's increased efforts to include more hands-on coding activities and time during PD than in 2017-2018.

Table 4.3.2.a

Teachers' confidence for teaching computational practices

	Persistence	Creating	Collaborating	Tinkering/ Remixing	Debugging
Overall	4.85	4.75	4.60	4.45	4.30
Suburban_1	4.69	4.46	4.62	4.77	4.77
Urban_1	4.00	4.56	3.67	3.00	3.33
Suburban_2	5.25	5.30	4.88	4.83	4.80
Rural_1	4.67	4.33	4.17	3.83	3.67
Rural_2	4.73	4.20	4.67	4.20	3.00
Urban_2	4.40	5.40	5.20	4.80	5.20
Rural_3	4.50	4.50	3.67	4.00	4.17
Rural_4	5.09	4.36	4.73	4.27	3.91
2017-2018	5.09	4.73	4.68	3.73	4.59

Computational perspectives emphasize attitudes toward computing overall, and mirror 21st-century learning, such as collaborating, communicating, and creating, as well as several coding-specific attitudes, such as fostering an inclusive computing culture, recognizing computational problems, and developing the ability to use abstractions to solve these problems. Teachers' ratings of these attitudes were not reported in the 2017-2018 report.

Overall, teachers were less confident in fostering these perspectives than in other areas, with teachers in several school districts reporting a lack of confidence in their abilities to develop and use abstractions, test and refine computational artifacts, and create computational artifacts (see Table 4.3.2.b). Collaborating and communicating about computing and fostering an inclusive culture all appear to be areas where teachers are most confident. Without further investigation, it is difficult to understand why teachers are less confident in teaching computational perspectives than they are with any other area measured in this evaluation. It is curious that teachers rated "creating" as a 4.75 in regards to computational practices, but rated "created computational artifacts" as only a 3.56. Hopefully, through repeated exposure to teaching computing and to creating computational artifacts themselves, teachers will increase their confidence in these areas. BootUp may want to pay attention to how teachers are fostering computational perspectives and provide tips for fostering these moving forward.

Table 4.3.2.b

Teachers' confidence to foster computational perspectives

	collaborate around computing	communicate about computing	foster an inclusive computer culture	recognize and define computational problems	develop and use abstractions	test and refine computational artifacts	create computational artifacts
Overall	4.38	4.33	4.31	3.95	3.47	3.58	3.56
Suburban_1	4.77	4.62	4.85	4.04	3.56	4.04	3.81
Urban_1	2.89	3.22	3.11	3.00	2.56	2.33	2.33
Suburban_2	4.87	4.97	4.67	4.44	4.03	4.11	4.18
Rural_1	3.83	3.83	4.00	3.50	3.33	3.17	3.00
Rural_2	3.53	3.27	3.20	3.07	2.67	2.60	2.53
Urban_2	4.80	4.60	5.00	5.40	4.80	4.60	4.40
Rural_3	4.17	3.83	4.17	3.50	3.17	2.83	3.17
Rural_4	4.36	4.18	4.18	3.82	2.73	3.27	3.27

4.4. Teachers' Ratings of BootUpPD components

BootUp Professional Development consists of several different components: coder resources, hands-on learning, model teaching, sharing projects/ideas, site visits, topic discussions, and videos.

Teachers were asked to rate the 8 primary elements of BootUp's professional development model on a scale of 1-10 (see Table 4.4). These include resources as well as practices. In 2017-2018, 22 teachers from Suburban_2 School District rated these BootUp elements. In 2018-2019, 115-117 teachers across all 8 participating districts rated these elements. Each BootUp element is briefly described below, including their 2017-2018 rating.

Coder Resource: A set of materials for teaching and learning each pre-planned lesson on Bootuppd.org. These include sample files, video walk-throughs and presentation slides for guiding the lesson in the classroom. In 2017-2018, teachers rated this a 7.86/10.

Hands-on Learning: BootUp's professional development model emphasizes allowing teachers to focus on practicing coding as much as possible. Even though this was rated the highest of all of BootUp's PD elements in 2017-2018 (8.82/10), BootUp increased the amount of hands-on learning in their PDs in 2018-2019.

Model Teaching: To help teachers understand how they might implement coding lessons in the classroom, BootUp professional development facilitators will demonstrate a lesson as the teacher. In 2017-2018, teachers rated this element as an 8.32/10.

Peer Coaching: Research indicates that many coding teachers feel isolated and do not get to collaborate with their peers as often as they would like (since they are often the only coding teachers at their schools). To combat this, BootUp gathers together several teachers from the same district to visit and observe one of their colleagues teaching in their own classroom. They then discuss and provide feedback to each other as a group. This practice was not rated in 2017-2018.

Sharing Projects/Ideas: During the BootUp training sessions, teachers are encouraged to share projects they have created or ideas they have had about teaching coding. In 2017-2018, these were rated 8.55 overall.

Site Visits: To help teachers in the context of their own schools, BootUp facilitators visit them individually on site. The intent is to answer any questions and offer formative feedback and support (this is not a summative evaluation). This element was in development and was not rated in 2017-2018.

Topic Discussions: BootUp provides both in-person and online venues for teachers to discuss specific coding or computational thinking in elementary education topics. In 2017-2018, this element was rated at 8.36.

Videos: One of the primary resources offered to teachers is a video walkthrough of what a completed project could look like. There are also videos that demonstrate how to complete each coding project. In 2017-2018, these were rated 7.57 overall.

Table 4.4
Teachers' ratings of BootUp elements

	Hands-on Learning	Coder Resource	Videos	Sharing projects/ ideas	Model teaching	Topic discussions	Site visits	Peer coaching
Overall	9.43	8.85	8.60	8.37	8.26	7.90	7.23	7.10
Suburban_1	9.46	9.62	9.31	8.92	8.79	8.60	8.30	8.38
Urban_1	8.67	7.44	8.00	6.78	7.25	7.00	7.17	6.56
Suburban_2	9.60	9.55	8.58	8.44	8.22	7.93	6.28	6.51
Rural_1	9.83	9.83	9.83	10.00	9.00	9.67	8.67	9.00
Rural_2	9.60	7.40	7.73	8.13	8.60	8.07	8.40	7.47
Urban_2	9.40	9.80	8.20	8.80	6.80	8.20	7.80	6.00
Rural_3	8.83	7.00	8.00	7.67	8.00	5.50	6.83	6.33
Rural_4	9.25	7.75	8.50	7.83	7.92	7.00	5.36*	6.33*
2017-2018	8.82	7.86	7.57	8.55	8.32	8.36	N/A	N/A

* this location did not implement these practices

As an evaluative framework, I would consider anything rated between 8-10 as highly rated, 7-7.99 as well-rated with room for improvement, and anything <7.00 as needing revision. Across the board, teachers rated all BootUp training elements favorably (see Table 4.4). Hands-on learning remained the highest-rated element, increasing .61 points in its overall favorability. Coder resources (+.99) and Videos (+1.03) increased appreciably in their ratings. While sharing projects (-.18) and model teaching (-.06) decreased, they effectively remained the same, both being rated above an 8.0. In 2018-2019, teachers rated Topic Discussions (-.46) significantly lower, dipping just below the highly rated threshold set above. Site visits (7.23) and peer coaching (7.10) were newly rated in 2018-2019 and received the lowest ratings, respectively. Discussions with BootUp facilitators revealed this may be due to the higher personal accountability that teachers feel with these methods because they require others to observe their teaching. In open-ended comments, teachers expressed a lot of initial hesitation at participating in these exercises. However, follow-up comments reveal that teachers who do participate in these activities end up appreciating them and losing some hesitation to be observed.

While the rating trend at each site generally rates BootUp elements in the same order, it is useful to analyze trends at individual sites to make targeted changes. For example, Rural_1 and Suburban_1 teachers appear to have rated almost all elements highly across the board, with the sharing of ideas even scoring a perfect 10/10 in Rural_1. In contrast, teachers in Urban_1, Rural_3 and Rural_4 appear to have been more conservative in their ratings. Urban_2 provided very high ratings for the hands-on learning and coder resources, but much lower ratings for model teaching and peer coaching. It might also be

best to compare overall ratings to Suburban_2' ratings, since all 2017-2018 ratings came from 22 of their teachers (compared to 40 ratings for 2018-2019).

4.5. Analysis of Teacher Comments Following each Professional Development

Following each professional development experience, BootUp facilitators provide teachers with the opportunity to provide feedback on their experience. Teachers respond to a single question that simply states, "*Please provide any suggestions, frustrations, confusions, and/or desires here.*" From 2018-2019, teachers provided 485 responses to this question. BootUp facilitators read these comments and use it as formative feedback for future trainings with specific groups.

I analyzed teachers' comments for trends over time and to see if there were common themes that may help to provide an overall evaluation for BootUp's trainings. If I were to characterize the responses into a single word, I would use, "**responsive**." Comments tended to follow trends depending on the group receiving the training. For example, many computing teachers in Suburban_2 and Rural_4 compared their trainings to those they received in 2017-2018, stating that they felt the trainings were much improved. Rural_2 and Urban_1 teachers were initially concerned with how they were going to integrate coding with their existing curricula. The vast majority of Rural_3 teachers indicated that they were a fairly tech-savvy group and that the first training went too slowly. Suburban_1 teachers were generally very pleased with their first training, as it was the first time they had ever been exposed to coding at all. In looking at later comments for each of these groups, it was clear that BootUp facilitators used this feedback to shape future trainings with each group. The Rural_3 group—initially the most vocal about the training moving along too slowly—nearly unanimously commented that the pace the second day was much quicker. Teachers recognized that this was in response to their shared concerns, as illustrated in the below comments:

> Today was fabulous. Thank you for taking the feedback about pacing yesterday and making immediate changes today. I really appreciated everything we got to work on today and felt that we had just the right of time to work, explore, create, share and then move on.
>
> The pacing was great today! I am still unsure of how to integrate the content this first year. I know later in the school year it will be easier to wrap my head around, but initially I feel that the students are going to need time to figure out scratch.
>
> I appreciate your consideration of pacing today. I was also excited to get into more challenging algorithms and assessment. Great day!

Meanwhile, other groups' comments from later trainings indicated that BootUp facilitators were responding to their individual needs. Rural_2 teachers comments at the 2nd training they received focused on how useful it was to work in grade-level groups. Many

teachers felt that this collaboration helped to address their concerns about integrating coding into their busy schedules (an oft-expressed concern following their first training).

> I loved being in a group with just our grade level! It made it much easier to bounce ideas off of each other and get practical ideas to take back to the classroom. I thought project one and two were maybe a little too similar (which was maybe the point!) but one could be eliminated if needed to save time. But, again, I loved having time to work on it and explore on our own as a way to learn.
>
> I really enjoy having this PD suggestion with my grade level colleagues, with a smaller group, too. We can really share ideas that are so applicable to our students. I like also taking this training in smaller steps. Then we can learn a bit more each time and practice with our students. Then come back together again to share what we have done, learn more and then extend learning with our students.
>
> It was awesome to have PD with just our grade level! Thank you for helping us individualize our curriculum to this age level!
>
> Love having our own kindergarten focused time--we are different than other grades and not everything they can do is applicable to K. Love the ideas, excited to implement some in my classroom in the next few weeks!

Teachers in Suburban_1, Suburban_2, Rural_1, and Urban_1 counties expressed their appreciation for the pacing and **hands-on** practice. They regularly praised the trainings and indicated they were an effective use of professional development time.

> I LOVED this training! Even though these were all day trainings, they went by fast because everything was so fun. Loved having the time to work on projects ourselves.
>
> Today I feel went really well and I learned many more blocks I can use when teaching Scratch. Many great ideas on being inclusive.
>
> This has been a great course. Each time I go away with a better understanding not only how to teach scratch, but how to teach in other areas. Techniques, tools and ideas are shared that are valuable in all teaching areas. I love that as teachers we can share, McKay gives us great ideas, and we learn from the Bootup material. Thanks for this opportunity!
>
> I'm so happy with this training! No suggestions. McKay is fabulous! Glad we are given time to try these things because it's becoming easier to use little by little!
>
> This is the best training I go to. Keep it up!
>
> I appreciate how the training's were modified to allow us more time to work at our own level (pick a project, and seek help as needed). I found this helpful to sort through topics I haven't had time to do otherwise.

> Although we spent a lot of time in projects, I appreciated the time because I was able to complete projects to actually use in my classes with my students rather than having to complete them at home.
>
> I like the hands on training. Learn by doing and collaborating.
>
> Fantastic! I loved being able to focus on Scratch projects. The hands on experience is priceless!
>
> This session was super fun. I got more ideas to take to the classroom in this session than the other 3!

Hands-on training and practice was so often praised that teachers also pointed out when they felt like the training did not provide the same opportunity as prior trainings.

> I know it's so important that we need the pedagogy and content, but this one wasn't as interactive.
>
> I felt like there was A LOT of time spent on unnecessary discussion. People were visiting with each other or surfing the internet waiting on the lesson to progress. In all honesty, I can read long passages of guidelines on my own time. I attend these trainings to learn how to do things in Scratch.

As trainings progressed, teacher comments tended to include more requests for how to use specific features of the software or for tips on how to handle specific classroom issues they had encountered throughout the year. Middle/Jr. High school teachers (mainly in Urban_2) were concerned that the trainings were targeted for younger grades. As BootUp continues to fine-tune its PD model, they may want to find ways to address teacher-specific needs in later trainings, while at the same time providing the same opportunity for hands-on training that teachers appear to like so much.

Overall, comments such as the following indicate that BootUp's model appears to be working for many teachers.

> (2nd PD) WONDERFUL CLASS, INFORMATIVE, I WENT FROM KNOWING NOTHING TO ACTUALLLY CODING. I REALLY APPRECIATE MCKAYS KNOWLEDGE AND PATIENCE AND THE EASE OF UNDERSTANDING EVERYTHING.
>
> (8th PD) I have really enjoyed the BootUp training sessions. Each time I leave I have a more clear understanding of coding concepts, ways to teach coding in Scratch and exciting projects for students to develop.
> I really appreciate the time invested in the Video walkthroughs and screen shots that are available, it gives me something to help teach the students and to refer to as I learn with them.
> I would like another year of BootUp training and would enjoy having the class time to learn and ask questions to prepare for future lessons.

At this time I do not have any confusions, when I come up on them I email McKay or check through the coder resources to figure out what I need to. I have enough to know how to problem solve.

5. The Effect of Learning Coding on Children

This section presents findings from secondary questions:

1. What is the effect of learning coding on students' computational thinking?
2. What is the effect of learning coding on students' attitudes toward coding?

5.1. Secondary Question 1 Findings: Computational Thinking

The Computational Thinking test (CTt) is one of the only tests that has been developed to measure young students' computational thinking (Román-González, Pérez-González, & Jiménez-Fernández, 2017). The CTt has been validated against other problem-solving tests and through expert review. It is a 28-item tests taken online. For each question, a graphic is presented that shows different actors. Students must solve a given problem that often involves moving the actor from one location to another. The CTt takes roughly one class period to complete.

The CTt was designed to scale well across grade-levels. It was originally tested in Spain and subsequently translated into British English, which is the version used with Suburban_2 school district students. The original test validation involved test-takers from 5th through 10th grades. Out of 28 questions, The average score for Spanish 5th graders was 13.09, 6th graders was 14.70, 7th & 8th graders was 16.24, and for 9th & 10th graders it was 18.05. It is against these scores that we place our expectations for Suburban_2 students' initial performance.

At the end of the second quarter of the 2018-19 school year, computing teachers in Suburban_2 School District administered the CTt to upper elementary students (grades 4-6). They again administered the exam at the end of the school year to gauge the extent to which student scores increased as a result of their computational training. Student scores on the exam did not count toward their grades in any class. They were encouraged to complete the test to the best of their ability in the time allotted (either 30 or 45 min. classes, depending on the school; average time to complete the test was under 20 min).

5.1.1. Demographics

On the initial assessment Over 4700 students from 23 different Suburban_2 county schools completed the initial CTt (See Table C.1. in Appendix C). There were 4684 usable responses, after dismissing results from 7th-8th grade students and from students with incomplete information. There was a high degree of parity between boys (50.28%) and girls (49.72%) who took the test. Scores were also fairly evenly distributed across the three grade levels (see Table 5.1.1).

On the year-end administration, there were 7405 usable responses provided by students from 33 Suburban_2 School District students from 4th-6th grade. Girls provided 51% of the final responses, while boys provided 49% (a slight reversal from the initial assessment).

Table 5.1.1
Distribution of CTt test-takers on initial and year-end CTt by grade and gender

	Initial Assessment			Year-end Assessment		
	Grade 4	**Grade 5**	**Grade 6**	**Grade 4**	**Grade 5**	**Grade 6**
Overall	1,484	1,574	1,626	2,494	2,636	2,756
Boys	713	816	826	1,238	1,363	1,426
Girls	771	758	800	1,256	1,273	1,330

5.1.2. CTt Scores

Table 5.1.2.a presents the raw test scores achieved by students, broken down by grade and gender. Suburban_2 county students scored higher initially per their grade level than their Spanish counterparts in 5th and 6th grades, performing at the expected grade level for the next grade (when compared to initial Spanish CTt validation scores). On average, students exhibited modest gains between the initial and year-end administrations, with 6th graders performing at the expected level for 7th graders by the end of the year.

Table 5.1.2.a
Comparison of students' initial and year-end CTt scores by grade and gender

	Initial Assessment			Year-end Assessment		
	Grade 4 (N = 1,484)	**Grade 5 (N=1,574)**	**Grade 6 (N=1,626)**	**Grade 4 (N=2,494)**	**Grade 5 (N=2,636)**	**Grade 6 (N=2,756)**
overall	13.35	14.88	16.05	13.825	15.820	16.510
boys	13.88	15.37	16.51	15.048	16.351	16.881
girls	12.87	14.36	15.58	13.825	15.252	16.112

Figure 5.1.2. demonstrates the distribution of scores across the entire spectrum of possible points. On average, boys performed nearly 1-point better on the exam than girls; this observation held true at each grade level across both administrations of the CTt. This finding is consistent with Moreno-León et al.'s (2017) findings, which indicated that this difference may be due to the emphasis on spatial reasoning in the CTt. Other research has also found that boys tend to score slightly better than girls on tests of spatial reasoning. That being said, this performance gap on computational thinking between

boys and girls is important to watch. Is it possible that, as more students participate in coding activities, the performance gap between genders will diminish?

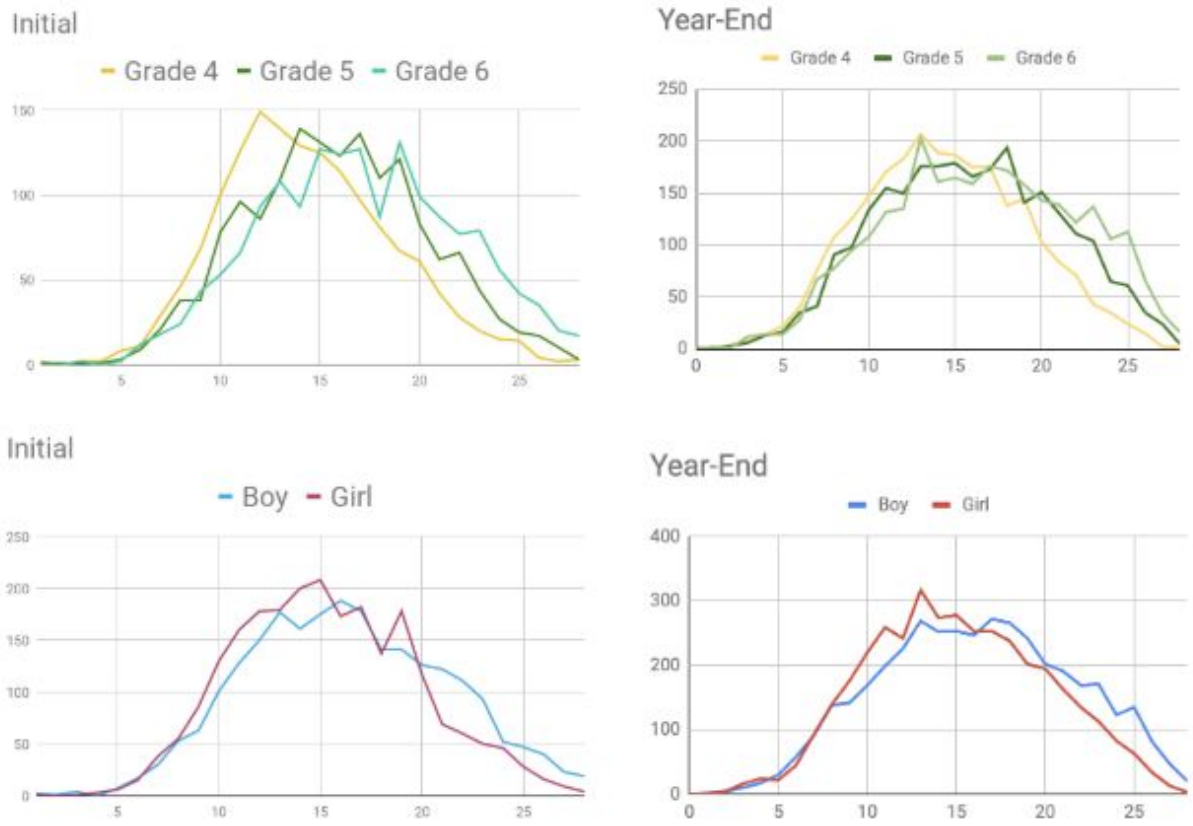


Figure 5.1.2. **Distribution of CTt scores by sub-group and administration**

While the performance gap between genders is certainly important to observe, there was another performance gap that is worth mentioning. Table 5.1.2.b compares the performance on the year-end exam between the students who also took the initial exam (n = 2565) and those who only took the year-end CTt (n = 5491). The effect of having taken the exam previously was greater than grade or gender effects. In fact, 6th graders performed on par with Spanish 9th graders by the end of the year. This difference may be due to several factors. The first consideration may be that students' familiarity with having taken the CTt earlier in the year helped them to remember the questions. This is known as a test-retest effect. However, research has demonstrated that allowing for sufficient time between administrations (around 8 weeks) eradicates the test-retest effect. In this case, it was four months or more between administrations. Furthermore, once students took the test, they never saw their scores, and neither did their teachers. I consulted with an educational measurement expert and he felt the likelihood of a test-rest

effect under these conditions was highly unlikely.Thus, it is unlikely that the difference between scores with the post-test only group is due to the test-retest effect.

Another possibility for the difference between test scores is that the teachers who administered the initial assessment were more proactive not only in administering the CTt, but also in their teaching of coding. Whatever the reason for the difference in scores, it will be important to continue to track these students' progress on their computational thinking to see if it continues to improve at this accelerated rate. Or, likewise, if the students of these teachers continue to outperform other students in the same district with similar demographics. If so, it would be worthwhile to study these teachers' practices to better understand why their students are outperforming their counterparts in computational thinking.

Table 5.1.2.b
Comparison of year-end CTt scores between those with a pre-assessment and those without

				Grade 4	Grade 5	Grade 6
pre/post				(N = 844)	(N = 823)	(N = 898)
	(N = 2565)	*Overall*	16.398	14.479	16.547	17.489
	(N = 1293)	*girls*	15.894	14.479	16.010	17.198
	(N = 1272)	*boys*	16.910	15.771	17.067	17.775
post only				(N = 1650)	(N =1813)	(N =1858)
	(N = 5491)	*Overall*	15.248	14.095	15.490	16.037
	(N = 2645)	*girls*	14.676	13.469	14.899	15.566
	(N = 2843)	*boys*	15.781	14.703	16.034	16.465

5.1.3. Performance Estimation

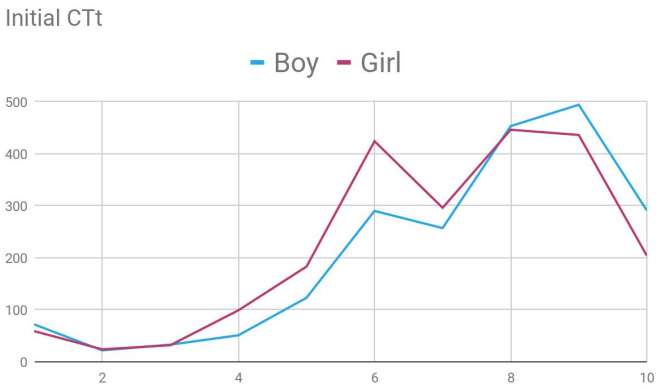


Figure 5.1.3.a. Students' estimation of how well they believe they performed on the CTt

In addition to basic demographics (gender and grade level), the final two questions on the CTt asked students to estimate how well they thought they did on the exam. This question was asked on a 0-10 point scale. Figure 5.1.3.a demonstrates that, in general, boys estimated their performance at a higher level than girls on the initial CTt.

I compared students' performance estimation with students' actual performance on the 28-item CTt by standardizing the CTt score to a 10-point scale by dividing their CTt scores by a factor of 2.8. Table 5.1.3.a demonstrates the average difference between students' actual score and estimated performance using this standardization. A negative score indicates that students performed that many points below where they predicted they would perform (on a 10-point scale).

Table 5.1.3.a
Average difference between students' actual and estimated performance on a 10-point scale

	Initial Administration			Final Administration		
	Grade 4	**Grade 5**	**Grade 6**	**Grade 4**	**Grade 5**	**Grade 6**
Overall	-2.36	-2.06	-1.93	-2.409	-2.066	-1.778
Boys	-2.41	-2.11	-2.00	-2.488	-2.051	-1.946
Girls	-2.32	-2.00	-1.87	-2.330	-2.081	-1.597

Curiously, even though boys outperformed girls by 1 question (on average), they also overestimated their performance at a greater rate than girls, with about 5% of boys in each sub-group overestimating their scores more than girls in the same sub-group. An important observation with these predictions is that all students appear to become more self-aware as they grow.

One limitation with this comparison is that the scales for estimating performance (10-point) were different than the actual CTt (28-point). To accommodate for this fuzziness, we included one additional question on the year-end administration: "Out of the 28 questions, how many do you think you got right?" This gives a more accurate and fine-grained prediction, which can be compared to students' actual scores (see Table 5.1.3.b). Again, older students and girls tended to be more accurate at predicting their scores (though there was no statistically significant difference between 5th grade boys' and girls' predictions). On average, 4th graders predicted they would score 4 points better than they did, while this gap narrowed to about 2 points better for 6th graders.

Table 5.1.3.b
Average difference between students' predicted and actual performance out of 28 points on the year-end CTt

	Grade 4	**Grade 5**	**Grade 6**
Overall	-4.101	-2.980	-2.299
Boys	-4.258	-2.925	-2.917
Girls	-3.947	-3.038	-1.636

5.2. Secondary Question 2 Findings: Student Attitudes Toward Coding

While the CTt seeks to measure the effect of learning to code on students' cognitive development, we also sought to better understand its effect on students' attitudes. For example, are students more likely to see themselves as coders or as using coding, in their future education and jobs? Do students see themselves as coders? How do they characterize coding? We sought to measure these and other attitudes students might have about coding.

To answer this question, we initially sought to use an existing scale. We searched scholarly literature for instruments intended to measure students' attitudes toward coding, computing, or computational thinking. We identified 16 scales meeting this criteria (see Appendix D, Table D.1). Unfortunately, as we examined each scale, we found them to be lacking overall in one area or another. For example, the majority of scales that measured attitudes toward STEM treated computing too generically, grouping it with all sorts of other technologies. Scales that did focus on coding specifically were typically developed for college-level students in computer science courses. In light of this shortcoming, we decided to create our own scale to measure elementary students' attitudes toward computing, with language that could be understood by students as young as 8 years old as well as those who had not yet had much exposure to coding (so that the scale could be used to measure changes from before to after learning to code).

To develop this scale, we curated questions from the 16 existing scales that we felt would help us to better understand young students' attitudes toward coding. We added our own questions to fill in areas where we felt there were gaps. This resulted in over 100 separate questions. We narrowed these down by having 4 different researchers prioritize the questions from 1-3 as being highly important or not important to ask. We then had 3 educational measurement experts read the questions and offer their feedback regarding the question's appropriateness and construction.

These efforts resulted in the Elementary Student Coding Attitudes Survey (ESCAS). The survey measures 6 constructs that contribute to a student's attitude toward coding: (a) self-efficacy, (b) interest, (c) usefulness, (d) perception of profession, (e) perception of gender, and (f) social value. While most questions consisted of Likert-type responses, questions that addressed social bias were open-ended (so that the question itself didn't bias students' responses). We piloted the scale with 324 4th-6th grade students in a local school district. Following the pilot, we analyzed students' open-ended responses for emergent categories. We then re-coded these into multiple-choice items so that the ESCAS could be used and easily analyzed at scale. The resulting ESCAS was comprised of 52 items.

5.2.1. ESCAS validation results

We administered the ESCAS to a sample of 5725 Suburban_2 School District students in 4th-6th grades who had participated in coding classes once a week throughout the school year. Inasmuch as we were validating an instrument, we did not collect any personally identifying information about students for this administration. All responses were anonymous. Thus, it is not possible to compare these students' scores to their CTt scores or any other measure. The goal of the validation was to create an instrument that could reliably provide answers about students' attitudes toward coding.

To validate the survey, we split the data into two groups. With the first group, we conducted a confirmatory factor analysis. With the second group, we conducted a structural equation model to verify the relationship(s) between the different constructs measured to represent students' attitudes toward coding. This model was based on initial CFA and SEM analyses conducted with pilot study data and the group 1 data. The final model is represented in Figure 5.2.1

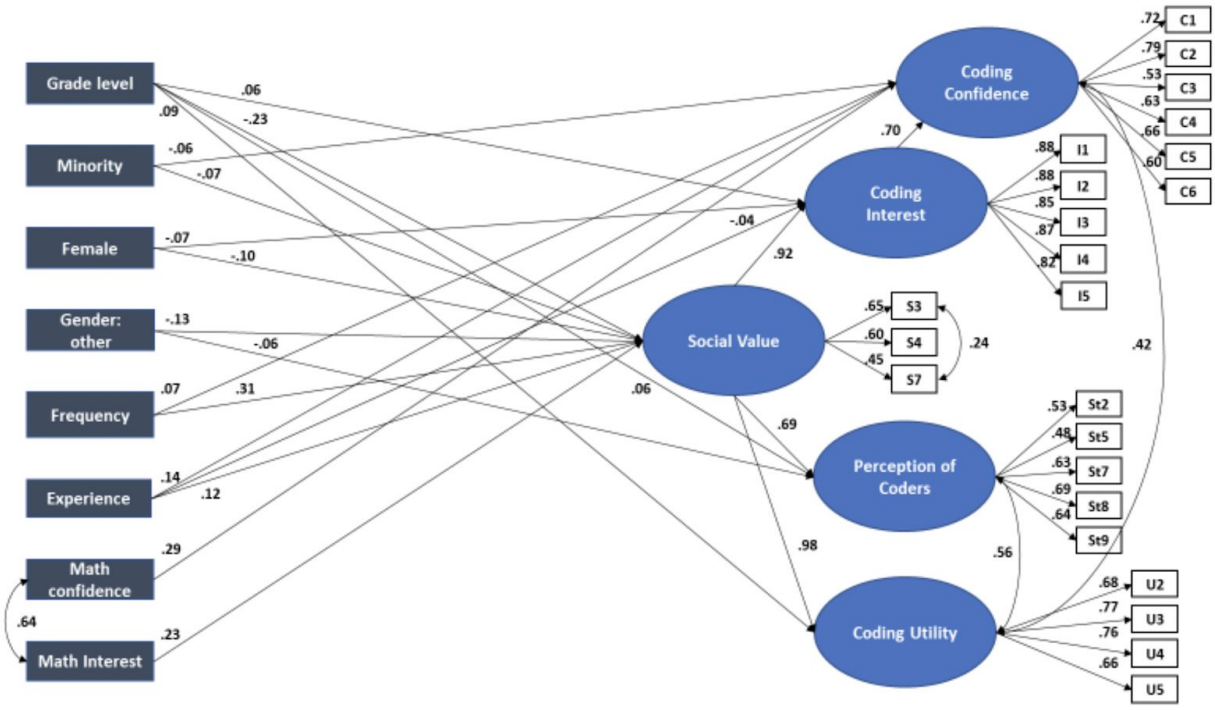


Figure 5.2.1. Final model of student attitudes toward coding as measured by the ESCAS

The resulting model revealed 5 factors as measured by 23 questions (see Appendix D, Table D.3) The most revealing result was that Coding Interest, Perception of Coders, and Coding Utility were all predicted by Social Value. In other words, the more that students felt their parents and friends valued coding, the greater interest they showed in coding themselves. What is just as telling is the social value item that is missing from this factor—teacher influence. That is, it did not matter if a student believed his or her teacher

thought coding was important. Thus, students with a social sphere of friends and parents who value coding more highly are more likely to value coding themselves.

Several other findings are worth mentioning.

Frequency Matters: We hypothesized that as coding frequency increased, coding attitudes would become more positive. Among all observable variables, frequency had the greatest influence on social value (β = .31), which in turn substantially influenced all other factors. Besides its indirect effect on coding confidence, coding frequency had an additional, direct, slightly positive effect on coding confidence (β = .07). As expected, coding experience also had a net positive effect on coding attitudes, with direct effects to social value (β = .12), coding confidence (β = .14), and coding interest (β = -.04).

Gender and Age Have Negative Effects: Female students demonstrated slightly lower coding interest (a small, but statistically significant result). Likewise, older students also valued coding less than younger students. This confirms findings from other research that indicates a bias toward (or against) coding as students grow older, particularly among females (i.e., females become less interested in coding). Future use of the ESCAS may serve as a measure to see if, over time, students who engage in coding develop these gender biases at a lesser rate than those who do not (or, hopefully, reverse the trend!).

Math Interest Matters: We hypothesized that math confidence and interest would predict coding confidence and interest. In our model, math confidence was the variable with the greatest predictive power for coding confidence (β = .29). Math interest had high predictive power for social value (β = .23), which substantially influenced coding interest (β = .92). This correlation between math and coding reinforces findings from others' research (Scherer, Siddiq, & Sánchez Viveros, 2018), and serves to strengthen the notion that those who feel comfortable with and interested in mathematics are more likely to also feel that way toward coding, even at the elementary level.

A full description of the development of the ESCAS has been submitted for review in an academic journal. A draft can be read at: https://www.researchgate.net/publication/335083392_Development_and_Analysis_of_the_Elementary_Student_Coding_Attitudes_Survey.

6. References

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

Reeve, S., Kitchen, E., Sudweeks, R. R., Bell, J. D., & Bradshaw, W. S. (2011). Development of an instrument for measuring self-efficacy in cell biology. *Journal of applied measurement*, *12*(3), 242-260. Retrieved from http://europepmc.org/abstract/med/22357126

Rich, P. J., Browning, S. F., Perkins, M., Shoop, T., Yoshikawa, E., & Belikov, O. M. (2018). Coding in K-8: International Trends in Teaching Elementary/Primary Computing. *TechTrends*. doi:10.1007/s11528-018-0295-4

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. doi:10.1016/j.chb.2016.08.047

7. Appendix A. Pre-Post Questions to Measure Teachers' Beliefs

Values:

1. Computing should be taught in elementary/primary school.
2. Learning about computing can help elementary students become more engaged in school.
3. Computing is like art, you are either born with the ability to think that way or you are not.
4. Computing content and principles CAN be understood by elementary school children.
5. My current teaching situation does NOT lend itself to teaching computing concepts to my students.
6. Knowledge of computer programming is NOT needed in most careers.
7. Providing more computing activities is NOT necessary to enrich my students' overall learning.
8. Computing is an important 21st-century literacy.
9. Computational thinking is an important part of today's science standards.
10. My current students are going to need to know how to code to remain competitive for jobs by the time they are adults.
11. Computing is NOT something that should be taught to special needs students.

CT Self-Efficacy:

1. When I'm presented with a problem, I have difficulty breaking it down into smaller steps.
2. I struggle to generalize solutions that can be applied to many different problems.
3. I am NOT good at solving puzzles.
4. I can read a formula (e.g., algorithm, equation, input/output process) and explain what it should do.

Coding Efficacy

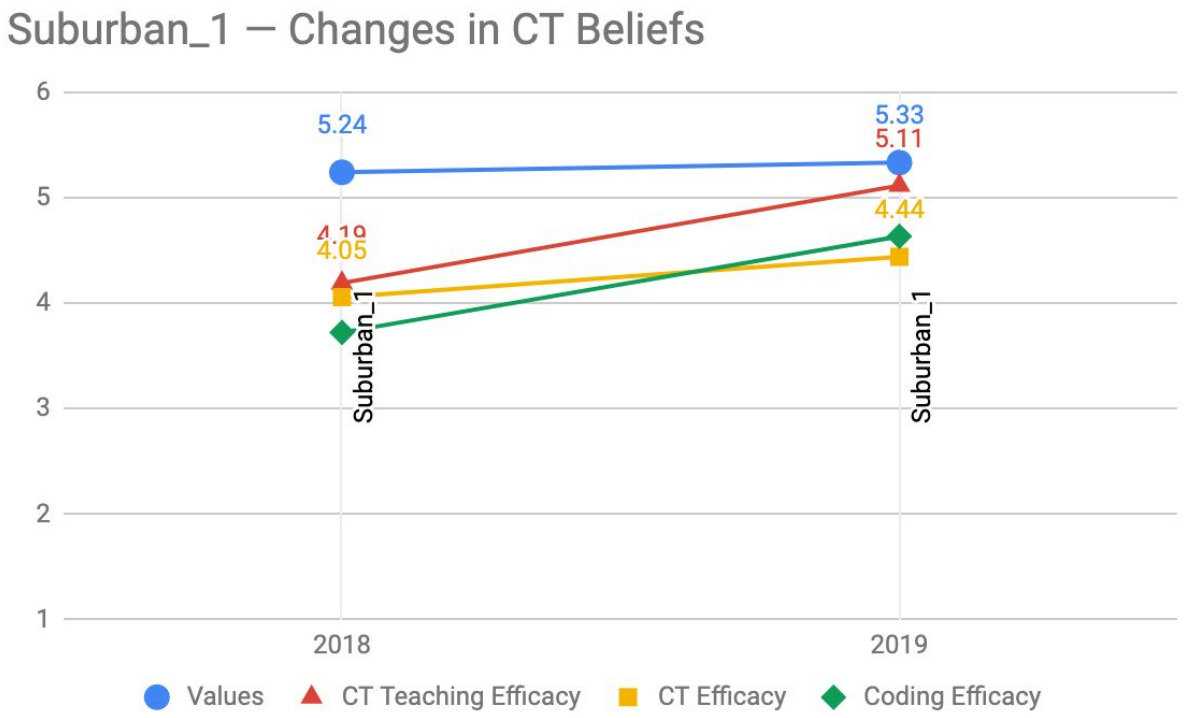
5. I can describe fundamental computing concepts (e.g., loops, variables, algorithms, conditional logic).
6. I can correct mistakes in the coding of a computer program on my own.
7. I can look at a process and figure out how to make it more efficient.
8. I can suggest different solutions in order to solve coding problems.
9. I can look at a computer program and explain the purpose of each command.
10. The thought of having to write a computer program intimidates me.
11. I am good at finding patterns in data.
12. I can apply Boolean logic (e.g., IF, AND, NOT, OR) to solve problems with multiple conditions.
13. I struggle to identify where and how to use variables in the solution of a problem.

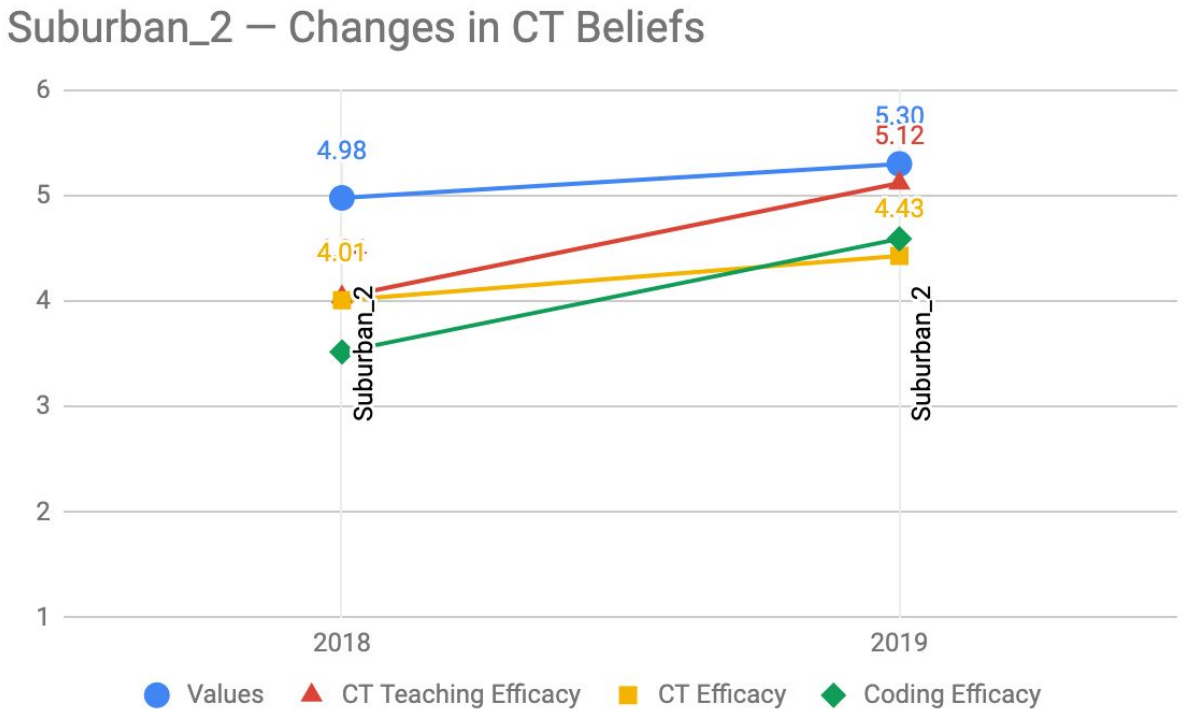
14. I can plan out the logic for a computer program even if I don't know the specific programming language.

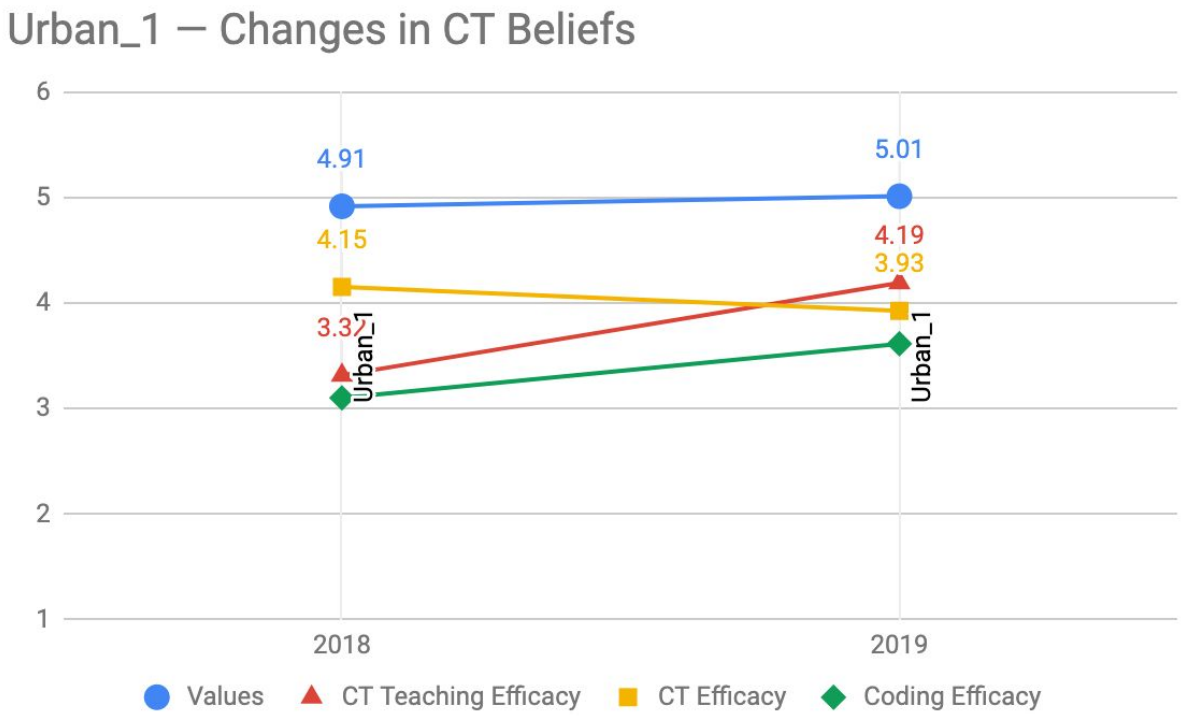
CT Teaching Efficacy:

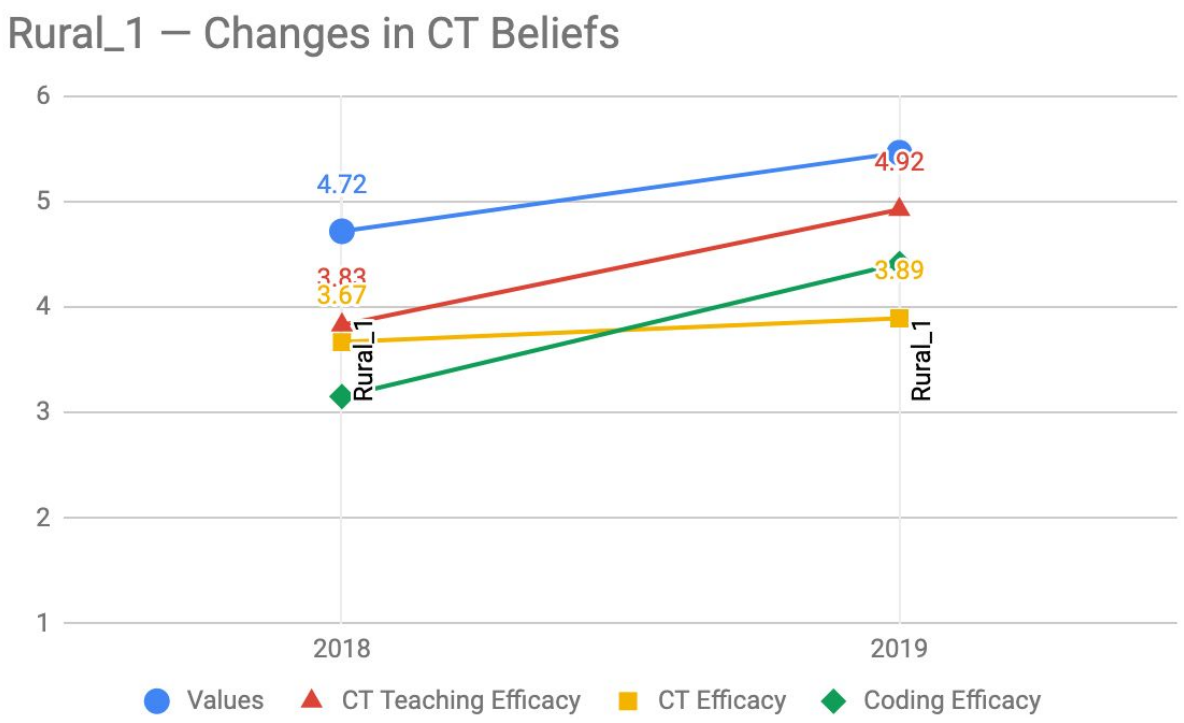
1. I can explain basic computing concepts to children (e.g., algorithms, loops, conditionals, functions, variables, debugging, pattern-finding).
2. I can help students debug their computer programs.
3. I can find uses for computer programming that are relevant for students.
4. I can integrate computer programming into my current curriculum.
5. I know where to find the resources to help students learn to code.
6. I believe that I have the requisite computer programming skills to integrate computing content into my class lessons.
7. I can recognize and appreciate computing concepts in all subject areas.
8. I can create computing activities at the appropriate level for my students.
9. I can explain computing concepts well enough to be effective in teaching computing.
10. I can explain how computing concepts are connected to daily life.
11. I can develop and plan effective computing lessons.

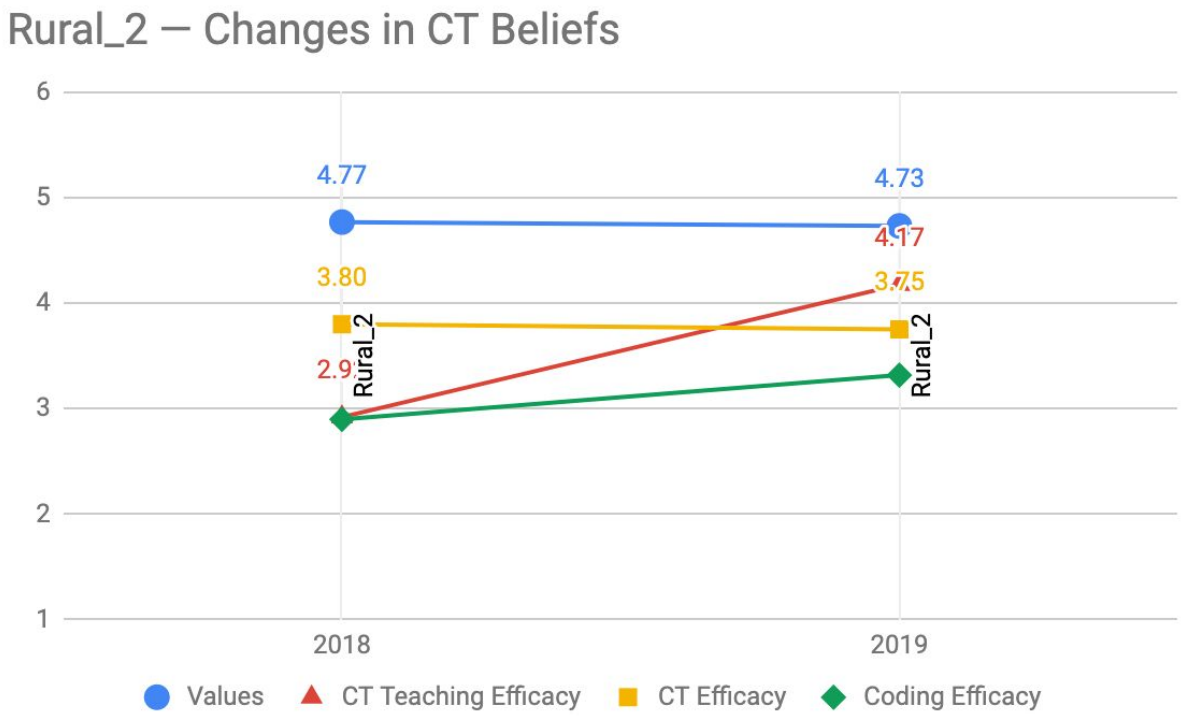
8. Appendix B. Pre-Post Belief Scores by District

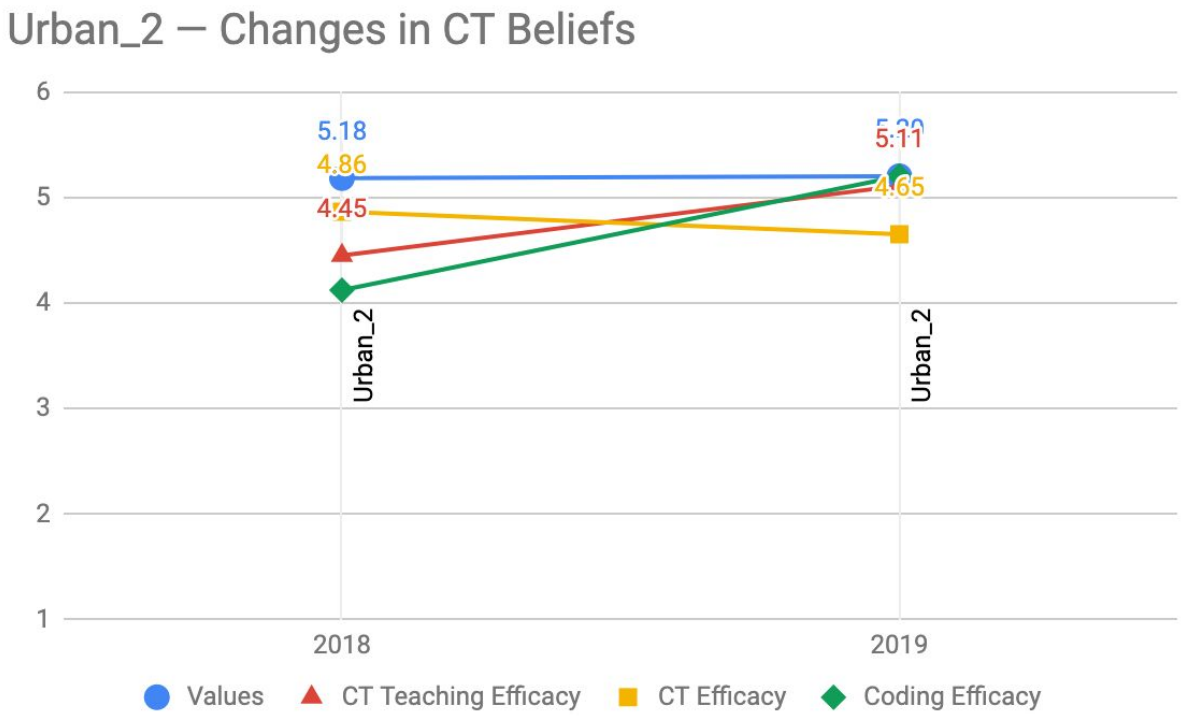


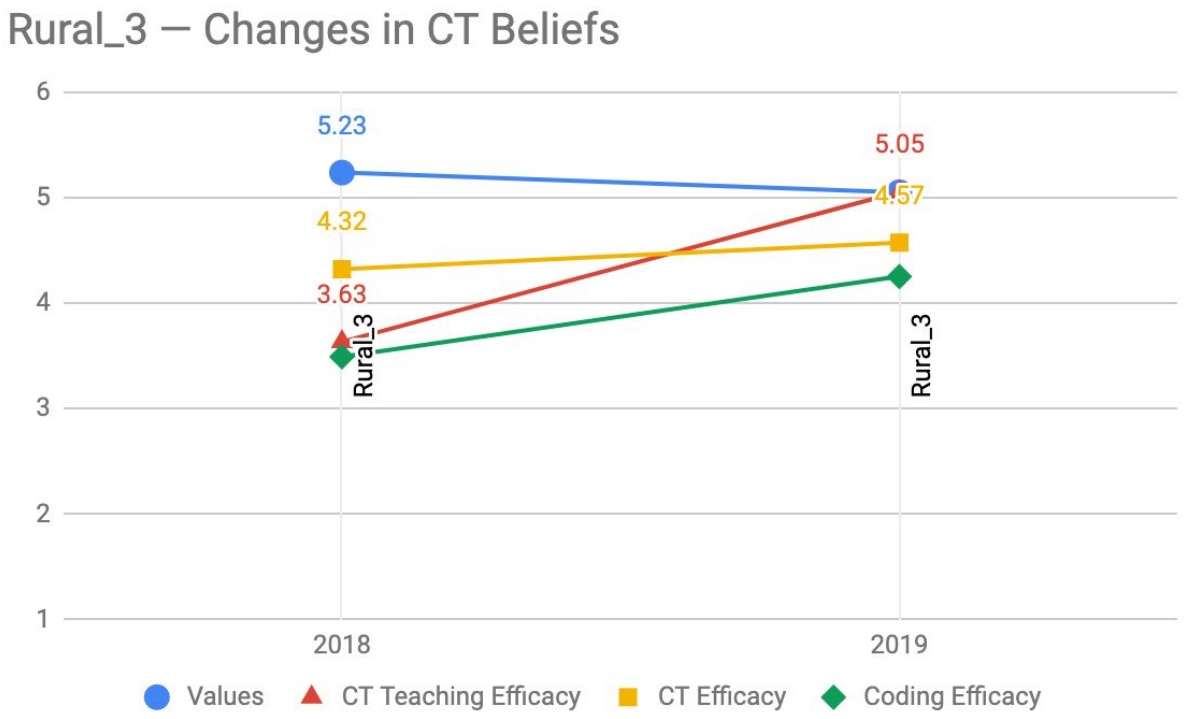


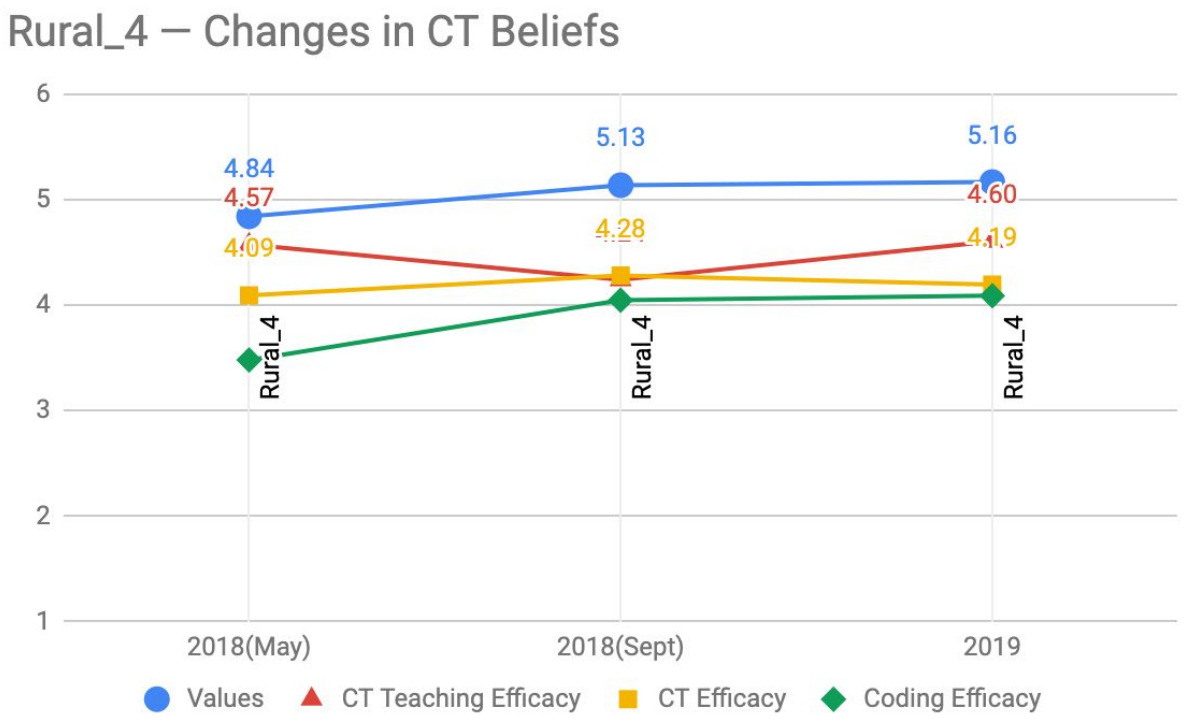

Urban_1 — Changes in CT Beliefs
6
5
4
3
2
1
4.91
5.01
4.15
4.19
3.93
3.32
Urban_1
Urban_1
2018
2019
Values
CT Teaching Efficacy
CT Efficacy
Coding Efficacy


Rural_1 — Changes in CT Beliefs
6
5
4
3
2
1
4.72
4.92
3.83
3.67
3.89
Rural_1
Rural_1
2018
2019
Values
CT Teaching Efficacy
CT Efficacy
Coding Efficacy


Rural_2 — Changes in CT Beliefs
6
5
4
3
2
1
4.77
4.73
4.17
3.80
3.75
2.9
Rural_2
Rural_2
2018
2019
Values
CT Teaching Efficacy
CT Efficacy
Coding Efficacy


Urban_2 — Changes in CT Beliefs
6
5
4
3
2
1
5.18
5.11
4.86
4.65
4.45
Urban_2
Urban_2
2018
2019
Values
CT Teaching Efficacy
CT Efficacy
Coding Efficacy


Rural_3 — Changes in CT Beliefs
6
5
4
3
2
1
5.23
5.05
4.57
4.32
3.63
Rural_3
Rural_3
2018
2019
Values
CT Teaching Efficacy
CT Efficacy
Coding Efficacy


Rural_4 — Changes in CT Beliefs
6
5
4
3
2
1
4.84
5.13
5.16
4.57
4.60
4.09
4.28
4.19
Rural_4
Rural_4
Rural_4
2018(May)
2018(Sept)
2019
Values
CT Teaching Efficacy
CT Efficacy
Coding Efficacy

9. Appendix C. CTt Response data

Table C.1.

Distribution of test-takers across schools for the INITIAL administration of the CTt

School	Total	Boy	Girl	Grade 4	Grade 5	Grade 6
Elementary 1	193	94	99	61	73	59
Elementary 6	267	134	133	97	95	75
Elementary 35	286	152	134	96	98	92
Elementary 7	95	51	44	24	34	37
Elementary 8	234	112	122	81	78	75
Elementary 9	291	157	134	87	98	106
Elementary 10	117	57	60	43	35	39
Elementary 36	23	15	8	4	19	0
Elementary 11	176	90	86	54	63	59
Elementary 13	106	49	57	24	46	36
Elementary 14	212	96	116	41	76	95
Elementary 37	302	152	150	91	114	97
Elementary 15	246	111	135	73	80	93
Elementary 16	173	80	93	56	52	65
Elementary 38	169	90	79	46	66	57
Elementary 19	302	145	157	106	83	113
Elementary 20	221	107	114	94	51	76
Elementary 23	247	129	118	76	79	92
Elementary 24	169	82	87	54	64	51
Elementary 26	228	123	105	76	72	80
Elementary 27	51	23	28	14	17	20
Elementary 28	298	160	138	98	101	99
Elementary 32	218	115	103	72	62	84
Elementary 39	60	31	29	16	18	26
Total	4624	2324	2300	1468	1556	1600

Table C.2.
Distribution of test-takers across school for the YEAR-END administration of the CTt

School	Total	Boys	Girls	Grade 4	Grade 5	Grade 6
Elementary 1	227	113	114	70	86	71
Elementary 2	171	83	88	54	49	68
Elementary 3	403	235	168	76	156	171
Elementary 4	392	192	200	148	134	110
Elementary 5	60	32	28	0	37	23
Elementary 6	310	160	150	108	105	97
Elementary 7	181	97	84	54	66	61
Elementary 8	192	95	97	56	57	79
Elementary 9	310	162	148	100	109	101
Elementary 10	441	216	225	141	144	156
Elementary 11	237	115	122	66	91	80
Elementary 12	386	202	184	126	134	126
Elementary 13	197	92	105	77	49	71
Elementary 14	260	127	133	87	72	101
Elementary 15	222	103	119	81	65	76
Elementary 16	185	83	102	57	63	65
Elementary 17	226	107	119	80	61	85
Elementary 18	362	194	168	96	136	130
Elementary 19	330	150	180	116	99	115
Elementary 20	275	137	138	108	78	89
Elementary 21	1	1	0	0	1	0
Elementary 22	239	135	104	65	78	96
Elementary 23	299	156	143	94	98	107
Elementary 24	214	105	109	75	67	72
Elementary 25	318	162	156	103	99	116
Elementary 26	230	127	103	54	94	82
Elementary 27	157	81	76	53	62	42
Elementary 28	314	173	141	110	103	101
Elementary 29	4	1	3	1	1	2
Elementary 30	135	71	64	46	42	47
Elementary 31	127	70	57	48	40	39
Elementary 32	230	123	107	72	69	89
Elementary 33	250	126	124	72	91	87
Elementary 34	1	1	0	0	0	1
Totals	7405	3777	3628	2350	2476	2579

10. Appendix D. Measuring Student Attitudes toward Coding

Table D.1

Validated Scales to Assess Students' STEM Self-efficacy, Attitudes, and Perceptions

Authors, year	Name	Population	Subject	Constructs	Items	Scale	N
Dorn & Tew (2015)	Computing Attitudes Survey (CAS)	College students	Computer science	Problem solving transfer, program solving strategies, problem solving fixed mindset, interest, real-world connections	26	Five-point	794
Hoegh & Moskal (2009)	Computing Survey	Colorado School of Mines undergraduate students	Computer Science	Confidence, interest, perceptions of gender, usefulness, perceptions of profession	38	Four-point	276
Ramalingam & Wiedenbeck (1998)	Computer Programming Self Efficacy Scale (CPSES)	University students	C++ program-ming	Independence and persistence, complex programming, self-regulation, simple programming	32	Seven-point	421
Washington, Grays, & Dasmohapatra (2016)	Computer Science Cultural Attitude and Identity Survey (CSAIS)	Undergraduate students of color	Computer science	Confidence, interest, gender, professional, identity	40	Four-point	65
Forssen, Moskal, & Harringer (2011)	Information Technology (IT) Attitude Survey	High school students in summer IT program	Informa-tion technology	General interest, gender stereotypes	20	Four-point	142

Hirsch, Gibbons, Kimmel, Rockland, & Bloom (2003)	High School Students' Attitude to Engineering and Engineering Self-Efficacy	High school students in funded summer program	Engineer-ing	Positive aspects of engineering, negative opinions of engineering, interest, job issues	33	Six-point	317
Mahoney (2010)	Student Attitudes Toward STEM	High school students (grades 9-12)	Science, technology, engineer-ing, mathema-tics	Awareness, perceived ability, value, commitment	96 (24 per content area)	Four-point	378
Erkut & Marx (2005)	Attitudes toward Engineering, Math, and Science	Eighth graders	Math, science, engineering	Attitudes toward math, attitudes toward science, attitudes toward engineering	35	Five-point	436
Gibbons, Hirsch, Kimmel, Rockland, and Bloom (2004)	The Middle School Students' Attitude to Mathematics, Science and Engineering Survey	Middle school students (Grades 5-8)	Mathema-tics, science and engineering	Interest (stereotypic), Interest (non-stereotypic), positive opinions, negative opinions, problem solving, technical skills	35	Six-point	1701
Hirsch, Carpinelli, Kimmel, Rockland, & Bloom (2007)	Adapted Middle School Students' Attitude to Mathematics, Science and Engineering Survey	Middle school students (Grades 5-8)	Math, science, engineering	Interest (stereotypic), Interest (non-stereotypic), positive opinions, negative opinions, problem solving, technical skills, engineering	36	Six-point	890

Kukul, Gökçearslan, and Günbatar (2017)	Computer Programming Self-Efficacy Scale	Middle school students	Program-ming	Self-efficacy	31	Five-point	233
Owen et al. (2008)	Revised Simpson-Troost Attitude Questionnaire (STAQ-R)	Middle school students (grades 6-8)	Science	Motivating science class, self-directed effort, family models, science is fun for me, peer models	22	Five-point	1754
Chambers (1983)	Draw A Scientist Test (DAST)	Elementary students (grades K-5)	Scientists	Stereotypic perceptions	1	Open-ended	4807
Hansen et al. (2017)	Draw-a-Computer-Scientist Test (DACST)	Children (grades 4-6)	Computer scientists	Perceptions	1	Open-ended	185/87
Knight & Cunningham (2004)	Draw an Engineer Test (DAET)	Grades 3-12	Engineers	Perceptions	5	Open-ended	384
Kong, Chiu, and Lai (2018)	Programming empowerment survey	Primary school students (grades 4-6)	Programming	Interest, collaboration, meaningfulness, impact, creative self-efficacy, programming self-efficacy	23	Five-point	287

Table D.2
ESCAS Constructs Assessed by Existing Scales

Authors, year	Self-Efficacy	Interest	Useful-ness	Perception of profession	Perception of gender	Social value
Scales for University Students						
Dorn & Tew (2015)		x				
Hoegh & Moskal (2009)	x	x	x	x	x	
Ramalingam & Wiedenbeck (1998)	x					
Washington, Grays, & Dasmohapatra (2016)	x	x		x	x	
Scales for High School Students						
Forssen, Moskal, & Harringer (2011)		x			x	
Hirsch, Gibbons, Kimmel, Rockland, & Bloom (2003)		x	x	x		
Mahoney (2010)	x	x	x			
Scales for Middle School Students						
Erkut & Marx (2005)	x	x	x			
Gibbons, Hirsch, Kimmel, Rockland, and Bloom (2004)		x	x	x		x
Hirsch, Carpinelli, Kimmel, Rockland, & Bloom (2007)		x	x	x		
Kukul, Gökçearslan, and Günbatar (2017)	x					
Owen et al. (2008)		x				x
Scales for Elementary School Students						
Chambers (1983)				x		
Hansen et al. (2017)				x		
Knight & Cunningham (2004)				x		
Kong, Chiu, and Lai (2018)	x	x	x			

Table D.3
Final 5 factors and 23 items measured by by the ESCAS

Factor	**Item**	
Coding Confidence	C1	I can learn to code.
	C2	I am good at coding.
	C3	I am good at problem solving.
	C4	I can write clear instructions for a robot or computer to follow.
	C5	If my code doesn't work, I can find my mistake and fix it.
	C6	I've been told I would be good at coding.
Coding Interest	I1	I like coding, or I think I would like coding.
	I2	I would like to learn more about coding.
	I3	Solving coding problems seems fun.
	I4	Coding is interesting.
	I5	I would like to study coding in the future.
Utility	U2	I can use coding skills in other school subjects.
	U3	Knowing how to code will help me to create useful things.
	U4	Knowing how to code will help me solve problems.
	U5	I think I will need to understand coding for my future job.
Social Influence	S3	My friends think coding is cool.
	S4	My parents think coding is important.
	S7	I am friends with kids who code.
Perceptions of coders	S2	Kids who are good at coding are smarter than average.
	St5	Kids who code enjoy doing sports.
	St7	Coders are good at math.
	St8	Coders are good at science.
	St9	Coders are good at language arts.