Athens Journal of Sciences- Volume 6, Issue 2 – Pages 97-122

The Effects of Computational Thinking Professional Development on STEM Teachers' Perceptions and Pedagogical Practices

By Sarah Haines[*], Michael Krach[†], Arkhadi Pustaka[‡], Qing Li[♦] & Laila Richman[**]

The authors investigated the impact of exposure to computational thinking activities and professional development on inservice teachers' perceptions and teaching practices in elementary and secondary school education. The participants of this 2017/2018 research study were STEM teachers from the Baltimore County (Maryland) Public Schools in the United States of America. The major focus of this study was on the impact of professional development activities on the inclusion of computational thinking activities in the k-12 mathematics and science classrooms. The analysis of the data indicated that most of the participating teachers felt that the professional development activities were valuable and made a positive impact on the quality and quantity of computational thinking activities they implemented in their classrooms. Most of these teachers stated that they would implement computational thinking activities on a weekly or monthly basis. Based on the analysis of the data and the results of this study, a list of future research questions is included.

Keywords: *Computational Thinking, Mathematics Education, Science Education, STEM, Professional Development.*

Introduction

In order for CT integration into k-12 mathematics and science classrooms to be successful, teachers at both the preservice and inservice levels must be provided with the appropriate coursework and professional development (PD) to allow them to master the pedagogical skills involved. Barr and Stephenson (2011) recommend the inclusion of a CT class across disciplines in teacher preparation programs. They further recommend providing inservice teachers with PD opportunities in the use of CT in k-12 classrooms. Research shows that education majors' views toward CT are more favorable if they are exposed to CT during their undergraduate careers. Additionally, they are more likely to use CT in their own classrooms, and see the applications to k-12 education more readily than those not exposed to CT in their teacher preparation programs (Yadav et al. 2011, Yadav et al. 2014).

Wozney et al. (2006) state that one of the greatest predictors of the extent of teachers' technology use and integration of CT in the classroom is their

[*] Professor, Towson University, USA.
[†]Professor, Towson University, USA.
[‡]Towson University, USA.
[♦]Professor, Towson University, USA.
[**]Associate Dean, College of Education, Towson University, USA.

https://doi.org/10.30958/ajs.6-2-2 doi=10.30958/ajs.6-2-2

belief that they can reach their instructional goals using this approach. This finding suggests that providing PD that assists teachers in gaining successful personal experiences in CT will help in this endeavor. We designed our PD based on best practices in the literature:

- Giving teachers time to use technology and to become comfortable with the CT approach through the online modules (Somekh 2008).
- Focusing new uses on teachers' immediate needs (Kanaya et al. 2005, Zhao and Cziko 2001).
- Starting with small successful experiences, through the online modules and in person PD (Ottenbreit-Leftwich 2007).
- Establishing a professional learning community (Putnam and Borko 2000).
- Situating professional development programs within the context of teachers' ongoing work (Cole et al. 2002, Snoeyink and Ertmer 2001/ 2002).

The focus of this study is on professional development in computational thinking for inservice teachers of science and mathematics. Our primary focus is to answer the following research question:

- ➢ What is the impact of professional development activities on the inclusion of CT activities in k-12 mathematics and science classrooms?

Literature Review

According to Wing's seminal manuscript on computational thinking (2006), "computational thinking is a fundamental skill for everyone, not just for computer scientists. Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science" (p. 33). Over the past few decades, advances in computer technology have revolutionized the STEM fields. The application of computer technology to virtually every field of study has changed the way STEM research is done today. Wing observes that the application of computer technology in the STEM fields facilitates the spread of (and the need for) computational thinking (CT) skills in the STEM disciplines. Barr et al. (2011) expand upon this foundational definition of CT, providing educators with an operational definition of CT that can be used to build on skills across grade levels and content areas. They define CT as a problem-solving process involving several steps: formulating a problem in such a way that the use of computer technology can help us solve it; analyzing data and representing that data through models or simulations, identifying possible solutions to the problem posed, and generalizing this process to a wide variety of situations and issues.

Following the premise of teaching STEM content as the STEM disciplines are practiced in real-world settings, it then makes sense that STEM teaching practices must also evolve. In the United States, many states have adopted the Next Generation Science Standards (NGSS, National Research Council 2013). The NGSS are k-12 science content standards. These standards set the expectations for what students should know and be able to do. The standards are designed to give local educators the flexibility to create classroom learning experiences that will stimulate student interest in science and prepare them for college, careers, and citizenship. The NGSS include the science and engineering practice of "Using Mathematics and Computational Thinking". According to the NGSS, the integration of quality mathematical and computational thinking can (and should) enhance science instruction. Scientists and engineers use computational thinking processes as tools for representing relationships between and among variables. Computational thinking can help scientists and engineers accomplish tasks such as simulating phenomena, analyzing data sets, and studying a multitude of quantitative relationships appropriate to the discipline.

The mathematics counterpart to the NGSS in the United States is the Common Core State Standards-Mathematics (CCSS-M). These standards state what each student in grades k-12 should know and be able to do in mathematics. Within the CCSS-M, there are no direct references with respect to integrating CT into the current framework (content clusters) for students in grades k-12 (CCSSI 2010). However, based on an understanding of the CCSS-M and the authors' working definition of CT (see Appendix 4), mastering concepts and skills in the following areas can certainly be considered major contributors to the development of a student's CT skills: generalize and analyze patterns, reason specifically and abstractly (especially with ratios and proportions), represent and interpret data, solve real-world problems. Since the task of integrating CT into existing curricula should be one of the many current teacher responsibilities, the teacher must be extremely knowledgeable with respect to mathematics content as well as how to best integrate CT into the curriculum. Websites, such as http://ctmath.ca/ (Computational Thinking in Mathematics Education), which provide effective classroom-tested CT activities, would be extremely beneficial for teachers in grades k-12 to utilize.

Besides the inclusion of CT in the NGSS science and engineering practices, CT is linked to science in other important ways. For example, Barr and Stephenson (2011) align core computational thinking concepts and capabilities with scientific experimentation. For example, the CT concept of abstraction is aligned with the science skill of building models, and the CT concept of algorithms is aligned with the scientific skill of creating and following sound experimental procedures. Similarly, the authors link data collection with collecting data in a scientific experiment; data analysis by analyzing the data collected during experimentation; automation by using probeware in a scientific setting; parallelization by running two scientific experiments with different parameters at the same time; and simulation by simulating solar system movement.

Sengupta et al. (2013) connect abstractions involved in computational thinking with the thought and skill processes involved in scientific inquiry. The authors draw similarities between algorithm design and the scientific process of developing mechanistic reasoning and explanations, or constructing models. In the same manner, test driven software development is seen as a parallel process to the scientific processes of hypothesis testing or validating/testing models.

In k-12 mathematics, many of the definitional components of CT can be directly related/linked to concept and skill areas in the CCSS-M. For example, algorithmic thinking can be linked to the study and analysis of computational algorithms and to the introduction and use of a problem solving protocol, such as Polya’s Problem Solving Model (Polya 1957), logical thinking and rigorous argument and proof can be linked to the study, analysis, and construction of both informal and formal proofs, and the collection and analysis of relevant data, and abstraction can be linked to the introduction and study of pattern recognition, creation, and analysis.

Baytak and Land (2011) found that the use of Scratch (scratch.mit.edu) increased computational skills in 5^{th} grade students. Lambert and Guiffre (2009) found that the use of Scratch improved elementary school students’ attitudes about computing and computer science.

The eight CCSS-M Standards of Mathematical Practice (SMP) can also be directly linked to CT, especially with respect to solving both traditional and non-traditional problems (see Table 1).

Table 1. *Standards of Mathematical Practice*

SMP#1	Make sense of problems and persevere in solving them
SMP#2	Reason abstractly and quantitatively
SMP#3	Construct viable arguments and critique the reasoning of others
SMP#4	Model with mathematics
SMP#5	Use appropriate tools strategically
SMP#6	Attend to precision
SMP#7	Look for and make us of structure
SMP#8	Look for and express regularity in repeated reasoning

Each of these SMPs can be linked to a current definition of CT, of which there are many. If one accepts the authors’ working definition of CT, then these eight “habits of mind” can be critical to the understanding of the proposed problem and instrumental to the creation of a solution to the problem.

The use of open-ended problems or scenarios encourages complete and meaningful responses, which requires the understanding of specific mathematical concepts and skills, the use of a problem solving protocol, constructing generalizations and proofs, modeling, and utilizing algorithms, all of which are components of CT.

The authors have established the ways in which CT is connected to the mathematics and science disciplines. Given that this connection exists, are there benefits associated with integrating CT into mathematics and science teaching? The answer to this question is multi-faceted. The nature of CT as described above places this way of thinking and problem-solving into the higher levels of Bloom’s taxonomy (Bloom et al. 1956, Gal-Ezer and Stephenson

2010). Besides encouraging higher level thinking in students, there are several practical reasons for integrating CT into science and mathematics classrooms. Weintrop et al. (2016) found that including CT in science classrooms builds on the interconnected relationship between CT and science. It also addresses practical concerns of reaching students - students must enroll in a certain number of science classes to graduate, but computer science is most often an elective course. If students are exposed to CT in their science courses, there is no need to worry about lack of computer science course offerings or students not electing to take the course. Moreover, currently, many schools do not have certified teachers in the area of computer science, but most do have teachers certified in the science and mathematics disciplines. These teachers should be more comfortable with the content, since CT aspects are embedded into their content areas. Last, integrating CT into science content courses means that the teaching methodology will more closely mirror the methods used by scientists in the field ("doing science the way it is done in the real world"). Jona et al. (2014) support these arguments. The authors found that CT interest and appeal is higher if the CT content is embedded into STEM courses instead of a stand-alone computer science course. Further, the integration of CT in science and mathematics classrooms has been shown to enhance the learning of STEM content (National Research Council 2011, Repenning et al. 2010, Sengupta et al. 2013, Wilensky and Reisman, 2006).

The integration of CT can be an effective instructional approach for learning science and mathematical concepts that students traditionally find difficult (Sherin 2001, Hambrusch et al. 2009, Blikstein and Wilensky 2009, di Sessa 2000, Kynigos 2007). Research suggests that programming in context (i.e., within an area of STEM discipline) makes it easier to learn. Hambrusch et al. (2009) found that using CT in college-level science classes with non-computer science majors resulted in higher learning gains and increased student engagement. Similar results have been reported for k-12 students. For example, Calao et al. (2015) found that in a sixth-grade mathematics class with CT embedded activities (for example, algorithmic thinking), student comprehension increased compared to a control group.

Methodology

Participants

At the beginning of the program, the participants consisted of 28 teachers. For various reasons, there was attrition and the final number of participants for this study consisted of 17 teachers. The participants taught in grades Kindergarten (n = 1), 1st (n = 1), 2nd (n = 2), 4th (n = 1), 5th (n = 1), 6th (n = 1), 7th (n = 1), 9th (n = 1); and several taught students in multiple grades: Kindergarten - 5th (n = 2), 9th - 12th (n = 4), and 10th - 12th (n = 2). The participants taught mathematics (n = 3), science (n = 4), both mathematics and science (n = 5), non-STEM related courses (n = 4), and special education (n = 1).

Description of Professional Development

The PD sessions focused on preparing teachers to integrate computational thinking and computer science activities, aligned with state standards including NGSS & Maryland College and Career Ready Standards (MCCRS, i.e. Common Core State Standards), into their daily lesson plans. Participants were provided with activities that supported their ability to serve as teacher leaders related to integrating computational thinking and computer science in their schools and district. The professional development sessions consisted of four main parts: 1) Introductory Session, 2) Online Modules, 3) Summer Institute, and 4) Colloquium. The three-hour Introductory Session was conducted face-to-face in early Spring 2017. Then, participants completed three online modules during Spring – Summer 2017 to learn the basics of computational thinking and coding using Star Logo Nova. All of the online modules were offered using EdX, a massive open online course provider. Next, during the Summer Institute 2017, participants attended a series of workshops to work collaboratively on the fourth module in order to apply computational thinking skills learned from the online modules and then to create lesson plans that incorporated CT in their classrooms. In addition, the participants completed a fifth online module focused on enhancing teachers' understanding of the content and pedagogical knowledge related to CT integration. Scratch, another software application, was introduced and used to support diverse learner needs. Finally, the participants were expected to implement their lesson plans and to report on their experiences during the 2018 Spring Colloquium.

Data Analysis

<u>Discussion Board Posts</u>

Participants were given access to an **edX** (for a detailed description see edge.edx.org) course site to access and provide reflections on the modules they were completing. For a sample list of discussion prompts, see Appendix 1. Science and mathematics content specialists, who were also completing the modules at the same time as the participants, responded daily to the discussion board posts with additional probing comments and questions that were intended to elicit further pedagogical ideas from the participants.

The content specialists used a modified version of Hord et al.'s stages of concern for adult learners (1987) to categorize and analyze the discussion posts. Hord et al. identified seven categories in their original study. However, due to the nature of the current study, the authors dropped one category and redefined the remaining six, and included an "Other" category, to better reflect the issues that were being reported in the Discussion Board Posts by the participating teachers (see Appendix 2 for a description of each category). To control for inter-rater differences in reliability, discussion posts were analyzed concurrently by the content specialists. The seven categories are described below.

- **Awareness:** teachers have an awareness of CT, but are not yet fully comfortable with it.
- **Informational:** teachers are comfortable with CT and are more interested in becoming involved with it.
- **Management:** teachers are fully committed to CT, but their attention is focused on the processes and tasks involved in implementing it.
- **Consequence:** teachers focus on the impact that CT has on their students' achievement.
- **Collaboration:** teachers have a need to collaborate and share ideas concerning CT.
- **Refocusing:** teachers explore current and future ways to incorporate CT into their lesson plans.
- **Other:** response does not fit into any of the above categories.

Professional Development Surveys

At the conclusion of the summer PD sessions, a seven question Likert-type instrument (referred to as Survey #1) was administered to the participating teachers. Responses were aggregated and analyzed for emerging trends. Two additional open-ended questions were also included and analyzed with respect to main ideas and experiences participants had during the CT professional development sessions (see Appendix 2).

Participants were surveyed again at the conclusion of the project during two focus group interview sessions; participants attended one of these sessions (referred to as Survey #2). The interviews were approximately 60-90 minutes long. The groups were semi-structured in that there was a list of questions that were intended to be covered, but the conversation was allowed and encouraged to take its course to include other areas of interest. See Appendix 3 for the focus group interview protocol.

Participant Lesson Plans

Participating teachers were required to write and submit a lesson plan appropriate for the grade level and students that they were currently teaching and were also required to integrate at least one appropriate CT activity into the lesson plan. A modified rubric based on the ISTE Computational Thinking operational definition was utilized to score each of the fourteen lesson plans (ISTE 2011). The iterative development process started with a discussion about various definitions of computational thinking; for example, ISTE's operational definition of computational thinking, from which an initial rubric was created. Using the created rubric, two of the authors independently evaluated five representative lesson plans, which were written and submitted by the participating teachers, and then met in a series of discussions to compare the results of their evaluation. A careful analysis of the inconsistencies and ambiguous aspects of the instrument (with respect to the evaluation process) led to a revision of the initial rubric. The authors then independently evaluated another three representative lesson plans using the revised rubric, from which a further refined rubric was created. The final rubric was developed after several rounds of testing in which each category was scrutinized. All five authors met as a team to discuss each category during this "fine tuning" process until reaching a consensus.

Categories used to score the lesson plans included Decomposition, Pattern Recognition, Abstraction, Algorithmic Thinking, Optimization, and Generalization (see Appendix 4 for the detailed scoring rubric). Sample lesson plans can be viewed in Appendix 5.

Results

Discussion Board Posts

After analyzing the discussion posts, the authors concluded the following:

Awareness: The teachers were aware of CT early in the PD sessions, but were not yet fully comfortable with it. As the teachers interacted with more modules, their comfort level increased, but started to decease as the content of the modules became more complex.

Informational: Many teachers felt comfortable with CT and were more interested in learning more about it as additional modules (that contained more complexity) were provided.

Management: Initially, the teachers were fully committed to CT, but their attention was focused on the processes and tasks involved in implementing it. As additional modules were introduced, the focus on processes lessened, but increased as the modules complexity increased.

Consequence: The teachers' focus on the impact of CT on student achievement was an initial concern, became less so as additional modules were provided, but returned when certain modules refocused their attention.

Collaboration: Based on perusing the 18 modules, the teachers did not seem to indicate that collaboration was important. However, based on the responses to Questions #8 (Survey #2), 44% of the responses indicated that the teachers profited professionally by collaborating with colleagues and/or requested more collaborative PD sessions.

Refocusing: Certain modules (for example, CT & Mathematics Education and Pattern Recognition) seemed to provide an incentive for teachers to explore current and future ways to incorporate CT into their lesson plans.

Survey #1: Overall Feedback Survey Following Summer Professional Development

Upon examination of the Likert-type scale responses included in the Overall Feedback Survey (Survey #1), a large majority of participants answered "Strongly Agree" or "Agree" for each question (see Appendix 1).

Based on analysis of the comments provided by the participating teachers on Survey #1, the authors made the following observations for each question:

Q1 & Q7: 100% of the respondents (n = 17) felt that the PD was important and worthwhile for their professional growth.
Q2, Q3 & Q5: 86% of the respondents felt that the PD activities and discussions were engaging, positively influenced their related content knowledge (mathematics and science content), and positively impacted their teaching of the related content.
Q4 & Q6: It seems that the PD has positively impacted the attitude of the participating teachers (88% answered Agree or Strongly Agree). However, 6% answered Disagree. This positive impact has influenced many of the teachers to apply information learned during the PD sessions on a Daily (12%), Weekly (47%), Monthly (35%), or Every 3 Months (6%) with their students.
Q8: Below are a few sample responses from Question 8 (a free response question):

> *I thought that the whole event was helpful as a whole. Going through the programming on SLNova was great because I experienced that it wasn't so hard to do and can pass it along to my students. I think the whole plan was good and each part was important.*
>
> *The activities and discussions we had regarding how to connect UDL and math/science to CT skills were very helpful.*
>
> *I really enjoyed learning how to connect computational thinking to science and math curriculum. The brain rules and UDL was extremely interesting. I feel much more comfortable with coding in SLNova. I found code.org and the unplugged lessons extremely beneficial.*
>
> *It was awesome to hear the different ways to incorporate CT into the classroom. It was nice to learn different ways to teach the students on basic topics.*

Q9: Below are a few sample responses from Question 9 (a free response question):

> *I liked having the opportunity to pair program and partner up to build lessons.*

> *I really enjoyed the collaborative aspect that the workshop was able to provide. I would have liked to have that same collaboration when working through Modules 1-3.*
>
> *It was awesome to hear the different ways to incorporate CT into the classroom. It was nice to learn different ways to teach the students on basic topics.*

Overall Survey #1 observations are stated below:

- Teachers learned about CT resources available to them and to their students.
- Teachers felt the collaborative aspect of the PD was positive.
- Teachers suggested more face-to-face meetings (to create lesson plans/ activities/games).
- Overall teacher satisfaction was positive – PD provided helpful/ beneficial experiences.

Survey #2: Exit Interviews

Of the 17 participating teachers who completed the entire PD sequence, only 5 were available for exit interviews. Exit interviews were transcribed and coded using NVivo. Based on the analysis of the transcribed information from the Exit Interviews (Survey #2), four main themes emerged: Skill Set, Teacher Attitude, Pedagogical Approach, and Logistics. **Skill Set** includes new ways of thinking with respect to the participating teachers. **Teacher Attitude** includes how the participating teachers felt about CT, coding, and the CTforAll project. **Pedagogical Approach** includes teaching methods, inclusion of CT activities in lesson plans, and administrative issues. **Logistics** includes issues with the professional development sessions (overall and individual sessions), the individual modules, and lesson plans. Some sample responses taken from the transcript appear below:

> *But I think it is that mindset shift - kind of like one of those things where I've always known that this stuff is the right way to do it. But now I'm hearing it and I'm learning a few ways to do it- in that way it's helping me as teacher (Skill Set, teacher #1).*
> *[Students] are interested in being hands on. I had kids that really, it's very hard to get them engaged. But if I tell them I know that with the coding they, they love the games, and the electronics, things like that. And so when you tell them things like, well coding is basically like making your own game. They were like wait, what does that have to do with school? And you know, and that might seem obvious to you or I, but those connections are the ones that get the kids from being not really interested to wanting to get involved and even going above and beyond. So I think it's helpful because a lot of kids know a lot about the stuff and they really want*

> *to sharpen their skills and they want to be able to explore these things that they may not have access to or opportunity to really learn from. How is it affecting students' learning? I think it gets them, it makes them see a point to their learning and it makes them tie in the classroom to their interests and things like that (Pedagogical Approach, teacher #2).*
> *And then I think it's also helpful, at least in my mind to think through…you try something and if it doesn't work that debugging or kind of repetitive iteration process for me was a big thing (Logistics, teacher #3).*
> *I felt at first, I was afraid because when they said you're going to do coding I thought "how did I get into this situation"? And um, and then I thought, well, I'm going to give it a try because I always liked computer science and then I found out it was much more than that (Teacher Attitude, teacher #4).*

Lesson Plans

Based on the nature of the collected data (categorical and discrete), the mode value was used as the appropriate measure of central tendency for data analysis (see Tables 2 and 3).

Table 2. *Overall Ratings for Lesson Plans by Category (see Appendix 4 for a Description of the Lesson Plan Rubric)*

Category	Rating	Sum[+]
Decomposition	1	13
Pattern Recognition	1	13
Abstraction	1	13
Algorithmic Thinking	1	14
Optimization	0	4
Generalization	1	10

\+ This number is the total number of lesson plans that were rated as "1" by the authors for the listed category. The total number of lesson plans that were evaluated was 14.

Overall Ratings – Lesson Plans by Categories (Mode, Total)

- The inclusion of Optimization (0,4) & Generalization (1,10) activities was not overwhelmingly apparent in the lesson plans. Most of the lesson plans did not even incorporate one example of an activity in either of these categories (71% of the lesson plans did not include an Optimization activity, 29% of the lesson plans did not include a Generalization activity).
- Decomposition (1,13), Pattern Recognition (1,13), Abstraction (1,13), and Algorithmic Thinking (1,14) were overwhelmingly apparent (93% of the lesson plans included Decomposition, Pattern Recognition, and Abstraction activities, 100% of the lesson plans included Algorithmic Thinking activities).

Table 3. *Overall Ratings for Lesson Plans*

Lesson Plan	Rating	Sum+
#1	0	1
#2	1	6
#3	1	5
#4	1	5
#5	1	5
#6	1	5
#7	1	6
#8	1	4
#9	1	4
#10	1	5
#11	1	4
#12	1	6
#13	1	5
#14	1	6

+ This number is the sum, over categories, for the indicated lesson plan. The maximum number that could be earned is 6.

Overall Ratings – Lesson Plans (Mode)

- Thirteen lesson plans (93%) were rated “1” except for Lesson Plan #1, which was rated “0”, primarily due to the lack of sufficient details in order to be properly evaluated. Totals across categories ranged from 1 to 6, with “5” being the most common rating, which means that the lesson plans were rated “1” in all categories except for one. The category that was often rated “0” was Optimization (see Table 4).

Table 4. *Overall Ratings for Lesson Plans by Content Area*

Mathematics		
Lesson Plan	**Rating**	**Sum+**
#1	0	1
#2	1	6
#3	1	5
#4	1	5
#6	1	5
#9	1	4
#13	1	5
Science		
Lesson Plan	**Rating**	**Sum+**
#5	1	5
#7	1	6
#8	1	4
#10	1	5
#11	1	4
#12	1	6
#14	1	6

+ This number is the sum, over categories, for the indicated lesson plan. The maximum number that could be earned is 6.

Overall Ratings – Lesson Plans by Content Areas (Mode)

- Mathematics – Six of the seven lesson plans were rated "1", which indicated that 86% of the lesson plans contained more than three of the six rated categories. The lesson plan that was rated "0" did not include enough specific information to be properly evaluated (see Table 5).
- Science – All seven lesson plans were rated "1", which indicated that 100% of these lesson plans contained more than three of the six rated categories (see Table 5).

Table 5. *Overall Ratings for Lesson Plans by Grade Level Band*

Mathematics		
Elementary+	**Rating**	**Sum**++
#1	0	1
#2	1	6
#6	1	5
#8	1	4
#9	1	4
#14	1	6
Science		
Secondary+	**Rating**	**Sum**++
#3	1	5
#4	1	5
#5	1	5
#7	1	6
#10	1	5
#11	1	4
#12	1	6
#13	1	5

+ Elementary: grades kindergarten through 5, inclusive
  Secondary: grades 6 through 12, inclusive
++ This number is the sum, over categories, for the indicated lesson plan. The maximum number that could be earned is 6.

Overall Ratings – Lesson Plans by Grade Level Band+ (Mode)

- Elementary Level – Five of the six lesson plans were rated "1", which indicated that 83% of these lesson plans contained more than three of the six rated categories. Lesson plan "1" was rated "0" mainly due to insufficient details.
- Secondary Level – All eight lesson plans were rated "1", which indicated that 100% of these lesson plans incorporated more than three of the six rated categories.
- 67% (4 of 6) of the Elementary Level lesson plans were mathematics lesson plans. 63% (5 of 8) of the Secondary Level lesson plans were science lesson plans.

Discussion

The analysis of the data demonstrated that most of the participants felt that the professional development activities were worthwhile and made a positive impact on the quality and quantity of CT activities they implemented in their k-12 mathematics and science classrooms. Most participants stated that they would be implementing CT activities with their students weekly or monthly.

Lesson plan analysis indicated that Optimization and Generalization were not routinely integrated into participants' classroom teaching. Upon further examination of the comments made by the teachers on the online discussion board, it is logical to conclude that this occurred due to the teachers' lack of their own experiences and comfort level with different computational thinking tasks. On the other hand, participants seemed comfortable with integrating decomposition, algorithmic thinking, pattern recognition, and abstraction into their teaching, as these categories were more prevalent in written lesson plans. It seemed easier for teachers to understand these types of activities and, therefore, incorporate them into their lesson plans (more experience with these types of computational thinking activities).

Based on the total sum of ratings across lesson plan categories, the science lesson plans earned a higher rating than the mathematics lesson plans (36 points to 31 points, out of a possible 42 points). This may have been due to the mathematics lesson plan that was rated "0". However, three of the science lesson plans earned a total score of "6" (the highest possible total), while only one of the mathematics lesson plans earned this total. Perhaps, since the modules were primarily science-based, the science teachers were better prepared to adapt or create lessons that were science-based. It may have been more difficult for the mathematics teachers to adapt and/or create a lesson plan that focused on a purely mathematical concept, such as "how best to conceptually teach the Pythagorean Theorem", with respect to computational thinking. In fact, using aspects of CT to develop a conceptual, as opposed to a procedural, understanding of specific mathematical content may not be the best way to accomplish this goal (Lye and Koh 2014).

Examination of the lesson plan total sum of ratings across categories, reveals that the Secondary Level lesson plans earned a higher rating than the Elementary Level lesson plans (41/48 = 85% to 26/36 = 72%). The authors felt that the Secondary Level teachers were better prepared to incorporate CT activities into their lessons due to their familiarity and consistent use of the scientific method and a mathematics-related problem solving protocol (for example, Polya's Problem Solving Model). This may not have been the case with the Elementary Level teachers, even though a few of these teachers have used both the scientific method and Polya's Problem Solving Model with their students.

Since the research question of this paper focused on the impact of PD on the inclusion of CT activities in the mathematics and science classrooms of participating teachers, the authors only reported on this particular issue under the Pedagogical Approach theme.

The qualitative analysis of the teacher interview data showed that the following themes were emerged:

- Many teachers used the terms “coding” and “CT” interchangeably, which contributed to the difficulty that most teachers experienced when creating, selecting, and/or implementing appropriate CT activities into their lesson plans.
- After the professional development, the participating teachers had a “mindset” shift. They were more willing to take risks with their students with respect to CT projects/activities (for example, the prosthesis project) and get out of the usual pedagogical routine. Most teachers were optimistic about CT, not only about how well it can be integrated into science content, but other content areas as well.
- The participating teachers also indicated that assigning and completing CT projects/activities were important for their students. Students began to understand the reasons for studying and learning concepts and skills in mathematics and science. There was an increase in engagement with respect to learning content so they could complete the project. The CT projects/activities could be used to illustrate the connection between “school” life and “real” life, sharpen academic skills, provide career opportunities, and create a viable product. According to one participating teacher “Including CT projects/activities in lesson plans will encourage students to think for themselves and isn’t that what CT is all about?” By participating in CT projects, teachers reported their students became better problem solvers and became more proficient with the logic of coding. In addition, since many students want to become programmers/ gamers, they were excited about getting more experience in writing code when completing certain CT activities.
- The teachers realized that CT projects/activities can be used to engage students in the SMPs and NGSS (SMP #1 & #3, scientific process).
- The teachers acknowledge that CT projects/activities can be integrated into the existing curriculum, but not every day and not everywhere. The activities must fit naturally, not be forced. Most students are not currently prepared to write code. Therefore, teachers must provide the opportunity and the necessary steps in order to enhance student success. The incorporation of CT activities/projects will increase the chance of this occurring. Many teachers felt that routinely introducing CT activities into their lesson plans would help students reach a higher level of thinking.
- Many teachers wanted more examples of how to incorporate CT projects/activities into their classrooms, especially with respect to teaching specific content. They would like ideas as to how best to resolve the issue of using more CT activities/projects and teaching specific content that is on the state mandated tests.
- The PD sessions helped the participating teachers learn how to teach their students to write code in StarLogo and Scratch. By participating in the PD sessions, the teachers learned it was okay to make mistakes

when writing code because this encourages open dialogue between and among teachers and this approach can and should be encouraged with their students. Teachers and students can approach writing code as an adventure that they can share and enjoy together by making and correcting mistakes!

Future Research Questions

Based on a careful consideration of the original research question, and the data generated from the present study, the following additional questions should be actively pursued in future research studies:

- Does content discipline impact the quantity of CT activities that teachers incorporate into their lesson plans?
- Does content discipline impact the quality of CT activities that teachers incorporate into their lesson plans?
- Do grade-level bands (elementary and secondary) impact the quantity of CT activities that teachers provide their students?
- Do grade-level bands (elementary and secondary) impact the quality of CT activities that teachers provide their students?
- What is the impact of the teachers' CT professional development activities on their students' attitudes and achievement towards mathematics and science?
- Is there a difference in teacher-attitudes toward incorporating CT activities into lesson plans based on grade-level bands (elementary and secondary)?
- Is there a difference in teacher-attitudes toward incorporating CT activities into lesson plans based on content specialty (mathematics and science)?
- What are the most pedagogically effective ways to include more emphasis on classroom integration of Optimization and Generalization activities in order to increase teachers' confidence in their ability to effectively integrate these skills into their current classroom teaching?

Acknowledgments

The activities conducted in this research study were performed in the State of Maryland in the United States of America from spring 2017 to spring 2018.

References

Barr V, Stephenson C (2011) Bringing Computational Thinking to K-12: What is Involved and what is the Role of the Computer Science Education Community? *ACM Inroads* 2(1): 48-54.

Baytak A, Land SM (2011) An Investigation of the Artifacts and Process of Constructing Computer Games about Environmental Science in a Fifth Grade Classroom. *Educational Technology Research and Development* 59(765-782): 48-54.

Blikstein P, Wilensky U (2009) An Atom is known by the Company it Keeps: A Constructionist Learning Environment for Materials Science using Agent-Based Modeling. *International Journal of Computers for Mathematical Learning* 14(2): 81-119.

Bloom BS, Krathwohl DR, Masia BB (1956) *Taxonomy of Educational Objectives, Classification of Educational Goals, Handbook I: Cognitive Domain*. New York: McKay.

Calao LA, Moreno-Le´on J, Correa HE, Robles G (2015) Developing Mathematical Thinking with Scratch an Experiment with 6th Grade Students. In *Design for Teaching and Learning in a Networked World*. Springer International Publishing, 17-27.

Cole K, Simkins M, Penuel WR (2002) Learning to Teach with Technology: Strategies for Inservice Professional Development. *Journal of Technology and Teacher Education* 10(3): 431-455.

Common Core State Standards Initiative - CCSSI (2010) *Common Core State Standards for Mathematics (CCSSM)*. Washington, DC: National Governors Association Center for Best Practices and the Council of Chief State School Officers. Retrieved from http://www.corestandards. org/wp-content/uploads/Math_Standards.pdf.

Di Sessa AA (2000) *Changing Minds: Computers, Learning, and Literacy*. Cambridge, MA: MIT Press.

Gal-Ezer J, Stephenson C (2010) Computer Science Teacher Preparation is Critical. *ACM Inroads* 1(1) Doi>10.1145/1721933.1721953.

Hambrusch S, Hoffmann C, Korb JT, Haugan M, Hosking AL (2009) *A Multidisciplinary Approach towards Computational Thinking for Science Majors*. Proceedings of the 40th ACM Technical Symposium on Computer Science Education (SIGCSE '09). New York, NY, USA: ACM, 183-187.

Hord SM, Rutherford WL, Huling-Austin L, Hall GE (1987) *Taking Charge of Change*. Alexandria, Va.: Association for Supervision and Curriculum Development.

ISTE (2011) *Operational Definition of Computational Thinking for K–12 Education*. https://bit.ly/2IbefQg.

Jona K, Wilensky U, Trouille L, Horn MS, Orton K, Weintrop D, Beheshti E (2014) *Embedding Computational Thinking in Science, Technology, Engineering, and Math (CT-STEM)*. Presented at the Future Directions in Computer Science Education Summit Meeting, Orlando.

Kanaya T, Light D, Culp KM (2005) Factors Influencing Outcomes from a Technology Focused Professional Development Program. *Journal of Research on Technology in Education* 37(3): 313-329.

Kynigos C (2007) Using Half-Baked Microworlds to Challenge Teacher Educators' Knowing. *Journal of Computers for Math Learning* 12(2): 87-111.

Lambert L, Guiffre H (2009) Computer Science Outreach in an Elementary School. *Journal of Computing Sciences in Colleges* 24(3): 118-124.

Lye SY, Koh JH (2014) Review on Teaching and Learning of Computational Thinking through Programming - What is Next for K-12? *Computers in Human Behavior* 41(Dec): 51-61.

National Research Council (2011) *Report of a Workshop of Pedagogical Aspects of Computational Thinking*. Washington, DC: The National Academies Press.

National Research Council (2013) *Next Generation Science Standards: For States, By States*. Washington, DC: The National Academies Press. https://doi.org/10.17226/18290.

Ottenbreit-Leftwich AT (2007) *Expert Technology-Using Teachers: Visions, Strategies, and Development*. Unpublished Dissertation. Purdue University, West Lafayette.

Pólya G (1957) *How to Solve it: A New Aspect of Mathematical Method*. Princeton, N.J.: Princeton University Press.

Putnam RT, Borko H (2000) What do New Views of Knowledge and Thinking have to Say about Research on Teacher Learning? *Educational Researcher* 29(1): 4-15.

Repenning A, Webb D, Ioannidou A (2010) *Scalable Game Design and the Development of a Checklist for Getting Computational Thinking into Public Schools*. In Proceedings of the 41st ACM Technical Symposium on Computer Science Education, 265-269.

Sengupta P, Kinnebrew JS, Basu S, Biswas G, Clark D (2013) Integrating Computational Thinking with K-12 Science Education using Agent-Based Computation: A Theoretical Framework. *Education and Information Technologies* 18(2): 1-30.

Sherin B (2001) A Comparison of Programming Languages and Algebraic Notation as Expressive Languages for Physics. *International Journal of Computers for Mathematics Learning* 6(1): 1-61.

Snoeyink R, Ertmer PA (2001/2002) Thrust into Technology: How Veteran Teachers Respond. *Journal of Educational Technology Systems* 30(1): 85-111.

Somekh B (2008) Factors Affecting Teachers' Pedagogical Adoption of ICT. In J. Voogt and G. Knezek (ed), *International Handbook of Information Technology in Primary and Secondary Education*, 449-460. New York: Springer.

Weintrop D, Beheshti E, Horn M et al. (2016) Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology* 25(Oct): 127-147.

Wilensky U, Reisman K (2006) Thinking Like a Wolf, a Sheep, or a Firefly: Learning Biology through Constructing and Testing Computational Theories—An Embodied Modeling Approach. *Cognition and Instruction* 24(2): 171-209.

Wing JM (2006) Computational Thinking. *Communications of the ACM* 49(3): 33-35.

Wozney L, Venkatesh V, Abrami PC (2006) Implementing Computer Technologies: Teachers' Perceptions and Practices. *Journal of Technology and Teacher Education* 14(1): 173-207.

Yadav A, Zhou N, Mayfield C, Hambrusch S, Korb JT (2011) *Introducing Computational Thinking in Education Courses*. In Proceedings of ACM Special Interest Group on Computer Science Education, Dallas, TX.

Yadav A, Mayfield C, Zhou N, Hambrusch S, Korb JT (2014) Computational Thinking in Elementary and Secondary Teacher Education. *ACM Transactions on Computing Education* 14(1): 1-16. Doi: 10.1145/2576872.

Zhao Y, Cziko GA (2001) Teacher Adoption of Technology: A Perceptual Control Theory Perspective. *Journal of Technology and Teacher Education* 9(1): 5-30.

APPENDIX 1: Sample Discussion Board Question Prompts

1. How do you currently make connections between mathematics and science?
2. Which of the activities mentioned have you done in your classroom?
3. Which of the activities mentioned do you want to do in your classroom? Why?
4. How does Computational Thinking relate to the Standards for Mathematical Practice?
5. How does Computational Thinking relate to the Next Generation Science Standards?
6. What kinds of models do you currently use in your teaching?
7. What kinds of computer models can be helpful in your teaching?
8. What are the benefits of students not just using computer models, but being able to modify and create models?
9. What are your concerns with teaching computer modeling in the science classroom?
10. Which part(s) of your current curriculum can be replaced or supplemented with unit(s) on computer modeling?

APPENDIX 2: Survey #1

Q1: The PD sessions throughout the project constituted meaningful and important skill/knowledge.
Q2: Activities and discussions at PD sessions enhanced my knowledge and understanding of related content
Q3: Activities and discussions during the PD sessions I attended have positively influenced my teaching of related content.
Q4: The information learned from participating in this project has/will positively impact the achievement of my students.
Q5: Project activities and discussions were engaging.
Q6: Approximately, how often do you believe you have applied the information learned from the PD sessions with your students?
Q7: Participating in the project workshops has been a worthwhile and valuable professional development experience for me.
Q8: What was the most beneficial/meaningful experience you had during the workshops?
Q9: Please provide any additional comments/feedback.

APPENDIX 3: Survey #2 Semi-Structured Interview & Focus Group Questions

Thanks for your participation. The purpose of the Focus Group/Interview is to evaluate the CT for All project, not the participants. I will be creating an audio recording of the conversation, and a typewritten transcript of the interview will be created. I will review the transcript to try and understand what impressions you have about the program to understand whether and how well it worked, and how future programs might be improved. Pseudonyms will be used in place of your names to maintain confidentiality. Please feel free to be candid as you share your thoughts.

There are two broad elements I would like to ask about: 1) the content, computational thinking; and 2) your thoughts about the implementation of the professional development. This includes the materials, the sessions, or the professional learning community online. I am interested in any impressions you care to share about your experience in the CT for All program.

What have you gained from participating in this project?

How do you feel your participation in this project has impacted your students' learning and performance?

How have your thoughts and perceptions of your role in teaching computational thinking in your content area changed? How have they remained the same?

What were the most beneficial components of the project?

What suggestions/recommendations would you make for future projects with a similar focus to this one?

How has your participation in this project impacted other teachers at your school?

What feedback have you received from your administrators around your participation in the project?

What are your perceptions toward computational thinking, specifically, how is computational thinking implemented? How well do you think computational thinking can be integrated? How have your perceptions changed from before ITQ until now?

APPENDIX 4: Lesson Plan Rubric Computational Thinking Rubric for CT for All Participant's Lesson Plans

Lesson Plan: ____

	0	1	#
Decomposition	No example of decomposition learning activities in the lesson plan	Lesson plan contains at least one example of a learning activity on decomposition	
Pattern recognition	No example of pattern recognition learning activities in the lesson plan	Lesson plan contains at least one example of a learning activity on pattern recognition	
Abstraction	No example of abstraction learning activities in the lesson plan	Lesson plan contains at least one example of a learning activity on abstraction	
Algorithmic thinking	No example of algorithmic thinking learning activities in the lesson plan	Lesson plan contains at least one example **of** a learning activity on algorithmic thinking	
Optimization	No example of optimization learning activities in the lesson plan	Lesson plan contains at least one example **of a** learning activity on optimization	
Generalization	No example of generalization learning activities in the lesson plan	Lesson plan contains at least one example of a learning activity on generalization	

Category Descriptions

Computational thinking is defined as "a problem-solving process that includes (but is not limited to) the following characteristics:

- Formulating problems in a way that enables the use of a computer and other tools to help solve them.
- Logically organizing and analyzing data.
- Representing data through abstractions such as models and simulations.
- Automating solutions through algorithmic thinking (a series of ordered steps).
- Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources.
- Generalizing and transferring this problem solving process to a wide variety of problems".

Based on the authors' working definition of computational thinking, there are six skill sets that are essential with respect to computational thinking:

1. **Decomposition:** Decomposition is defined as "the ability to break a large problem down into many smaller problems". Decomposition helps formulate the problem in such a way that computers or other tools can be used in the solution process. For example, evidence of decomposition in a lesson plan can be that the students are asked to "identify parts", or to make a "list of examples" of the elements of a specific activity.
2. **Pattern Recognition:** Pattern recognition is defined as "the ability to notice similarities or common differences in data". For example, evidence of pattern recognition in a lesson plan can be that "order matters", or that the students are asked to determine "similarities and/or differences" between and among cases/scenarios.
3. **Abstraction:** Abstraction is defined as the process of "deciding what details are needed and which details can be ignored". The process of abstraction is often used when creating models or the simplified versions of complex systems. For example, evidence of abstraction in a lesson plan can be that the students are asked to "describe", "illustrate", or "use model(s)".
4. **Algorithmic Thinking:** Algorithmic thinking is defined as "a way of getting to a solution through the clear definition of the steps needed". Developing an algorithm means creating a step-by-step solution to solve a problem. For example, evidence of algorithmic thinking in a lesson plan can be that students are asked to create a "step-by-step" process for making a ham and cheese sandwich.
5. **Optimization:** Optimization is defined as "identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources". For example, evidence of optimization in a lesson plan can be that students are asked to identify "limitations" in a solution process, or to think of how to make a solution process more efficient.
6. **Generalization:** Generalization is defined as "generalizing and transferring the problem solving process to a wide variety of problems". For example, evidence of generalization in a lesson plan can be that the students are asked to consider two or more contexts for a given task.

APPENDIX 5: Sample Lesson Plans

Grade Level: Kindergarten
Standards:

- *Common Core State Standards*
    - Literacy
        - R.K.1; W.K.1; W.K.2; W.K.3; W.K.7; SL.K.5
    - Math
        - MP 2; MP 4
        - K.CC; K.MD.A.2
- *Next Gen Science*
    - K.LS.1-1 and K.ESS.3-1

Objectives: Today we will decompose large activities into a series of smaller events. Today we will arrange sequential events into their logical order.
Vocabulary: algorithm
Lesson Plan: (selected from the unit plan)

- First discuss with your students the vocabulary algorithm. Show them the word. You say the word, they say the word. Explain to students the definition of an algorithm. An algorithm is a precise sequence of instructions for processes that can be executed by a computer. Have definition posted with a picture. Refer to this frequently throughout the lesson. Have students continuously practice saying the word.
    - *Extension: Algorithm comes from the Greek word arithmos which means number*
- Explain to students that today we will be learning about the concepts of algorithms and how we follow algorithms in real life every day.
- Go over the objective for today.
- Have a class discussion about the steps that they take to get ready for school in the morning. As students are saying things they do to get ready write them up on the board. Once finished writing out the ideas, put numbers next to their responses to indicate order. Be sure to point out places where order matters and where order does not matter. Explain to students how we just created an algorithm for how they get ready for school in the morning. Refer back to the definition of algorithm. Introduce that it is possible to create algorithms for things we do every day.
    - *Extra Support: Create an algorithm for another every day activity such as making a peanut butter jelly sandwich, how to walk through the door and return, or making chocolate milk.*
- Next day in class we are going to create an algorithm for planting a seed. Refer back to the definition of algorithm. Have the students watch the brainpop video on plant life cycle in order to gain background knowledge on how to plant a seed. As the video is playing feel free to stop the video and ask students questions in order to continue their engagement and motivation for watching the video.
- Once the video ends have the students split up into small groups/pairs. In the small groups/pairs each will need a plant a seed worksheet, scissors, glue, and a sentence strip. Option for choice: have this in an electronic format (wixie/interactive whiteboard lesson). Students will cut out each of the 9 squares. Their job is to determine which 6 of them you will utilize in order to plant a seed, so they will not use 3 squares
    - *Extra Support: Pull students who may need a little extra support at a small group with the teacher to help them with the activity. Pair the students up with a stronger person who can help them with the activity. Instead of giving the group/pair of students all 9 give them just the 6 that they need to sequence. If needed complete activity as a whole class.*

  - *Fine Motor: Students who may struggle to cut them out have the squares already pre-cut so they just have to focus on gluing.*
- Once groups are finished their sequence of events (algorithm) to plant a seed they will swap sentence strips (or electronic formats) and follow the sequence of steps the other group created in order to actually plant a seed.
- After all groups have planted a seed following their algorithm have a class discussion about the process. What did you learn? How many of you were able to follow your algorithms to plant your seeds? Did the exercise leave anything out? What would you have added to make the algorithm even better? What if the algorithm had been only one step: plant the seed? Would that have been easier or harder? What is it were forty steps? What was your favorite part of the activity?

Intro to Computational Thinking

Date	Aug 18, 2017	**Time Frame**	50 minutes
Unit	Computational Thinking	**Unit Theme**	

NGSS Standards	**Developing and Using Models** Modeling in 9–12 builds on K–8 and progresses to using, synthesizing, and developing models to predict and show relationships among variables between systems and their components in the natural and designed worlds. • Develop a model based on evidence to illustrate the relationships between systems or between components of a system. (HS-PS1-4),(HS-PS1-8)
Disciplinary Core Ideas	**ETS1.C: Optimizing the Design Solution** • Criteria may need to be broken down into simpler ones that can be approached systematically, and decisions about the priority of certain criteria over others (trade-offs) may be needed. *(secondary to HS-PS1-6)*
Objective	Students will be able to draw pixels into a grid in order to demonstrate how a computer can create an image from a set of numbers.
Useful links	http://csunplugged.org/wp-content/uploads/2015/03/CSUnplugged_OS_2015_v3.1.pdf

Drill: (5 min.) Propose the following questions to the class. Have them write their : In what situations would computers need to store pictures? (A drawing program, a game with graphics, or a multi-media system.) How can computers store pictures when they can only use numbers?

Engagement (not boring): 5 minutes	
__Captures students' attention? __Activates students' prior knowledge? __Connects to a complex question, global issue, or real world problem?	Provide time for students to discuss their thoughts about the above discussion. Some students may know a lot, others may need help. If students are aware of what a fax machine does, that can help in the explanation. To go even further, a fax machine could be brought into the classroom.
Exploration (think labs, hands-on, student driven): 10 minutes	
__Analyzes other disciplines to answer a complex question, global issue, or real world problem? __Applies a systematic approach to address the real world connection? __Selects and employs relevant technological tools?	Students will be given the handout with the "a" letter drawn in pixelated form. Students will need to draw this image and then to the side of the image write the coded numbers that correspond to the pixels. Students can help each other to understand and ask questions as needed. Most students will catch on to this very quickly and begin to move onto the next part.

Explanation (multi-sensory): 15 minutes	
__Analyzes data and draws conclusions? __Communicates understandings and possible solutions?	There are numbers and a grid on the next page of the worksheet. Using the numbers, the students need to figure out which pixels to shade in. The result will be an image of a little picture. Students will know if they are getting the answers correct if their picture makes sense.

Extension/**E**laboration (building complexity, real world application, frequent review): Remaining time in lesson.	
__Modifies experimental procedures, prototypes, models, or solutions? __Analyzes related STEM careers?	Students can now create their own picture, turn it to code, and then trade with a classmate who will have to decipher the picture. Students can draw their favorite character or emoji in order to make it unique and personalized to their interests. There is also an included extension activity where the students can include color to their picture, again using numbers to indicate the colors being used.
Evaluation (multiple modalities): During/After class	

UDL Connections:
Students are taking a picture and turning it into a number. Then students take a number and turn it into a picture. Abstract concepts are explained through this process.

__Demonstrates understanding of concepts through rubric-based performance assessments? __Participates in peer reviews?	Rather than a formal evaluation, I would circulate when the students are making their own pictures to see the engagement level and understanding of the concepts of this lesson.

AVID Strategies			
Writing to Learn	Inquiry	Collaboration	Reading to Learn
__Cornell Notes __Notes (Right Hand) __Other ______________	_X_Analyze __Extend _X_Apply __Seek Clarification __Other ______________	_X_Problem Solve _X_Work together __Other ______________	__Pre-Reading Activities __Summarize/Reflective __Highlight/Underline __Other ______________