In this chapter, we focus on assessing computational thinking. To do so, we answer questions such as what does computational thinking look like when we look at student performance? How can we identify computational thinking across subjects? What considerations should be made when developing computational thinking assessments? We also discuss what teachers must know in order to assess students’ abilities to think computationally. The chapter reviews different approaches to the assessment of computational thinking and discusses how assessment can inform learning and teaching practices. Finally, we discuss to what extent new forms of assessment require new teacher competencies.

Introduction

With the increased presence of computational thinking (CT) in classrooms around the world, there is a growing need for teachers to be able to assess this essential 21st-century skill. To date, no single, unified definition of CT has emerged but there is growing consensus on constituent CT skills, such as abstraction, problem decomposition, algorithms, and debugging. Given the nature of CT and the wide array of situations where it can be employed and the ways it can be enacted, there is no “one-size-fits-all” or universal approach to assessing the concepts and practices that constitute CT. Instead, the nature of a CT assessment must be tailored to the specific contexts in which the assessment is being administered. This chapter reviews efforts to assess CT and also draws on assessments designed to measure other hard-to-assess constructs (Mislevy et al., 2009; Cheng et al., 2010). In this chapter, we provide guidance to help educators develop and administer CT assessments that yield accurate measures of important CT concepts and practices.

DOI: 10.4324/9781003102991-6
This chapter begins with a high-level discussion on the goals of assessment and what that means in the context of CT. We then lay out approaches for planning a CT assessment for both researchers and practitioners. This chapter then shifts to more practical guidance, beginning with a discussion of assessment approaches for CT, followed by how the specifics of a given context can impact the assessments used, and finally, we provide a series of examples of CT assessment that span age-level, assessment strategies, and contexts. The chapter concludes with a discussion of challenges associated with CT and the ways that assessment can inform teaching. Collectively, this chapter is intended to equip teachers with an overview of considerations and approaches for creating appropriate and effective CT assessments for their classrooms.

Goals of Assessment

Assessment is described as the art of reasoning from evidence about what a student knows and/or can do (National Research Council, 2001). From this evidence-centric view, assessments are designed to give students opportunities to provide observable evidence of their abilities and knowledge. Assessment users – such as researchers or educators – rely on these observations and an underlying model of student cognition to make inferences about student’s understanding. To determine the type of evidence that is needed, assessment designers must consider the overall purpose of the assessment. The National Research Council (2001) defines three general purposes for assessment. The first purpose of assessment is to assist learning. This role of assessment, often referred to as formative assessment, is intended to provide information to educators about a student’s progress, particular challenges they are facing, and identify areas of strengths and weaknesses. The next purpose is the assessment of individual achievement, often referred to as summative assessment. Here, the goal of the assessment is to assign a score that can be used as a grade in a classroom, or as an identification of a students’ ability along a continuum, such as a score on a large-scale assessment. The final purpose of assessment is to evaluate a program or curriculum, where the conclusion drawn is not related to specific students, but instead used as a tool to determine the effectiveness of a particular program. In this chapter, we will focus on student-focused assessments, namely formative and summative assessments.

It is important to think about the purpose of the assessment as it can provide insight into what the overall structure of the assessment should be. This includes considerations for what (and how many) constructs will be included in the assessment, what type of (and how much) evidence is needed from the student, and the type and form of tasks that should be included in the assessment. A summative assessment is often given at the end of a unit or course and therefore covers a breadth of constructs. In contrast, a formative assessment is given during instruction to provide teachers with information they can use to
determine the next instructional steps (Cizek, 2010). Given this goal, formative assessments often go into depth on one construct to allow students to demonstrate a range of abilities and/or to display challenges on that construct. How the purpose of an assessment relates to its design is discussed in greater detail below.

**Considerations for Assessing Computational Thinking**

Developing an assessment involves first identifying the knowledge, skills, and abilities to be covered, the type of evidence needed, and how assessment tasks can be defined (Messick, 1994; Mislevy, 2007). It is critical to align these three aspects of the assessment to ensure that the assessment is measuring the desired constructs and meeting its goals (Mislevy, 2007). By construct, we mean a particular set of knowledge, skills, or other attributes that are identified as desired outcomes of instruction, and thus, the focus of the assessment.

When defining the constructs to be measured, a place to start is defining the scope of the assessment. As mentioned above, this may differ depending on the purpose of the assessment. If the assessment is measuring progress toward meeting a learning standard (i.e., a standards-aligned assessment) then a starting point is the standard itself and supporting materials. For a classroom assessment (i.e., an assessment that measures learning objectives derived from instructional materials), the learning objectives and the curriculum lessons provide guidance for defining the main constructs to be covered. Here, the goal is to assess the constructs as they are taught in the specific classroom (as opposed to an assessment that could be used more generally). However, even with these starting points, the constructs may need to be further elaborated and defined. This elaboration highlights which aspect(s) of the construct are most important to include and how students are expected to engage with the construct during the assessment. Included in defining the construct of interest is specifying the boundaries of the assessment. The boundaries provide information on what is outside of the range of expectations for the assessment. These can help focus the assessment on specific aspects of a construct and also are a way to differentiate the expectations of students at different grade levels.

For example, when developing an assessment that focuses on the CT practice of developing and using abstractions, many constructs within that practice could be included (Bienkowski et al., 2015). At a high level, the designer of the assessment should decide if they want to include fundamental knowledge of abstractions (e.g., why abstractions are useful), the ability to analyze or use a given abstraction, the ability to create abstractions, or the ability to design solutions that include an abstraction. For each construct chosen, further elaboration and specifications of what to include on the assessment can be defined.

For an assessment task that is measuring students’ ability to create abstractions, further clarification would attend to how students would be creating the
abstraction, what type of abstraction they would be given, and what the resulting abstraction would look like. Aspects to consider when determining these clarifications are the expectations of students at that grade level and the overall context of the course or assessment. Expectations for students in the early grades will differ from students in older grades. For example, an abstraction for an early elementary student might involve observing a simple pattern and finding what repeats within the pattern. For older students, this may involve identifying places from a description of a situation that can or should be abstracted.

Once the boundaries are defined, the assessment developer can identify required vs. additional content knowledge. In the context of CT, on frequently encountered version of this consideration is whether or not the student is expected to have any prior knowledge of programming. If the assessment is solely for CT, and the developer wants to test CT with a programming task, then they must recognize that the assessment is requiring knowledge of the programming environment in addition to CT skills. If an additional knowledge or skill is not part of what should be measured then some method should be used to support students’ performance, either by providing them with additional scaffolds or resources related to the supplemental knowledge/skill, removing the ancillary content from the assessment, or allowing for errors in the scoring rubrics.

This decision is particularly critical in an integrated learning context where the CT skills are taught alongside disciplinary content in a core subject area (e.g., a CT-enhanced math or science lesson). In such contexts, there is the additional challenge of determining what it means for students to engage with CT in the context and how it is reflected in the assessment. The first decision, in this case, is whether or not the assessment is focusing solely on CT or if it will include CT as well as content from the related subject area. If the assessment focuses on integration, then it is critical to define what it means for CT to be integrated with the discipline. This could mean that the focus of the tasks is on CT, and the context for the tasks comes from the discipline, or it could mean that there are additional aspects of the discipline that must be incorporated into the assessment. Aligning the way that CT is framed and how it is integrated into the curriculum ensures that students have had the opportunity to learn the constructs being measured on the assessment.

Once the constructs and boundaries of the constructs have been defined, other practical aspects of the assessment administration should also be considered, including how much time is allotted for the assessment, and how the assessment will be delivered to the students. The constraints of the assessment delivery can shape what can be covered on the assessment. For example, the time allotted for students will limit the number of tasks that can be included, which then may limit the number of constructs to be measured. It may also limit the depth of the assessment within a particular construct.

Access to technology and the capabilities of the available technology may also affect the set of constructs that are to be measured. Some constructs may
be more appropriate in an environment where students have access to technology and are allowed to work through situations such as fixing and testing code errors, iteratively solving problems in a simulation-based environment, or collaboratively producing a design. Other constructs, such as justifying why a problem may be solved computationally, may be suitable for paper/pencil format to help students generate explanations. The degree to which the test environment matches the learning environment should be considered. For example, if students are learning and doing all of their work in an online system, but the assessment is paper/pencil then the assessor needs to take into account how the knowledge transfers from these two environments.

Strategies for Assessing Computational Thinking

Once there is a clearly defined set of constructs to be measured and the set of constraints have been identified, it is time to develop or choose an appropriate assessment. The considerations previously described provide guidelines for assessment development. This can be used to inform the selection of the assessment form and then shape the development of tasks that measure the set of constructs. This section will introduce some common forms of assessment and discuss their trade-offs with respect to assessing CT.

**Paper-and-Pencil Multiple Choice and/or Constructed Response Assessments**

Paper-and-pencil written assessments are one of the most widespread forms of classroom assessment. However, not all paper/pencil assessments are created equal. Paper/pencil questions can take several forms, including multiple choice (MC) questions, constructed response questions, or can involve larger performance or scenario-based tasks. While some argue that higher-order thinking skills (such as CT) cannot be measured with MC items (Darling-Hammond, 2014), there is still a use for this type of item format. MC items can quickly evaluate a base-level of content knowledge or, if designed correctly, they can identify challenges that students have. When deciding to use MC items it is important to identify what parts of CT can be measured in this format, what challenges students might face (to help identify distractors) and to recognize what aspects of CT are not being measured (Basu et al., 2020).

Not all constructs are appropriate for MC items. For example, to determine if students have the knowledge/skills to create artifacts or generate explanations, open-ended tasks are more appropriate. Examining the verbiage of the construct can help identify what type of item is most appropriate. If the desire is to measure a student’s ability to explain, then having students write an explanation would align with the desired construct. If students are creating an artifact, then further defining what it means to create in a paper/pencil format...
Assessment of Computational Thinking is critical. This could mean identifying the representations they can use (e.g., can students use pseudo-code to create an algorithm, can they use a picture to convey a solution), and specifying the boundaries of the task (e.g., while a task that requires students to design is aligned to the creation of an artifact it does not mean that all parts of creation are covered in a design task). For tasks that might ask students to represent patterns, a method that would allow students to visually represent these patterns may be important. Assessing these more complex concepts often benefit from situating the task within a scenario to help motivate and situate the activity being done (Rusmann & Bundsgaard, 2019).

It may still be challenging to measure students’ process skills such as decomposing problems, modularizing solutions, debugging program errors, and iterating through test cases using a paper/pencil assessment. Students may be able to describe the process they would use, but this may not always be how they would implement the process in practice. For tasks that require students to produce code when they do not have access to the tools they would normally use, the task may be measuring recall and not the ability to apply knowledge. For example, if students learned to code in a block-based language that offers visual supports such as displaying the set of available blocks, but the assessment task asks them to just produce code, the tasks may measure more of student’s ability to recollect the code than their ability to use the blocks.

**Technology-Enhanced Assessments**

While technology-enhanced assessments may not be available in all situations, they can provide a way to measure aspects of the process and enactment of CT practices. While some technology-enhanced assessments are essentially paper/pencil assessments in an online environment, others take advantage of technology through the use of simulations, real-time feedback, or game-based assessments (Shute & Moore, 2017). These assessments have the advantage of allowing students to interact in a simulated real-world environment, providing a more authentic CT experience for the student. Through the use of log-files, data can be collected and analyzed to examine not only students’ final products but also the process they followed to arrive at that product. Data analytic methods can be used to determine what set of skills students show evidence of, allowing for integrated skills (CT skills in the context of specific content) to be measured.

In simulation and game-based environments, it is critical that there is a strong theory of what is being measured and how actions in the environment relate to the constructs of interest as students have more ability (and are often encouraged) to play around and try things. Students are not always aware that they are being assessed and therefore may not act in a way that displays their skills. Not only should the action sequences be defined, but these action sequences should allow for multiple pathways and different student strategies. Actions that provide no additional information should also be identified, and
students should not be penalized for exploring, as students may have different goals in mind when interacting with the assessment environment.

While these assessments are able to measure complex skills (Quellmalz et al., 2008; Scalise, 2012; Shute & Moore, 2017), they can also be challenging to build and validate. They require the use of strong evidence models (i.e., how the tasks will be scored and what these scores mean) to identify the relationship between student actions and the constructs of interest. Considerations must also be incorporated around different student strategies and goals when interacting with the technology-enhanced assessment. These assessments, however, can be engaging for students and may generate information that more accurately reflects how students would interact in the real world than would a paper/pencil assessment.

**Portfolios/Artifact Analysis**

Portfolio assessments are conducted by collecting user-created artifacts over a period of time which are then evaluated for student understand and enactment of desired skills or practices. There are two main purposes for portfolio assessments: to measure growth and to highlight mastery skills or practices (Danielson & Abrutyn, 1997). The set of constructs to be measured in the portfolio should be pre-determined. Artifacts that students generate are then aligned to these constructs to ensure that the full set is represented. When using a portfolio to demonstrate growth, not all artifacts have to align to all constructs, but there must be opportunities for students to demonstrate growth over time and therefore multiple artifacts should be collected for each construct. When using a portfolio to demonstrate mastery, this same restriction may not apply, because one artifact can show mastery of one or more constructs.

When reviewing the portfolio and/or individual artifacts, good rubrics are critical. Rubrics should reflect a range of abilities for each of the constructs of interest. The rubric should make it clear either what constitutes growth or mastery depending on the purpose of the portfolio. Relating the aspect of CT that is of interest to the observations that can be made from the artifacts allows for inferences to be drawn about student performance. The rubrics may call for the identification of whether certain features are included in the artifact or may call for an evaluation of how well these features were incorporated into the artifact. For rubrics that call for the user to evaluate an artifact, clear guidance on the boundaries for the evaluation must be included in order to ensure that the rubric is applied uniformly.

**Surveys and Interviews**

For measuring students’ overall conceptions of CT and/or attitudes, surveys may be an appropriate tool. Surveys have the advantage that they are generally quick to administer, but they often are not able to get at deep conceptual
understanding and face self-report bias. Interviews can also be useful tools, as they allow for more probing into why and how students are thinking. Interviews, however, require a significant investment of time to administer and quantify and can be challenging to use in a classroom setting.

**Other Assessments**

The list above is not meant to be comprehensive, but instead to highlight how assessment formats can align with the purpose. Classroom teachers may also use other types of formative assessments, such as student journaling and student discourse. With formative assessment, there is an additional layer that should be considered, which is determining how the results of the assessment will be used to inform instruction (Cizek, 2010). Here again, the purpose of the formative assessment should be specified before development, the type of evidence that will be collected should be clear, and the actions performed based on that evidence should be specified prior to the administration of the assessment.

**Rubrics**

Any of the above approaches, with the possible exception of multiple-choice instruments, may require the use of a rubric. Rubrics are tools designed to assist in objectively evaluating the level of performance on an assessment item that does not have a fixed answer. Rubrics are used to score assessments and should be developed along with the assessment and should reflect the purpose and constraints of the assessment. Developing rubrics along with the tasks can highlight areas where the tasks are not aligned to the constructs, the tasks require skills that are not the focus of the assessment, or the prompts of the task may be misinterpreted. During rubric development, it is important to think about the range of possible responses and to highlight the desired student response. The desired response can be used to specify the top level of the rubric, and to check that the question is prompting for everything that is currently specified in this response.

**Instrument Validity**

An important aspect of assessment development, independent of the specific form the assessment takes, is providing evidence of the validity of the assessment for the stated purpose. Validity is the collection of evidence to support the inferences made as a result of an assessment (AERA, APA, NCME, 2014). While there are different frameworks for specifying the type of evidence that should be collected, in general, there is agreement that evidence should include ensuring that the questions on the assessment are measuring the construct of interest and not requiring students to use constructs, not of interest (construct validity), the inferences made from the assessment are appropriate, and the
internal structure of the assessment (e.g., reliability and factor structure) is appropriate. The degree to which evidence around the reliability is collected may vary depending on the stakes of the assessment. However, ensuring that there is alignment between the tasks that are developed, how student responses are evaluated, and the desired construct of interest provides support for the overall validity of the assessment for its purpose.

**Relationship between Context and Assessment**

One of the challenges of assessing CT is that it can be taught in various ways and across different contexts. As discussed previously, the context in which the CT concepts and skills are taught has direct implications for assessment. In this section, we discuss the relationship between context and the form that assessment might take. In exploring this relationship, we discuss three distinct contexts – physical computing, integrated CT, and programming-based CT – and for each context, present examples of assessments aligned to that context. We also vary the age range of learners to further highlight how to align assessment to the specifics of the context. These examples are intended to serve as demonstrations of the relationship between the operationalization of CT, the context in which learners engage with it, the characteristics of the learners, and the form of assessment used. While only a small sample of available assessments are presented here, researchers have compiled computer science and CT assessments to serve as resources for educators, including an online repository at https://csedresearch.org (with 181 assessments at the time of writing) and an academic systematic literature review of the state of CT assessment that includes 96 studies of assessments, categorized by grade level, subject domain, educational setting, and assessment format (Tang et al., 2020).

**Assessment in Physical Computing Contexts**

We begin by discussing approaches for assessing CT in physical computing contexts with younger learners. In particular, we look at young learners interacting with robots and how performance tasks can shed light on learners’ knowledge of and capabilities with CT concepts and practices. There is a growing body of research investigating how to support young learners in exploring CT concepts and practices through robots such as the Bee-Bots, Spheros, and Ozobots. For example, the TangibleK Robotics Program is a six-lesson curriculum designed to support young learners (ages 3–8) in designing, programming, and debugging basic programs written for a physical robot (Bers et al., 2014). To assess learners as they progress through the curriculum, Bers and colleagues developed a series of rubrics to evaluate learners’ developing mastery of CT concepts (e.g., sequencing instruction) and practices (e.g., debugging). The rubrics were then applied to the programs that learners wrote as a means of identifying students’ proficiency with CT concepts and practices.
A second example of assessing young learners using physical computing can be seen in the work of Angeli and Valanides (2020), who taught young learners foundational CT ideas using the Bee-Bot (Figure 6.1A). The Bee-Bot can be programmed by pressing the buttons on its top and then run by pressing the “Go” button. In this way, learners can program a robot to move without needing to type any commands or read sets of instructions. To assess learners with the Bee-Bot, Angeli and Valanides created a mat with a series of images on it (Figure 6.1B) and then gave the young learners a series of problems to solve that got progressively more difficult. For example, the Bee-Bot was placed on

![Figure 6.1](image-url)
the beehive at the bottom of the mat, and learners were asked to program it to visit the flowers in a specific order. Subsequent tasks then required more complex sequences of commands to solve. To assess performance, the researchers watched learners work through the problems and used a rubric to evaluate both the practices associated with programming (e.g., problem decomposition) as well as the conceptual understanding in terms of the correctness of the solution and the appropriateness of the blocks used.

In presenting these two examples of assessment with young learners using physical computing, it is important to highlight the overall appropriateness of the approach used. Given the age of the learners, neither assessment relies on literacy skills, and students were not asked to read long sets of instructions or to write out sequences of steps. Likewise, there is alignment between the technology used (programmable robots), the tasks learners carried out (defining sets of instructions for their robot), and the form of assessment (evaluating resulting programs and the process of creating those programs).

**Assessment in Integrated Contexts**

Our second set of examples of assessments looks at strategies for assessing CT in contexts where it is embedded within other disciplines. Specifically, here we look at two sets of assessments developed as part of a project seeking to embed CT into high school math and science curricula from the CT in science, technology, engineering and math (CT-STEM) project at Northwestern University. One set of assessments was designed to be curriculum-independent and the second to align to a specific set of activities. Both assessments use the CT-STEM Taxonomy as its operationalization of CT (Weintrop et al., 2016), which focuses on four categories of CT practices in math and science contexts: Data Practices, Modeling Practices, Computational Problem Solving Practices, and Systems Thinking Practices.

To create an assessment not coupled to a specific curriculum, the research team developed a suite of online assessments in which learners interact with embedded computational models, interactive visualizations, data collection and analysis tools, and basic programming environments (Weintrop et al., 2014). Each set of questions is situated within a short STEM-themed scenario. For example, a series of questions asked students to explore a basic ecosystem model with little purple bugs and grass (Figure 6.2A). This assessment focused on concepts related to computational modeling, data analysis, and systems thinking practices. As part of this assessment, students are asked to run the model, test out different configurations and interpret the resulting data and respond to questions like: “what happens to the purple bug population after a drought occurs?” Answering this question requires no knowledge of droughts or insect behavior, rather, it requires learners to test out the Drought behavior of the model and see how the purple bug population responds. The questions in this assessment include multiple choice questions as well as constructed response
questions that were scored using a rubric. The second version of this assessment replaces bugs with flower pickers and the grass with flowers, the underlying logic of the model, as well as the questions students answered are identical. The fact that the model logic and questions remained constant while swapping out the context highlights how the STEM content was decoupled from the CT skills being assessed.
The second CT-STEM assessment developed for this project uses a similar approach, that of embedding interactive models and posing multiple-choice and short answer questions; however, in this case, the STEM content was aligned to a curriculum called “From Ecosystems to Speciation” (Arastoopour Irgens et al., 2020). As part of this assessment, students were asked to analyze models related to topics drawn from the curriculum, including models exploring how a virus spreads through a community (Figure 6.2B), the relationships between different species within an ecosystem, and population dynamics. An example question from this assessment is: “List at least two ways that this model makes simplifications compared to how viruses spread in the real world?” Again, thematic analyses were conducted to evaluate the correctness of student responses, however, unlike the purple bugs example above, prior disciplinary knowledge is needed to answer the question. Like the previous assessment example, this assessment focused on CT practices related to modeling, data analysis, and systems thinking.

With these two examples, we show how CT can be assessed when situated within other domains, in particular, providing two examples of science-situated assessment assessing CT practices from the CT-STEM Taxonomy (Weintrop et al., 2016). In the CT-STEM project, that domain in high school math and science and the approach to assess CT is to use interactive simulations situated within an assessment which can either be coupled to a specific curriculum and ask CT questions that depend on learners’ knowledge of that domain or be situated within that context but not be coupled to specific disciplinary knowledge.

**Assessment in Programming Contexts**

The final context we look at for assessing CT is situated in more conventional computer science instruction, specifically, CT as it relates to learning to program. Here, we highlight two approaches: traditional written assessments and assessment based on artifact evaluation. There are a growing number of CT assessments based on conventional classroom assessment approaches. Such multiple choice/short answer assessments often ask students to read short programs and respond with the output of the program or to fill in blanks in the programs. Several such assessments exist, including Foundations for Advancing Computational Thinking (Grover et al., 2015) (Figure 6.3A), the Commutative Assessment (Weintrop & Wilensky, 2015) (Figure 6.3B), and the Computational Thinking Assessment (Román-González et al., 2017) (Figure 6.3C). Using a multiple choice/short answer approach has the benefit of aligning to conventional assessment practices and being easily administered and scored.

The second form of CT assessment common in programming contexts is the analysis of student-constructed artifacts, be they programs, computational crafting projects, or some other form of portfolio project. Analysis approaches can include automated tools, such as Dr. Scratch (Moreno-León et al., 2015),
Assessment of Computational Thinking

Hairball (Boe et al., 2013), and REACT (Koh et al., 2014) or frameworks for hand-scoring artifacts, such as the Progression of Early Computational Thinking (PECT) model for earlier grades (Seiter & Foreman, 2013) or Brennan and Resnick’s (2012) CT Assessment framework for Scratch projects. Alongside assessments based on the evaluation of learner-created programs, other assessment approaches use a similar approach but focus on learner-constructed physical artifacts, such as e-textiles (Lui et al., 2020) and digitally fabricated

---

(a) What is the value of $x$ at the end of the script:

- 12

(b) What will be the value of $x$ and $y$ after this script is run?

- $\text{var } x = 10$;
- $\text{var } y = x$;
- $x = (x + 5)$;

A) $x$ is equal to 15 and $y$ is equal to 15
B) $x$ is equal to 5 and $y$ is equal to 10
C) $x$ is equal to 15 and $y$ is equal to 10
D) $x$ is equal to “$x + 5$” and $y$ is equal to “$x$”
E) $x$ is equal to 10, 15 and $y$ is equal to 10
Maker-related constructions (Yin et al., 2020), we can see conceptual understanding enacted as well as assess CT practices in ways that are difficult through conventional assessments. However, it is important to note that emerging research suggests that the presence of some concepts in student-created artifacts is a weak proxy for conceptual understanding (Salac & Franklin, 2020), suggesting that artifact analysis is best used alongside other forms of assessment such as written assessments.

Across these examples, our intent is to highlight the importance of aligning assessment to the specifics of the context in which the learning is happening (including the technologies used, age of the learner, and the relationship between CT and other disciplines). Further, the goal of this section was to provide several examples of existing CT assessments as well as to highlight the diversity of approaches and tools that currently exist for assessing CT.

**How Assessments Can Inform Teaching**

As discussed, assessments can serve a summative purpose to provide an overall score for a student that represents an estimation of the students’ level of understanding or ability in a given subject area at a given point in time. The score by itself represents the average level of performance across the entire range of topics in a particular subject. Such a score is useful for making judgments about the level of competence of individual students. However, the score is not useful for determining which areas need improvement. In contrast, formative assessments can provide rich information about areas of strength and weakness for individual students. That information can be used by teachers to make

![FIGURE 6.3 Examples of three traditional CT assessments: (A) FACT, (B) the Commutative Assessment, and (C) the Computational Thinking Assessment. Credit line only for part C: Román-González et al. (2017).]
judgments about the areas of the curriculum in which students need more work. Good formative assessments will also provide teachers with a picture of student growth over time.

There are several key considerations for a robust approach to formatively assessing CT. First is the use of a variety of assessments. Different assessment types and contexts of assessments provide different information about the students. CT instructional experiences are typically performance-driven: students are designing, building, and debugging computational artifacts. These kinds of performance-based activities provide rich information about students’ ability to complete a task. The extent to which the task captures intermediate steps in the design process also provides information about students’ approaches to problem solving. Rubrics are extremely valuable for making judgments about students in these situations, as the rubric highlights what to look for throughout the activity as well as define the dimensions that are to be evaluated in the final product. Only evaluating the resulting artifacts does not reveal student’s thinking about how artifacts are produced. Providing opportunities for teachers to observe students as they engage in design and allowing students to explain their approach can enhance the assessment as long as the rubric provides information to teachers for making sense of their observations. Since rubrics provide different dimensions of quality and the range of quality within each dimension, student scores on the rubric provide information to the students about how the artifact can be improved.

A significant limitation of performance- or artifact-based assessments is that they do not reveal students’ thinking about how the artifacts are produced. Therefore, these types of assessments can be paired with evidence from observations and dialog. Teachers can observe students as they engage in design and ask them to explain their approach in the moment of design. After the fact, the teachers can probe students’ rationales behind the design and how the system works.

The second consideration is the validity of the formative assessment. An important assumption for any assessment is that the task can only be accomplished if the students apply the targeted concepts and skills. For example, if a task is meant to demonstrate a student’s ability to apply a particular programming construct, but the task can be accomplished without it, then successful completion of the task does not necessarily demonstrate student competency in applying the targeted programming structure. An important way to test the validity of a performance task is to combine it with student explanations for their process. Teachers can probe students as they engage with the performance task to gain information on how students are approaching the assessment task.

The third consideration is productive feedback. In prior research on formative assessment across a variety of subjects, the quality of formative feedback had a substantial influence on student learning growth (Black & Wiliam, 1998). For feedback to be effective (i.e., useful for learning), the students need to be aware of the standard of performance and receive information about their level
of performance relative to the standard. Providing students with only an overall score, or just the set of correct answers does not provide students with enough information on how they can improve. Feedback becomes productive when the teacher provides the mechanism by which the students can be successful. For example, in a programming task, the students can readily understand the standard for the desired program output given a specific input. If the student completes the program and it does not achieve the designed outcome, the student will need feedback on how to improve the code. If the teacher simply provides a grade or examines the code and tells the student what they did wrong, there is less opportunity for learning. If instead, after evaluating the program errors, the teacher suggests particular debugging strategies that will uncover the bug, the feedback becomes more useful for guiding future learning.

### Challenges of Assessing Computational Thinking

We have described several approaches to assessing CT, and we also want to highlight some challenges. In this section, we describe some of these challenges as a way to point to future directions for efforts seeking to improve the state of computational thinking assessments.

**Challenge 1:** Being clear about what is being assessed. We began this chapter by noting that the purpose of the assessment, as well as the construct(s) being measured, must be clearly defined. Given the relative newness of CT and the ongoing debates over definitions (Shute et al., 2017), specifying what to measure can get muddled. Researchers and practitioners might measure attitudes toward CT or look for interest and value in CT or computer science as an area of study. To match the right assessment to the right construct of interest, assessment developers have to be very clear about what is of interest. Lack of clarity is particularly noticeable when assessment developers purport to measure a construct beyond programming. Assessment may start with a goal to measure higher-level CT concepts in a way that is independent of programming and focus on constructs such as abstraction, generalization, problem solving, and pattern recognition. However, even if they successfully define these constructs, when they get down to the business of actually developing tasks for students to perform to show evidence of CT, they may end up devolving to measuring, for example, “abstraction” by counting how many variables students use or whether they understand a for a loop. Clarifying not only the definition of the concepts but also how students are expected to engage in these concepts, can help ensure assessments are measuring the concepts.

**Challenge 2:** Assessing practices vs. knowledge. Computer science has a long history of developing auto-graders that will test solutions for programming problems submitted by students, especially for large introductory classes at the postsecondary level where grading by hand is tedious. Early work in the assessment of computer science in K-12 also looked at program correctness and the
use of programming constructs such as loops, and conditionals (Werner et al., 2012). However, as described above, recent work argues that looking at the final product does not yield valid evidence for inferences about student knowledge as does measuring the students’ process of applying CT to create a program (Piech et al., 2012; Grover et al., 2017). However, conducting such measures at scale is difficult. Developments in learning analytics and educational data mining have revived interest in assessing student’s programming as an instance of open-ended problem solving and the creation of novel artifacts (Blikstein et al., 2014). Other efforts have focused on top-down, hypothesis-driven methods that look for a priori patterns that represent programming problem-solving strategies (Grover et al., 2017). Still, issues remain with analyzing the volumes of programming data to discern/abstract the strategies that students use, or the practices they pursue. Even more challenging is to then give guidance to students to help them get on more correct paths to a solution.

Challenge 3: Assessing CT in the context of other subjects. Computation is an important ancillary skill in disciplines such as science, mathematics, and, increasingly, social sciences. Because CT may be taught within the context of those disciplines, teasing out separate assessments of CT while assessing content knowledge can be difficult. However, it is necessary because (1) CT is recognized as an important skill and may be taught in an integrated way to meet state standards, and (2) teachers need to be sure that they are measuring the right skills and that construct-irrelevant knowledge is not interfering with the intended assessment.

What Teachers Need to Know to Assess Computational Thinking

K-12 computer science teacher professional development (PD) has progressed in developing educators who can teach computer science without having a CS degree or even college-level disciplinary courses. PD for various K-12 curricula focuses on equity and leveling the playing field for CS, on learning the ins and outs of a programming environment, or on developing teaching strategies for helping students overcome difficulties in accomplishing programming tasks. Teachers trained in this way may subsequently focus on assessing what they know: programming over CT, and interest over knowledge. Teachers need to understand the computational concepts and underlying CT practices, and how to discern evidence of knowledge of these concepts and practices in what their students do and say. For example, in a programming project, grading rubrics that focus on surface features of a problem solution (does the student have an information box on their web page?) show that a student can follow directions but do not necessarily assess CT.

When CT is integrated into other subjects, or not taught by a computer science educator in a course devoted solely to computer science, teachers may not know what kind of CT students at a particular grade level are capable of.
To assess CT, educators need to know what students are capable of knowing and doing so that adjustments or scaffolding can be provided for students to complete complex assessments. Likewise, it is important to decide how central assessing CT is to the ultimate learning goal of the class, as in integrated contexts, the primary assessment objective may be related to the domain content rather than CT outcomes.

Finally, teachers need to understand equity concerns in assessment. Like teaching, assessing can be inequitable, especially when it draws on prior knowledge that only certain students would be likely to have. It can also be inequitable if students have not had equal opportunities to learn the concepts being assessed. If teachers understand that they can scaffold students’ performance in completing an assessment item just as they scaffold students’ performance in completing a class activity, they can more equitably assess CT.

Conclusion

Given the growing presence of CT in K-12 classrooms, teachers tasked with teaching these important skills must be able to effectively assess the students in their classrooms. This is important not just for assigning a grade but as a means for gaining insight into the capabilities of students to inform instruction to ensure all students have foundational CT skills. As discussed in this chapter, assessment of CT is not a simple, straightforward exercise, instead, teachers must make several consequential decisions to tailor the assessment approach to fit the context and needs of their situation. With this chapter, we lay out what various dimensions to consider, the implications of the outcomes of these decisions, and provide examples and guidance in hopes of successfully equipping teachers to effectively assess CT in their classrooms.

Note

1 http://ct-stem.northwestern.edu.

References


Assessment of Computational Thinking  109


