Book Review

Neumann, M.D., Dion, L., & Snapp, R. (2021). *Teaching computational thinking: An integrative approach for middle and high school learning*. The MIT Press. ISBN: 9780262045056 (paperback), $45.00, 200 pages; ISBN: 9780262366144 (electronic), open-access. https://doi.org/10.7551/mitpress/11209.001.0001

Introducing students to computer science (CS) and programming, primarily through the development of coding skills, has emerged as a top priority in policy and practice in recent years. The impact on Technology and Engineering Education is clear: The ability to use, create, and understand computer programs is essential for success in subjects such as automation and robotics, digital and integrated manufacturing, media production, and more. However, the conversation surrounding the implementation of these 21st-century skills has moved beyond the ability to use computers or create code toward the development of a broader set of *computational thinking skills* that support critical thinking on- and offline. Maureen D. Neumann, Lisa Dion, and Robert Snapp (2021) define computational thinking skills as "a set of mental and cognitive skills that are applied to the problem-solving process to help individuals discover and apply different strategies and algorithmic solutions to challenging and complex problems" (p. 1). This book offers educators a hands-on, evidenced-based approach to introducing middle and high school students to the application of computational thinking skills using various integrative approaches to learning that supports this premise. Research-based suggestions for activities are presented using traditional mechanisms of CS (computers with programming software) and through "unplugged" (offline) methods – often in parallel. In each activity, emphasis is placed on making connections between real-life situations and engaging students in rich tasks while addressing the skills needed for programming, working with data, and following a methodology of inquiry and action.

Neumann et al. (2021) state that "computer programming, computer science, and computational thinking are often intertwined in the academic literature, but they are not equivalent concepts" (p. 2). Instead, they suggest that while "we must be wary of learning to code for coding's sake; simple coding does not require computational thinking" (p. 11). The authors continue by suggesting that the following skills are essential elements of computational thinking: abstraction, sequencing, loops, events, modularizing, conditionals,

Williams, J. (2022). Review of the book *Teaching computational thinking: An integrative approach for middle and high school learning*, by Neumann, M. D., Dion, L., & Snapp, R. *Journal of Technology Education*, *33*(2), 43-47. https://doi.org/10.21061/jte.v33i2.a.4

algorithmic processes, data usage, scalability, transfer of skills to new situations, and reading, understanding, and updating previously written code (pp. 9-10). Throughout the text, the authors provide context for these essential elements and offer suggestions and strategies for practitioners interested in implementing engaging and creative computational thinking skill development activities in their curriculum.

Summary of Teaching and Learning Activities

Chapter 2: Creating Algorithmic Art focuses on two key aspects of computational thinking: The development of algorithms and abstraction. Algorithms "are a step-by-step set of instructions that can be applied more generally—not just one specific computational problem that you are working on at the moment" (Neumann et al., 2021, p. 10). Abstraction, in essence, is the simplification of a problem into its basic parts so that it can be understood more clearly and applied to other contexts. An "unplugged" version of this algorithm development activity is presented that does not require computers, instead opting for creating drawings using grid paper and colored markers. However, the outcome of the paper-based activity closely resembles the intent of the computer-based approach, achieving similar educational outcomes. The authors offer an example of engaging students in computational thinking skills for each subsequent chapter in this way – focusing on the deeper concept of the skill applied in context rather than the use of technology as a means of accomplishing a task. From the outset of activities designed to support computational thinking skill development, opportunities are present for cross-disciplinary integration. The book touts teaching with an *integrative approach* in its title, and this is clearly the case immediately and throughout the following chapters.

Chapter 3: Applying Graph Theory to Analyze Literature and Social Networks offers an opportunity for liberal arts integration with traditional STEM-based concepts such as using graphing to visualize relationships and patterns. The activity focuses on developing the skills of abstraction, data usage, sequencing, and conditionals. An example is provided from the book *Harry Potter and the Sorcerer's Stone.* Additionally, examples are provided of how topographical maps can expose connections and data points of events on social media tools like Snapchat or Facebook. Digital tools can be used to work with data and demonstrate the concept, such as spreadsheets and desktop publishing software. However, non-digital "unplugged" resources can also be used, such as poster board and markers or a pin-and-string plot on a bulletin board.

Chapter 4: Using Abstraction, Iteration, and Recursion in Labyrinths and Mazes presents learners with an opportunity to integrate concepts of world history, culture, and mathematics within learning activities that draw upon the concepts of the previous chapters while adding concepts of iteration and recursion. An emphasis is placed on planning and organization and begins by generating hand-drawn mazes and labyrinths, differentiating between the two.

Next, students begin to apply the mathematical calculations that can be used to describe their designs' appearance and how to solve them. Attention is given to the cultural relevance of the design and the significance to society through a historical lens. Again, the focus of this chapter is not on digital tools or technology, but rather the development of the computational thinking skills needed to not only develop a maze, but to examine the potential ways to solve it and expand on the patterns that are present in each design by using algorithms – in this case mathematical language (i.e., formulas) as a means to describe a phenomenon.

Chapter 5: Simulating the Different Laws of Physics in Video Games revisits the utilization of software as the primary means to accomplish CS-focused, computational learning activities. However, the authors suggest that students work in collaborative teams to explore unplugged activities that physically engage students in the major concepts in parallel with the simulation activities. The computational thinking skills in this chapter pertain to the physics concepts: Newton's Laws of Motion and the associated kinematic equations dealing with displacement, velocity, and acceleration. Additionally, the authors stress the value of teamwork, collaboration, and communication, as students work together during the programming process. This acknowledgment connects with earlier suggestions found in the text's introduction that STEM students should be working together, focusing on the concept of productivity from a position of learning from mistakes and collaboratively engaging in learning with a growth mindset. Neumann et al. (2021) argue that these actions work to "…empower more students to be proficient in technology prolific environments" (p.7) and encourage support for diversity and equity in STEM learning environments. The authors argue that these impacts play a critical role in a student's potential career path, especially as it relates to choosing careers in CS.

Chapter 6: Critically Examining and Analyzing Data focuses on the foundations of research inquiry and provides a methodical approach to investigating primary data to arrive at evidence-based conclusions. This approach includes the formulation of questions, data selection (e.g., sampling techniques) and collection, data processing methods, displaying and analyzing results, and description and discussion of the results. Neumann et al. (2021) describe the purpose of the chapter:

> To be informed citizens in a democratic society, we need to be educated consumers of the information we absorb and use. The electorate's ability to critically examine, analyze, and use data to make good judgments and decisions is paramount to the health of any nation. At the heart of computational thinking is the ability to apply a problem-solving process using different strategies and algorithmic solutions to complex problems. (p. 116)

Digital tools such as spreadsheets are widely used throughout the activities with an emphasis on how different types of graphs and various techniques of manipulating data can reveal a clearer picture and provide enhanced context with a dataset. Programming examples using Python are also presented, affording students an opportunity to go further within the CS context. While climate data is used in these activities, it is noted that the transfer of skills to process different datasets, for example, is a broader goal of the lessons, with an emphasis on creating new lines of inquiry and investigation.

Chapter 7: Incorporating Computational Thinking in the Classroom closes the book by briefly providing examples and reflections from five teachers (one was a middle school technology teacher) who have incorporated computational thinking skills into their classroom activities. The authors condense their feedback into five practical tips for educators who are considering incorporating computational thinking skills into their lessons:

1. Start small: Have a final project that contains computational thinking.
2. Scale up: Create lessons or units of learning that include computational thinking based on what your students need to know and do.
3. Learn all that you can.
4. Collaborate with colleagues across disciplines, schools, and/or states.
5. Keep calm and struggle on. (pp. 146-151)

Throughout the text, the reader is reminded that struggling is a part of the process of productivity and as teachers, we should both model and embrace the phenomena just as we would expect our students to do.

> Computational thinking is a way of thinking that is not new to a lot of people. Many people recognize the different components in the language of their content areas. However, integrating coding into your teaching is the newest piece of learning for many teachers. It takes time to learn new things. (p. 148)

Conclusion

Computational Thinking: An Integrative Approach for Middle and High School Learning has received several positive endorsements from CS educators, but it is clearly applicable to non-CS educators as well. Middle and high school Technology and Engineering Education and other STEM non-CS teachers will likely find this text useful when carefully applied to their teaching and learning environment. Many of the activities presented in this book appear to be well suited for high-achieving students in high school, but the content and instructional methods could be adjusted to serve most students in both middle and high school, as the title of the book suggests. While the text offers four

direct examples of implementation, the final chapter elaborates on how these examples provide opportunities for expansion and creativity. This is especially important to consider, as the individual teacher often knows what is best for their own students.

About the Author

Jeritt J. Williams (jjwilli@ilstu.edu) is an Instructional Assistant Professor in the Department of Technology and doctoral student in the School of Teaching and Learning at Illinois State University in Normal, Illinois.