Section 2:
Teaching and assessing computing in the curriculum

A framework for formative assessment and feedback to support student learning in CS classrooms

Shuchi Grover (Looking Glass Ventures, USA)

Grover, S. (2021). A framework for formative assessment and feedback to support student learning in CS classrooms. In Understanding computing education (Vol 1). Proceedings of the Raspberry Pi Foundation Research Seminar series.

Available at: rpf.io/seminar-proceedings-2020

Raspberry Pi

Section 2: Teaching and assessing computing in the curriculum

A framework for formative assessment and feedback to support student learning in CS classrooms

Shuchi Grover (Looking Glass Ventures, USA)

Abstract

This is a conceptual paper that makes a case for making formative assessment an integral part of computing in classrooms and formative assessment literacy a key part of computer science (CS) teacher training and preparation. The paper distils key ideas of formative assessment from education research that are key to understanding the what and why of this crucial classroom practice that can help improve teaching and learning in computer science classrooms. Drawing on prior research in CS education on assessment (albeit summative assessment, mainly) and programming comprehension, as well as ongoing research led by the author, the paper also presents dimensions of a preliminary framework that can help guide the adoption of formative assessment in K-12 CS and progress on three key aspects of formative assessment in K-12 CS: formative assessment design, formative assessment literacy in teacher professional development (PD), and leveraging community and community-developed resources for formative assessment.

Introduction

Over the last half decade, teaching of computer science has become widespread in school settings with curriculum, tools, teacher preparation, and classroom implementation concurrently making steady progress. Programming is central to K-12 CS experiences. Research literature on student difficulties in learning programming, even in easy-to-use block-based environments, continues to grow, as is the large body of literature on addressing novice programmer misconceptions that transcend age, context, and even programming environments. Formative (or classroom) assessment — aimed at assessment for learning, and often targeting student misconceptions — is a critical omission from K-12 CS education discourse and practice, especially in the US. Several studies have identified huge gaps in formative assessment and assessment literacy for K-12 CS teachers (e.g. Vivian & Faulkner, 2018; Vivian et al., 2020; Yadav et al., 2015), even when evidence in research asserts that attention to classroom formative "assessment can produce greater gains in student achievement than any other change in what teachers do" (Wiliam & Leahy, 2012).

Even though formative assessment has been a topic of scholarship for a long time since the 1960s (see Bloom, 1969), it was Paul Black and Dylan Wiliam’s seminal research in 1998 that crystallised the importance of formative assessment to improve learning and launched a very active field of educational research. Black & Wiliam (1998) defined formative assessment as “all those activities undertaken by teachers, and/

or by their students, which provide information to be used as feedback to modify the teaching and learning activities in which they are engaged." About ten years later, they established formative assessment as an explicit domain of assessment practice and defined it as follows, "Practice in a classroom is formative to the extent that evidence about student achievement is elicited, interpreted, and used by teachers, learners, or their peers, to make decisions about the next steps in instruction that are likely to be better, or better founded, than the decisions they would have taken in the absence of the evidence that was elicited" (Black & Wiliam, 2009, p. 9).

This paper addresses the need and rationale to push for more deliberate use of well-designed formative assessment in CS classrooms as well as formative assessment literacy for CS teachers. It first offers ten principles of formative assessment distilled from education research. These principles clarify what formative assessment is, and more importantly, is not, and serve to provide the rationale for why we need formative assessment. It then presents a preliminary framework to guide the K-12 CS education community. The framework highlights three key aspects or dimensions that need attention to bring formative assessments to classrooms — design of assessments, teacher preparation and formative assessment literacy, and community participation and resource repositories.

Ten principles of formative assessment

Seminal papers and groundbreaking research in education around three decades ago argued for and demonstrated that formative assessment improves student learning (Black & Wiliam, 1998; Crooks, 1988; Sadler, 1989). Since then, disciplinary-based education research in all core subjects has paid much attention to understanding and implementing formative assessment in classroom teaching to improve student learning. Google Scholar search results on formative assessment in school education run into hundreds of thousands of articles. Only a handful of these are situated in K-12 CS contexts. This section helps build a foundational understanding of formative assessment based on decades of education research through presenting ten principles of formative assessment distilled from education research. These ten principles essentially also describe what formative assessment is, and why it is important.

1. *Formative assessment is assessment **for**, rather than **of**, learning*. Assessment of learning is often referred to as summative assessment. Assessment for learning privileges the learning aspect, whereas assessment of learning privileges the assessment aspect.

2. *Formative assessment is all about feedback*. The raison d'être of formative assessment is to provide evidence and feedback to improve learning. Feedback is a key element in assisting the learning process for both instructors and students (Hattie & Timperley, 2007). In educational research, formative assessment is often not considered complete until it has resulted in feedback and action on the part of the teacher (or teaching agent) and/or the learner. However, feedback in formative assessment is mainly targeted at the student and is most valuable when students have the opportunity to use it to revise their thinking as they are working (Bransford, Brown, & Cocking, 2000). The feedback provided to the learner must impact:
    a. Learner's perception that there may be a gap between goal and where they are at currently, and
    b. What learners do to close the gap

3. *Formative assessment is not a "test"*. It

is NOT aimed to give a student a grade regardless of what pedagogy is being used in the classroom. W.J. Popham famously said “For some teachers, test is a four-letter word, both literally and figuratively” (Popham, 2009, p.5). However, the word assessment has often mistakenly led teachers to assume that formative assessment is about evaluating a student’s performance rather than view it as a part of the ongoing teaching and learning process. Teachers also harbor misperceptions around assessment in project-based classrooms. Barron & Darling-Hammond (2008) asserted that “The most effective inquiry-based approaches use a combination of informal ongoing formative assessment and project rubrics that communicate high standards and help teachers make judgments about the multiple dimensions of project work” (p.6). This widespread misperception of formative assessment as ‘tests of student learning’ is troubling. Heritage (2010) rues that the education community is at the risk of losing the promise of formative assessment for teaching and learning because of the false, but widespread, assumption that formative assessment is a kind of measurement instrument rather than a process that is integral to the practice of teaching and learning.

4. *Formative assessment is a process*. For the teacher, this process involves monitoring *(Is learning taking place?)* to diagnosis (*What is learned / not learned?*) to action (*What to do about it?*). For the learner, the formative assessment process helps them understand — *Where am I going? Where am I now? What are my next steps?*

5. *Formative assessment* is a form of *regulation* — at the classroom level, it helps a teacher regulate the learning process (Hudesman et al., 2013). At the student level, it serves as a way of self-regulation. Many scholars have written about how the external feedback of formative assessment triggers internal processing for the student. Monitoring and external feedback generates internal feedback at a variety of levels such as cognitive, motivational, and behavioral (Nicol & Macfarlane-Dick, 2006). The seminal work on *How People Learn* (Bransford, Brown, & Cocking, 2002) asserts that a formative interaction is one in which an interactive situation influences cognition, i.e. it is an interaction between external stimulus and feedback, and internal production by the individual learner. Crooks (1998) perhaps articulates it best — classroom assessment guides students’ judgment of what is important to learn, affects their motivation and self-perceptions of competence, structures their approaches to personal study, and affects the development of enduring learning strategies and skills.

6. *Formative assessment is critical for sharing learning goals with students and what constitutes “good” work*. A key part of formative assessment is to share day-to-day learning goals with students. If improvement in learning is to take place, students need to come to hold a concept of quality in line with that held by the teacher and the community (via standards, for example). This growing concept of what “good work” is forms part of the learning itself (Brookhart, 2003). As students begin to understand their intended learning goals, they develop the skills to make judgments about their learning in relation to a learning standard or instructional outcome and implement a variety of strategies to regulate their learning.

7. *Formative assessment is closely related to teacher pedagogical content knowledge* (PCK; Shulman, 1987). According to Heritage & Wylie (2018), who have done extensive

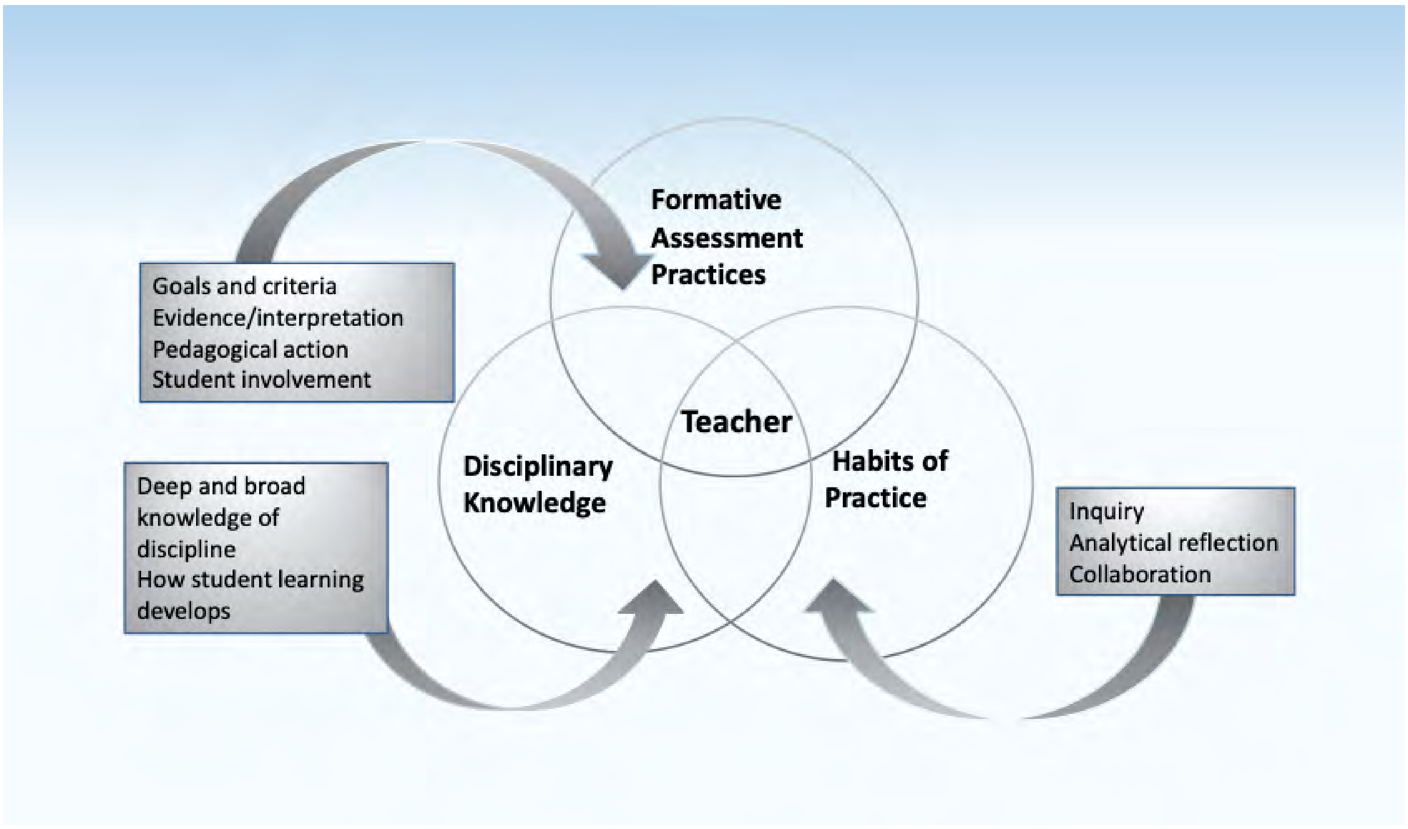


Figure 1. Teacher PCK is intertwined with classroom assessment and habits of practice (Image source: Heritage, 2018)

work on formative assessment in school mathematics teaching, teachers' formative assessment practices are closely intertwined with their disciplinary knowledge and classroom habits of practice (Figure 1).

8. *Formative assessment can and should take many forms*. Formative assessment can range from informal moves such as observation or a show of hands or even informal questions and conversations, to formal assessments that are administered to probe student understanding. Formal assessments can take many forms, including quick "quizzes" (including Entry/Exit Tickets), multiple choice (MC) and fixed answer probes, other innovative question types (such as Parson's Puzzles), open-response types questions (that would need manual grading), programming assignments (with rubrics to guide student work), peer and self-assessment, project showcase, self-explanation and reflection (written or audio-video recorded), and portfolios as well as artifact-based interviews. Examples of each of these are provided in Grover, Powers, & Sedgwick (2020). Ideally, teachers should employ "systems of assessment" that include various forms, target various cognitive, interpersonal, and intrapersonal learning goals of teaching CS, and provide a holistic multi-faceted view of student development

(Grover, 2017). These varied forms of assessment are key for equity and inclusion as well, since different forms of assessment privilege different learners.

9. *Formative assessment needs to be speedy and timely*. This aspect of formative assessment is key. Given that formative assessment is proximal to the learning process, it is key that a teacher and student receive feedback in time to remedy any lack of understanding. Research suggests that teachers' day-to-day classroom practices with an explicit focus on short-cycle assessment have been found to be most impactful (Wiliam, 2006). When teachers want to quickly survey student thinking, multiple choice items are efficient and have utility in terms of taking little time to ask, collect responses and process them. William & Black (2009) suggest attending to "Moments of Contingency", which are critical points where learning changes direction, depending on the information gleaned from the assessment.

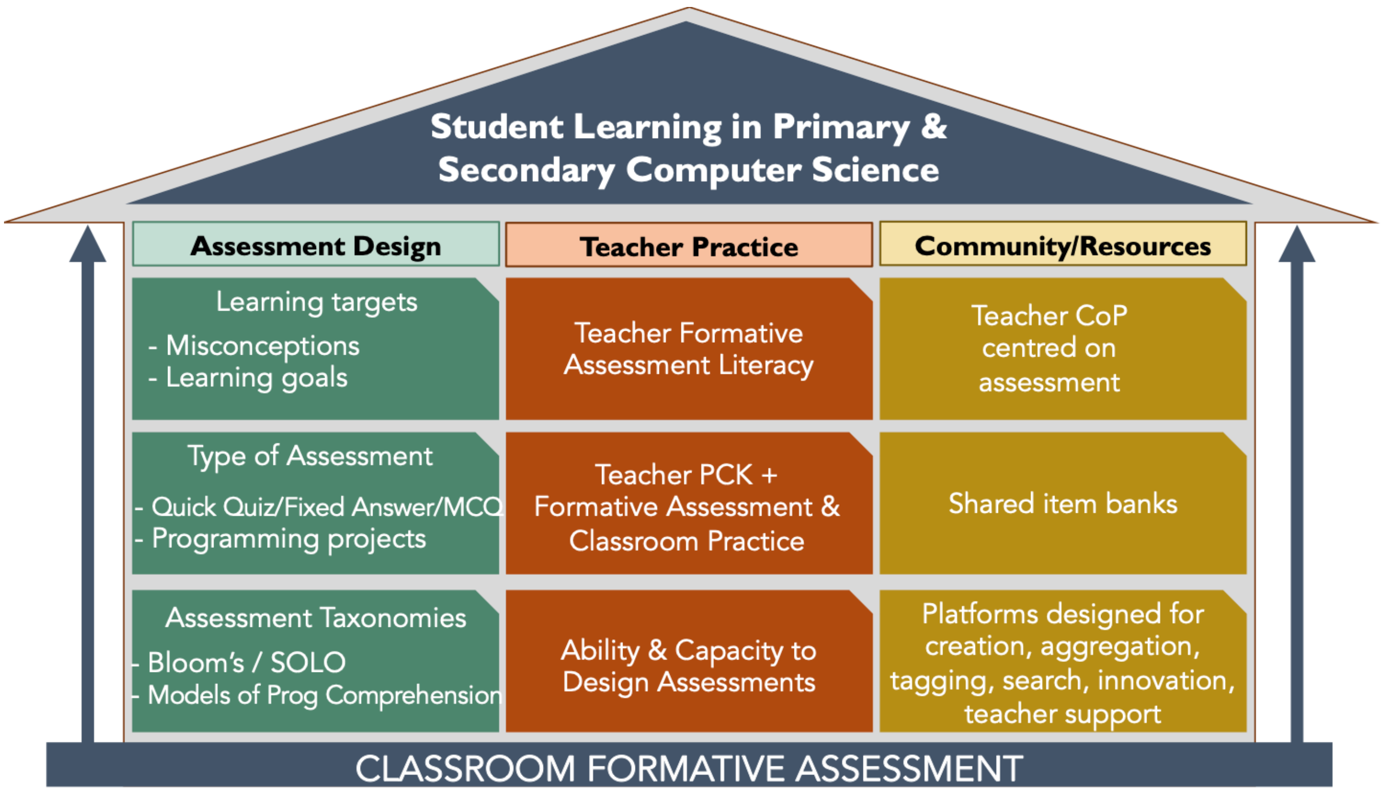
10. *Formative assessment should target known misconceptions*. Research in computer science education over the last four decades has documented several difficulties that novice learners face when they first encounter programming. Several of these difficulties and misconceptions are "sticky" and have been observed in learners of various ages and across various programming languages and environments. "Diagnostic items" that target known misconceptions are ideal candidates for formative assessment to probe whether or not students have understood key concepts (Ciofalo & Wylie, 2006; Wylie & Ciofalo, 2008).

Toward a framework of formative assessment for computing in schools

This section outlines a preliminary framework for successful integration of formative assessment in computer science teaching and learning. This framework has been developed as part of an ongoing research project funded by the National Science Foundation (DRL-1943530) aimed at examining formative assessment design, aggregating and creating a community resource or "assessments hub" (leveraging an online platform and homework system, Edfinity.com[7]), and building teacher awareness and formative assessment literacy. This framework also draws on prior research in formative assessment in broader education research such as work on diagnostic assessments by Ciofalo & Wylie (in the context of mathematics). It also draws on prior research in CS education on assessments, albeit mostly summative assessment (e.g. Clear et al., 2008; González, 2015; Grover, 2017, 2020; Lister, 2005; Lister et al., 2006; Schulte et al., 2010; Tang et al., 2020; Wiebe et al., 2019, among others), formative assessments (Grover, 2017; Grover, Sedgwick, & Powers, 2020); programming comprehension and learning trajectories of programming (e.g. Armoni, 2014; Izu et al., 2019; Schulte et al., 2010); and teacher preparation literature in CS (Vivian & Faulkner, 2018; Vivian et al., 2020) and more broadly (deLuca et al., 2018). It should be noted that research in formative assessment specifically in the context of CS and in K-12 settings is very thin. Among the few that exist, many are in the context of automated tools for assessing student programs (e.g. Basawapatna et al., 2015; Moreno-León et al., 2015; Von Wangenheim et al., 2018). These studies are (a) not generalisable as they are restricted to a specific programming language or environment or programming tasks, (b) provide little guidance on identifying specific areas of difficulty or misconceptions, and (c) may not be completely accurate in truly assessing student understanding (Salac & Franklin, 2020).

How successful formative assessment is in impacting student learning in K-12 computer

[7] http://edfinity.com


Student Learning in Primary & Secondary Computer Science
Assessment Design
Teacher Practice
Community/Resources
Learning targets
- Misconceptions
- Learning goals
Teacher Formative Assessment Literacy
Teacher CoP centred on assessment
Type of Assessment
- Quick Quiz/Fixed Answer/MCQ
- Programming projects
Teacher PCK + Formative Assessment & Classroom Practice
Shared item banks
Assessment Taxonomies
- Bloom's / SOLO
- Models of Prog Comprehension
Ability & Capacity to Design Assessments
Platforms designed for creation, aggregation, tagging, search, innovation, teacher support
CLASSROOM FORMATIVE ASSESSMENT

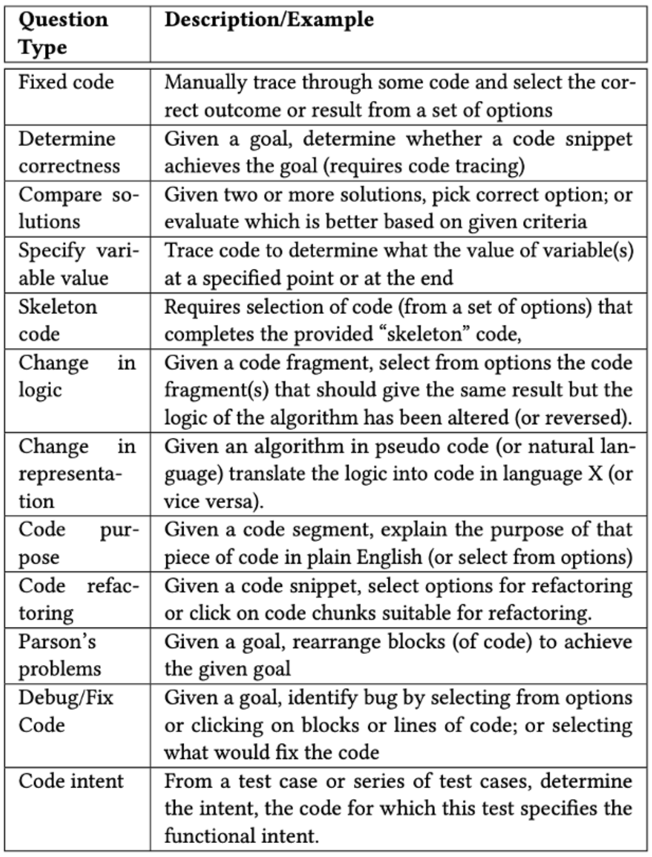
Grover, S. (2021, March). Toward A Framework for Formative Assessment of Conceptual Learning in K-12 Computer Science Classrooms. In Proceedings of the 52nd SIGCSE Technical Symposium. ACM.

Figure 2. Dimensions of a framework for formative assessment in CS school classrooms

science depends on three pillars or dimensions – the design of assessments, teacher assessment literacy and their classroom practice, and the broader community (Figure 2). These dimensions are interconnected but work at different levels of the 'computing at schools' enterprise. The remainder of this section describes each of these briefly. Grover (2021) provides more details on each.

Design of assessments for formative feedback

Formative assessments for K-12 CS classrooms could be formal or informal and target conceptual and/or affective learning goals. This effort recognises that programming assignments (in a programming language or environment) are popular as formative tasks, they are time-consuming to score especially on good rubrics that also shed light on exactly what specific concepts a student may or may not have understood (Grover, et al., 2018). This paper currently focuses mainly on formal, designed formative quiz-like check-ins for speedy feedback on conceptual understanding. These are strategic, targeted, autogradable, frequent, low-stakes, and provide quick feedback and explanation. Such items are suitable for probing understanding of key programming concepts (such a sequence, loops, conditionals, functions, expressions, variables, and other data structures) and CT practices (such as debugging, problem decomposition, algorithmic thinking, pattern recognition, and abstraction). These need not, however, always involve a code snippet or programming language. Well-designed multiple



Question Type	Description/Example
Fixed code	Manually trace through some code and select the correct outcome or result from a set of options
Determine correctness	Given a goal, determine whether a code snippet achieves the goal (requires code tracing)
Compare solutions	Given two or more solutions, pick correct option; or evaluate which is better based on given criteria
Specify variable value	Trace code to determine what the value of variable(s) at a specified point or at the end
Skeleton code	Requires selection of code (from a set of options) that completes the provided "skeleton" code,
Change in logic	Given a code fragment, select from options the code fragment(s) that should give the same result but the logic of the algorithm has been altered (or reversed).
Change in representation	Given an algorithm in pseudo code (or natural language) translate the logic into code in language X (or vice versa).
Code purpose	Given a code segment, explain the purpose of that piece of code in plain English (or select from options)
Code refactoring	Given a code snippet, select options for refactoring or click on code chunks suitable for refactoring.
Parson's problems	Given a goal, rearrange blocks (of code) to achieve the given goal
Debug/Fix Code	Given a goal, identify bug by selecting from options or clicking on blocks or lines of code; or selecting what would fix the code
Code intent	From a test case or series of test cases, determine the intent, the code for which this test specifies the functional intent.

Table 1. Item types for programming (Grover, 2021; Schulte et al., 2010)

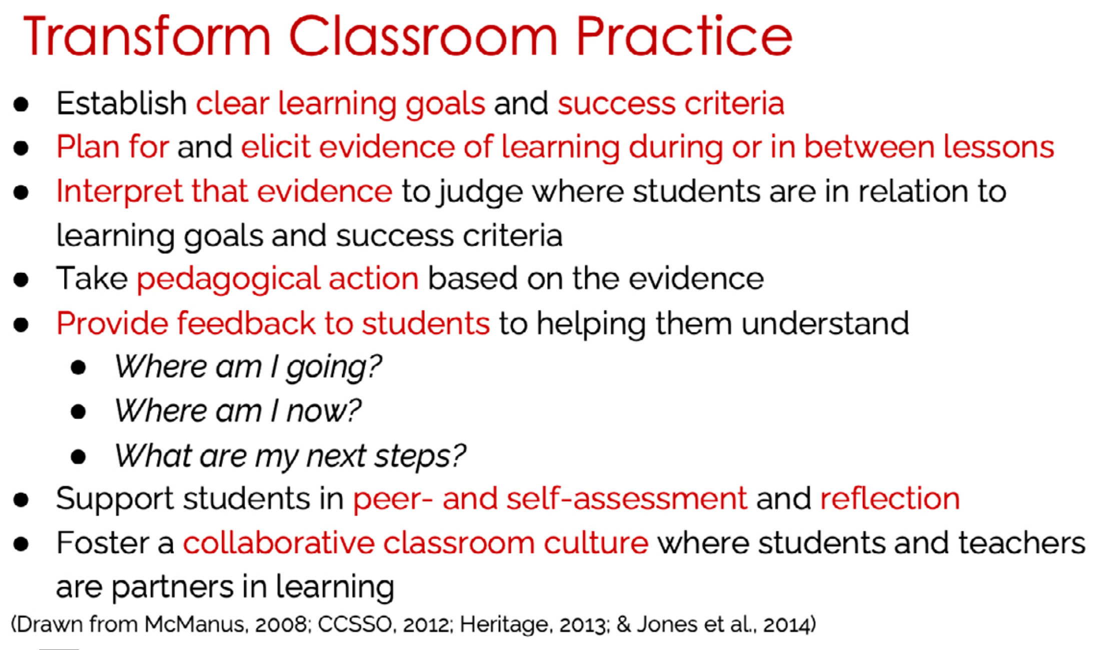
choice questions and easily gradable fixed answer types can probe and shine a light on conceptual understanding, and surface student difficulties and gaps in understanding. Past research in CS education has identified various kinds of good question types (Schulte et al., 2010). These, along with additional ones added by the author, are presented in Table 1.

Formative assessments, and specifically diagnostic items, should target the several known misconceptions and difficulties highlighted in CS education research (e.g. Soloway & Spohrer, 2013; Sorva, 2020). According to Ciofalo & Wylie (2006) and Wylie & Ciofalo (2008), what makes a diagnostic item particularly formative is that an incorrect response not only provides information about gaps in student understanding; it also provides insight into what it is that the student does not understand — in other words, the nature of their misconceptions.

Given the nature of formative assessment and its role in providing feedback on ongoing learning, formative assessments should also target learning progressions and the building blocks of programs that contribute to building program comprehension skills. Formative assessments can query student understanding of single concepts especially when a concept is first introduced. Given that learning of programming is intertwined with program syntax and semantics, it is also important that formative assessments target learning goals that encompass both structure and function as defined in Schulte's Block Model (Schulte, 2008, 2010) rather than only learning goals-oriented trajectories and progressions (such as those articulated by Rich et al., 2017). Examples of such items are shown in Figure 4. Bloom's taxonomy and SOLO taxonomy (Biggs & Collins, 1982; Clear et al., 2008; Lister et al., 2006; Thompson et al., 2008) have been used extensively in tertiary CS education assessment research and could similarly provide guidance on design of formative assessment items target varying levels of program comprehension and CT practices such as debugging, algorithmic thinking, and abstraction.

Teacher practice and formative assessment literacy

K-12 CS teachers' lack of confidence or knowledge and skills has impacted the implementation of assessments and the depth of feedback they provide (Vivian et al., 2020). It is therefore crucial to develop teachers' capacity and influence their habits of practice to make formative assessment integral to their teaching.



Transform Classroom Practice

- Establish clear learning goals and success criteria
- Plan for and elicit evidence of learning during or in between lessons
- Interpret that evidence to judge where students are in relation to learning goals and success criteria
- Take pedagogical action based on the evidence
- Provide feedback to students to helping them understand
  - *Where am I going?*
  - *Where am I now?*
  - *What are my next steps?*
- Support students in peer- and self-assessment and reflection
- Foster a collaborative classroom culture where students and teachers are partners in learning

(Drawn from McManus, 2008; CCSSO, 2012; Heritage, 2013; & Jones et al., 2014)

Figure 3. Classroom practice for supporting formative assessment (Source: Linquanti, 2014)

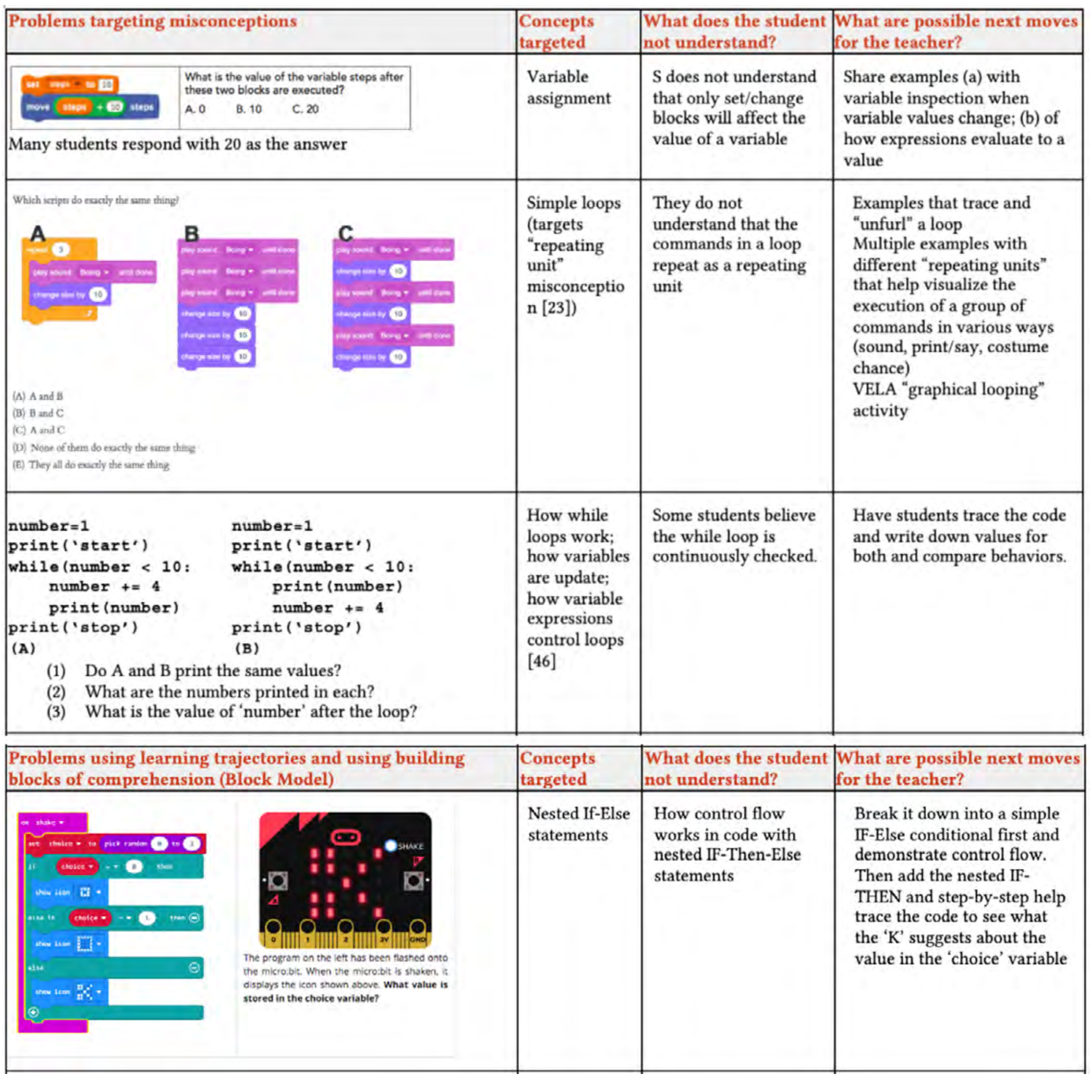
Teachers need to incorporate the formative assessment process as part of their routine classroom practice (Figure 3).

As part of the development of teacher PCK, teacher preparation needs to also help build in teachers an awareness of common targets of, and check students' understanding of, known targets of difficulty and misconceptions from CS education research. Teachers should have an understanding of how programming learning develops in novices and use items that target granular learning goals and elements of programming as guided by the Block Model (Schulte, 2008). Formative assessment literacy also provides teachers with the understanding of how to enact the formative assessment cycle of assessment, diagnosis, and formative action. Figure 4 provides examples of items targeting known misconceptions or granular learning goals along with the next moves teachers can make as part of the formative cycle.

Community resources: assessment repositories, feature-rich platforms, and collaboration

Teacher learning communities are a powerful mechanism to improve teachers' capabilities in using assessment in the service of learning. Teacher communities of practice (CoP) have been shown to sustain themselves around a shared need, and the give and take of shared resources for the benefit of all (Hoadley, 2012). Assessment item repositories are a useful mechanism, but only when they are well-designed to support a CoP of CS teachers (Fincher et al., 2010).

Extant and ongoing efforts for CS assessment item banks and repositories include Edfinity (edfinity.com), Project Quantum (https://diagnosticquestions.com/Quantum), Viva (Giordano et al., 2015), and the Canterbury Question Bank focused on introductory college-



Problems targeting misconceptions	Concepts targeted	What does the student not understand?	What are possible next moves for the teacher?
What is the value of the variable steps after these two blocks are executed? A. 0 B. 10 C. 20 Many students respond with 20 as the answer	Variable assignment	S does not understand that only set/change blocks will affect the value of a variable	Share examples (a) with variable inspection when variable values change; (b) of how expressions evaluate to a value
Which scripts do exactly the same thing? A B C (A) A and B (B) B and C (C) A and C (D) None of them do exactly the same thing (E) They all do exactly the same thing	Simple loops (targets "repeating unit" misconception [23])	They do not understand that the commands in a loop repeat as a repeating unit	Examples that trace and "unfurl" a loop Multiple examples with different "repeating units" that help visualize the execution of a group of commands in various ways (sound, print/say, costume chance) VELA "graphical looping" activity
`number=1` `print('start')` `while(number < 10:` `    number += 4` `    print(number)` `print('stop')` (A) `number=1` `print('start')` `while(number < 10:` `    print(number)` `    number += 4` `print('stop')` (B) (1) Do A and B print the same values? (2) What are the numbers printed in each? (3) What is the value of 'number' after the loop?	How while loops work; how variables are update; how variable expressions control loops [46]	Some students believe the while loop is continuously checked.	Have students trace the code and write down values for both and compare behaviors.
Problems using learning trajectories and using building blocks of comprehension (Block Model)	**Concepts targeted**	**What does the student not understand?**	**What are possible next moves for the teacher?**
The program on the left has been flashed onto the micro:bit. When the micro:bit is shaken, it displays the icon shown above. **What value is stored in the choice variable?**	Nested If-Else statements	How control flow works in code with nested IF-Then-Else statements	Break it down into a simple IF-Else conditional first and demonstrate control flow. Then add the nested IF-THEN and step-by-step help trace the code to see what the 'K' suggests about the value in the 'choice' variable

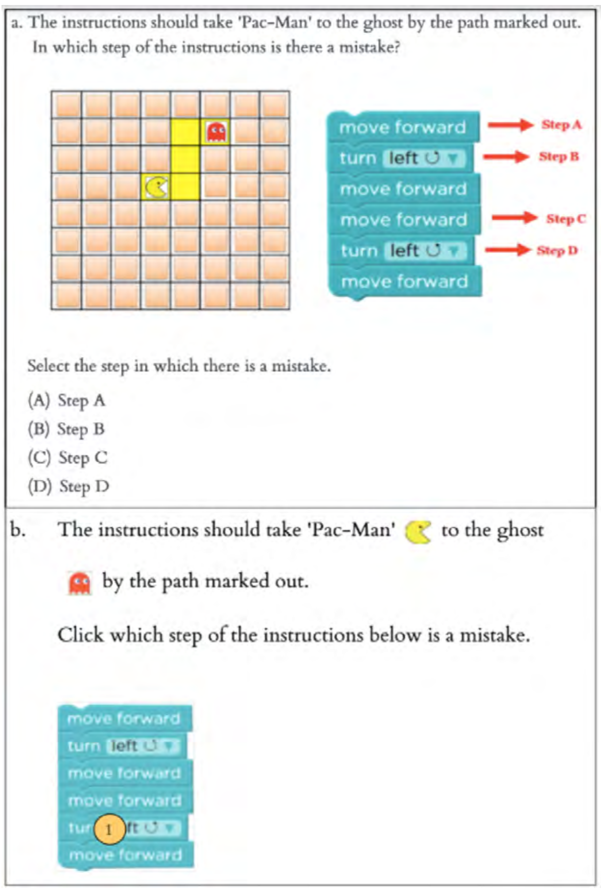
Figure 4. Examples based on research on misconceptions, learning trajectories, and levels of program comprehension, along with teacher diagnosis and formative action (Grover, 2021).

level CS (Sanders et al., 2013), with the author's current research contributing to the development of formative assessments on Edfinity.

In order to support formative assessment practice by teachers, assessment platforms and homework systems must be feature rich to support aggregation, creation, curation, and cataloging or taxonomising of assessments based on multiple and multi-level taxonomies relevant to CS teachers. This section uses Edfinity.com as an exemplar to describe such a platform. Taxonomies on Edfinity include CS/CT topics, learning standards (such as those from CSTA, 2017) or learning goals by curricula (such as AP CS Principles), grade, difficulty level, and ad hoc metadata (such as programming language) to support easy search and discovery. Edfinity also aids with assessment delivery, administration, auto-grading, and teacher dashboards. Backend data and analytics on student performance provide teachers crucial insights into students' learning and understanding at individual and aggregate levels (Grover et al., 2014). Edfinity also provides for multiple attempts of a question, hints, and feedback (or explanation) for correct and incorrect options. Solution explanations accompany the item and serve as (a form of) feedback. These explanations, as also the question stem, support rich text, graphics, and video for better learner engagement and multiple modes (and languages) of presentation to equitably support diverse learners. Edfinity item types include technology-enhanced assessments that push the boundaries to include interactivity (such as hotspot and point-and-click items), drag-drop (for items types such as Parson's Problems), microworlds, and in-browser code entry and testing. Such items are not only engaging but also help reduce cognitive load (Figure 5). Assessment creation and aggregation functions on Edfinity also support features for teacher collaboration (à la Google Docs), contribution, attribution, and sharing, as well as interfaces for creation of both simple and technology-enhanced items. Furthermore, technology platforms could innovate with randomised variants of items, solution validation, and customised feedback to students. Such technology platforms should be affordable. However, tools such as Google Forms, while free and popular for formative assessment, do not auto-grade or afford many of the features important for formative assessment. Similarly, paper formative assessments cannot be autograded or leverage aforementioned affordances of technology.

In Closing

This position paper makes a persuasive argument for formative assessment and teacher formative assessment literacy in K-12 CS,



a. The instructions should take 'Pac-Man' to the ghost by the path marked out. In which step of the instructions is there a mistake?

```
move forward      → Step A
turn left ↺       → Step B
move forward
move forward      → Step C
turn left ↺       → Step D
move forward
```

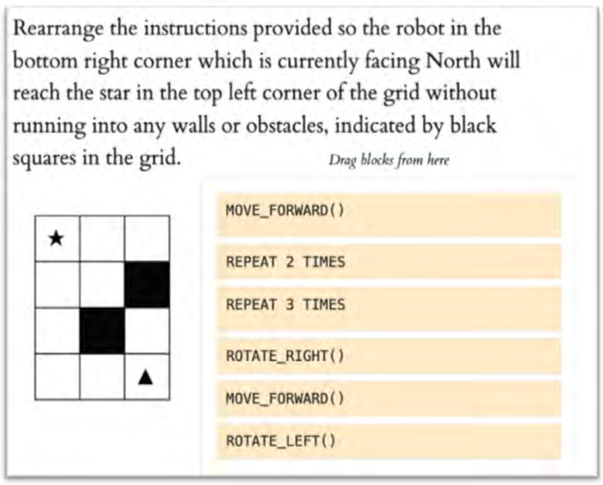
Select the step in which there is a mistake.

(A) Step A
(B) Step B
(C) Step C
(D) Step D

b. The instructions should take 'Pac-Man' to the ghost by the path marked out.

Click which step of the instructions below is a mistake.

```
move forward
turn left ↺
move forward
move forward
tur 1 ft ↺
move forward
```



Rearrange the instructions provided so the robot in the bottom right corner which is currently facing North will reach the star in the top left corner of the grid without running into any walls or obstacles, indicated by black squares in the grid.

```
MOVE_FORWARD()
REPEAT 2 TIMES
REPEAT 3 TIMES
ROTATE_RIGHT()
MOVE_FORWARD()
ROTATE_LEFT()
```

Figure 5a. An MCQ item from CTt () adapted into a point- and-click item (more intuitive and lower cognitive barriers) on Edfinity.com. (5b) A Parson's Puzzle Problem for AP CS Principles.

keeping the goal of robust student learning in mind. The framework presents several key ideas that can serve to provide guidance on taking important first steps to make both CS classroom teaching and learning as well as CS teacher preparation more robust through attention to formative assessments and formative assessment literacy, respectively. More work is needed, however, especially around classroom research on the use of formative assessments in different contexts and for various concepts. The framework as presented and explicated is focused on conceptual learning of programming. Although CS is certainly more than conceptual learning of programming, through focusing the framework and its dimensions on conceptual learning and examples of formative assessment forms, along with designs, tools, and guidance for providing convenient and powerful formative feedback, this paper makes a start in addressing a crucial lacuna.

References

Armoni, M. (2014). Spiral thinking: K--12 computer science education as part of holistic computing education. *ACM Inroads*, 5(2), 31-33.

CSTA et al. (2017). *CSTA K-12 computer science standards*, revised 2017. Computer Science Teachers Association, USA, 2017.

Barron, B., & Darling-Hammond, L. (2008). How can we teach for meaningful learning? In L. Darling-Hammond et al., editors, *Powerful learning: What we know about teaching for understanding*, 1, 11-16. Jossey-Bass, San Francisco, 2008.

Basawapatna, A. R., Repenning, A., & Koh, K. H. (2015, February). Closing the cyberlearning loop: Enabling teachers to formatively assess student programming projects. *In Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 12-17).

Biggs, J. B., & Collis, K. (1982). *Evaluating the Quality of Learning: the SOLO taxonomy*. New York: Academic Press..

Black, P., & Wiliam, D. (1998). Assessment and classroom learning. *Assessment in Education: principles, policy & practice*, 5(1), 7-74.

Black, P., & Wiliam, D. (2009). Developing the theory of formative assessment. *Educational Assessment, Evaluation and Accountability (formerly: Journal of Personnel Evaluation in Education*), 21(1), 5.

Bloom, B. S. (1969). Some theoretical issues relating to educational evaluation. *Educational evaluation: new roles, new means: the 63rd yearbook of the National Society for the Study of Education*, 69, 26-50.

Bransford, J. D., Brown, A. L., & Cocking, R. R. (2000). *How people learn* (Vol. 11). Washington, DC: National academy press.

Brookhart, S. M. (2003). Developing measurement theory for classroom assessment purposes and uses. *Educational measurement: Issues and practice*, 22(4), 5-12.

Ciofalo, J., & Wylie, C. E. (2006). Using diagnostic classroom assessment: one question at a time. *Teachers College Record*, 108(1).

Clear, T., Whalley, J., Lister, R. F., Carbone, A., Hu, M., Sheard, J., ... & Thompson, E. (2008). Reliably classifying novice programmer exam responses using the SOLO taxonomy. *National Advisory Committee on Computing Qualifications*.

Council of Chief State School Officers (CCSSO). (2012). *Distinguishing formative assessment from other educational assessment labels*. Washington, DC: Author. Retrieved from https://www.michigan.gov/documents/mde/CCSSO_Assessment__Labels_Paper_ada_601108_7.pdf

Crooks, T. J. (1988). The impact of classroom evaluation practices on students. *Review of educational research*, 58(4), 438-481.

DeLuca, C., Valiquette, A., Coombs, A., LaPointe-McEwan, D., & Luhanga, U. (2018). Teachers' approaches to classroom assessment: A large-scale survey. *Assessment in Education: Principles, Policy & Practice*, 25(4), 355-375.

Denny, P., Luxton-Reilly, A., & Simon, B. (2008, September). Evaluating a new exam question: Parsons problems. *In Proceedings of the fourth international workshop on computing education research* (pp. 113-124).

Fincher, S., Kölling, M., Utting, I., Brown, N., & Stevens, P. (2010, August). Repositories of teaching material and communities of use: nifty assignments and the greenroom. *In Proceedings of the Sixth international workshop on Computing education research* (pp. 107-114).

Giordano, D., Maiorana, F., Csizmadia, A. P., Marsden, S., Riedesel, C., Mishra, S., & Vinikienė, L. (2015). New horizons in the assessment of computer science at school and beyond: Leveraging on the viva platform. In *Proceedings of the 2015 ITiCSE on Working Group Reports* (pp. 117-147).

González, M. R. (2015). Computational thinking test: Design guidelines and content validation. In *Proceedings of EDULEARN15 conference* (pp. 2436-2444).

Grover, S. (2017). Assessing algorithmic and computational thinking in K-12: Lessons from a middle school classroom. In *Emerging research, practice, and policy on computational thinking* (pp. 269-288). Springer, Cham.

Grover, S. (2020, February). Designing an Assessment for Introductory Programming Concepts in Middle School Computer Science. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 678-684).

Grover, S. (2021). Toward A Framework for Formative Assessment of Conceptual Learning in K-12 Computer Science Classrooms. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. ACM.

Grover, S., Basu, S., & Schank, P. (2018, February). What we can learn about student learning from open-ended programming projects in middle school computer science. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 999-1004).

Grover, S., Pea, R., & Cooper, S. (2014, March). Promoting active learning & leveraging dashboards for curriculum assessment in an OpenEdX introductory CS course for middle school. In *Proceedings of the first ACM conference on Learning@ scale conference* (pp. 205-206).

Grover, S., Sedgwick, V., and Powers, K. (2020) Feedback through formative check-ins. In S. Grover, Ed., *Computer Science in K-12: An A to Z Handbook on Teaching Programming*. Edfinity, 2020.

Hattie, J., & Timperley, H. (2007). The power of feedback. *Review of educational research*, 77(1), 81-112.

Heritage, M. (2010). Formative Assessment and Next-Generation Assessment Systems: Are We Losing an Opportunity?. *Council of Chief State School Officers*.

Heritage, M. (2013). *Formative assessment in practice: A process of inquiry and action*. Harvard Education Press.

Heritage, M. (2018). Supporting Teachers' Successful Implementation of Formative Assessment. Presentation for Assessment Learning Network, Michigan Assessment Consortium.

Heritage, M., & Wylie, C. (2018). Reaping the benefits of assessment for learning: Achievement, identity, and equity. *ZDM*, 50(4), 729-741.

Hoadley. C. (2012). What is a community of practice and how can we support it? In Land, S., & Jonassen, D. (Eds.). *Theoretical Foundations of Learning Environments* (2nd ed.). Routledge.

Hudesman, J., Crosby, S., Flugman, B., Issac, S., Everson, H., & Clay, D. B. (2013). Using formative assessment and metacognition to improve student achievement. *Journal of Developmental Education*, 37(1), 2.

Izu, C., Schulte, C., Aggarwal, A., Cutts, Q., Duran, R., Gutica, M., ... & Weeda, R. (2019). Fostering Program Comprehension in Novice Programmers-Learning Activities and Learning Trajectories. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education* (pp. 27-52).

Jones, B., Chang, S., Heritage, M., Tobiason, G., & Herman, J. O. A. N. (2014). Supporting students in close reading. *National center for research on Evaluation, Standards, and Student Testing.*—Los Angeles: University of California, 5.

Linquanti, R. (2014). Supporting formative assessment for deeper learning: A primer for policymakers. *Formative Assessment for Students and Teachers/State Collaborative on Assessment and Student Standards. Washington, DC: CCSSO*. Retrieved from https://www.michigan.gov/documents/mde/CCSSO_Supporting_Formative_Assessment_for_Deeper_Learning_601111_7.pdf

References

Lister, R. (2005, January). One small step toward a culture of peer review and multi-institutional sharing of educational resources: a multiple choice exam for first semester programming students. In *Conferences in Research and Practice in Information Technology Series*.

Lister, R., Simon, B., Thompson, E., Whalley, J. L., & Prasad, C. (2006). Not seeing the forest for the trees: novice programmers and the SOLO taxonomy. *ACM SIGCSE Bulletin*, 38(3), 118-122.

McManus, S. (2008). *Attributes of effective formative assessment*. CCSSO.

Moreno-León, J., Robles, G., & Román-González, M. (2015). Dr. Scratch: Automatic analysis of scratch projects to assess and foster computational thinking. RED. *Revista de Educación a Distancia*, (46), 1-23.

Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in higher education*, 31(2), 199-218.

Parsons, D., & Haden, P. (2006, January). Parson's programming puzzles: a fun and effective learning tool for first programming courses. In *Proceedings of the 8th Australasian Conference on Computing Education-Volume 52* (pp. 157-163).

Popham, W. J. (2009). Assessment literacy for teachers: Faddish or fundamental?. *Theory into practice*, 48(1), 4-11.

Rich, K. M., Strickland, C., Binkowski, T. A., Moran, C., & Franklin, D. (2017, August). K-8 learning trajectories derived from research literature: Sequence, repetition, conditionals. In *Proceedings of the 2017 ACM conference on international computing education research* (pp. 182-190).

Salac, J., & Franklin, D. (2020, June). If They Build It, Will They Understand It? Exploring the Relationship between Student Code and Performance. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 473-479).

Sanders, K., Ahmadzadeh, M., Clear, T., Edwards, S. H., Goldweber, M., Johnson, C., ... & Spacco, J. (2013, June). The Canterbury QuestionBank: building a repository of multiple-choice CS1 and CS2 questions. In *Proceedings of the ITiCSE working group reports conference on Innovation and technology in computer science education-working group reports* (pp. 33-52).

Sadler, D. R. (1989). Formative assessment and the design of instructional systems. *Instructional science*, 18(2), 119-144.

Schulte, C. (2008, September). Block Model: an educational model of program comprehension as a tool for a scholarly approach to teaching. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 149-160).

Schulte, C., Clear, T., Taherkhani, A., Busjahn, T., & Paterson, J. H. (2010). An introduction to program comprehension for computer science educators. In *Proceedings of the 2010 ITiCSE working group reports* (pp. 65-86).

Shulman, L. (1987). Knowledge and teaching: Foundations of the new reform. *Harvard educational review*, 57(1), 1-23.

Soloway, E., & Spohrer, J. C. (Eds.). (2013). *Studying the novice programmer*. Psychology Press.

Sorva, J. (2020). Naive conceptions of novice programmers. In S.Grover, Ed., *Computer Science in K-12: An A to Z Handbook on Teaching Programming*. Edfinity, 2020.

Thompson, E., Luxton-Reilly, A., Whalley, J. L., Hu, M., & Robbins, P. (2008, January). Bloom's taxonomy for CS assessment. In *Proceedings of the tenth conference on Australasian computing education-Volume 78* (pp. 155-161).

Vivian, R., & Falkner, K. (2018, October). A survey of Australian teachers' self-efficacy and assessment approaches for the K-12 digital technologies curriculum. In *Proceedings of the 13th Workshop in Primary and Secondary Computing Education* (pp. 1-1)

Vivian, R., Franklin, D., Frye, D., Peterfreund, A., Ravitz, J., Sullivan, F., ... & McGill, M. M. (2020, June). Evaluation and Assessment Needs of Computing Education in Primary Grades. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 124-130).

Von Wangenheim, C. G., Hauck, J. C., Demetrio, M. F., Pelle, R., da Cruz Alves, N., Barbosa, H., & Azevedo, L. F. (2018). CodeMaster--Automatic Assessment and Grading of App Inventor and Snap! Programs. *Informatics in Education*, 17(1), 117-150.

Wiebe, E., London, J., Aksit, O., Mott, B. W., Boyer, K. E., & Lester, J. C. (2019, February). Development of a lean computational thinking abilities assessment for middle grades students. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 456-461).

Wiliam, D. (2006). Formative assessment: Getting the focus right. *Educational assessment, 11*(3-4), 283-289.

Wiliam, D., & Leahy, S. (2012). Sustaining formative assessment with teacher learning communities. In *PERIHA Professional Learning Series Workshop, Ministry of Education*.

Wylie, E. C., & Ciofalo, J. (2008). Supporting teachers' use of individual diagnostic items. *Teachers College Record*.

Yadav, A., Burkhart, D., Moix, D., Snow, E., Bandaru, P., & Clayborn, L. (2015). Sowing the seeds: A landscape study on assessment in secondary computer science education. *Comp. Sci. Teachers Assn.*, NY, NY.