Pedagogical Practices that Facilitate the Development of Computational Thinking Skills and Dispositions in Makerspaces

Abstract: Focusing on computational thinking (CT) as the process to integrate computer science across subjects, and making as the activity to engage in this process, is a promising way to introduce students to computing. However, there is little guidance on what practices instructors should employ in maker activities to support CT skill/disposition development. This case study takes place in a makerspace program, using the cognitive apprenticeship conceptual framework as analytical framing to answer the question: Within makerspace activities what evidence exists regarding promising practices that support youth development of CT skills and dispositions? Findings inform our understanding of practices instructors can utilize in making activities to assist students in advancing their CT skills/dispositions: tinkering, embodiment, walkthroughs, drawing, and debugging.

Introduction

Computational thinking, or CT, highlights the *practice,* the problem-solving skills and knowledge, needed for effective use of computing. CT builds upon problem solving skills that most students already have or can relate to, and promotes challenging assumptions through the evaluation of an artifact, encouraging critical thinking (Wing, 2008; Grover, 2018).

If CT is the thinking process needed to fully engage in our digital culture, making is the activity that helps carry out this process. Making, as it pertains to the maker movement, involves developing an idea into a product. Makerspaces are informal community workspaces with access to various materials and tools where people can congregate to make and innovate. Importantly, makerspaces attract people with varying levels and areas of expertise, making them an excellent environment for understanding the interplay among various disciplines and, therefore, demonstrating the versatility and power of CT (Halverson & Sheridan, 2014; Litts, 2015).

The purpose of this study was to examine the structure of makerspace environments and activities that contribute to CT skills. Specifically, this study addresses promising practices that instructors can employ to support student development of CT skills and dispositions within makerspace activities.

Theoretical Framework

For making and CT to happen cohesively to provide opportunities for learning in a classroom or makerspace, an appropriate structure must be put in place. While we have reason to believe that making has the potential to teach CT to children, we don’t know how this might actually be accomplished. Making for learning cannot be limited to just providing students with materials and a room, and hoping for the best. The learning must be supported and scaffolded in a way that develops expertise.

Collins, Brown, and Newman’s (1989) cognitive apprenticeship model focuses on the higher-order metacognitive and strategic processing skills employed by experts in a domain. Facilitators (or in school settings, teachers), make the implicit processes experts utilize explicit, introducing the domain to their students through appropriate strategic processing methods. These methods are often not learned didactically, but informally through apprenticeship methods like observation and coaching, with the learning of skills and knowledge in their specific context.

Cognitive apprenticeship places an emphasis on the decontextualization of knowledge, situating learning in a variety of settings so that students can practice applying the skills they have learned in diverse contexts. This model was proposed as a way to integrate the effective scaffolded and domain-specific practices of traditional apprenticeships with more traditional learning settings that do not have the luxury of small student-teacher ratios of apprenticeship models and have a more generalized focus on knowledge and cognitive skills (Collins, Brown, & Newman, 1989). In order to make this integration possible, cognitive apprenticeship approaches learning environments holistically, composed of four dimensions: content, method, sequence and sociology (Collins & Kapur, 2014).

In addition, cognitive apprenticeship makes novice and expert strategies explicit by employing a variety of pedagogical methods. These methods, namely *modeling, coaching, scaffolding, articulation, reflection,* and *exploration,* are intended to “help students acquire and integrate cognitive and metacognitive strategies for using, managing, and discovering knowledge” (Collins, Brown, & Newman, 1989, p. 480).

The effective investigation of the practices instructors could use to help students develop CT skills in a makerspace needs to be structured by a conceptual framework. ewi

Methods

The data collection came from four activities conducted in one four-hour day in a summer program. This day was selected because it encapsulated four of the types of activities participants were asked to do at different points in the span of the summer program: (1) work in large groups to review the material, using their own bodies to recreate a circuit on a breadboard, as seen in Figure 1; (2) play with the provided materials to make an interesting working circuit; (3) use computing tools (Arduino) to build on what was learned about basic circuits; and (4) consider the ways they could use the material they learned to create a final project aligned with their own interests. The observed subjects included 19 high school students from a large Midwestern school district and 6 facilitators from a large Midwestern library system. The collected data includes field notes, video and audio recordings, screen recordings, participant notebooks, and project artifacts. From those notes and transcripts, I conducted open coding, noticing facilitator practices and participant actions. I also coded for instances of where CT skills and dispositions were noticed in participants, as well as what cognitive apprenticeship method was being used by facilitators, if any. Finally, I coded for the impact of different instances, measuring them on a scale of 1-4, with 1 being not a very effective moment, and 4 being a very effective moment (Table 1). Impact in this case refers to how clearly the CT skill or disposition was demonstrated by instructor, or how well it was adopted by the participant.

The analysis focused on the resources and scaffolds instructors employ CT skills and dispositions, as defined by Selby and Woollard (2014) and ISTE and CSTA (2011), using the cognitive apprenticeship framework. Triangulation occurred both through involving another researcher in the interpretation of data (investigator triangulation) as well as using multiple sources of evidence (data triangulation) (Patton, 1987).

Findings

Open coding revealed a set of practices that the facilitators and participants engaged in that kept reappearing throughout the activities. The most common seen throughout the day were tinkering, embodying, walkthroughs, drawing, and debugging.

Tinkering was identified when participants were seen to be or encouraged to solve problems by "messing around" through play and discovery (Martinez & Stager, 2013).

The *embodying* code was applied to instances where participants or facilitators were using their own body or the body of another participant, to solve a problem or clarify an idea. The term *walkthrough* is in reference to a review process that programmers do, in which they walk their peers through the code. The *drawing* code was applied to instances where participants used drawing to plan, document, or work through a problem with a circuit. *Debugging* refers to when participants performed the systematic application of analysis and evaluation to determine why something is not working (Csizmadia et al., 2015).

Tinkering was the most widely used of the practices throughout all three activities, as participants were encouraged by facilitators to “play around” or “explore”. However, the tinkering mindset was not always a comfortable place for the participants (Table 2). The freedom could be overwhelming and the facilitators had to remind them that “this is not school” and responding “you’re making whatever you want to make” when asked what they were supposed to do. Aside from the discomfort from not having a clear goal, at the start of each activity, the participants were a lot of times guessing, plugging themselves into the tabletop breadboard, plugging components into the actual breadboards, or entering code into the Arduino software, without understanding what they were doing. But, the facilitators did not just work within the tinkering mindset to help participants develop more sophisticated learning strategies. Instead of just relying on participants arriving at an understanding of electricity or programming organically, they encouraged other practices to expand the participants' learning, namely, embodying, walkthroughs, drawing, and debugging.

Embodiment took place when participants had to use their own bodies and the bodies of their classmates as props to construct an algorithm for how electrons would move through their circuit, step-by-step (Figure 1). The traced articulation from the tabletop breadboard activity then extended throughout the day, coded as walkthrough. Facilitators kept using phrases like “talk me through it” “what do you mean” or “let’s hear it”, prodding participants to explain themselves in detail, and therefore pushing their thinking. This was especially true in developing their algorithmic thinking skills, constructing sequences in order to solve problems or better understand situations.

Drawing also served as a way for participants to further clarify their thinking. The drawing of circuits focused participants on removing unnecessary information when determining the structure of their circuit and how it worked (Figure 2). The physical location of a component on the breadboard sometimes confused participants as to where the component existed in the electrical path of a circuit, so the drawing forced them to better understand how the circuit worked.

Where drawing served as a sort of map for the tinkering, debugging provided an engine for it to continue. The clear structure of identifying a problem, determining the various possible reasons for the problem, and then coming up with solutions lent itself to various targeted instructional opportunities for CT that happened alongside the playful nature of tinkering and prevented it from getting frustrating or confusing.

By the end of the session, in which participants are finishing up working with the Arduino and brainstorming ideas for their final project, the tinkering matured with the facilitators moving from doing less modeling, coaching, and scaffolding, and allowing for more exploration, with participants incorporating CT into more intentional tinkering on their own. They still needed to confer with one another, but the conversations were purposeful and with direction.

Participants also began to get comfortable with the Use-Modify-Create progression (Lee et al. 2011), borrowing code and changing it to fit their purposes, experimenting with what's possible, taking risks, and trying a different tack if it did not work.

Discussion

By promoting various forms of expression for participants to "think aloud" their understanding in conjunction with cognitive apprenticeship methods, i.e. embodiment/walkthroughs, debugging, and drawing, the facilitators were able to encourage more strategic and focused tinkering to produce artifacts that both met the participants' goals for what they wanted to create and the facilitators' goals for what they wanted them to learn, as conceptualized in Figure 3.

Where the cognitive apprenticeship methods are placed on Figure 19 is only where they were most commonly observed, not that they were only observed during those practices, or that those practices did not coincide with other methods. For example, other methods besides articulation were utilized during embodiment/walkthrough and drawing practices, and articulation was used in the other practices, but articulation was most utilized during embodiment/walkthrough and drawing practices.

Embodiment served to present breadboard circuits in a larger, physical way that participants could identify with and collaborate on. This embodiment generated a habit of walkthroughs, with participants becoming comfortable with articulating the path of electrons through their breadboard circuit, and then beginning to articulate the sequence of their code in their Arduino projects, throughout the tinkering process.

Drawing schematics of their circuits also provided opportunities for articulation, while encouraging the development of the skill of abstraction, helping participants to focus on what was important in their circuit and think more strategically on how their tinkering should move forward. The facilitators' focus on drawing provided the participants with objects-to-think-with as they talked them through their process, as well as focusing their attention on the abstraction of problem representation. Facilitators also used drawing as reflection, connecting the student work to that of industry.

Once their circuit or their program was put together, the practice of debugging promoted constant inquiry and refinement as facilitators utilized modeling and coaching methods, instituting a more systematic process to the tinkering. Debugging served to help participants analyze and evaluate their work continually throughout the activities. Through embodiment, walkthroughs, drawing, and debugging, in conjunction with the utilization of the cognitive apprenticeship methods of scaffolding and exploration, participants' tinkering became more sophisticated, as did demonstrations of their CT skills and dispositions.

Acknowledgement: This material is based on work supported by the National Science Foundation under Grant [removed for blind submission]. Special thanks to the [removed for blind submission] team for their thoughtfulness and dedication.

References

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. Knowing, Learning, and Instruction: Essays in Honor of Robert Glaser, 18, 32-42.

Collins, A., & Kapur, M. (2014). Cognitive apprenticeship. In R. Keith Sawyer, ed., The Cambridge Handbook of the Learning Sciences, 2nd edn., Cambridge University Press. pp. 109-27.

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., Woollard, J. (2015). Computational thinking: a guide for teachers. Retrieved October 1, 2016, from https://community.computingatschool.org.uk/resources/2324.

Grover, S. (2018, November 5). A Tale of Two CTs (and a Revised Timeline for Computational Thinking). Retrieved from https://cacm.acm.org/blogs/blog-cacm/232488-a-tale-of-two-cts-and-a-revised-timeline-for-computational-thinking/fulltext

Halverson, E. R., & Sheridan, K. M. (January 01, 2014). The Maker Movement in Education. *Harvard Educational Review, 84,* 4, 495-504.

International Society for Technology in Education & Computer Science Teachers Association. (2011). Operational Definition of Computational Thinking for K-12 Education. Retrieved from http://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf on April 1, 2014.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*. 2(1), 32-37.

Litts, B. (2015). Making learning: Makerspaces as learning environments Retrieved from http://www.informalscience.org/sites/default/files/Litts_2015_Dissertation_Published.pdf

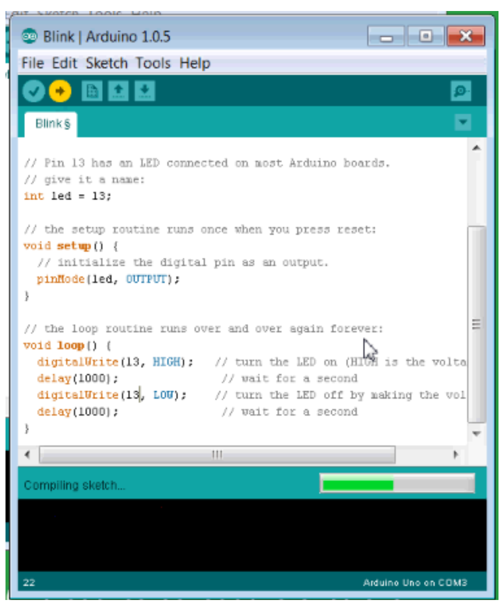
Martinez, S. & Stager, G. (2013a). *Invent to learn: making, tinkering and engineering in the classroom.* Constructing Modern Knowledge Press: Torrence, CA.

Patton, M. Q. (1987). *How to use qualitative methods in evaluation* (No. 4). Sage.

Selby, C. and Woollard, J. (2014) Computational Thinking: The developing definition. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education, SIGCSE 2014.* ACM.

Wing, J. M. (January 01, 2008). Computational thinking and thinking about computing. *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences, 366,* 1881, 3717-25.

Table 1. Examples of participants blindly guessing while tinkering

Activity	Example	Notes on example
Tabletop breadboard	Sam [to Ezra]: There seems to be a little bit of confusion on how to connect batteries in series just using the breadboard. They seem to just be taking the top out.	One facilitator (Sam) notices that despite working with the breadboards the day before, the participants are still unclear on how to power their breadboards and are randomly placing the battery at the top. He mentions this to the lead facilitator (Ezra).
Breadboard exploration	Researcher notes: Kids are just fiddling around … not completely understanding how the circuit works. [Another researcher] is reminded of paper on "Hands On, Minds On"	A researcher's observation notes point out the participants plugging components into their breadboards randomly without thinking through their actions.
Arduino	Denise and Ana change where the variable "led" is called to turn on the light to "13" then to "14" then back to "13" and then change where variable "led" is called to turn off to "13".  	After lead facilitator led the participants through the process of opening the boxes with the Arduinos, plugging them into their computers and sending them code to make an LED on a breadboard blink, he asked the participants to "change some of these variables and see what happens." He explains how the code calls for certain pin numbers in the breadboard to be activated and suggests they change the values for the delay. Two participants (Denise and Ana) demonstrate the haphazard way the participants began exploring the Arduino environment. Instead of changing the value assigned to the delay in order to change the blink pattern, the participants began changing where the variable was supposed to be called to turn on (HIGH) to the pin number in the breadboard where their LED was plugged in (13). They did this where it was called to be turned off (LOW), as well. They do not have an

		understanding of how programming variables work and how they should interact with the LED they are controlling.

Table 2. Examples of various impact scores

Impact Score	Example of coded selection	Notes
1	Juan: I added some stuff to your code and it might help you do what you were trying to do. It's really nice.	The facilitator did the work for the participant, missing out on a learning opportunity.
2	Ezra: Once you have this going, you're going to see your LED blinking, right? Now, I want you to try changing some of these variables and see what happens. Think, this is the number 13 and your positive is plugged into 13. What happens if you change this?	The facilitator indicates that he wants them to begin tinkering with the variables, but participants are unclear on which variables and what they should be doing with them.
3	Sam: What do you want your circuit to do? Ezra: Yeah what does this circuit do? Jackie: Turn on a light Angela: Turn on a light bulb. Ezra: You have two light bulbs in parallel? Or one light bulb? Jackie: We have one that turns on here.	Facilitators model evaluation for participants to consider the appropriate design for their circuit.
4	Sam: Is there anything unnecessary in this circuit? Ezra: Yeah, can we get rid of jumpers? Will it work with less jumpers? Well, we have to connect things differently than right? Julio: She has the one... Ezra: Yeah, so what are you connecting to? Julio: I'm just connected to [inaudible] so you can get rid of me, I'm [inaudible].	The participant evaluates the circuit and recognizes how to make it more efficient.

Figure 1. Facilitator demonstrating where the path of the electrons along the participants' bodies could go

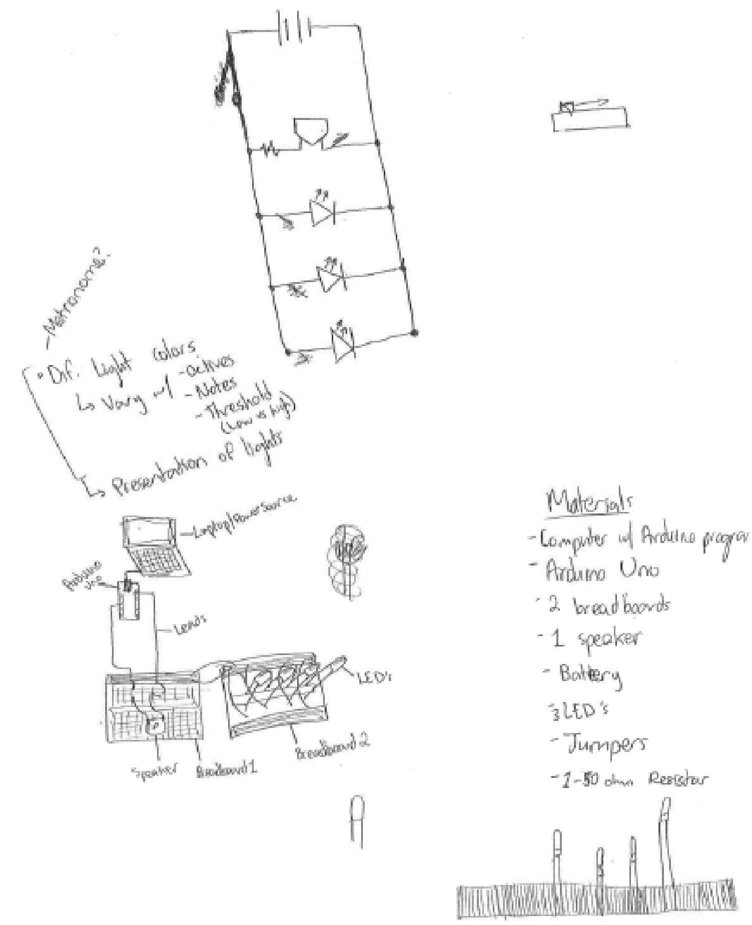


Figure 2. Using drawing to plan a circuit

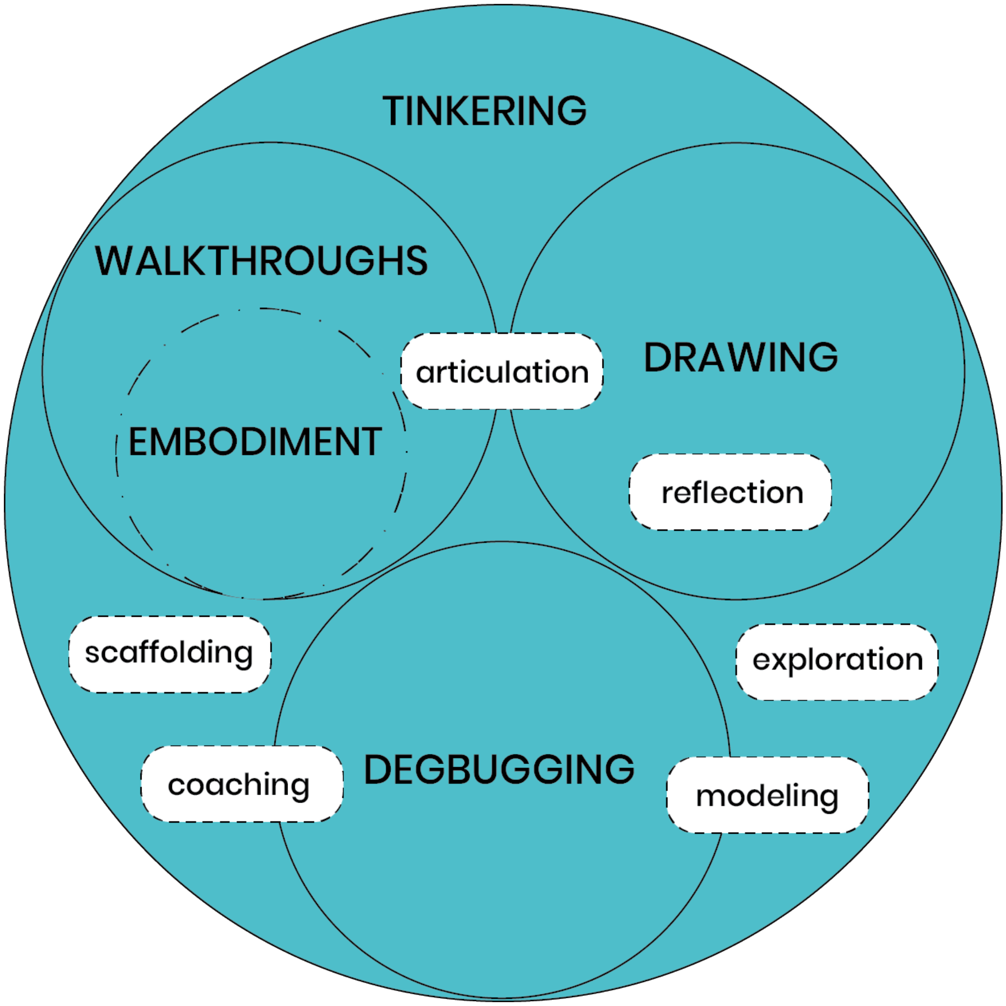


Figure 3. Practices and associated cognitive apprenticeship methods that supported the development of CT skills and dispositions in the observed maker activities