Suters, L., Suters, H., & Anderson A. (2021). Coding connections at the interface of algebra and physical world concepts. *Contemporary Issues in Technology and Teacher Education, 21*(2), 441-490.

Coding Connections at the Interface of Algebra and Physical World Concepts

Leslie Suters
Tennessee Technological University

Henry Suters
Carson Newman University

Adam Anderson
Oak Ridge National Laboratory

This paper describes a 54-hour summer institute for grades 6-12 mathematics and science teachers (*N* = 19) with a comprehensive approach to preparing teachers to use computational thinking (CT) in their classrooms, including training in Python computer programming with Lego® Mindstorms® robotics, mathematics content sessions, and opportunities to solve real-world robotics challenges. Results of an assessment used to measure content knowledge and CT skills and the Technological Pedagogical Content Knowledge survey both yielded statistically significant increases. Participant reflections revealed they developed an enhanced understanding of programming and the ability to integrate programming into the curriculum. The authors propose an innovative approach to teaching disciplinary CT within the context of programming robots capable of interacting with the outside world to address real-world challenges.

The rapid changes in societal and work environments in response to the Fourth Industrial Revolution (4IR) are indicators of the growing need to prepare students for a workforce with skills in science, technology, engineering, and mathematics (STEM) and, in particular, computer science (CS; Committee on STEM Education of the National Science & Technology Council, 2018; Schwab & Davis, 2018; World Economic Forum, 2016). Emerging technologies such as artificial intelligence, biotechnology, the internet of things, and autonomous vehicles, together with the ways humans interact with these technologies, are "blurring the lines between the physical, digital, and biological spheres" (Schwab, 2016, para. 2).

Embedding computational thinking (CT) practices within mathematics and science curriculum provides opportunities to prepare students better to meet the future needs of the job market in which these emerging technologies are being developed and used (Grover & Pea, 2013; National Governors Association Center for Best Practices & Council of Chief State School Officers, 2010; National Research Council [NRC], 2012).

CT has importance beyond addressing the needs of the future job market, particularly as a fundamental skill for solving problems that is essential within every discipline, not only within CS (Yadav et al., 2017; Wing, 2011). CT is characterized by problem solving, modeling, data mining, networking, algorithmic reasoning, programming, designing solutions, communicating thoughts in a creative, organized way, and debugging (Computer Science Teachers Association [CSTA], 2016; Sneider et al., 2014). The International Society for Technology in Education (ISTE) standards for educators and students emphasize CT competencies as a way to simultaneously increase content knowledge of disciplines such as mathematics and science while developing digital learning skills (ISTE, 2016, 2017, 2018).

While CS concepts and CT skills are outlined clearly in current standards, they are new to students, teachers, administrators, and parents, who often incorrectly label basic computer literacy activities such as creating documents and searching the Internet as CT skills (CSTA, 2016; ISTE 2018). Sands et al. (2018) surveyed teachers and found that many lacked understanding of the core components of CT and lacked awareness of how these skills can be implemented in classrooms. In a systematic literature review of teacher preconceptions of CT, Cabrera (2019) found that teachers conflated CT with technology integration, CS or programming, other nonspecific problem-solving strategies, and "thinking like a computer."

Current research endeavors have recognized the need to contextualize CT within specific disciplines (Gadanidis et al., 2017; Grover & Pea, 2013; Koehler et al., 2013; Qin, 2009; Weintrop et al., 2016; Yaşar, 2013). Within this context, we developed a summer institute aimed at addressing the need for high quality professional development (PD) in disciplinary CT strategies in mathematics and science. The institute, called Coding Connections at the Interface of Algebra and Physical Science, used computer programming, the technological interface of the real and virtual worlds, as an analogous interface for algebra and physical science.

Participants were introduced to the Python programming language through Lego® Mindstorms® EV3 robotics. This paper describes our investigation of teachers' experiences as participants in the institute in which we examined the following questions.

1. What conceptual and attitudinal changes did teachers experience in terms of mathematics and science-specific technological pedagogical content knowledge?
2. How did participation in this institute impact teacher use of disciplinary CT strategies as applied to algebra and physical science?
3. What did teachers perceive as the areas of improvement needed for the design of the institute as articulated in post-program reflections?

Literature Review

Need for CT Training for Preservice and In-service Teachers

Sands et al. (2018) claimed the need to prepare both preservice and in-service teachers in CT practices, regardless of their respective academic discipline. PD for noncomputing in-service teachers should be sustained and should focus on supporting integrating CT through the use of problem-solving activities within disciplines through communities of practice rather than as an instructional add-on to the curriculum. Mouza et al. (2017) observed that PD efforts focused on embedding CT in K-12 education have occurred primarily at the high school level within dedicated CS classes using CS curricula.

Menekse (2015) reviewed the effectiveness of 21 K-12 teacher PD programs focused on computer science or computational thinking conducted between 2004 and 2014. Six factors were used as a gauge of effectiveness in terms of the potential of the PD to change teaching practices and enhance student learning, including (a) duration of at least 50 hours; (b) support for classroom implementation; (c) active learning methods; (d) explicit focus on pedagogical content knowledge; (e) collaboration with local district or school administration; and (f) student learning data or evidence. This study revealed that the majority of the programs lacked the requirements for high-quality and effective PD through three key findings. There was minimal collaboration between higher education institutions and local education agencies to develop PD, most of the programs were less than or equal to 1 week with limited ongoing support for implementation, and a lack of discipline specific pedagogical content knowledge development.

TPACK and SAMR Models

The Technology, Pedagogy, and Content Knowledge (TPACK) Framework (Mishra & Koehler, 2006) and Substitution Augmentation Modification Redefinition Model (SAMR; Puentedura, 2014) served as guiding frameworks for this project. The TPACK and SAMR models ultimately support the challenge of integrating technology and CS effectively with mathematics and science instruction. The TPACK framework builds on the work of Shulman (1986) and explores how technology is integrated with expert knowledge of best practices within specific disciplines (Bull et al., 2019).

The SAMR model designed by Dr. Ruben Puentedura (2014) is a framework used to assess and evaluate digital technology use in the classroom. The first two levels, Substitution (technology acts as a tool substitute) and Augmentation (adds a functional change) comprise an Enhancement section. The Modification (task redesign) and Redefinition (creation of new tasks) levels in the Transformation section can lead to greater student engagement, involvement, and ultimately increased student achievement and learning.

PD Design

Coding Connections provided PD for 19 teachers (six middle school and 13 high school) from 11 high-need partnering school districts that served not fewer than 10,000 children or not less than 20% of children from families below the poverty line. Enrollment was open to any teacher who applied from the partner districts, and all teachers who applied were accepted. There were six algebra, 10 science, two STEM, and one robotics teachers.

Two teachers had participated in a CT and middle grades mathematics PD opportunity offered by the project team the previous year focused on the use of Bootstrap Algebra and programming robotics (Suters & Suters, 2020). The current project included a 2-week summer institute, along with one follow-up Saturday during the fall semester, for a total of 54 contact hours. Coding Connections used Python programming language with sensors and robots created using Lego Mindstorms to bridge the interface between the mathematics of algebra and real-world problems of physical science.

Teacher participants received three Lego Mindstorms EV3 Core Sets with sensors, a Chromebook, publications geared for algebra and physical science instruction, and a $75 daily stipend. They received explicit instruction with the TPACK Framework and SAMR Model to engage them intentionally in thinking about the intersection of content, pedagogy, and technology and how it can help them with the Tennessee Educator Acceleration Model, or TEAM, which is the evaluation system required by the state (Tennessee Department of Education, 2019).

PD Framework

The summer institute was planned using characteristics of effective PD, including clear goals, content and practice, active learning experiences, facilitators with appropriate expertise, alignment with state/district goals and standards, and collaborative and collective participation of teams of teachers (Council of State Science Supervisors [CSSS], 2018; Demonte, 2013; Desimone, 2009; Koba et al., 2013; Loucks-Horsley et al., 1996). Each participating district identified at least two teachers to form a professional learning community (PLC). As recommended by Loucks-Horsley et al. (1996), teachers worked in learning communities and were prepared to serve in leadership roles. During the workshop teachers worked in groups of three to four as they experienced activities that embedded CT practices that mirrored methods they could use with students.

Coding Connections was conducted jointly by education methods, engineering, mathematics, and CS faculty members to effectively model pedagogy and focus on building content knowledge within the context of embedding CT. All project team members had experience working with in-service teachers in preparing them to teach mathematics, science, and computer programming.

Institute Schedule and Design

Weintrop et al. (2016) developed a CT in mathematics and science practices taxonomy that included four major categories: data practices, modeling and simulation practices, computational problem-solving practices, and systems thinking practices. We followed a similar structure.

In the data practices category Lego Mindstorms robots were used to generate experimental data to illustrate concepts, including rates of change, functional relationships, and statistical concepts. These data were then manipulated and displayed graphically and the results were analyzed. In the modeling and simulation practices category, functional models were created using the data collected from the robots, and these models were tested.

In the computational problem-solving category, the Lego Mindstorms involved computer programming with Python. This activity often involved breaking the problem down into functional modules (or smaller problems) and then developing computational abstractions (such as a math equation or computer code) to create these modules.

Finally, the programs needed to be tested and debugged. The final category in Weintrop's taxonomy is systems thinking practices, which was addressed with programming the robots with Python. A robot, together with its programming is a complex system, and the relationships between the various components must be understood and managed for the system to operate correctly.

On Day 1 teachers completed preassessments and did an unplugged CS activity, called My Robotic Friends, as an icebreaker. It highlighted the use of algorithms, written code, functions, and debugging while building structures with disposable cups (Thinkersmith, 2013). Subsequent days featured blocks of instruction, including modules for mathematics, Python programming introduction, and Lego Mindstorms Python programming challenges as described below.

Block 1: Mathematics Modules

The first module introduced the idea of modeling a real-world situation either physical (e.g., using a Lego Mindstorms robot as a model for a more complex system) or mathematical (e.g., using mathematics to make predictions about how a system will behave) and then used a robot to gather real-world data about its position over time and used this to calculate the rate of change of position.

These data were used to create a mathematical model that could be used to predict behavior of the robot; furthermore, the predictions of this model were then compared to the actual behavior observed in further experiments. Module 2 addressed the mathematical definition of a function, graphing linear functions, and slopes and intercepts of lines. Module 3 focused on measuring distance and applying this to the path followed by a robot, including the Pythagorean theorem and the distance formula.

The topic of the fourth module was, again, modeling and the fact that data may not be perfectly linear. In this situation it is necessary to select an appropriate approach to modeling, and linear regression was discussed to fit a linear model to noisy data. Quadratic models were introduced to fit data where the rate of change is linear (e.g., constant acceleration).

Block 2: Python Modules

While roughly a third of the teachers had previous experience coding Lego Mindstorms using Lego's visual coding software, none of the teachers had exposure to Python programming or other types of text-based programming. With that in mind we asked the teachers to complete an online hour of code using Lightbot (http://lightbot.com/hour-of-code.html). Lightbot teaches specific coding constructs, including sequential control flow (step by step), procedures, loops, and debugging.

We also played coding board games including Robot Turtles: The Game for Little Programmers! (Shapiro, 2013), Code Master Programming Logic Game (Code Master, 2015), Littlecodr (Littlecodr Games Inc., 2015), and Code Monkey Island (Code Monkey Island, n.d.). We purchased a copy of Python for Kids: A Playful Introduction to Programming by Jason Briggs (2013) for each teacher and allowed time during the sessions to work through examples in the book; however, we asked them to continue working through chapters as homework.

The institute included three Python programming modules, and the first of these contained a general overview of CS: software vs. hardware, a conceptual definition of computer programming, and an overview of computer hardware. Next, this module provided a brief introduction to the Python language and its programming environment, including installing and starting the environment, basic syntax of output statements, simple function definitions and function parameters, and loading and running programs.

In the second Python module the participants learned how to load and run Python programs on the Lego Mindstorms hardware, and the syntax of the commands to control the robot's motors was introduced. The third Python module returned to the mathematical definition of a function, with the focus on the translation of these functions to Python code and a brief introduction to the concepts of object-oriented programming.

Programming the robot is inherently object oriented, because the motors and sensors of the robots are treated as objects. Finally, the syntax of the commands to get data from the robot's sensors was introduced.

Block 3: EV3 Robotics Challenges

One of the primary goals of these challenges was to get teachers excited about coding by creating connections to real-world problems that might be interesting to both teachers and students alike. To this end, a set of challenges was designed in increasing difficulty to have teams arrive at a practical challenge-solution combination important to the world at large.

For each challenge, unless stated otherwise, the "arena" refers to a 4' x 8' map of the world, on a table with raised borders or walls, upon which the robots completed the challenge. The challenges included open-loop movement (the robot performs a set series of actions without sensor input), closed-loop movement (the robot continually corrects its actions based on sensor input), the robotic ocean sweep, and line follower. Appendix A, Part 1, describes each challenge, including the rules and results. Parts 2 and 3 in Appendix A include the Robotic Sweep Python Code and READ Me file, respectively. Parts 4 and 5 include the Line Follower Python Code and associated READ Me file.

Block 4: Lego Mindstorms Curriculum-Aligned Group Projects

After participating in the challenges, teams were given the opportunity to develop their own group projects that they could use in the classroom. These projects provided the opportunity to make explicit connections between what they had learned up to this point of the institute to curriculum and instruction and served as a means of developing disciplinary CT skills. The teachers formed six small groups, and each selected a lesson originally designed to use with Lego Mindstorms visual drag-and-drop programming that they modified to use with Python programming. A brief overview of each project is given here with details on two of the projects given afterward. The first four projects were adapted from the Polytechnic Institute of New York University (2010).

- The Zipliner's Delight.
- Sonar Is Batty: adapted a lesson titled "measuring sound waves," which used biomimicry echolocation with ultrasonic sensors to map out a geographical area.
- Batbots: adapted a lesson titled Biomimicry: Echolocation in Robotics, which used ultrasonic sensors to maneuver robots in an arena.
- Acceleration Due to Gravity: constructed a device using the touch and rotational sensors and EV3 brick to measure the time of flight for a falling ball at different heights.
- What's Your Function: adapted a lesson titled Linear Functions from Rensselaer Polytechnic Institute's Center for Initiatives in Pre-College Education (2017).

- The Perfect Pitcher: built a motorized pitching arm that was optimized for distance and accuracy. Adapted from STEM by Design: Teaching With Lego Mindstorms EV3 (Bratzel, 2014).

The teams were tasked with determining the best way to construct the model robot described in their selected lesson. They programmed the robot to collect data using Python and performed all robot controls in code while applying an understanding of the underlying mathematics. The summer institute culminated with each group demonstrating how their robot functioned and sharing a presentation of connections to mathematics and science curriculum as well as the Python programming, including an explanation of the data collected and graphs created.

The Zipliner's Delight

This team was challenged with creating a safe way to set up a zipline and attach their EV3 brick without damage to their equipment. The challenge took perseverance, with numerous trials for setup of both hardware and Python programming to collect the desired data. The key equation of the zipline problem is defined by

$$x(t) = (h(t) - h_0)csc(\theta)$$

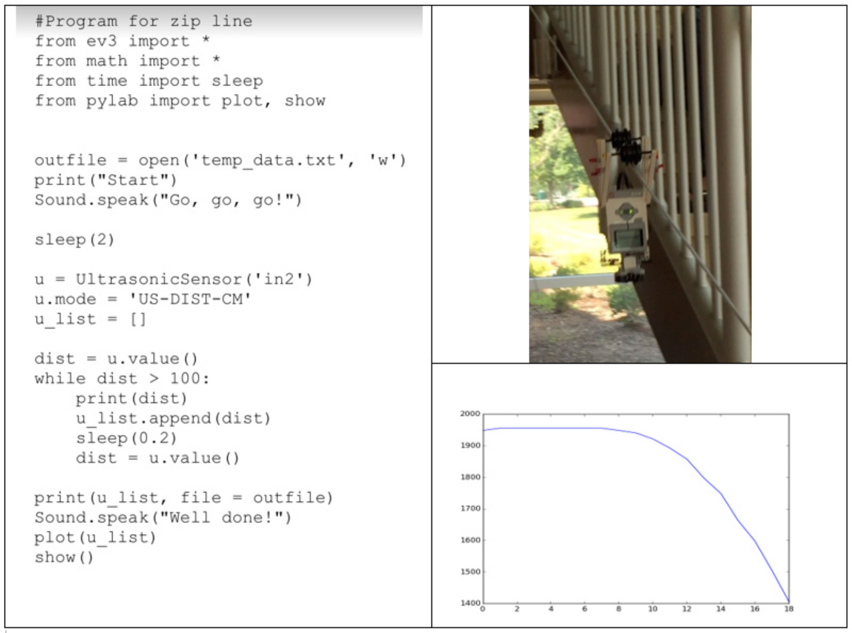
where $x(t)$ is the distance traveled across the zipline, $h(t)$ is the height above the ground, and Θ is the angle between the zipline and ground. Using the experimental setup to measure $h(t)$, and employing derivatives from algebra, the team was able to calculate interesting quantities such as velocity and potential and kinetic energies.

Figure 1 illustrates sample code, an image of the robot, and sample graph collected from the Zipliner's Delight project. Video 1 includes a sample video of the zipline robot in action. See Appendix B, Parts 1 and 2 for the Zipliner's Delight Python Code and associated READ ME file.

This group of physical science teachers located a number of curriculum standards that aligned with this project: (These were physical science standards in Tennessee prior to a transition to using standards based upon the Framework for K-12 Science Education, NRC, 2012).

- CLE 3202.Inq.3 - Use appropriate tools and technology to collect precise and accurate data.
- CLE 3202.Inq.6 - Communicate and defend scientific findings.
- CLE 3202.Math.2 Utilize appropriate mathematical equations and process to solve basic physics problems.
- CLE 3202.3.1 Investigate the relationship between speed, position, time, velocity, and acceleration.
- CLE 3202.4.2 Relate gravitational force to mass.

Figure 1 The Zipliner's DelightPython Code, Image and Graph



```
#Program for zip line
from ev3 import *
from math import *
from time import sleep
from pylab import plot, show


outfile = open('temp_data.txt', 'w')
print("Start")
Sound.speak("Go, go, go!")

sleep(2)

u = UltrasonicSensor('in2')
u.mode = 'US-DIST-CM'
u_list = []

dist = u.value()
while dist > 100:
    print(dist)
    u_list.append(dist)
    sleep(0.2)
    dist = u.value()

print(u_list, file = outfile)
Sound.speak("Well done!")
plot(u_list)
show()
```

Video 1 Zipline Example Video (https://youtu.be/9kNbRcto-bQ)

Additionally, the project aligns closely with the following CS Standards for Grades 9-10 (CSTA, 2017):

- 3A-DA-11 Create interactive data visualizations using software tools to help others better understand real-world phenomena.
- 3A-DA-12 Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.
- 3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
- 3A-AP-21 Evaluate and refine computational artifacts to make them more usable and accessible.
- 3A-AP-22 Design and develop computational artifacts working in team roles using collaborative tools.
- 3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.

The key takeaways from these results are the ability of teachers with no prior coding experience to write functional Python code to aggregate the experiment data. Also, the measured data is smooth and shows the quadratic behavior that results from acceleration and gravity.

What's Your Function?

This project (game) was created by the teachers as a fun way to mix robotics, coding, and algebra for the students. As a bonus to the project, students are required to use critical thinking to solve the final problem. The game involves attempting to guess a line equation based on the behavior of the robots. A large number line is created with discrete real-numbered tick marks. The robot is placed on one of these numbers with the same number entered into the program. After executing the program, the robot moves to the output number determined by the hidden equation. The student is given any number of robot uses to find the underlying equation.

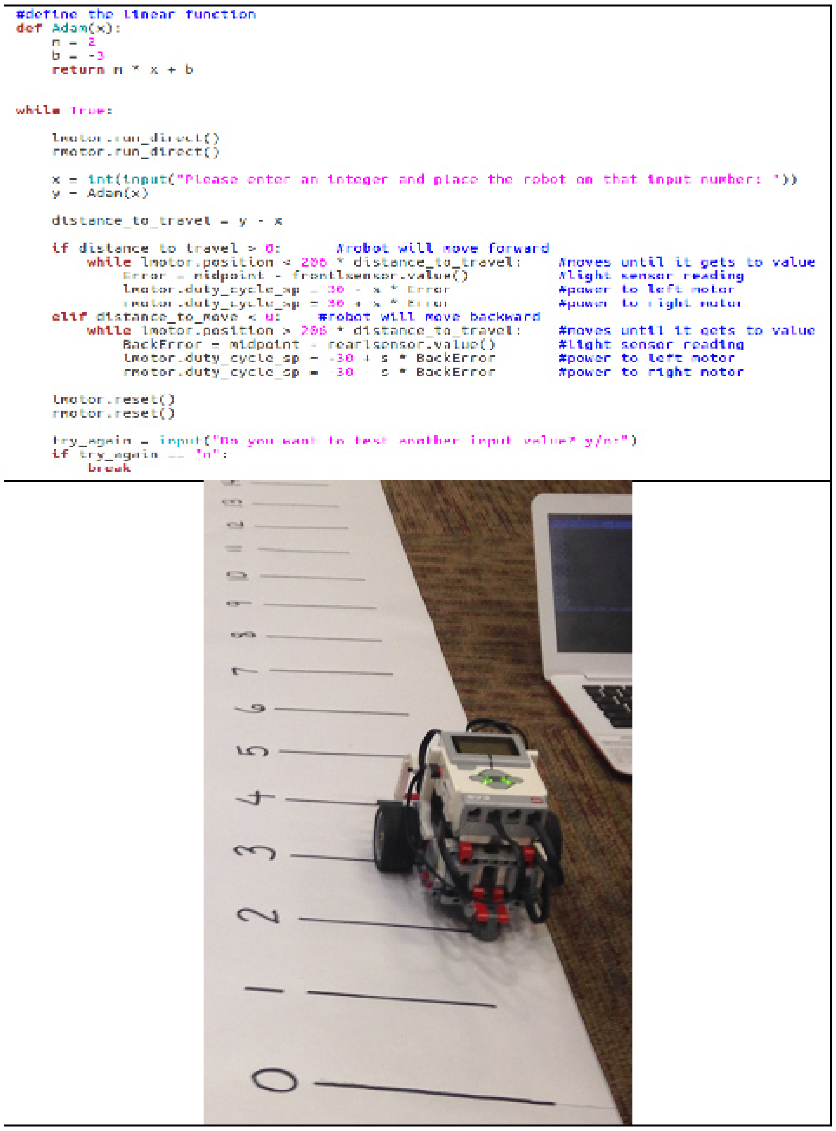
Figure 2 shows the number line and robot created for this group project as well as a sample of the corresponding code for this project. Video 2 includes a sample video of the function machine robot in action. See Appendix C, Parts 1 and 2 for the Python Code and associated READ ME file for the What's Your Function project.

Video 2 Function Machine Example Video (https://youtu.be/Wi1YymkGpMM)

This group of algebra teachers used suggested Common Core Math Standards (National Governors Association Center for Best Practices & Council of Chief State School Officers, 2010) that aligned with this project from the source lesson plan including:

- 8.F.1 Understand that a function is a rule that assigns to each input exactly one output. The graph of a function is the set of ordered pairs consisting of an input and the corresponding output.
- 8.F.2 Compare properties of two function each represented in a different way.
- 8.F.3 Interpret the equation $y = mx + b$ as defining a linear function, whose graph is a straight line; give examples of functions that are not linear.
- 8.F.4 Construct a function to model a linear relationship between two quantities. Determine the rate of change and initial value of the function from a description of a relationship or from two (x, y) values, including reading these from a table or from a graph.

Figure 2 What's Your Function? Project Sample Code and Hardware Setup



```
#define the linear function
def Adam(x):
    m = 2
    b = -3
    return m * x + b


while True:

    lmotor.run_direct()
    rmotor.run_direct()

    x = int(input("Please enter an integer and place the robot on that input number: "))
    y = Adam(x)

    distance_to_travel = y - x

    if distance to travel > 0:        #robot will move forward
        while lmotor.position < 206 * distance_to_travel:    #moves until it gets to value
            Error = midpoint - frontlsensor.value()           #light sensor reading
            lmotor.duty_cycle_sp = 30 - s * Error             #power to left motor
            rmotor.duty_cycle_sp = 30 + s * Error             #power to right motor
    elif distance_to_move < 0:      #robot will move backward
        while lmotor.position > 206 * distance_to_travel:    #moves until it gets to value
            BackError = midpoint - rearlsensor.value()        #light sensor reading
            lmotor.duty_cycle_sp = -30 + s * BackError        #power to left motor
            rmotor.duty_cycle_sp = -30 - s * BackError        #power to right motor

    lmotor.reset()
    rmotor.reset()

    try_again = input("Do you want to test another input value? y/n:")
    if try_again == "n":
        break
```

Additionally, the project aligns closely with the following CS Standards for grades 9-10 listed below (CSTA, 2017).

- 3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
- 3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
- 3A-AP-21 Evaluate and refine computational artifacts to make them more usable and accessible.

- 3A-AP-22 Design and develop computational artifacts working in team roles using collaborative tools.
- 3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.

The teachers were able to combine code from the Line Follower challenge to help the robot travel along the edge of the number line and the Open-Loop Challenge as the robot traveled a fixed distance between number ticks. The hidden function is contained in "Adam(x)" and is not shown to the students but must be learned indirectly by playing with the robot. Players quickly learn that the game only needs to be run twice in order to guess the correct values of m and b.

Saturday Workshop Fall Semester

The teachers were asked to select a book to read in grade or subject level teams to focus on improving and strengthening pedagogy emphasized by the state evaluation system, TEAM. The six middle school science, STEM, and robotics teachers selected the book STEM Lesson Essentials: Integrating Science, Technology, Engineering, and Mathematics (Vasquez et al., 2013). The seven high school physical science teachers chose Teaching High School Science Through Inquiry and Argumentation (Llewellyn, 2013). The six high school algebra teachers selected Making Sense of Algebra: Developing Students' Mathematical Habits of Mind (Goldenbury et al., 2015). The teachers used an online discussion forum to post responses to the reading between the summer institute held in June and the Saturday follow-up session held in September.

During the Saturday follow-up teachers took their postassessments, debriefed their book study, and planned conference proposals to disseminate information they learned. Two algebra teachers presented sessions for the Tennessee Mathematics Teachers Association conference, "Math and Programming Mindstorms – Mastering Mathematical Processes" and "Acceleration Due to Gravity: An EV3 Programming Project From the Coding Connections Workshop."

Five science and STEM teacher participants presented a 6-hour workshop, "K-12 Coding Connections at the Interface between Science and Mathematical Practices," at the Tennessee Science Teachers Association conference. They shared unplugged activities and games, Hour of Code, Python coding, ways in which they had integrated programming and robotics into their own classrooms, and specifics of programming projects and challenges they completed during the summer workshop.

Methods

This project was designed using a mixed methodology approach of collecting qualitative and quantitative data and a convergent parallel design was used to collect both types of data concurrently (Buchholtz, 2019; Creswell & Clark, 2017). Quantitative data were collected using a mathematics and science content assessment and a CT assessment designed specifically for this institute by grant staff as well as the TPACK assessment designed by Mishra and Koehler (2006). The 20-item mathematics and science content assessment used two items that were modified from released test items from the TIMSS Assessment (1995), which evaluated design of experimental inquiry specific to the use of robotics to explore physical science concepts, and two items that determined understanding of scientific models from the Project 2061 Science Assessments (American Association for the Advancement of Science, 2020).

The remaining 16-items were not externally validated because they were designed by grant staff specifically to measure understanding of contextual situations involving the use of robotics and Python programming, including data analysis, programming outputs, and robot functions. See Appendix D, Part 1, for a copy of the assessment. Additional quantitative sources included a pre- and postinstitute evaluation form and a post-TEAM assessment. Quantitative data was analyzed using two-sample *t*-tests with the use of a Bonferroni correction to determine the statistical significance of changes.

Narrative analysis was used to discover emergent themes within the qualitative data collected pre- and postparticipation (Patton, 1990). Participant responses to an open-ended prompt included on the TPACK survey, as well as responses to a final project evaluation form and TEAM assessment (both designed by project staff), served as the qualitative data. The TPACK survey prompt was provided pre- and postparticipation and asked participants to describe a specific teaching episode in which they effectively demonstrated or modeled combining content, technologies, and teaching approaches in a classroom lesson.

The responses to the TPACK open-ended prompt were categorized into teacher-focused and student-focused use of technology and organized by stages of the Mathematics Teachers TPACK Developmental Model (Niess et al., 2009). It included reference to the SAMR Model level (Puentedura, 2014) where applicable. The final project evaluation form required participants to describe the “top three take-aways” from participation, the most helpful part of training, and what could have been done to improve their experiences. The responses to the project evaluation form were analyzed to search for similarities and differences between participant ideas to identify the emergent themes for top take-aways, what was most helpful, and what could have been improved.

The TEAM assessment asked participants to describe three indicators positively impacted by participation. The responses were sorted by indicator, and the top five indicators for the entire group were reported along with representative comments.

Findings

In this section we share a comparison of teacher performance on the CT, mathematics and physical science pre- and postassessment, both the quantitative and qualitative results for the TPACK assessment and the Coding Connections evaluation form, and qualitative analysis for the TEAM assessment form.

Content Knowledge Pre- and Postassessment

The content assessment included 20 questions aligned with CT, mathematics, and physical science. The pretest average was 44.5 with a range in scores from 25 to 92.5. The posttest average was 59.5 with a range of scores from 27.5 to 85. The 15-point increase in the average scores was statistically significant at the $p < 0.001$ level with the use of a two-sample *t*-test. The assessment included a mixture of question types that represented a range of the practices, as developed by Weintrop et al. (2016).

To determine which questions had the largest contributions to the overall statistical significance, individual pre- and postassessment two-sample *t*-tests were performed. When a Bonferroni correction is applied to this collection of 20 individual *t*-tests, $p < 0.0025$ is required for an individual test to be significant at the 0.05 level. Only two of the questions were independently statistically significant at this level. To outline their contributions, five out of the 20 questions were selected for discussion and illustrated in Appendix D, Part 2. Again, note that the significance levels reported may not show that improvement is independently statistically significant, but they do illustrate the relative contributions of each question to the overall significant result.

The first of these problems illustrated in Part A combines several of the categories in Weintrop's (2016) taxonomy. In this problem the participants are asked to complete a Python function that converts from the raw output of a temperature sensor to degrees Fahrenheit. This task involves understanding the relationships within a system (System Thinking Practices) as well as programming (Computational Problem Solving Practices) and constructing a computational model (Modeling and Simulation Practices). This problem does not require a high level of knowledge about Python syntax, but rather emphasizes the functional relationships between different values (raw sensor data, Celsius and Fahrenheit temperatures) as well as presenting them in a format that differs slightly from the usual mathematical notation. The pretest average on this question was 19%, while the posttest average was 44% with a *p*-value of 0.014.

Other problems involved creating computational models about real-world situations. For example, the problem illustrated in Part B involves understanding the relationship between the straight-line distance between two objects and the motion of one of the objects. This problem is highly dependent on designing and constructing an appropriate computational model (Weintrop's Modeling and Simulation Practices). The pre-posttest improvement on this question was from 19% to 56.5%, with a *p*-value of 0.0045.

The next problem illustrated in Part C involves analyzing Python code to determine the behavior of a robot. Again, this problem does not require a high level of knowledge about Python syntax, but it does require the participant to think abstractly and create a mental model of how the robot will behave. The left wheel is turning faster than the right wheel, so the robot will turn right. This action involves Weintrop's Systems Thinking Practices by investigating a complex system as a whole. The pretest average on this question was 28% and the posttest average was 50%, which was statistically significant with a *p*-value of 0.0024.

Part D requires the participant to think about creating a complex system (a robot functioning in a physical environment, which also must be created) to construct an experiment to collect data about potential and kinetic energy. This activity involves considerations about data collection (Weintrop's Data Practices), as well as thinking of the system as a whole (Weintrop's System Thinking Practices). The improvement on this question was from 28% to 44% with a *p*-value of 0.05.

Some of the questions depended more on a knowledge of Python syntax. For example, the problem shown in Part E involves knowledge of a Python if-else flow control structure. This action is programming knowledge and falls within Weintrop's Computational Problem Solving Practices. The pre-posttest results on this problem improved from 37.5% to 81.5%, which was statistically significant with a *p*-value of 0.0019.

TPACK Survey

TPACK Quantitative Results

The TPACK assessment included 32 Likert-scale items divided into seven categories taken from the Survey of Preservice Teachers' Knowledge of Teaching and Technology (Schmidt et al., 2009). Each item response was scored with a value of 1 for *strongly disagree* to 5 for *strongly agree*. The participant's responses were averaged over all 32 questions. Additionally, the participant's responses were averaged over each construct. For example, the six questions under Technology Knowledge (TK) are averaged to produce one score.

A paired two-sample *t*-test was computed for the participant's average responses over all the questions to show a significant change ($p = 0.0082$). To determine the individual contributions, separate two-sample *t*-tests were performed on each construct. Once a Bonferroni correction was imposed, technological pedagogical knowledge (TPK) was the only construct that showed a statistically significant change. Table 1 includes the participant average results for the pre- and post-TPACK, standard deviation, and the *p*-value to help determine the contribution of each construct to the overall statistical significance.

TPACK Qualitative Results

On the pre- and postassessments, the participants were asked to describe a specific teaching episode in which they effectively demonstrated or modelled combining content, technologies and teaching approaches in a classroom lesson. On the preassessment participants responded in the following ways: unable to describe a teaching episode using technology ($n = 4$), teacher use of technology for instruction with an interactive whiteboard or document camera ($n = 2$), student use of robotics that was not aligned with content instruction ($n = 2$), student use of graphing calculators ($n = 1$), student use of a spreadsheet to solve math problems ($n = 1$), student use of internet for research and to create power point presentations (n = 1), student use of content simulations to model science concepts (n = 2), and student use of Vernier probeware to collect and analyze data (n = 2).

Table 1 Pre- and Post-TPACK Assessment Results

TPACK Subscale	**Pretest**		**Posttest**		***p* value**
	Mean	*SD*	Mean	*SD*	
TK (6 items)	3.96	0.46	4.16	0.49	0.009
Math & Science CK (6 items)	4.17	0.46	4.40	0.45	0.007
PK (7 items)	4.29	0.52	4.43	0.54	0.110
PCK Math & Science (2 items)	3.97	0.62	3.94	0.58	0.359
TCK Math & Science (2 items)	3.75	0.68	4.06	0.70	0.027
TPK (7 items)	4.04	0.53	4.34	0.51	0.006
TPACK Math & Science (2 items)	3.84	0.70	4.06	0.70	0.110
Note. Pretest and posttest scores are averages between 1 and 5. n = 16.					

One teacher did not have an instructional position at the time of the postassessments, so her data is not included in the qualitative results. On the postassessment participants responded as follows: no use of technology at this point in school year or major modifications to integrate inquiry-based science with no mention of technology (n = 2), teacher use of technology including interactive whiteboard and video (n = 3), student use of coding that was not aligned with content (n = 3), student use of iPad apps for metric conversion (n = 1), and student use of coding and or robotics aligned with disciplinary content (n = 6).

Responses from seven of the teachers are included in Table 2 as representatives of how the teachers started using robotics or programming in their classrooms and after-school programs. The responses are organized by stages of the Mathematics Teachers TPACK Developmental Model by categorizing how teachers used technology (Niess et al., 2009) and include reference to the SAMR Model level (Puentedura, 2014), where applicable.

Table 2 TPACK Postassessment Response Analysis: TPACK Stage and SAMR Model Level

Stage/Level	Representative Quotes
Accepting or Persuasion Stage n = 2	• "I used chromebooks to teach coding to my Algebra I students. We did basics with codecombat.com. They learned through trial and error and through help from others who were 'getting' it." High school algebra, 15 years, female • "I implemented the python 3 coding with a coding club." Eighth-grade science, 11 years, female
Adapting or Decision Stage SAMR: Modification Level n = 2	• "During the robotics club... used EV3 coding to conduct tests to determine the # of wheel rotations required to go a certain distance. This was replicated 5 times and for 3 wheel sizes. Students created data tables and did final average calculations. All measurements used metric system." High school physical science, 8 years, female • "My physics students are being given the opportunity to integrate EV3 programming with Vernier probeware. Since my school does not offer programming as a course, many of my students have little or even no experience in this field. I am confident that providing this experience is leading to a more enriched physics course for my students!" High school physical science, 8 years, female
Exploring or Implementation Stage SAMR: Modification or Redefinition n = 3	• "Combined writing inequalities with the Lego Robot following a line program that uses the light sensor. I explained the general outline of the python program and how the light sensor works. I had groups of students predict what the inequalities should be to make the robot follow the line. Each group then tried out their inequalities with the robot. Smaller classes had the opportunity to modify their inequalities and try several times to find the correct inequalities to make the robot follow the line." High school algebra, 6 years, female • "I recently used Nearpod to introduce a Robotics lesson which combined math (ratio and rate), and science (force and momentum) topics in a challenge that requires the students to drive onto and balance on a bridge." Middle school Robotics, 3 years, female • "Modeling the parallels between functions in python and functions in mathematics. I taught students how to build functions in python to highlight the key concepts of functions changing an input to one specific output. Student used a class set of chromebooks and the website repl.it, and I used a projector to share my own code. I started with the concept we're going to make a function that behaves like $f(x) = 2x$ ^ 2 + 3. How do we do that? The students then generated ideas, and I provided the specific code they needed to accomplish. Most students were then able to adapt the code for other functions." High school algebra, 1 year, female

Two of the teachers asked students to use Python coding but used them for activities that were peripheral to classroom instruction, which is representative of the “Accepting or Persuasion” stage of the Mathematics Teachers TPACK Developmental Model. The SAMR Model level is not indicated for this group of teachers because they were not using the technology for disciplinary (mathematics or science) instruction.

Two of the teachers described scenarios at the “Adapting or Decision” stage of the Mathematics Teachers Developmental model by having students use the technology to enhance or reinforce ideas they have previously learned. Additionally, they asked students to use technology to redesign a task which would normally be completed without technology which is at the “Modification” level of the SAMR Model. Three teachers described teaching scenarios within the “Exploring or Implementation” stage of the Mathematics Teachers Developmental model by engaging the students in high-level thinking by using computer programming as a learning tool for mathematics or science. They asked students to use technology at the “Modification” (redesigned a task with technology) or possibly even the “Redefinition” (creation of a new task) level of the SAMR Model.

Coding Connections Pre- and Postevaluation Form

Eight Likert-scale questions were asked to determine perceptions regarding preparation for teaching as a result of participation in the project. Two open-ended questions were asked at the conclusion of the project to determine three take-aways gained from their experience and what could have improved their experience in the project. Each Likert-scale item response was scored with a value of 1 = *strongly disagree* to 5 = *strongly agree*.

A two-sample paired t-test was completed to determine the level of significance of any changes for each item. When a Bonferroni correction as applied to this collection of eight individual t-tests, $p < 0.00125$ is required for an individual test to be significant at the 0.01 level. Only two of the questions, were independently statistically significant at this level. Table 3 includes the participant average results for the pre- and postevaluation form, standard deviation, along with the p value.

All responses to the Likert-scale items were highly favourable, and after participation in grant activities all participants felt better prepared to use mathematics practice standards (statistically significant change at $p < 0.01$), as well as use computer programming and robotics aligned with algebra or physical science instruction (statistically significant at $p < 0.01$). No change was found in their perceived ability to integrate critical thinking activities (4.19 average) or use TEAM pedagogy in their classrooms (3.94 average). The teachers were, however, able to name a number of TEAM indicators they felt better prepared for with supported examples in response to an open-ended prompt.

Table 3 Coding Connections Pre- and Postevaluation Form

Question Prompt	**Pretest**		**Posttest**		***p* value**
	Mean	*SD*	Mean	*SD*	
I feel that I am prepared to teach CCSS-Mathematical Content standards	2.75	1.35	3.5	0.94	0.0065
I feel that I am prepared to teach CCSS-Mathematical Practice Standards.	2.56	1.17	3.5	0.94	0.00054
I feel that I am prepared to integrate Algebra I and Physical Science Content in my classroom.	3.25	1.15	3.88	0.48	0.0229
I feel that I am prepared to use the Science and Engineering Practices as outlined by the K-12 Framework for Science Education.	3.5	1.06	4.13	0.60	0.014
I feel that I am prepared to teach using TEAM pedagogy.	3.94	0.75	3.94	0.56	0.5
I feel that I am prepared to select challenging mathematical tasks.	3.19	1.18	3.56	0.79	0.041
I feel that I am prepared to use critical thinking and problem-solving activities in my classroom.	4.19	0.73	4.19	0.63	0.5
I feel that I am prepared to use computer programming and robotics aligned with CCSS-Math and physical science in my classroom.	2.69	1.31	4.31	0.77	0.00017

Note. Pre- and postassessment scores are averages between 1 and 5. *n* = 16.

The first open-ended prompt included on the postproject evaluation form was, "What are the top three takeaways you gained from your experiences with this project?" We isolated five themes from the 16 responses received, including enhanced understanding of programming/coding, ability to integrate coding into curriculum, introduction to resources for teaching science with inquiry or mathematics manipulatives that teachers had not used before, ways to interest and motivate students to learn, and an enjoyment for the challenge provided by programming. Table 4 includes representative quotes from teachers for each isolated theme. Each quote is labeled by subject/grade level, number of years teaching experience, and gender.

Table 4 Top Takeaways From Coding Connections

Theme	**Representative Quotes**
Enhanced understanding of programming/ coding *n* = 9	"I have a deeper understanding of programming and working with sensors. I could only help my robotics team with teaching them basic coding last year and any advanced programming and troubleshooting they had to learn how to do on their own (which they did, with encouragement from me!). But now, if there is a problem with a program and they get stuck, I can lend my expertise to help them possibly come up with a solution. I am better able to recognize issues with the programs now since we were "made" to troubleshoot our own problems in the workshop!" Sixth-grade science, 10 years, female.
Integrate programming curriculum *n* = 8	"How to add CS into my existing Algebra I curriculum." High School Algebra, 6 years, female. "Physical science can be integrated into biology using the robots." High school physical science, 18 years, female.
Introduction to resources *n* = 5	"I enjoyed seeing the many different types of manipulatives that I could use within my classroom." High school algebra, 1 year, female.
Ways to interest & motivate students to learn *n* = 4	"I have new ideas to present to my students to peak [sic] their interest in Mathematics. I have used multiple coding activities in my classroom." High school algebra, 15 years, female. "I enjoyed the robotic competitions as a fun way to engage students in learning." High school algebra, 1 year, female.
Valued & enjoyed the challenge provided by programming *n* = 4	"Coding with Python is easier than it looks. Enjoyed the challenge that coding and constructing the robots offered. It felt good to be able to say, I(We) did that!" High school physical science, 18 years, female. "Sometimes the struggle leads to more learning. Curiosity can be an excellent motivator in learning." High school algebra, 6 years, female.

The second open-ended prompt on the post-evaluation form was, "What could we have done to improve your learning experience?" We isolated three areas of the training for which teachers suggested improvements, including a need to differentiate instruction for participants with varied ability levels for programming, a request to share more ideas for ways to integrate Lego Mindstorms with curriculum, and ways to manage wait time when there were technology glitches. Table 5 includes representative quotes from teachers for each isolated theme. Additionally, two teachers wanted more time to learn about the Lego Mindstorms software that is packaged with the robotics, and two teachers wanted more time, in general, for activities.

Table 5 Suggested Areas of Improvement for Coding Connections

Theme	Representative Quotes
Differentiate instruction for varied programming levels *n* = 5	"As a science teacher, I was lost with the portion that required the math theorem, which took a while to catch up in order to follow the rest of the lecture." High school physical science, 8 years, female.
More ideas to integrate Lego® Mindstorms® with curriculum *n* = 3	"I wish we had had more opportunity to discuss how best to implement the EV3 for our specific classes. I appreciate learning about coding and the usages of the robot, but I feel like I have had to do a lot of my own research to figure out how to use the robot effectively in my class that has limited time resources." High school physical science, 5 years, female.
Time management for technology glitches *n* = 3	"Troubleshooting was a large problem in the beginning, but I'm not sure what you could do to improve that. "High school algebra, 1 year, female.

TEAM Postassessment

The TEAM Instruction rubric is used by supervisors of instruction in the state of Tennessee to evaluate teacher instruction. The teachers were asked to select three indicators out of 12 possible, for which they felt they made positive changes as a result of grant participation. Representative quotes of the five most mentioned indicators are included in Table 6.

Discussion

Key Findings

The results of this study revealed several important findings as related to the research questions. First in terms of conceptual and attitudinal changes (Research Question 1) the participants showed marked improvement in their confidence in mathematics content knowledge and use of the mathematics practice standards, which aligns well with the mathematics content modules emphasized during the institute and statistically significant improved performance from the pre to post content assessment.

Additionally, the participants showed statistically significant improvement on the overall TPACK assessment. Due to the high demands of teaching the participants how to use discipline-specific Python programming during the 2-week institute, mathematics and science-specific pedagogical strategies (such as inquiry-based science and using manipulatives for math instruction) were relegated to the book study conducted between the summer institute and September follow-up session.

Table 6 TEAM Indicator Improvement Coding Connections

Indicator	**Representative Quotes**
Activities and Materials *n* = 12	“I can incorporate activities of coding for students to apply their knowledge of inequalities, functions, recursive functions… hoping to add more as the year progresses and new ideas develop.” High school algebra, 6 years, female. “As a science teacher learning more about math concepts, I can reinforce things they learn in math class (like ratios and algebraic thinking) as they relate to the science concepts.” Sixth-grade science, 10 years, female.
Motivating Students *n* = 10	“… it showed me how to present mathematics to students who think they're never going to use this again and show them that math is in everything. Computers are a very tangible way for them to see math in the real world.” High school algebra, 1 year, female.
Problem Solving *n* = 10	“The coding has resulted in a better understanding for my students of why problem solving is so important. If one thing is wrong, they have to go back and find that area and fix it for the whole thing to work.” High school algebra, 15 years, female.
Thinking *n* = 5	“Learning about coding has helped me see how logical thinking needs to be taught to our students and I feel like I have been given tools to help my students learn about logical step by step thinking that are needed to be proficient at coding and science in general.” High school physical science, 5 years, female.
Teacher Content Knowledge *n* = 5	“I feel that I am more prepared in an area of STEM that prior to the training I was completely unaware. I will use this new knowledge and these new skills to teach my students and open their eyes to what computer coding is.” Eighth-grade science, 11 years, female.

Participation in the institute impacted teacher use of disciplinary CT strategies in a number of ways (Research Question 2). Considering common themes across qualitative assessments, participation in this institute prepared over half of these teachers to integrate programming with curriculum (Top Takeaway, *n* = 8) by enhancing their understanding of programming (Top Takeaway, *n* = 9) and describing coding and robotics as a contributing factor to their top-rated indicator for improvement on the TEAM assessment, Activities and Materials (*n* = 12). Six teachers also described a specific situation in which they used coding and or robotics as part of mathematics, science, or STEM instruction in the TPACK postassessment.

Additionally, a large number of the participants felt that programming and using robotics for real-world applications is a useful way to motivate students to learn mathematics and science (TEAM assessment, *n* = 10 and Top Takeaway, *n* = 4). The amount of troubleshooting and perseverance that the robotics challenges aligned with programming in Python involved perhaps prompted the high number of participants who perceived "Problem Solving" as one of the TEAM indicators for which they were most prepared (n = 10), which is a central component of CT (CSTA, 2016).

Teacher participants suggested three areas of need for improvement (Research Question 3), all of which could be addressed efficiently with sustained PD over multiple years that provides repeated opportunities to plan, implement, and reflect on teaching strategies and for teachers to discuss and coordinate efforts to design and implement coherent teaching (CSSS, 2018). An area of need mentioned by five teachers was differentiating instruction for participants with varied ability levels for programming and mathematics content knowledge. Three teachers suggested finding ways to manage wait time when there were technology glitches.

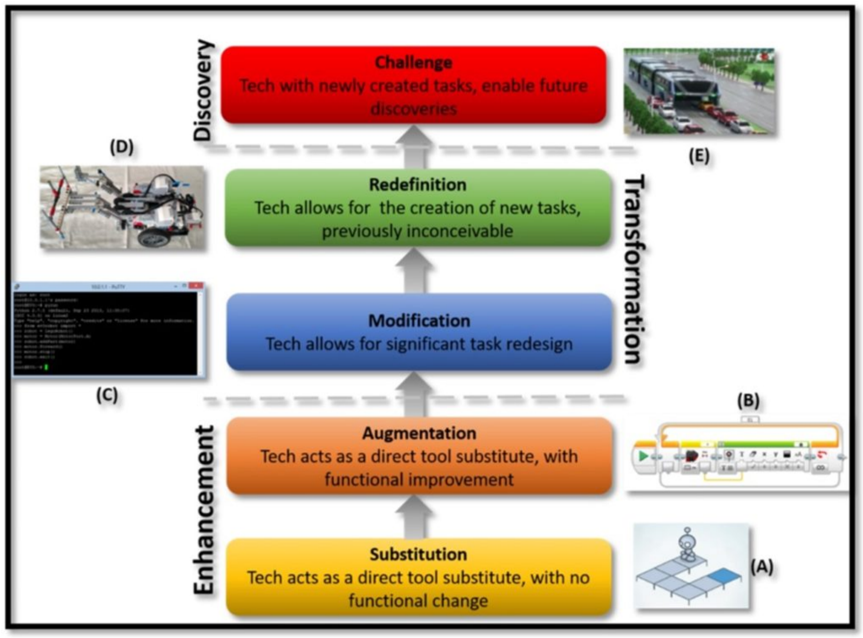
We noted that glitches tended to occur with teachers who were novice programmers as well as insecure with using the technology. One solution that could be pursued to address ability levels and wait time is setting up breakout rooms for beginning, intermediate, and novice programmers, which would allow for targeted instruction. Another ideal technique would be to ensure that heterogeneous ability grouping is planned for challenges as a model for how to enact these activities in the classroom and as a means to develop readiness for participants to serve as mentors, coaches, and leaders to sustain and support ongoing professional learning (CSSS, 2018).

Three teachers requested opportunities to learn more ways to integrate Lego Mindstorms with curriculum. Participants began to develop activities to use Lego Mindstorms in the classroom at the beginning of the next school year. It would be ideal, however, to extend the PD and teacher collaboration beyond the original training period to ensure that all teachers have the necessary social supports and curriculum materials to implement the teaching approaches introduced (CSSS, 2018).

Implications for Teacher Education

Preparing teachers to use digital technologies and CT concepts effectively requires adaptations to the design of PD for teachers who may not have had exposure to these tools and ideas in their preparation programs, particularly since few programs currently focus on training preservice teachers to incorporate CT into K-12 classrooms (Sands et al., 2018; Yadav et al., 2017). We propose an innovative approach to teaching disciplinary CT and expansion of the SAMR Model that can be used for mathematics and science PD and classroom instruction found in Figure 3.

Figure 3 Coding Connections and SAMR Levels


Discovery
Challenge
Tech with newly created tasks, enable future discoveries
(D)
(E)
Redefinition
Tech allows for the creation of new tasks, previously inconceivable
Transformation
Modification
Tech allows for significant task redesign
(C)
(B)
Augmentation
Tech acts as a direct tool substitute, with functional improvement
Enhancement
Substitution
Tech acts as a direct tool substitute, with no functional change
(A)

Note. A. Lightbot; B. Lego Mindstorms EV3 Programming; C. Python Programming; D. Lego Mindstorms EV3 Robot programmed to address a challenge; E. Urban Infrastructure Grand Challenge. *Note*: Image modified from the creation of Dr. Ruben Puentedura, Ph.D. http://www.hippasus.com/rrpweblog

To reiterate, we suggest the use of TPACK to support meaningful selection of SAMR levels. At the beginning of the institute, teachers played Lightbot (A) as a substitution for unplugged programming, which helped them learn basic algorithm design and function calls. This introductory coding could then be augmented with visual coding (B) through the Lego Mindstorms visual coding environment with more complex blocks and capabilities. Where A moves a virtual robot, B moves a robot in the world. Due to the limited time available in this institute we did not use the Lego visual coding tools to allow for time to work with the Python programming language. Python is introduced (C) that matches all the capabilities of B, but then opens up a new world of task design. Indeed, the final robot (D), as redefined by the participant to solve a challenge, was inconceivably difficult to bring to task until after A-C.

Throughout the institute our teacher participants completed robotics-based challenges similar to those used in First Lego League robotics competitions representing the redefinition level (D). We propose the next step is to complete challenges similar to the National Academy of Engineering (2020) Grand Challenges. For example, the grand challenge, "Restore and Improve Urban Infrastructure" as depicted by E represents our modification of the SAMR Model to include an additional level, a challenge. At this level, in the context of our training, learners are provided an opportunity to solve preferably real-world challenges incorporating the use of designing, troubleshooting, and programming robots to help discover solutions to these future challenges.

An important consideration is the notion of equitable STEM practices and access to resources to develop disciplinary CT skills for teachers and their students from high-need school districts, such as the population served in this study. As noted in the discussion section, limitations included the lack of funding to purchase classroom sets of robotics and a limited time to complete PD training. A number of aspects of our PD experience can be replicated without funding or by reallocating funding to support more students over time with the use of virtual robots and online and unplugged coding options, as described in Table 7.

Virtual robots allow users to create settings for and program robots without having access to the expensive hardware. Robot Virtual Worlds and Virtual Robotics Toolkit, in particular, offer perpetual licenses for a set number of students that can be used year after year at about the cost of two of the Lego Mindstorms kits. The hands-on kits provide a certain motivation factor to students; therefore, we recommend having at least one kit on hand for groups of students to rotate using while the majority of the class programs and interacts virtually.

We introduced the participants to a number of free online coding options including Code.org, CodeCombat, and Repl.it, and teachers designed instruction for student use of these tools particularly in mathematics classrooms. Microsoft's open-source platform, MakeCode, for micro.bit, Minecraft, Lego Mindstorms EV3, and more allows for creating CS learning experiences with both block and text editors for learners at different levels. And finally, unplugged programming with the use of board games and free or low-cost materials found at CS Unplugged can introduce students to important aspects of CT without the use of a computer.

Yadav et al. (2016) suggested pedagogical approaches to embedding CT-based problem-solving in K-12 classrooms that do not necessarily rely on programming by using algorithmic thinking to break down tasks within science and mathematics, collecting, representing, and analyzing data, using abstraction to develop and represent models of the real world, and using automation with simulation tools. With equity and access in mind, these suggested modifications could increase the sustainability of PD opportunities offered in high-need settings.

Conclusions

Our institute design suggests a number of recommendations for teacher education. The Coding Connections institute provided teachers the opportunity to increase TPACK and disciplinary CT skills by coding robots using Python to perform specific and adaptable tasks to solve challenges. Teacher reflections illuminated a number of activities they developed for using programming and robotics at the Augmentation, Modification, and Redefinition levels of the SAMR Model. As a result of participation, these teachers were able to share their experiences at state teacher conferences, and they served as resources on discipline-specific mathematics and science CT within their schools. Burrows et al. (2017) highlighted a number of recommendations to increase the use of integrated STEM activities that are also supported by this study, including accessible information to get teachers started, ongoing PD through workshops and conferences, and support and commitment from administrators.

Table 7 Low-Cost or Free Alternatives for Developing CT Skills

Options	**Tools**
Virtual Robots	• Robot Virtual Worlds: NXT-G, EV3, or LabView for Lego, Vex http://www.robotvirtualworlds.com/ (*For Cost*) • Virtual Robotics Toolkit: Lego EV3 Mindstorms https://www.virtualroboticstoolkit.com/ (*For Cost*) • Birdbrain Technologies: multiple devices https://www.birdbraintechnologies.com/remote-robots/three-ways-remote-robots/
Online Coding Options (All Free)	• Microsoft Makecode https://www.microsoft.com/en-us/makecode • Code.org https://code.org/ • Repl.it: Free in-browser coding in 50+ languages https://repl.it/ • CodeCombat: Python & JavaScript https://codecombat.com/ • Scratch https://scratch.mit.edu/
Unplugged Coding	• CS Unplugged https://csunplugged.org/en/ • Board Games – Code Master, Code Monkey Island, Turing Tumble https://www.turingtumble.com/ (*For Cost*)

Mathematics and science content can be aligned with CT practices at the middle and high school level in lieu of standalone CS courses as a meaningful way to address the top three skills needed in the workplace in 2020, including complex problem solving, critical thinking, and creativity (ISTE 2017; World Economic Forum, 2016). Developing CT practices within disciplines, or using integrated STEM activities through authentic projects, can improve critical thinking skills, cultivate interest in STEM careers, and can empower learners with a more realistic view of STEM disciplines (Burrows et al., 2017; Qin, 2009; Weintrop et al., 2016).

Author Note

This material is based upon work supported by an Improving Teacher Quality Grant from the Tennessee Higher Education Commission under Grant No. 33201-08816 FY 16-17.

References

American Association for the Advancement of Science (2020). *Science assessment*. http://assessment.aaas.org/pages/home

Bower, M., Wood, L. N., Lai, J. W., Howe, C., Lister, R., Mason, R., Highfield, K., & Veal, J. (2017). Improving the computational thinking pedagogical capabilities of school teachers. *Australian Journal of Teacher Education, 42*(3), 53-72

Bratzel, B. (2014). *STEM by design: Teaching with LEGO Mindstorms EV3*. College House Enterprises LLC.

Briggs, J. R. (2013). *Python for kids: A playful introduction to programming*. No Starch Press.

Bull, G., Hodges, C., Mouza, C., Kinshuk, Grant, M., Archambault, L., Borup, J., Ferdig, R.E., & Schmidt-Crawford, D. A. (2019). Conceptual dilution. *Contemporary Issues in Technology and Teacher Education, 19*(2). https://citejournal.org/volume-19/issue-2-19/editorial/editorial-conceptual-dilution

Burrows, A.C., Garofalo, J., Barbato, S., Christensen, R., Grant, M., Kinshuk, Parrish, J., Thomas, C., & Tyler-Wood, T. (2017). Editorial: Integrated STEM and current directions in the STEM community. *Contemporary Issues in Technology and Teacher Education, 17*(4). https://citejournal.org/volume-17/issue-4-17/science/editorial-cite-journal-science-education-3-0

Cabrera, L. (2019). Teacher preconceptions of computational thinking: a systematic literature review. *Journal of Technology and Teacher Education*, *27*(3), 305-333.

Code Master. (2015). https://www.thinkfun.com/products/code-master/

Code Monkey Island. (n.d.). https://www.kickstarter.com/projects/rajsidhu/code-monkey-island-making-programming-childs-play

Computer Science Teachers Association. (2016). K–12 Computer science framework. http://www.k12cs.org.

Computer Science Teachers Association. (2017). *CSTA K-12 computer science standards,* Revised 2017. Computer Science Teachers Association.

Committee on STEM Education of the National Science & Technology Council. (2018). *Charting a course for success: America's strategy for STEM Education*. https://www.energy.gov/sites/prod/files/2019/05/f62/STEM-Education-Strategic-Plan-2018.pdf

Council of State Science Supervisors. (2018). *Science professional learning standards*. http://cosss.org/Professional-Learning

DeMonte, J. (2013). *High-quality professional development for teachers: Supporting teacher training to improve student learning*. Center for American Progress.

Desimone, L. M. (2009). Improving impact studies of teachers' professional development: Toward better conceptualizations and measures. *Educational Researcher, 38*(3), 181-199.

Goldenberg, E. P., Mark, J., Kang, J. M., Fries, M., Carter, C. J., & Cordner, T. (2015). *Making sense of algebra: Developing students' mathematical habits of mind*. Heinemann.

Grover, S., & Pea, R. (2013). Computational thinking in K–12 a review of the state of the field. *Educational Researcher, 42*(1), 38–43.

International Society for Technology in Education. (2016). *ISTE standards for students*. https://www.iste.org/standards/for-students

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

International Society for Technology in Education. (2018). *ISTE standards for educators: Computational thinking competencies*. https://www.iste.org/standards/computational-thinking

Koba, S., Wojnowski, B., & Yager, R. E. (Eds.). (2013). *Exemplary science: Best practices in professional development*. NSTA Press.

Koehler, M.J., Mishra, P., & Cain, W. (2013). What is technological pedagogical content knowledge? *Journal of Education, 193*(30), 13-19.

Littlecodr Games Inc. (2015). http://littlecodr.com/

Llewellyn, D. (2013). *Teaching high school science through inquiry and argumentation*. Corwin Press.

Loucks-Horsley, S., Stiles, K., & Hewson, P. (1996). Principles of effective professional development for mathematics and science education: A Synthesis of Standards. *NISE Brief, 1*(1), n1.

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers College Record, 108*(6), 1017-1054.

Mouza, C., Yang, H., Pan, Y. C., Ozden, S. Y., & Pollock, L. (2017). Resetting educational technology coursework for pre-service teachers: A computational thinking approach to the development of technological pedagogical content knowledge (TPACK). *Australasian Journal of Educational Technology*, *33*(3), 61-76.

National Governors Association Center for Best Practices & Council of Chief State School Officers. (2010). *Common core state standards for mathematics*. Authors.

National Research Council. (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. The National Academies Press.

Niess, M. L., Ronau, R. N., Shafer, K. G., Driskell, S. O., Harper, S. R., Johnston, C., Browning, C., Ozgun-Koca, & Kersaint, G. (2009).

Mathematics teacher TPACK standards and development model. *Contemporary Issues in Technology and Teacher Education, 9*(1), 4-24. https://citejournal.org/volume-9/issue-1-09/mathematics/mathematics-teacher-tpack-standards-and-development-model

Polytechnic Institute of New York. (2010). Applying mechatronics to promote science. http://engineering.nyu.edu/gk12/amps-cbri/html/resources/classroom_menu.html

Portnoy, L. (2018, December 27). How SAMR and tech can help teachers truly transform assessment. *EdSurge News*. https://www.edsurge.com/news/2018-02-01-how-samr-and-tech-can-help-teachers-truly-transform-assessment

Puentedura, R. (2014). SAMR: A contextualized introduction [Web log post]. http://www.hippasus.com/rrpweblog/archives/2013/10/25/SAMRAContextualizedIntroduction.pdf

Qin, H. (2009, March). Teaching computational thinking through bioinformatics to biology students. *ACM SIGCSE Bulletin, 41*(1), 188-191.

Rensselaer Polytechnic Institute. (2017). Rensselaer Center for Initiatives in Pre-College Education. http://www.rpi.edu/dept/cipce/index.html

Sands, P., Yadav, A., & Good, J. (2018). Computational thinking in K-12: In-service teacher perceptions of computational thinking. In M. Khine, (Ed.), *Computational thinking in the STEM disciplines* (pp. 151-164). Springer International Publishing.

Schmidt, D. A., Baran, E., Thompson, A. D., Koehler, M. J., Mishra, P., & Shin, T. (2009). Survey of preservice teachers' knowledge of teaching and technology. *Récupéré le*, *2*.

Schwab, K. (2016, January 14). *The fourth industrial revolution: What it means, how to respond*. World Economic Forum. https://www.weforum.org/agenda/2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/

Schwab, K., & Davis, N. (2018). *Shaping the future of the fourth industrial revolution*. Currency.

Shapiro, D. (2013). Robot turtles: The board game for little programmers. Kickstarter, Inc.

Shulman, L. (1986) Those who understand: Knowledge growth in teaching. *Educational Researcher, 15*(2), 4-14.

Sneider, C., Stephenson, C., Schafer, B., & Flick, L. (2014). Exploring the science framework and NGSS: Computational thinking in the science classroom. *Science Scope, 38*(3), 10.

Suters, L., & Suters, H. (2020). Coding for the Core: Computational thinking and middle grades mathematics. *Contemporary Issues in Technology and Teacher Education*, *20*(3), 435-471. https://citejournal.org/volume-20/issue-3-20/mathematics/coding-for-the-core-computational-thinking-and-middle-grades-mathematics

Thinkersmith (2013). Traveling circuits lesson 3: My Robotic Friends. https://csedweek.org/files/CSEDrobotics.pdf

TIMSS Assessment (1995). International Association for the Evaluation of Educational Achievement. TIMSS & PIRLS International Study Center, Lynch School of Education, Boston College. https://isc.bc.edu/timss1995i/TIMSSPDF/CitemPhy.pdf

Tennessee Department of Education. (n.d.). TEAM: Tennessee Educator Acceleration Model. https://team-tn.org/

Vasquez, J. A., Sneider, C. I., & Comer, M. W. (2013). *STEM lesson essentials, grades 3-8: Integrating science, technology, engineering, and mathematics*. Heinemann.

Weintrop, D., Beheshti, E. Horn, M., Orton, K., Jona, K., Trouille, L. & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127–147.

Wing, J. (2011). Research notebook: Computational thinking—What and why. *The Link Magazine*, 20-23.

World Economic Forum. (2016). *The future of jobs: Employment, skills and workforce strategy for the fourth industrial revolution. Global Challenge Insight Report*. http://www3.weforum.org/docs/WEF_Future_of_Jobs.pdf

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: pedagogical approaches to embedding 21st century problem solving in K-12 classrooms. *TechTrends*, *60*(6), 565-568.

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

Yaşar, O. (2013). Teaching science through computation. *Generations*, *13*, 15.

Appendix A
EV3 Robotics Challenges

Part 1 EV3 Robotics Challenges

Challenge 1: Open-Loop Movement.
With any experiment, there is an issue that all scientists and engineers run into - hysteresis. For example, a self-driving car can never assume that the same set of commands will always lead to the same results. Indeed, one of the first lessons for any team, from middle school robot teams to national science laboratory engineering teams, is to never assume the experiment will end the same way. Open loop solutions often suffer from hysteresis (an output change based on the history of the experiment) while closed loop solutions try to overcome this.

Rules. This challenge is to see how close a team can get to the far wall of the arena without touching the wall. The rules are as follows:

1. The robot must start touching the West wall.
2. The robot that gets closest to the East wall - without touching - is the winner. A robot that stops while touching the East wall will be penalized 8mm.
3. Each team will get three runs. The team with the smallest sum distance will be declared the winner.
4. You can use any robot design but no external sensors (e.g. ultrasonic or light) may be used.

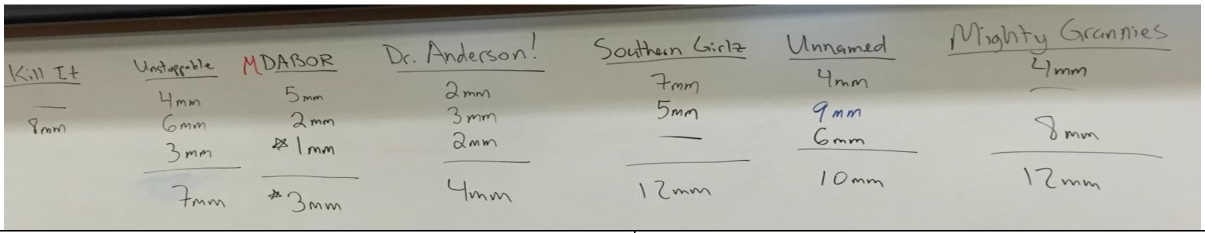
Results. The purpose of this challenge is to get a benchmark for the next challenge's results. Open loop systems will perform worse than closed loop systems and teams recorded their best performing distance to compare with the next challenge.

Challenge 2: Closed-Loop Movement.
Sensors are everywhere. And though they are a popular literary device for dystopian storylines, electronic sensors are just the method used to allow computers access to the real world. Without sensors you'd take the "smart" out of modern technology. Imagine a phone without a camera, microphone, speaker, or wi-fi (all sensors). In this challenge teams demonstrate the use of sensors as it emulates a version of some cutting-edge technology - automatic car breaking.

Rules. The Closed Loop challenge rules are exactly the same as the Open Loop challenge except for one important change to Rule #4:

1. The robot must start touching the West wall.
2. The robot that gets closest to the East wall - without touching - is the winner. A robot that stops while touching the East wall will be penalized 5mm.
3. Each team will get three runs. The team with the smallest sum distance, after the worst score is discarded, will be declared the winner.
4. You can use any robot design and ANY external sensors (e.g. ultrasonic or light).

Results. To get teams excited and into the "spirit" of the challenge, results were displayed real-time on the board (below) with teams attempting to beat the professor and scores from Challenge 1. Note that this was successful since all teams beat their previous scores with the use of sensors (and one team was able to beat the professor).



Kill It	Unstoppable	MDABOR	Dr. Anderson!	Southern Girlz	Unnamed	Mighty Grannies
—	4mm	5mm	2mm	7mm	4mm	4mm
8mm	6mm	2mm	3mm	5mm	9mm	—
	3mm	*1mm	2mm	—	6mm	8mm
	7mm	*3mm	4mm	12mm	10mm	12mm

Challenge 3: The Robotic Ocean Sweep.
Robots, coding, and machine learning are used everywhere and not just in self-driving cars. This challenge emphasized that it is actually not that hard to code a robot to autonomously cover some area such as in popular vacuum robots. For context, oil, and other spills are a constant worry in oceanography. In the Gulf of Mexico oil spill of 2010 over three million barrels of oil leaked out before the well was capped. One method of cleaning up these spills is to use_swimming robots. In this challenge teams design and code a robot to "clean up" the

Rules. After blocks are randomly scattered throughout the arena, teams must obey the following rules.
1. The robot can start anywhere in the world that you want.
2. The robot scores points by removing objects from the world's oceans. Green blocks are toxic garbage and worth 5 points while red blocks are normal trash and worth 1 point.
3. To count for your score, the block must be completely off the world map. The score for each round is the sum objects times their worth. Each round lasts at most 30 seconds or until the bot says "Stop!".
4. Since form factor is important with swarmbots, your score will also be divided by the maximum width of your bot.

world’s oceans.

5. Each team will get three runs. The team with the highest total score will be declared the winner.
6. You can use any robot design but make sure you include some kind of plow!

Results. Since teams were given total leeway on how to solve the problem of block (trash) collection, the solutions were varied in their presentations. Some teams used what they learned in the open and closed loop challenges to do an exhaustive sweep of the arena in order to get all blocks. Other teams were more hesitant and simply did a single-shot pass at cleaning. Finally, one team did minimal coding but invested their time into a mechanical solution with a large plow used to gather as many blocks as possible. Sample Python coding used to solve the challenge included below. Note the clever use of different motor powers to help blocks stay in the plow as they’re pushed off the map.

```
from ev3 import *
from time import sleep

#connect the motors
motorLeft = LargeMotor('outB')
motorRight = LargeMotor('outC')

#set the run time for each motor to 5 sec
motorLeft.time_sp = 5000
motorRight.time_sp = 5000

#set the power level for each motor
#Note Right has more power to the robot will turn left
motorLeft.duty_cycle_sp = 75
motorRight.duty_cycle_sp = 100

#run the motors
motorLeft.run_timed()
motorRight.run_timed()

#wait for the robot to stop
sleep(5)
```

Challenge 4: Line Follower. A line follower robot is able to follow some arbitrary track without any physical connection with the track except for the wheels. This method has great advantages over something like tracks or rails since the cost of laying down a new course is just the cost of paint. Line follower robots have many commercial applications such as autonomous cars on roads or forklifts in large factories. One of the main math concepts behind a popular line following algorithm is something called a PID controller. Though this controller can sound complex, the "I" stands for integral and the "D" stands derivative, we simplified the challenge to making a P controller ("P" stands for proportional).

Rules. This challenge was a double elimination tournament. Robots competed head-to-head on equal length tracks. The first robot to cross the finish line was declared the winner for that round. A couple rules for this challenge:

1. Any robot design can be used; however, you are only allowed to use one light sensor.
2. If your robot leaves the track, the judge will signal the team and they are allowed to place the robot back on the track from the point of its departure.

Results. The teachers learned that in order to keep the light sensor centered on the line, the sensor must read 50% at all times. If the sensor read greater or less than this value, the motors’ powers were corrected to force the robot back onto the line

$$\boxed{p_r = m_r s + b_r}$$
$$\boxed{p_l = m_l s + b_l}$$

where the teams were required to find correct constants so the robot functioned correctly. Teams quickly learned that incorrect values would cause the robot to oscillate wildly as it attempted to follow the line. Example code generated by one of the teams is included. Note how the team heuristically determined proportionality constants for the motor power functions so the resulting behavior would be more stable.

```
from ev3 import *
from nav import *

left = LargeMotor('outB')
right = LargeMotor('outC')

lsensor = ColorSensor('in3')
lsensor.mode = 'COL-REFLECT'

def rmotor(x):
    return x*0.58824 +20.09804

def lmotor(x):
    return x* (-0.58824) + 66.569

left.run_direct()
right.run_direct()

while True:
    print(lsensor.value())

    right.duty_cycle_sp= rmotor(lsensor.value())
    left.duty_cycle_sp= lmotor(lsensor.value())
```

Part 2

Robotic Sweep Python Code

```
from ev3 import *
from time import sleep

#connect the motors
motorLeft = LargeMotor('outB')
motorRight = LargeMotor('outC')

#set the run time for each motor to 5 sec
motorLeft.time_sp = 5000
motorRight.time_sp = 5000

#set the power level for each motor
#Note Right has more power to the robot will turn left
motorLeft.duty_cycle_sp = 75
motorRight.duty_cycle_sp = 100

#run the motors
motorLeft.run_timed()
motorRight.run_timed()

#wait for the robot to stop
sleep(5)
```

Part 3

Robotic Sweep READ Me Information

This program is designed for a wheeled robot that has two wheels, one connected to output ‘outB’ and the other connected to output ‘outC’. Both motors are set to run for the same amount of time. However the amount of power supplied to the right wheel is greater, resulting in a left-hand turn. The robot is allowed to roll to a stop.

Part 4

Line Follower Python Program Code

```
from ev3 import *
from nav import *

left = LargeMotor('outB')
right = LargeMotor('outC')

lsensor = ColorSensor('in3')
lsensor.mode = 'COL-REFLECT'

def rmotor(x):
   return x*0.58824 +20.09804

def lmotor(x):
   return x* (-0.58824) + 66.569

blue = 14
white = 65
midpoint = (blue + white)/2.0

left.run_direct()
right.run_direct()

while True:
   print(lsensor.value())

   right.duty_cycle_sp= rmotor(lsensor.value())
   left.duty_cycle_sp= lmotor(lsensor.value())
```

Part 5

Line Follower program for EV3 robot READ Me Information

This program is designed for a wheeled robot with motors connected to outputs ‘outB’ and ‘outC’. There is also a color sensor connected to input ‘in3’. The program implements a very basic line follower. The robot drives forward using the color sensor to detect the left edge of blue painter’s tape on a white floor. The basic rule is that if the sensor is detecting more blue, then the robot should steer to the left. If the sensor is detecting more white, then the robot should steer to the right. The parameters in the code would need to be adjusted depending on the colors of the floor and tape.

The code is included here with explanatory comments added.

```
#A basic line follower program

#import libraries
from ev3 import *
from nav import *

#connect motors
left = LargeMotor('outB')
right = LargeMotor('outC')

#connect color sensor
lsensor = ColorSensor('in3')
lsensor.mode = 'COL-REFLECT'

#Determine the power for the right wheel as a function of the color sensed
def rmotor(x):
   return x*0.58824 +20.09804

#Determine the power for the left wheel as a function of the color sensed
def lmotor(x):
   return x* (-0.58824) + 66.569

#initialize the colors for the floor and tape.
blue = 14
white = 65
midpoint = (blue + white)/2.0

#Turn on the motors
left.run_direct()
right.run_direct()
```

```
#Run forever
while True:
   print(lsensor.value())

   #Balance the power to each wheel to stay centered on the line.
   right.duty_cycle_sp= rmotor(lsensor.value())
   left.duty_cycle_sp= lmotor(lsensor.value())
```

Appendix B
The Zipliner's Delight Activity

Part 1

```
#Program for zip line
from ev3 import *
from math import *
from time import sleep
from pylab import plot, show


outfile = open('temp_data.txt', 'w')
print("Start")
Sound.speak("Go, go, go!")

sleep(2)

u = UltrasonicSensor('in2')
u.mode = 'US-DIST-CM'
u_list = []

dist = u.value()
while dist > 100:
   print(dist)
   u_list.append(dist)
   sleep(0.2)
   dist = u.value()

print(u_list, file = outfile)
Sound.speak("Well done!")
plot(u_list)
show()
```

Part 2

Function Zipline program for EV3 robot.

This program is designed for a robot that is built in one of two ways. Either the robot itself slides along a zip line toward a stationary target, or the robot is stationary and a target is designed to slide along the zip line toward the robot. In either case, an ultrasonic sensor is connected to input port 'in2' and is used to measure the distance between the robot and the target.

As the distance between the robot and the target decreases, the distances are repeatedly measured as recorded. At the end of the experiment, the data is saved to a file for further analysis. This is a modern reinterpretation of Galileo’s famous experiment rolling balls down an inclined plane to discover uniform acceleration due to gravity.

Here is the code with added explanatory comments.

```
#Program for zip line
#Import library functions
from ev3 import *
from math import *
from time import sleep
from pylab import plot, show


outfile = open('temp_data.txt', 'w') #Open the output file

#The robot indicates that it is ready to begin
print("Start")
Sound.speak("Go, go, go!")

sleep(2)                        #Wait 2 seconds for the experiment to start

#Initialize the ultrasonic sensor
u = UltrasonicSensor('in2')
u.mode = 'US-DIST-CM'
u_list = []                     #Create a list to store the data

dist = u.value()                #Record the first distance
while dist > 100:               #Record until the distance gets small
   print(dist)                  #Print the distance and record it in the lis
   u_list.append(dist)
   sleep(0.2)                   #Wait 0.2 second before making the next recording
   dist = u.value()             #Read the next distance

print(u_list, file = outfile)   #Record the distances in the output file
Sound.speak("Well done!")       #The robot indicates it is done
plot(u_list)                    #Create a graph of the results
show()
```

Appendix C
What's Your Function Activity

Part 1

```
#FunctionMachine.py
#Robot moves along a numberline with user entering domain value.
#Robot moves to the correct range value.

from ev3 import *
#from time import sleep

#Setup light sensors for reflection
frontlsensor = ColorSensor('in3')
frontlsensor.mode = 'COL-REFLECT'
rearlsensor = ColorSensor('in2')
rearlsensor.mode = 'COL-REFLECT'

#Calibrate light sensor
s = 0.2
black = 11
white = 90
midpoint = (black+white)/2.0

#Turn motors on
lmotor = LargeMotor('outB')
lmotor.reset()
rmotor = LargeMotor('outC')
rmotor.reset()

#define the linear function
def Adam(x):
   m = 2
   b = -3
   return m * x + b


while True:

   lmotor.run_direct()
   rmotor.run_direct()

   x = int(input("Please enter an Integer between -4 and 11 and place the robot on that input
number: "))
```

```
y = Adam(x)

distance_to_travel = y - x

if distance_to_travel > 0:      #robot will move forward
    while lmotor.position < 206 * distance_to_travel:    #moves until it gets to value
        Error = midpoint - frontlsensor.value()         #light sensor reading
        lmotor.duty_cycle_sp = 30 - s * Error           #power to left motor
        rmotor.duty_cycle_sp = 30 + s * Error           #power to right motor
elif distance_to_move < 0:    #robot will move backward
    while lmotor.position > 206 * distance_to_travel:    #moves until it gets to value
        BackError = midpoint - rearlsensor.value()       #light sensor reading
        lmotor.duty_cycle_sp = -30 + s * BackError       #power to left motor
        rmotor.duty_cycle_sp = -30 - s * BackError       #power to right motor

lmotor.reset()
rmotor.reset()

try_again = input("Do you want to test another input value? y/n:")
if try_again == "n":
    break
```

Part 2

Function Machine Python program for EV3 robot.

This program utilizes a robot that has two light sensors and the ability to drive back and forth along a line. One of the light sensors is located at the front of the robot and connected to input port ‘in3’ while the other sensor is located at the rear of the robot and connected to input port ‘in2’.

A number line is marked out on a sheet of paper taped to the floor. The distance between the values on the number line is calibrated to the rotation of the robot’s wheels. The exact values used in the code will depend on the spacing of the lines and the diameter of the wheels used on the robot. The robot will follow the edge of the sheet of paper by sensing the color difference between the paper and the floor. The values used in the code will depend on the colors and reflectivity of both the floor and the paper. The driving motors for the robot are connected to output ports ‘outB’ and ‘outC’.

The robot is placed on a number on the number line. The value of this number is read into the robot as input. The robot uses this value as input for a linear function, producing an output value. Finally the robot drives along the edge of the paper until the location of the output value is located on the number line.

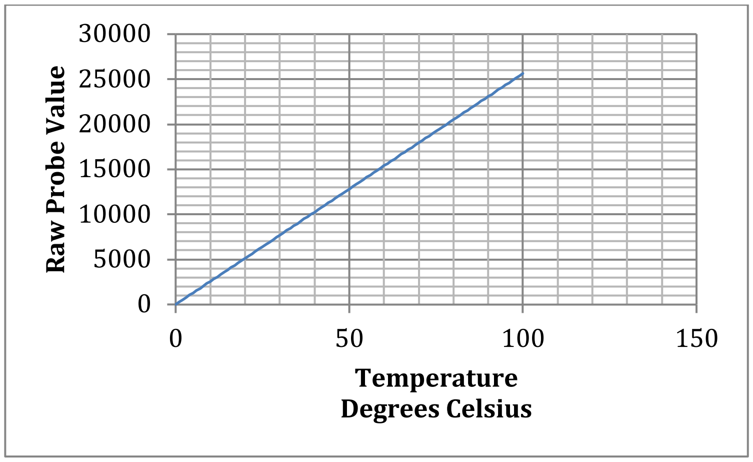
Appendix D

Part 1

Coding Connections at the Interface of Algebra I and Physical World Concepts Post-Test

Name: ___________________________________

1. **The following chart is from a datasheet describing the characteristics of the Lego temperature sensor. If the sensor is reading a raw value of 9000 what is the temperature in the room? Write your answer in Celsius.**



Temperature = ___________________C

2. **When you did the calculation in problem Problem 1, which variable is the independent variable and which is the dependent variable? Explain your reasoning.**

3. **The design of the temperature probe creates a relationship between the raw output value and temperature. In this relationship, which is the independent variable and which is the dependent variable? Explain your reasoning.**

4. **If the temperature is 60^o Celsius, what is the raw output value from the temperature sensor?**

Raw Value = ___________________

5. **Again using Problem 1, write an equation describing the input temperature / raw ouput value. Let the variable ‘x’ be the raw value and f(x) the temperature (in degrees Celsius).**

   f(x) = ______________________________

6. **Using your equation from Problem 5, write a Python function that returns the temperature in Fahrenheit given some raw value. The formula to convert degrees Celsius to degrees Fahrenheit is F = 9/5 C + 32.**

   ```
   def RawToFahr(r_value):

           tempC = _________________

           tempF =  _________________

           return tempF
   ```

7. **After running the following code on your EV3 robot, which direction (toward the left motor or the right motor) will the robot drift as it moves forward? Explain your answer.**

   ```
   steer = 10
   left = ev3dev.large_motor('outA')
   right = ev3dev.large_motor('outB')
   left.time_sp=5000
   right.time_sp=5000
   left.duty_cycle_sp= 50 + steer
   right.duty_cycle_sp= 50 - steer
   left.run_timed()
   right.run_timed()
   ```

8. What is output to the terminal window after running the following Python code?

```
x = 4
y = 2
z = 5
print(x**y)
```

9. A robot has wheels that are 5cm in diameter and an effective wheel base of 20cm. If the right wheel remains stationary, how many rotations must the left wheel perform to turn the robot 90^o to the right?

10. What is the output of the following Python code?

```
print(3 + 4 * 7 - 6 / 2)
```

11. Consider the following anecdotal story:

"Carl Friedrich Gauss was born in 1777 and later became one of the greatest mathematicians of all time. Once, as a youth in elementary school, his teacher made him add up all the numbers from 1 to 100 as punishment for misbehavior. He was able to solve the problem in seconds."

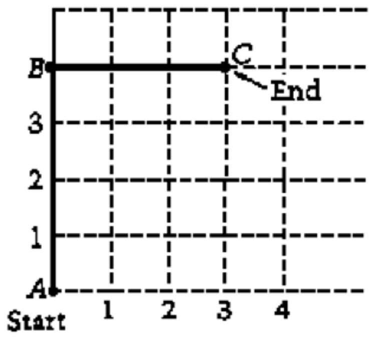
Finish the Python code below to find the same solution.

```
sum = ___________
for n in range(101):
     sum += ____
print(sum)
```

12. What is the output of the following Python code?

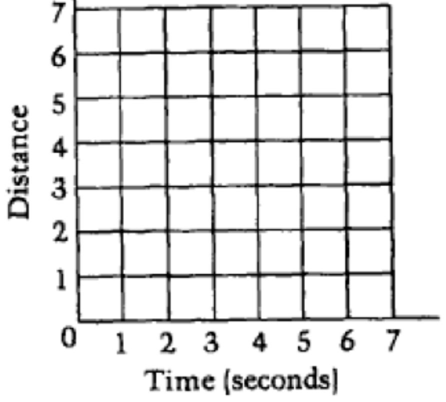
```
x = 23
if x < 12:
     print("Option 1")
else:
     print("Option 2")
```

13. The darkened segments in the figure below show the path of a robot that starts at point *A* and moves to point *B* and then on to point *C*. The robot moves at a constant rate of 1 unit per second. The robot's distance from a point is the <u>shortest</u> distance between the robot and the point.



What is the distance between point A and point C?

14. Referring back to Problem 13, sketch the graph of the distance of the robot from point *A* as a function of time on the 7-second interval.



15. **On your graph for Problem 14, sketch the distance of the robot from point *C* as a function of time on the 7-second interval.**

16. **Using your graph for Problems 14 and 15, determine between which two consecutive seconds is the robot equidistant from points *A* and *C.***

17. **Briefly outline an experiment Chloe could do at her school, using a robot with a zip-line, to explain the conversion of potential to kinetic energy. Indicate what materials Chloe will need, what measurements she will take, and what computations she will make**.

18. **Using the zip-line experiment described in the previous question, four teams in Chloe's class did the experiment you described. Each team got a different answer. Explain one reason why this might happen**.

19. An engineer made a model of a ship to help him think about how it works. He made sure that some characteristics of the ship were accurately represented, but he did not include all of the ship's characteristics in his model. Is it okay that he ignored some of the ship's characteristics?

A. It is okay, but only if he represented the characteristics that affect how the ship works, because models need to include the characteristics that are relevant to what is being studied.

B. It is okay, but only if he represented the characteristics that affected whether the model looks like the ship, because models should look like the things that they represent.

C. It is okay, but only if he represented the characteristics that people would be interested in knowing about, because models are only used to communicate information to others.

D. It is not okay that he ignored some of the ship's characteristics. A model should be like the object it is representing in every way possible.

20. An architect is designing a house and shows the plans to his coworker. The coworker likes the design but tells the architect that he now needs to make a three-dimensional (3-D) model of the house before the construction company can begin building it.

The architect says that even though the plans are just drawings on paper, they can be thought of as a model of the house. The coworker disagrees and says that a model of a house has to be three-dimensional.

As they discuss it further, they agree that the plans have all the information the construction company will need to build the house, including designs for building the floors and walls, but the architect and his coworker still disagree about whether the plans can be called a model.

Which of them is correct and why?

A. The architect is correct because he is the one who made the plans and therefore knows whether they can be considered a model.

B. The architect is correct because the plans represent the features of the house that are to be built.

C. The coworker is correct because a model needs to be three-dimensional.

D. Neither is correct because the house has not yet been built, and there cannot be a model of something that does not exist.

Items 17 & 18

TIMSS Assessment (1995). International Association for the Evaluation of Educational Achievement (IEA). Publisher: TIMSS & PIRLS International Study Center, Lynch School of Education, Boston College. *H29 a & b page 178*
https://isc.bc.edu/timss1995i/TIMSSPDF/CitemPhy.pdf

Items 19 & 20

American Association for the Advancement of Science (2020). Science Assessment.
http://assessment.aaas.org/pages/home

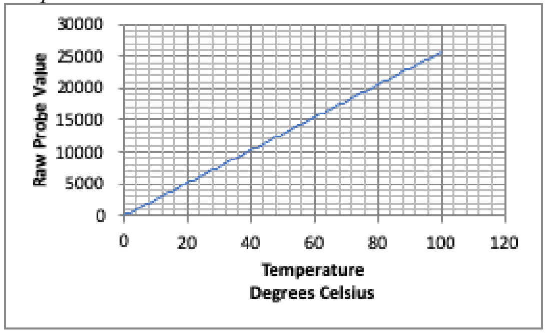
Part 2

(A) **Using your equation from Problem 5 (*included below*), write a Python function that returns the temperature in Fahrenheit given some raw value. The formula to convert degrees Celsius to degrees Fahrenheit is F = 9/5 C + 32.**

```
def RawToFahr(r_value):
        tempC = _________________
        tempF = _________________
        return tempF
```

Previous problems for reference.

The following chart is from a datasheet describing the characteristics of the Lego temperature sensor.



Write an equation describing the input temperature / raw output value. Let the variable 'x' be the raw value and f(x) the temperature (in degrees Celsius).

f(x) = ______________________________

Sample Correct Participant Response:

```
def RawToFahr(r_value):
        tempC = r_value/250
        tempF =  9/5 * tempC +32
        return tempF
```

(C) **After running the following code on your EV3 robot, which direction (toward the left motor or the right motor) will the robot drift as it moves forward? Explain your answer.**

```
steer = 10
left = ev3dev.large_motor('outA')
right =
ev3dev.large_motor('outB')
left.time_sp=5000
right.time_sp=5000
left.duty_cycle_sp= 50 + steer
right.duty_cycle_sp= 50 - steer
left.run_timed()
right.run_timed()
```

Sample Correct Participant Response:
It will veer right because the power of the left motor will be 60% and the power to the right will be 40%, making the left wheels turn faster.

(D) **Briefly outline an experiment Chloe could do at her school, using a robot with a zip-line, to explain the conversion of potential to kinetic energy. Indicate what materials Chloe will need, what measurements she will take, and what computations she will make.**
Sample Correct Participant Response:
An ultrasonic sensor will allow her to create a graph showing a function of time versus height. This will show the relationship that as time increases the height decreases at a faster rate. She can then calculate the acceleration of the object as a function of distance over time squared. Knowing the mass of the robot will allow her to calculate the potential and kinetic energies. PE=mgh & KE=1/2mv^2

(B) **Using your graph for Problems 14 and 15, determine between which two consecutive seconds is the robot equidistant from points *A* and *C*.**

Correct Response: Between 3 and 4 seconds

(E) **What is the output of the following Python code?**

```
x = 23
if x < 12:
     print("Option 1")
else:
     print("Option 2")
```

Correct Response: Option 2