K-8 Computational Thinking through Engineering (Fundamental)

Dr. Christine M. Cunningham, Pennsylvania State University

Dr. Christine M. Cunningham is a Professor of Practice in Education and Engineering at the Pennsylvania State University. She aims make engineering, science, and computational thinking education more equitable, especially for populations that are underserved and underrepresented in STEM. Christine is the founding director of Youth Engineering Solutions (YES), which develops equity-oriented, research-based, and field-tested curricula and professional learning resources for preK-8 youth and their educators. Her research focuses on articulating frameworks for precollege engineering education.

Dr. Darshita N. Shah, The Pennsylvania State University

Darshita (Dipa) Shah is the Curriculum Director for Youth Engineering Solutions at The Pennsylvania State University. Dipa has spent her career grappling with the challenge of how to best design motivating and engaging curriculum materials for students across the K-16 spectrum that can be practically implemented across the rich variety of our nation's educational contexts. Most recently, Dipa was the senior associate director with MIT's Teaching and Learning Lab where she facilitated workshops for campus educators on how to design curricular materials, implement evidence-informed pedagogies, and create welcoming classroom environments. Previously, Dipa was a manager for curriculum development for Engineering is Elementary and a lead science instructor with the Discovery Museum. Dipa received a bachelor's degree in chemical engineering from North Carolina State University and a doctorate in chemical and biological engineering from the University of Colorado at Boulder.

Ashwin Krishnan Mohan, Pennsylvania State University
Dr. Gregory John Kelly, Pennsylvania State University

Dr. Gregory Kelly is Senior Associate Dean for Research and Distinguished Professor at the College of Education at Pennsylvania State University. His research investigates classroom discourse, epistemology, and science and engineering learning. Greg is a member of the National Academy of Education and a fellow of the American Educational Research Association. He received the research awards including the Dr. John J. Gumperz Memorial Award for Distinguished Lifetime Scholarship from the American Educational Research Association and the Distinguished Contributions to Science Education through Research Award from National Association for Research in Science Teaching. Greg has a B.S. in physics from the State University of New York at Albany and a Ph.D. in Education from Cornell University.

K-8 Computational Thinking in and through Engineering (Fundamental)

Our society increasingly depends on computers and digital devices. Most of the technologies that we use daily—from toothbrushes to traffic signals to smartphones have involved computational tools in their conceptualization, manufacture, or operation. Increasingly, many engineered solutions rest heavily on computational thinking (CT). This raises the potential of using CT in educational settings. Carefully designed integration of epistemic practices and tools can foster opportunities for engineering education to be more authentic, powerful, and inviting.

Attempts to integrate computational thinking with engineering in educational settings are relatively recent. Initial research in this area suggests that the integration of CT with engineering, engineering design, and science in K-12 education has potential to develop children's engineering and design skills, improve science learning, and foster engagement. With this in mind, this theoretical paper examines the potential for the integration of engineering and computational thinking. It reviews existing literature and proposes a nascent framework for K-8 CT and engineering anchored in examples from middle school.

Introducing CT to K-12 Science and Engineering Education: Definitions and Perspectives

The current interest in CT and growth in empirical studies that consider integrated CT and STEM have led to an emergent research area. From this work, there are multiple definitions and frameworks for CT and how such thinking can be integrated into engineering education. CT has been defined and taken up in different ways across different domains and bodies of work within the literature. Wing (2006) framed CT as a set of fundamental skills that involve multiple levels of abstraction. Drawing from this characterization of CT as a set of processes that are applicable across domains (rather than as disciplinary practices for doing computer science) has been critical in the evolution of research on computational thinking. By situating CT in this broader domain, the value and use in engineering becomes clear. Broadly, CT can be used to refer to the thought processes involved in expressing solutions as computational steps or algorithms that can be carried out by a computer (Aho, 2011; Barr & Stephenson, 2011; Lee et al., 2014).

These definitions emphasize abstracting and representing solutions in particular ways that allow use by an information processing agent—the thinking processes themselves do not require computers but are often conceptualized with this in mind. Priami (2007) succinctly summarized the basic feature of CT: "abstraction of reality in such a way that the neglected details in the model make it executable by a machine" (p. 64). The field of CT research, which initially focused on completely analyzable problems and solutions, has evolved to consider real-world problems whose solutions are large, complex systems. Therefore, while computer science is concerned specifically with the study of "computers and algorithmic processes," CT encompasses a broader set of processes to apply computation to studying all areas of human endeavor (Aho, 2011). Over time, CT has come to be viewed as a form of literacy (diSessa, 2001; Jacob & Warschauer, 2018; Kafai & Proctor, 2022) that can be used as a set of practices to explore other domains (diSessa, 2001; Kafai & Proctor, 2022) and as a "third leg of science" that is a critical part of the inquiry process (Denning, 2019, Grover & Pea, 2013).

Scholars have proposed different sets of skills and practices and fundamentally different taxonomies for classification of practices and components of CT (Barr & Stephenson, 2011; Brennan & Resnick, 2012; Grover & Pea, 2018; K-12 Computer Science Framework Steering Committee, 2016; Shute et al., 2017, Weintrop et al., 2016). For instance, Weintrop et al. (2016) separated CT into four categories: data practices, modeling & simulation practices, computational problem-solving processes, and systems thinking practices. These models identify shared facets of CT such as troubleshooting or iterative refinement as universally important to CT, but differ in how they represent CT analytically, and what other practices are considered central to their models. The processes of abstraction, representation, modeling, simulation, and logic of algorithms situate CT in a set of broader inquiry practices. An expansive definition views CT as a problem-solving approach that involves breaking problems into parts, recognizing patterns, identifying a process of accomplishing a task, and generalizing patterns into rules (Barr & Stephenson, 2011; Grover & Pea, 2018; K-12 Computer Science Framework Steering Committee, 2016; Shute et al., 2017, Weintrop et al., 2016).

For the purpose of integrating engineering and CT in K-8 education, we drew from the varying definitions to focus on three central epistemic practices of CT for STEM integration. These practices tie together many of the key processes and activities entailed in uses of CT in engineering education. These practices are visualizing data; modeling processes and procedures; simulating events, functions, and mathematical relationships; and automating data collection, analyses, and representation. In what follows, we review empirical studies involving the various ways CT connects to engineering (and STEM more generally) in varying educational settings.

Uses of CT in Science and Engineering Education

Review of the Literature: Methodology

The review was conducted using the cross-section of the terms “Engineering education” or “Engineering design” or “Makerspace” along with the terms “Computation” or “Computational thinking” as well as with an independent search with the terms “Science Education” with “Computation” or “Computational thinking” The search was run on ERIC ProQuest and Google Scholar. Abstracts from ERIC ProQuest (N=755) and the first 20 pages on Google Scholar for both the searches (N=800) were read. The search also utilized snowball sampling to consider other articles of relevance. Abstracts were examined to understand if the study involved considerations of CT or computer programming within an in-school, out-of-school, or makerspace activity that could be an engineering or science design challenge. The final list of studies (N=20) that met our criteria for integrating CT within a design-based activity or problem in engineering or science, within any education setting, were read in full and became the core literature informing our work. Through the review of these studies, we identified additional research (through snowballing sampling) informing the framework presented subsequently.

Epistemic Practices and Tools of CT for Science and Engineering

Epistemic practices and tools are central to knowledge building. Epistemic practices can be discipline specific and concern the construction, communication, evaluation, and legitimation of knowledge claims (Kelly, 2008). Specific epistemic practices of engineering for education support student engagement and learning (Cunningham & Kelly, 2017). Through the engineering design process, students often draw from and apply epistemic tools which are physical, symbolic,

or discursive artifacts that facilitate the construction of knowledge (Kelly & Cunningham, 2019). Although there are epistemic practices that transcend specific disciplines (e.g., posing researchable questions), there are others that characterize particular, discipline-specific ways of engaging in the world and constructing knowledge. In CT, these practices include data collection, analysis, and representation; problem decomposition; abstraction; generalization; pattern recognition; and simulation (Lyon & Magana, 2021).

Tofel-Grehl, Searle, and Ball (2022) seek to map CT onto the practices associated with scientific reasoning as students in their study designed instruments to measure G-forces during a class field trip to an amusement park. While not explicitly framing any of the science and engineering practices in the study as an epistemic practice, students in the study engaged with 5 out of the 7 big ideas in CT (Wing, 2006) and the authors elaborate correlations between these ideas and science and engineering practices (NGSS Lead States, 2013) such as engaging in argumentation from evidence, and developing and using models. CT within the literature is framed as a set of practices to engage in problem-solving implying that, within K-12 settings, CT can serve as a disciplinary body of knowledge in its own right, and as a set of epistemic practices for problem-solving and meaning making in general.

This framing is echoed in student discourse and practices around CT and engineering design (Ardito et al., 2020; Tofel-Grehl, Searle, & Ball, 2022; Yang, Baek, & Swanson, 2020). Ardito et al. (2020) for instance found that students used CT as epistemic practices to problem-solve and make meaning of engineering challenges in robotics, often reflecting on this in their journals. Yang, Baek, & Swanson (2020), pulling from observations of student practice, relate components of CT as epistemic problem-solving practices, in an integrated framework they call the problem-solving process. Other studies also indicate the utility of framing CT as a set of epistemic practices through the development of curricula that leverage CT practices within an engineering design challenge (Bartholomew et al., 2018; Bartholomew & Zhang, 2019; Lilly et al., 2022; Love & Griess, 2020).

At a post-secondary level, Lyon & Magana (2021), starting from CT practices identified by Selby & Wollard (2013), integrate modeling into the context of a senior engineering course in food and pharmaceutical processing, with a high-level conjecture that modeling and simulation activities situated in real-world engineering contexts can promote CT. They found that nearly every CT practice in their conceptualization was used by students in the development of their computational model. Practices that are typically associated with the engineering design process, such as iterative improvements of designs, and productive failure, were found to be manifest in student practices as they engaged in computational modeling. Similar integration of engineering with CT was also reported by (Vieira et al., 2020) in a course on computation and programming for material scientists as part of a study investigating how students use different scaffolding strategies. Thus, across grade levels and subject domains, CT has been productively employed for the production of knowledge by engaging students in epistemic practices.

Integrating CT with Science and Engineering

A number of studies have sought ways to integrate CT into engineering and science. For young learners, CT has been shown to be effective in both in-school and out-of-school settings. Love & Griess (2020) examined elementary school students in an engineering design challenge involving

building enclosures for animals using electronic components connected and controlled via a device using a computer language (similar to Scratch). The lesson was designed around physical computing (Cápay & Klimová, 2019) which brings together CT and engineering design. Integrating a storyline from a children's book about an elementary school student aspiring to be an engineer, students learned not just standards-aligned practices from multiple content areas but also provided an opportunity to connect engineering with enhancing literacy skills. CT has also been used to drive student understanding of science content in fields such as biology (Arık & Topçu, 2021; Waterman et al., 2020) or climate science (Tucker-Raymond et al., 2019) either through working directly with computer programs (Tucker-Raymond et al., 2019; Waterman et al., 2020) or through 'unplugged' learning experiences that leverage computational thinking without the need for computing devices (Arık & Topçu, 2021; Hurt et al., 2023; Peel et al., 2022; Waterman et al., 2020). These approaches frame computational thinking as a form of practice and thinking that is integrable into disciplinary education in the sciences and engineering (Hurt et al., 2023). For instance, by defining computational thinking through algorithmic explanations, CT can be integrated into science education (Peel et al., 2022).

Informal centers and out-of-school (OS) opportunities can also offer spaces that are well-suited to engineering design-based work (Yang et al., 2021). At a science center, Ehsan, Rehmat, & Cardella (2021) analyzed the CT of 5-7-year-olds engaged in an engineering design challenge with their family. They connected engineering design actions to CT competencies from the literature and showed CT competencies and engineering design practices can empower each other when engaged in together. In the domain of physics, Yang, Baek, and Swanson (2020) used a project-based learning approach to integrate components of CT in eight lessons on airplane design to learn about physical forces in a class of 6th graders. The pre-post results of their study show that students' CT skills improved significantly after the course. In a follow-up study (Yang et al., 2021), similar gains are reported in design-based projects that integrate CT based activities into a science context (investigating life on Mars) and an engineering context (designing earthquake resistant bridges). Student thinking in both cases is scaffolded by a problem-solving process and connections are made to CT at each point in the process.

Increasingly, K-12 education has recognized the importance of integrating computing and inquiry in science or engineering in systematic ways (Peel et al., 2022; Tofel-Grehl et al., 2022; Yang et al., 2021), that seeks to move past the perception of CT as only being relevant within the domains of information technology (Arık & Topçu, 2021). The integration of CT in K-12 has the potential to improve learning and increase student engagement in STEM via "CT-embedded scientific inquiry" (Yang et al., 2021) and CT-embedded engineering. Tofel-Grehl et al. (2022) argued that this integration has not reached its full potential for two primary reasons. First, teachers often lack sufficient content knowledge to span the two disciplines of science and computing. Second, there are few models of systematic integration of science and CT that are available to guide teachers' classroom practice as the integration of CT in K-12 learning has rarely been studied (Sengupta et al., 2013). This conclusion can be extended to engineering as well, as teachers need models to demonstrate the potential of integrated engineering and CT to engage students in epistemic practices and knowledge construction.

Using CT to Expand Students' Identities and Engagement

Effective engineering and science programs engage students in authentic experiences with the knowledge, practices, and values of the disciplines. Such engagement needs to be attentive to the students' academic identities, cultural backgrounds, and interests in the subject matter. Recent works in CT call for expanding frameworks in CT to the "organizational, social, and cultural environment of computer systems" within the communities of the students (Tedre & Malmi, 2018). For example, Kafai & Proctor (2022) argued that efforts to make computer science, and CT more broadly, inclusive requires the development of robust frameworks that can attend both to students' identities and their respective communities. Engineering design affords students the opportunities to personalize designs and solutions that reflect their own sense-making, overriding the notion that there is only one homogeneous way of generating and arriving at a solution (Tofel-Grehl et al., 2022). By shifting the discourse to a student-centered, inquiry driven model, engineering-design activities have the opportunity to both integrate CT into STEM (Tofel-Grehl et al., 2022), and offers opportunities to democratize CT and CS by allowing for equitable participation by students from historically underrepresented identities.

Situating engineering in real-world contexts, building on students' experiences, and encouraging multiple, diverse solutions offers the potential to improve participation in engineering, and STEM fields more broadly. The current lack of diversity in STEM fields is derived from limited opportunities to "participate, develop interest, and have one's identities supported" (Kafai & Proctor, 2022, p.148). Students' opportunities to learn CT skills and practices as part of their curriculum are to date limited (Nager & Atkinson, 2016; Love & Griess, 2020). New models need to be developed that support science and STEM teachers to engage students across engineering and computational disciplines (Lilly et al., 2022; Vieira et al., 2020). Against this backdrop, engineering design has emerged as an effective approach for introducing computing to students (Peppler & Glosson, 2013). For example, Ardito et al. (2020) investigated how participation in a collaborative robotics program differed by gender, and how it shaped the development of CT in 6th grade students. The authors found that the students rarely viewed coding as a separate skill, divorced from the context of designing for the robotics challenge. The students also viewed team work as critical. Explicit reference to collaborative discursive practices and dialogues were central to engineering design challenges, and to CT within these settings (Ardito et al., 2020; Leonard et al., 2017). Productive discourse drawing from the discourses of the disciplines demonstrated how students can engage in the key practices of the respective disciplines. Additional research is needed to fully recognize the barriers to participation and ways to building positive affiliation and a sense of belonging among all students.

Emerging Themes and Implications for Curriculum Development

Our review for previous studies makes clear that CT and engineering should not theorized as being separate sets of practices and competences, but as mutually reinforcing each other both as practices for problem-solving (Yang et al., 2020, 2021) and for defining learning outcomes in our curricula. In this way, students use CT throughout the course of the engineering design cycle with a focus on problem-solving and meaning making (Ardito et al., 2020; Leonard et al., 2017). This body of evidence shows promise for the design of curricula that effectively integrates CT to drive engineering design, leading to learning outcomes in both sets of practices. In subsequent sections, we provide examples of such integration from an engineering and CT middle school

curriculum. We describe our approach to engineering and CT, provide examples from our Middle School curriculum as part of Youth Engineering Solutions (YES) and introduce our framework for CT and engineering.

Engineering and Computational Thinking: An Integrated Approach

Our interest in the relationships between engineering and CT stems from our work as curriculum developers. Our work in K-12 engineering education has shown that carefully designed curricular units that integrate engineering with science lead to better student outcomes in both engineering and science (Cunningham et al., 2020). Integration with multiple disciplines is challenging and does not always lead to learning outcomes (National Academy of Engineering and National Research Council, 2014). Recognizing the power and prevalence of computation in engineered solutions, we began to ask how we might develop CT ideas and skills in youth. We posit that, just as the application of science to engineering strengthens understandings, using CT to improve engineering solutions could result in meaningful, enhanced understandings of both disciplines.

For the past three years, the team at YES has examined the literature and worked closely with classroom teachers, engineers, and developers of computational tools to explore approaches to CT education. We asked: What do students need to know? How can we help students to understand the interconnections, possibilities, and pitfalls of CT? How could this work be structured in ways that could realistically be accomplished in classrooms across the country?

Our goal was to build students' and teachers' understanding that engineers use computational thinking, tools, and practices to solve problems. We are engaged in design-based research to develop curricular materials that integrate science and engineering. We aimed to integrate CT into our curricular offerings. This meant that CT ideas and skills would need to support or enhance the engineering units.

Our current framework grew from several cycles of development and testing. Presently, we are applying these to school-based, middle school engineering units. We have begun to consider how CT and the framework connect to elementary-level engineering and plan to eventually consider out-of-school engineering as well. We expect that, through this expanded application, the framework will continue to evolve. We describe our framework with an emerging set of principles for effective uses of CT in engineering. We demonstrate how these principles manifest themselves in curricular design. Before doing so, we provide a brief introduction to the YES Middle School curriculum.

YES Middle School

We have designed each YES Middle School (YES MS) unit to engage students in a seven-phase engineering design process (Figure 2**)** as they generate a solution to a problem. Engineering units and challenges are carefully designed so students use scientific ideas aligned with the NGSS. Six YES MS engineering units comprised of nine lessons each are currently under development.

In the YES MS Engineering Medicine Coolers, students design a cooler system for medication. They focus on designing for users who are most impacted by the problem of medication losing efficacy due to hot temperatures. The parts of the medicine cooler system must cool the air inside

the cooler and slow heat transfer into the cooler. Using their knowledge of thermal energy transfer, students brainstorm, plan, and create unique solutions to the problem (Anonymous, 2023a). The unit connects to physical science standards related to thermal energy transfer as well as the engineering design standards (see Table 1).

Table 1: Engineering Medicine Coolers Unit Connections to NGSS (YES, 2023a)

NGSS Performance Expectation		In this unit, students...
MS-PS1-6	Undertake a design project to construct, test, and modify a device that either releases or absorbs thermal energy by chemical processes.	investigate how dissolving different amounts of potassium chloride in water lowers the temperature of the mixture.
MS-PS3-3	Apply scientific principles to design, construct, and test a device that either minimizes or maximizes thermal energy transfer.	apply their knowledge of insulative materials to design a medicine cooler that minimizes thermal energy transfer.
MS-ETS1-1	Define the criteria and constraints of a design problem with sufficient precision to ensure a successful solution, taking into account relevant scientific principles and potential impacts on people and the natural environment that may limit possible solutions.	determine that their medicine coolers should meet the needs (criteria) of those who are most impacted by the problem of high temperatures being harmful to medications and utilize a cost constraint to ensure that their designs are accessible to all.
MS-ETS1-2	Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.	work with their teams to test and evaluate how well their designs meet the criteria and constraints of the problem. Based on this analysis, they iterate on their designs and consider which of the users' needs their design best meets.

Nine lessons structure the engineering unit. Students think about problem-solving approaches and compare their approaches to a seven-phase engineering design process that they will use throughout the unit. Students learn that hot temperatures damage medications. They think about who is most impacted by this problem. They ask questions about how coolers keep things cool and conduct investigations to understand how various materials and thickness of these materials slow heat transfer. The coolers the students engineer will have two components—insulation and the endothermic dissolution of potassium chloride. They learn more about potassium chloride as they experiment with solutions of different concentration. As students frame the problem and gather scientific data to inform their designs, they begin to consider the criteria and constraints they will weigh during cooler design. They plan and design a cooler and test it by manually collecting temperature data to see how well it meets the criteria of getting the system as cool as possible for as long as possible. After analyzing their data, teams redesign the cooler and compare their results to their original design. Finally, students display their coolers, consider which coolers meet users' needs, and reflect on their growth as engineers.

Two unit-specific, CT modules accompany each YES MS unit. In designing the YES units, we considered four broad categories of computational tools that engineers use as they progress through the engineering design process:

- data visualizations
- models of processes and procedures
- simulations of events, functions, and mathematical relationships; and
- automation of data collection, analyses, and representations.

Engineers engage in thinking computationally when using these tools in their approach to design. We describe next how these tools are employed during YES engineering activities.

Engineering Medicines Coolers is accompanied by two CT modules, one focused on visualizing heatwaves and a second focused on designing an alarm system. We illustrate the CT in each.

Visualizing Heatwaves: In this module, which has three lessons, students explore how using a computer to visualize data can help them further frame the problem of hot temperatures damaging medication. Working to identify which regions and people might be most impacted by heatwaves, they observe that the same data can be used to produce different data visualizations. This motivates students to better understand the algorithms that are being used to generate the visualizations. Aided by a MATLAB live script, students modify the algorithm to create their own data visualization, highlighting how peoples' experiences and beliefs can and do influence the instructions they give to a computer (YES, 2023b).

From the module it is expected that students learn that:

- Some people can be impacted by a problem more than others.
- Engineers can use computational tools to frame a problem.
- Scientists may consider one or more characteristics when defining "heatwave."
- A data visualization can be useful for summarizing a large dataset.
- A computer has to be provided with an algorithm (a set of instructions) written by a human in order to produce a data visualization.
- A human's experiences and beliefs may influence algorithms that they write. (YES, 2023b, p. 7)

As students work toward these educational goals, they learn ways that CT informs the details of their own technology and some of the broader ramifications of CT and engineering in society.

Medicine Cooler Alarm: In this two-lesson module, students think about how they might alert users that a medicine cooler has warmed to room temperature. Thinking about their experience of manually collecting temperature data when testing their coolers, they consider how a computer could be leveraged to design a practical solution to data collection. They learn to write an algorithm for a micro:bit to monitor the temperature of the medicine cooler and alert users when it crosses a threshold. Students also reflect on how a temperature sensor could have been useful during the development of the medicine cooler (in the Investigate and Test phases).

The goals for the module are that students will learn:

- micro:bits have a built-in sensor to measure the temperature.

- Computers output information based on the algorithm, or instructions we give them.
- micro:bits can output information by displaying it on an LED screen or by playing a sound.
- A loop can be used to do a task repeatedly.
- Conditional statements can be used in an algorithm to prompt the computer to evaluate if something is true. (YES, 2023c, p. 7)

As they work toward these goals, students learn that CT tools can reduce the labor of data collection, representation, and analyses and produce powerful ways of applying data to solve problems.

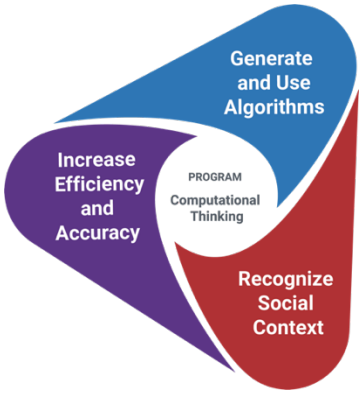
YES CT modules are optional. Teachers can choose to do one, both, or neither. The curricular resources for each module include a teacher guide, student engineering notebook, slides for classroom projection, and print hand-outs that students might use. As we created the YES CT curricula, we considered the literature and derived a set of principles for effective uses of CT in engineering.

YES Computational Thinking Framework

YES defines CT as *an approach to interpreting, creating, or refining algorithms to solve problems*. Based on this definition, we identified three key principles for effective use of CT in middle school engineering. YES MS engineering and CT curricula emphasize these principles as students interact with the four categories of computational tools outlined above. These principles, summarized in Figure 1 and listed below, describe why it is important for engineers to think computationally and for students to learn to think in this way:

1. Engineers use computational tools to solve engineering problems more efficiently and accurately.
2. Engineers must interact with computational tools in a knowledgeable way—understanding the algorithm, or the steps, the computer is executing to carry out the task and knowing how to generate new algorithms.
3. Engineers must recognize the human and socially embedded nature of computational tools and the biases that may exist in the tool itself and/or in the application and interpretation of the tool and its output.

Figure 1: Youth Engineering Solutions Principles for Computational Thinking



We turn now to describe each of these principles briefly and illustrate how they play out in our units and lessons.

Use CT to Solve Engineering or Science Problems More Efficiently and Accurately

People, including students, are often motivated to learn new ideas and skills when they understand why these are important or how they can help. CT is no exception. The pragmatic utility of CT for engineering is an underemphasized aspect of computational literacies. To nurture students' interest in CT, we aim to develop their understanding of the power of using computational tools by having them experience how they can help them better or more efficiently solve a problem. By automating processes, human effort and error can be reduced, resulting in more accurate and equitable outcomes.

In both CT modules supporting the Engineering Medicine Coolers unit, students experience how CT can produce tools that help solve tasks more accurately and efficiently. Initially students are asked to gather, collect, or analyze data. In the Visualizing Heatwaves module, students first manually identify how many heatwave days are in a small dataset according to a set definition. After they laboriously seek patterns in the raw data, not only do they have a firm understanding of how a computational algorithm may work, but they also understand how using a computer to automate the analysis and visualization of data saves human labor and time!

Similarly, the Medicine Cooler Alarms module introduces students to additional benefits of automation. In the engineering unit, students read and record data from a digital thermometer every minute. In the CT module, they build on that experience, identifying the algorithm they used and identifying possible mistakes they may have made completing this relatively mundane task. They then explore how their algorithm can be translated to a set of instructions that can tell a machine to do the same thing. A computer can repetitively and more accurately collect and log data while they are freed up to complete other engineering tasks. Having students experience how a computer can help them complete necessary tasks and the advantages of doing so can help students understand why developing their CT skills to instruct machines and be informed users of computational tools is beneficial.

To solve problems more accurately and efficiently, engineers use computational thinking and tools to solve problems throughout the many phases of their work. CT drives much more than analysis of data. CT might be used to frame or visualize a problem, model solutions, collect data more accurately or efficiently, analyze data, or generate improved solutions, for example. Students should understand that CT can be leveraged through the course of the engineering design cycle. Our YES middle school engineering design process involves seven phases (see Figure 2). Our CT modules demonstrate how CT practices can scaffold design thinking in various phases.

Figure 2: YES Middle School Engineering Design Process

For example, the Visualizing Heatwaves modules helps students more deeply frame the problem of heatwaves. In this module, students expand their understanding of who is most impacted by heatwaves. They explore a data visualization of heatwave days across the United States created using temperature data collected by weather stations in 2021. They identify where the most heatwave days occurred, and whether their region suffered from any.

Another CT module, Medicine Cooler Alarms, connects with both the "Iterate" and "Test" phases of the process. In this module, students learn how a digital tool (in this case, a temperature sensor on a micro:bit) can help them monitor the temperature inside of their medicine cooler. They also think about how such a computational tool could improve their system by alerting users when the temperature rises above a certain value. They are challenged to create an addition to their cooler system that will serve this purpose.

Understand, Employ, Interpret, and Generate Algorithms and CT Tools

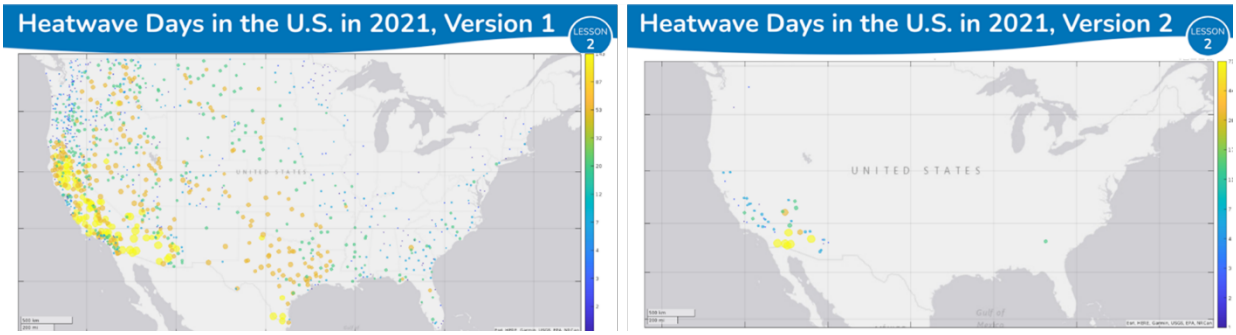
Creating and generating artifacts such as computer programs or algorithms is an important learning outcome, especially from within the disciplinary bounds of computer science. However, although engineers do sometimes solve a problem by articulating an algorithm within a programming language, their work is broader. Generating algorithms (physical or digital) does not reflect the multitude of ways in which engineers engage with and use CT tools. Engineers' work often rests on understanding, interpreting, and/or questioning the reasoning and logic underlying a computational tool or algorithm developed by someone else. These CT skills help them to better identify the affordance and limitations of the tools and select the appropriate one. Thus, we aim to help students understand or interpret algorithms and CT tools as well as generate them. We focus on developing students' abilities related to the concepts, routines, symbols of CT that are used in algorithmic thinking and expression. We want to educate informed users as well as generators of algorithms.

This important distinction between knowing how to think computationally and knowing how to be an intelligent consumer or interpreter of computational tools and artifacts underlies some CT modules in the YES MS curriculum. For example, in the Medicine Cooler Alarms module, students think about devices in their lives that monitor data and learn that they can do so because

they have sensors. They explore a micro:bit that is outputting temperature data and try to make sense of what the code is telling the machine to do. Working from a base of existing code, they draw from their own experiences of manually collecting data to modify the code to better suit their needs. They connect the repetitive task of reading a thermometer every minute to the programming concept of a loop. Later, students write out the logic that they would like their medicine cooler alarms to follow, "If the temperature gets above _____, make a sound." Through these written statements, students are introduced to the computational thinking concept of a conditional. The goal of the activity is not to teach students to program micro:bits, but rather to help them build their understanding of basic coding concepts so that they can understand existing code and modify it to help them accomplish their task.

In the Visualizing Heatwaves module, students learn that they cannot take computational tools at face value. They compare data visualizations generated from the same heatwave data and think about why they differ. Using a small subset of the data, student teams manually count the number of heatwave days as defined by a given definition. Teams have different definitions to help them recognize that applying different heatwave definitions to the same data produces different results. Teams compare the processes they used to count heatwave days to deepen their understanding of the algorithm used to generate the data visualizations they inspected earlier (see Figure 3). Using a MATLAB live script, students are able to manipulate the algorithm used to generate the data visualization and explore how changing the heatwave definition used impacts the output. In this module, students do not generate the algorithm, but rather come to realize the importance of understanding the algorithm that produces a computational product and how small adjustments to that algorithm can dramatically change the product.

Figure 3: MATLAB generate display of days reaching specified temperature



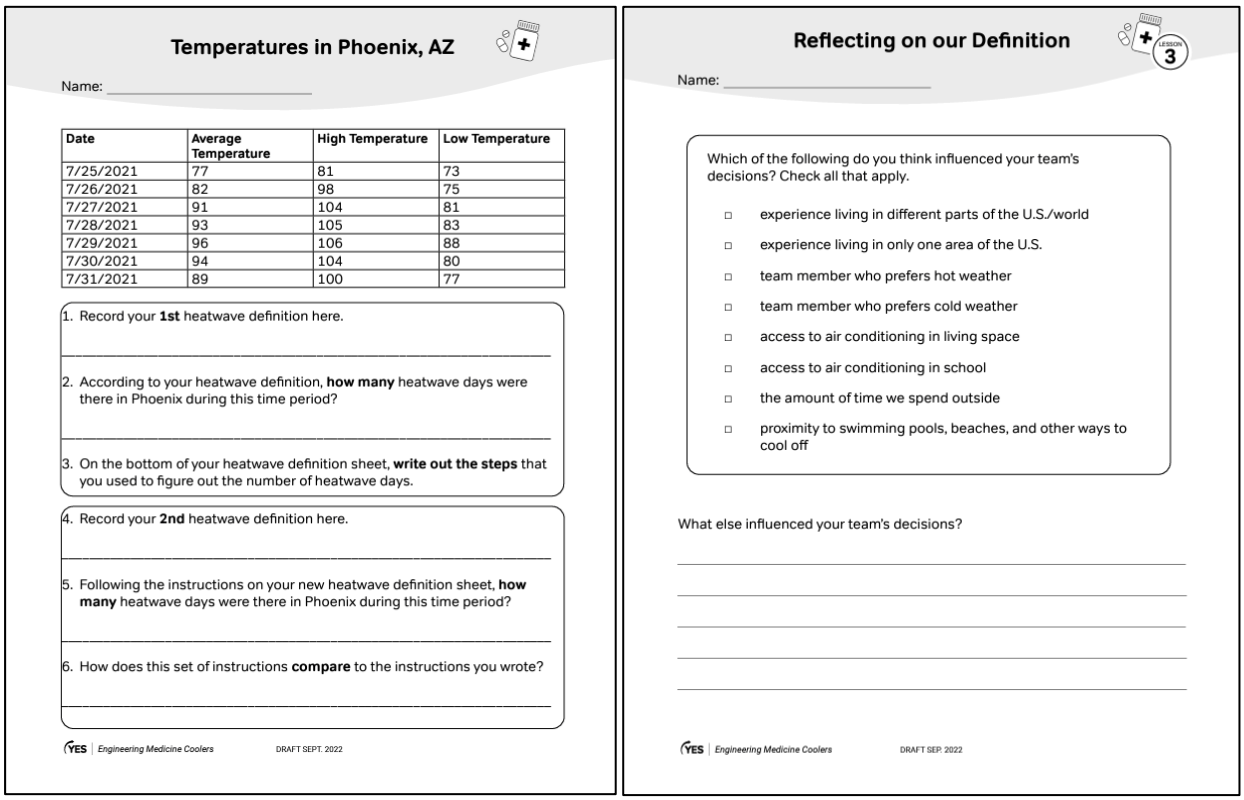
<u>Recognize the Human and Socially Embedded Nature of CT and Consider Biases that Might Exist</u>

CT and its products and solutions are created by humans for specific ends. In keeping with YES's overall approach to socially engaged engineering (Cunningham & Kelly, 2022), we aim to develop students' understanding that CT solutions are created by humans whose decisions can introduce bias to CT models and tools, that one algorithm may not work for all populations, and that CT impacts and is impacted by society.

In the Visualizing Heatwaves CT module, students are asked to think about how human biases can impact decisions that affect algorithms and outcomes. As described above, students are presented with varying definitions of a heatwave and learn these exist because scientists study

heatwaves for different reasons. They explore how changing the definition used in the algorithm results in different data visualizations impacting how they identify where the most heatwave days occur. The final lesson of the module has students consider the question, "How does a human's perspective influence a data visualization?" Teams write their own definition of a heatwave and then reflect on how they think their personal experiences may have influenced their definition (see Figure 4). After teams share their definitions and resultant data visualizations, students reflect upon how human perspectives can influence the algorithms used to produce data visualizations and ponder how adjusting the definition could support certain points of view. Students are asked, "As engineers who use data visualizations to frame a problem, what do we need to be cautious about/mindful of?" (YES, 2023b). Fostering such conversations among students can help students to become more informed and critical producers and consumers of technologies.

Figure 4: Prompt in Visualizing Heatwaves CT Module



Temperatures in Phoenix, AZ

Name: _____

Date	Average Temperature	High Temperature	Low Temperature
7/25/2021	77	81	73
7/26/2021	82	98	75
7/27/2021	91	104	81
7/28/2021	93	105	83
7/29/2021	96	106	88
7/30/2021	94	104	80
7/31/2021	89	100	77

1. Record your **1st** heatwave definition here.

2. According to your heatwave definition, **how many** heatwave days were there in Phoenix during this time period?

3. On the bottom of your heatwave definition sheet, **write out the steps** that you used to figure out the number of heatwave days.

4. Record your **2nd** heatwave definition here.

5. Following the instructions on your new heatwave definition sheet, **how many** heatwave days were there in Phoenix during this time period?

6. How does this set of instructions **compare** to the instructions you wrote?

YES | Engineering Medicine Coolers DRAFT SEPT. 2022

Reflecting on our Definition

LESSON 3

Name: _____

Which of the following do you think influenced your team's decisions? Check all that apply.

- ☐ experience living in different parts of the U.S./world
- ☐ experience living in only one area of the U.S.
- ☐ team member who prefers hot weather
- ☐ team member who prefers cold weather
- ☐ access to air conditioning in living space
- ☐ access to air conditioning in school
- ☐ the amount of time we spend outside
- ☐ proximity to swimming pools, beaches, and other ways to cool off

What else influenced your team's decisions?

YES | Engineering Medicine Coolers DRAFT SEP. 2022

Limitations and Future Work

Our work to date has focused on reviewing the literature to understand existing structures and approaches to CT with the aim of generating functional principles that can inform curriculum design. We have engaged in conceptualization and early classroom testing with teachers and modified our principles, approach, and curricula based on feedback. However, to date, we have worked with about a dozen or so educators and a few units. We need to work with more teachers to pressure-test the ideas. We also need to apply the principles to additional units to test their generalizability across engineering activities and challenges. Future work will involve working

with additional classrooms and units and collecting more robust sets of qualitative and quantitative data to analyze. Such research will illuminate how students can learn computational thinking in the context of engineering and provide evaluative information about students interact with the curricular units. Such valuable data will inform subsequent iterations.

Conclusion

CT in education is an emerging field with new developments, theories, and applications. We have demonstrated that CT can be applied and employed in engineering to develop students' tools for thinking and improve the processes of engineering analysis and design. Drawing from the varied, multidisciplinary literature, we examined those CT practices that best provided tools for thinking for middle school students. Our examples of the CT modules in the YES MS engineering curriculum provide avenues for examining the potential of the integration of CT and engineering to improve students' understanding of both knowledge domains. In the application of CT in YES, students move beyond just superficial application of CT artifacts to examine and understand the logic behind the computational tools. Four components and CT practices were foregrounded (visualization, models, simulation, automation) to foster students' engagement in engineering. These tools are made available in the engineering curriculum to build students' understanding and develop awareness of the usefulness of CT in situated, engineering problems.

We presented three principles for effective use of CT in engineering and illustrated these principles in two middle school engineering and CT curriculum units. These principles integrate CT and engineering in ways that improves students' reasoning in specific situated problems. By applying these principles to curriculum design, we help students understand the engineering challenge more deeply, create more efficient technologies, and understand how computational tools can enhance engineering work. The principles coalesce usefulness, applications, and understandings of CT in engineering. By considering engineering problems in context, the curriculum units show the human and socially embedded nature of CT and recognize that choices of tools, parameters, and criteria may enter bias into uses of engineering in society.

Acknowledgements

The ideas and thinking in the paper have been shaped through close collaboration with MathWorks staff including Mary Dzaugis and Emma Smith Zbarsky. A generous grant from MathWorks also supports this work.

References

Aho, A. (2011). Ubiquity symposium: Computation and computational thinking. *Ubiquity*, *2011* (January), 1–8. https://doi.org/10.1145/1922681.1922682

Ardito, G., Czerkawski, B., & Scollins, L. (2020). Learning computational thinking together: Effects of gender differences in collaborative middle school robotics program. *TechTrends*, *64*(3), 373–387. https://doi.org/10.1007/s11528-019-00461-8

Arık, M., & Topçu, M. S. (2022). Computational thinking integration into science classrooms: Example of digestive system. *Journal of Science Education and Technology*, 31(1), 99–115. https://doi.org/10.1007/s10956-021-09934-z

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

Bartholomew, S. R., Zhang, L., & Weitlauf, J. (2018). Engineering design and coding through quadcopters. *Technology and Engineering Teacher*, *78*(1), 14–21.

Bartholomew, S. R., & Zhang, L. (2019). Socially relevant contexts. *Technology and Engineering Teacher*, *79*(1), 13–19.

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada* (Vol. 1, p. 25).

Cápay, M., & Klimová, N. (2019, April). Engage your students via Physical Computing!. In *2019 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1216-1223). IEEE.

Cunningham, C. M., Lachapelle, C. P., Brennan, R. T., Kelly, G. J., San Antonio Tunis, C., & Gentry, C. A. (2020). The impact of engineering curriculum design principles on elementary students' engineering and science learning. *Journal of Research in Science Teaching. 57*, 423–453. DOI: 10.1002/tea.21601

Cunningham, C. M., & Kelly, G. J. (2017). Epistemic practices of engineering for education. *Science Education*, *101*(3), 486–505.

Cunningham, C. M., & Kelly, G. J. (2022). A Model for Equity-Oriented PreK-12 Engineering. *Journal of Pre-College Engineering Education Research (J-PEER), 12*(2), Article 3. https://doi.org/10.7771/2157-9288.1375

Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM*, *60*(6), 33–39. https://doi.org/10.1145/2998438

diSessa, A. A. (2001). *Changing minds: Computers, learning, and literacy*. Cambridge: MIT Press.

Ehsan, H., Rehmat, A. P., & Cardella, M. E. (2021). Computational thinking embedded in engineering design: Capturing computational thinking of children in an informal engineering design activity. *International Journal of Technology and Design Education*, *31*(3), 441–464. https://doi.org/10.1007/s10798-020-09562-5

Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

Grover, S., & Pea, R. (2018). Computational thinking: A competency whose time has come. *Computer science education: Perspectives on teaching and learning in school*, *19*, 1257–1258.

Jacob, S. R., & Warschauer, M. (2018). Computational thinking and literacy. *Journal of Computer Science Integration*, *1*(1). https://doi.org/10.26716/jcsi.2018.01.1.1

K-12 Computer Science Framework Steering Committee. (2016). *K-12 computer science framework*. ACM.

Kafai, Y. B., & Proctor, C. (2022). A revaluation of computational thinking in K–12 education: Moving toward computational literacies. *Educational Researcher*, 51(2), 146–151. https://doi.org/10.3102/0013189X211057904

Kelly, G. (2008). Inquiry, activity and epistemic practice. In R.A. Duschl, & R.E. Grandy (Eds.), *Teaching scientific inquiry* (pp. 99–117). Brill.

Kelly, G. J., & Cunningham, C. M. (2019). Epistemic tools in engineering design for K-12 education. *Science Education*, *103*(4), 1080–1111.

Lee, I. A., Martin, F., & Apone, K. (2014). Integrating computational thinking across K-8 curriculum. *ACM Inroads*. *5*(4), 72–75.

Leonard, J., Barnes-Johnson, J., Mitchell, M., Unertl, A., Stubbe, C. R., & Ingraham, L. (2017). Developing teachers' computational thinking beliefs and engineering practices through game design and robotics. In E. Galindo & J. Newton (Eds.), *Proceedings of the 39th annual meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education* (pp. 1289–1296). IN: Hoosier Association of Mathematics Teacher Educators.

Lilly, S., McAlister, A. M., Fick, S. J., Chiu, J. L., & McElhaney, K. W. (2022). Elementary teachers' verbal supports of science and engineering practices in an NGSS-aligned science, engineering, and computational thinking unit. *Journal of Research in Science Teaching*, *59*(6), 1035–1064. https://doi.org/10.1002/tea.21751

Love, T. S., & Griess, C. J. (2020). Engineering encounters. *Science and Children*, *58*(2), 51–57.

Lyon, J. A., & Magana, A. J. (2021). The use of engineering model-building activities to elicit computational thinking: A design-based research study. *Journal of Engineering Education*, *110*(1), 184–206. https://doi.org/10.1002/jee.20372

Nager, A., & Atkinson, R.D. (2016). *The case for improving U.S. computer science education*. Available at https://ssrn.com/abstract=3066335 or http://dx.doi.org/10.2139/ssrn.3066335

National Academy of Engineering and National Research Council. 2014. *STEM Integration in K-12 Education: Status, Prospects, and an Agenda for Research*. Washington, DC: The National Academies Press. https://doi.org/10.17226/18612.

National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking*. National Academies Press.

NGSS Lead States. (2013). *Next Generation Science Standards: For states, by states*. Washington, D.C.: The National Academies Press

Peel, A., Sadler, T. D., & Friedrichsen, P. (2022). Algorithmic explanations: An unplugged instructional approach to integrate science and computational thinking. *Journal of Science Education and Technology*, 31(4), 428–441. https://doi.org/10.1007/s10956-022-09965-0

Peppler, K., & Glosson, D. (2013). Stitching circuits: Learning about circuitry through e-textile materials. *Journal of Science Education and Technology*, *22*(5), 751–763.

Priami, C. (2007). Computational thinking in biology. In Priami, C. (Eds.), *Transactions on Computational Systems Biology VIII*. Lecture Notes in Computer Science, 4780. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-76639-1_4

Youth Engineering Solutions [YES]. (2023a). Engineering Medicine Coolers. Youth Engineering Solutions.

Youth Engineering Solutions [YES]. (2023b). Visualizing Heatwaves. Youth Engineering Solutions.

Youth Engineering Solutions [YES]. (2023c). Medicine Cooler Alarm. Youth Engineering Solutions.

Selby, C., & Woollard, J. (2013). *Computational thinking: the developing definition.* University of Southampton (E-prints).

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, *18*(2), 351–380. https://doi.org/10.1007/s10639-012-9240-x

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

Tedre, M., & Malmi, L. (2018). Changing aims of computing education: A historical survey. *Computer Science Education*, *28*(2), 158–186.

Tofel-Grehl, C., Searle, K. A., & Ball, D. (2022). Thinking thru making: Mapping computational thinking practices onto scientific reasoning. *Journal of Science Education and Technology*. https://doi.org/10.1007/s10956-022-09989-6

Tucker-Raymond, E., Puttick, G., Cassidy, M., Harteveld, C., & Troiano, G. M. (2019). "I Broke Your Game!": Critique among middle schoolers designing computer games about climate change. *International Journal of STEM Education*, 6(1). https://doi.org/10.1186/s40594-019-0194-z

Vieira, C., Magana, A. J., Roy, A., & Falk, M. (2020). Providing students with agency to self-scaffold in a computational science and engineering course. *Journal of Computing in Higher Education*. https://doi.org/10.1007/s12528-020-09267-7

Waterman, K. P., Goldsmith, L., & Pasquale, M. (2020). Integrating computational thinking into elementary science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning. *Journal of Science Education and Technology*, 29(1), 53–64. https://doi.org/10.1007/s10956-019-09801-y

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. https://doi.org/10.1007/s10956-015-9581-5

Wing, J. (2006). Computational thinking: What and why. *Communications of the ACM*, *49*(3), 33–35. https://doi.org/10.1109/MED.2008.4602144

Yang, D., Baek, Y., & Swanson, S. (2020). Developing computational thinking through project-based airplane design activities. *Proceedings - Frontiers in Education Conference, FIE*, *2020-October*. https://doi.org/10.1109/FIE44824.2020.9274021

Yang, D., Baek, Y., Ching, Y.-H., Swanson, S., Chittoori, B., & Wang, S. (2021). Infusing computational thinking in an integrated STEM curriculum: User reactions and lessons learned. *European Journal of STEM Education*, *6*(1), 04. https://doi.org/10.20897/ejsteme/9560