

# Chapter 6

## Combining Assessment Tools for a Comprehensive Evaluation of Computational Thinking Interventions



Marcos Román-González, Jesús Moreno-León and Gregorio Robles

**Abstract** Given that computational thinking (CT) is still a blurry psychological construct, its assessment remains as a thorny, unresolved issue. Hence, in recent years, several assessment tools have been developed from different approaches and operational definitions of CT. However, very little research has been conducted to study whether these instruments provide convergent measurements, and how to combine them properly in educational settings. In response, we first review a myriad of CT assessment tools and classify them according to their evaluative approach. Second, we report the results of two convergent validity studies that involve three of these CT assessment tools, which come from different perspectives: the Computational Thinking Test, the Bebras Tasks, and Dr. Scratch. Finally, we propose a comprehensive model to evaluate the development of CT within educational scenarios and interventions, which includes the aforementioned and other reviewed assessment tools. Our comprehensive model intends to assess CT along every cognitive level of Bloom's taxonomy and throughout the various stages of typical educational interventions. Furthermore, the model explicitly indicates how to harmoniously combine the different types of CT assessment tools in order to give answer to the most common research questions in the field of CT Education. Thus, this contribution may lead scholars and policy-makers to perform accurate evaluation designs of CT according to their inquiry goals.

---

M. Román-González (✉)

Universidad Nacional de Educación a Distancia (UNED) – Facultad de Educación,  
C/ Juan del Rosal, nº 14, Office 2.18, Madrid, Spain  
e-mail: [mroman@edu.uned.es](mailto:mroman@edu.uned.es)

J. Moreno-León

Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado (INTEF),  
C/ Torrelaguna, nº 58, Madrid, Spain  
e-mail: [jesus.moreno@educacion.gob.es](mailto:jesus.moreno@educacion.gob.es)

G. Robles

Universidad Rey Juan Carlos (URJC), ETSI Telecomunicación, Camino del Molino s/n,  
Fuenlabrada, Madrid, Spain  
e-mail: [grex@gsyc.urjc.es](mailto:grex@gsyc.urjc.es)

© The Author(s) 2019

S.-C. Kong and H. Abelson (eds.), *Computational Thinking Education*,  
[https://doi.org/10.1007/978-981-13-6528-7\\_6](https://doi.org/10.1007/978-981-13-6528-7_6)

**Keywords** Computational thinking · Assessment · Evaluation · Comprehensive model · Research designs

## 6.1 Introduction

In the last decade, computational thinking (CT) (Wing, 2006) has emerged as an umbrella term that refers to a broad set of problem-solving skills, which should be acquired by the new generations to thrive in our computer-based world (Bocconi et al., 2016). Thus, the use of the CT term has evolved and grown up, even without reaching a consensus about its definition (Kalelioglu, Gülbahar, & Kukul, 2016).

Moreover, the relation between CT and computer programming is blurry too. It is assumed that computer programming enables CT to come alive, and it is the main way to demonstrate CT skills (Lye & Koh, 2014). However, CT might be projected onto a wide range of tasks that do not involve programming (Wing, 2008). In other words, it is necessary to activate CT skills in order to program properly, but these skills could be used in other contexts that are disconnected from computer programming. Therefore, CT is a broader term than computer programming.

In a certain sense, the coining of CT as an umbrella term has been extremely useful, and its sudden success can be explained. First, the CT term has helped to place Computer Science Education beyond computer programming. Second, it has helped to lower the entry barriers to computer programming, in parallel of the appearance and rise of visual blocks languages; in the same vein, CT has provided the frame to focus not on the computer programming syntax, but on the underlying mental processes for it. As a result, CT is perceived as a friendly and nonthreatening term that has contributed to bring Computer Science (CS) closer to the teachers and to foster the *CS4all* movement. Third, CT is such a *liquid* term that it can be used more as an approach than as a concept; then, CT has enhanced the metaphor of “programming to learn” instead of “learning to program.” In other words, CT has made possible to imagine a computational approach for any of the subjects of the curriculum. Finally, CT term has gathered not only cognitive skills, such as decomposition, pattern recognition, abstraction, and algorithmic design, but also noncognitive variables (Román-González, Pérez-González, Moreno-León, & Robles, 2018) and related soft skills such as persistence, self-confidence, tolerance to ambiguity, creativity, and teamwork, among others. In summary, CT term has served as a response to a global phenomenon in which it has become evident that our lives, increasingly mediated by algorithms, need a new set of skills to relate properly with the ubiquitous machines (Rushkoff, 2010; Sadin, 2015).

Nevertheless, this lack of definition of CT term that has proved useful in the past could be a burden for its future survival and development. Thus, there is not only a lack of consensus on a CT formal definition but also disagreements about how CT should be integrated into educational curricula (Lye & Koh, 2014), and especially on how it should be properly assessed (Grover, 2015; Grover & Pea, 2013). The latter is an extremely relevant and urgent topic to be addressed because without

reliable and valid assessment tools, CT will have difficulties to consolidate in the educational system and it runs a serious risk of disappearing as a construct worthy of consideration for educational psychology.

Expressed in a metaphorical way, CT term has had its naïve childhood, it has gone through an impulsive (and productive) adolescence, and it is now entering adulthood through a process of demystification. As Shute, Sun, and Asbell-Clarke (2017) say, CT is being demystified and, if it wants to survive, it is time to study it scientifically and to define operational CT models that can be empirically validated.

Ironically, although CT assessment seems to be the thorniest issue in the field, we consider that it brings the biggest opportunity to reinforce CT as a serious and well-established psychological construct. Assessment and measurement imply to operationally define the construct, CT in this case, in order to design an assessment tool that must be consequently validated. Hence, advances in assessment can contribute decisively to consolidate CT as a solid concept, a solid variable worthy of being studied and developed. In this sense, this chapter aims to review the current state-of-the-art CT assessment tools and to propose a comprehensive evaluation model, which could combine these tools effectively.

The chapter is structured as follows: in the next section, we review a myriad of CT assessment tools and classify them according to their evaluative approach. In the third section, we report the results of two convergent validity studies that involve three of these CT assessment tools, which come from different perspectives: the Computational Thinking Test, the Bebras Tasks, and Dr. Scratch. In the fourth section, we propose a comprehensive model to evaluate CT interventions within educational scenarios, which includes the aforementioned and other reviewed assessment tools. Finally, in the fifth and last section, we offer our conclusions and we speculate about future lines of research.

## 6.2 Computational Thinking Assessment Tools

Without being exhaustive, and focusing on K-12 education, we can find the following CT assessment tools, which can be classified depending on their evaluative approach:

- **CT diagnostic tools:** They are aimed at measuring the CT aptitudinal level of the subject. Their major advantage is that they can be administered in pure pretest condition (e.g., subjects without any prior programming experience). Complementarily, the diagnostic tools can be also applied in posttest condition (i.e., after an educational intervention) in order to check if the CT ability has increased. Some of the CT diagnostic tools are the *Computational Thinking Test* (Román-González, 2015; Román-González, Pérez-González, & Jiménez-Fernández, 2017b), the *Test for Measuring Basic Programming Abilities* (Mühling, Ruf, & Hubwieser, 2015), and the *Commutative Assessment Test* (Weintrop & Wilensky, 2015). All the aforementioned tests are aimed at middle-school and/or high-school students; for

elementary-school students, the instrument developed by Chen et al. (2017) in the context of everyday reasoning and robotics programming can be used.

- **CT summative tools:** Their goal is to evaluate if the learner has achieved enough content knowledge (and/or if he is able to perform properly) after receiving some instruction (and/or training) in CT skills. Then, the main use of the CT summative tools is placed in the posttest condition. We can distinguish among several of these tools according to the learning environment used. Thus, we find the summative tools of Meerbaum-Salant, Armoni, and Ben-Ari (2013) in the Scratch context; the tool named *Quizly*<sup>1</sup> (Maiorana, Giordano, & Morelli, 2015) for assessing content knowledge in the context of App Inventor; the *Fairy Assessment* (Werner, Denner, Campe, & Kawamoto, 2012), a performance-based tool that runs in the Alice environment; or the summative tools used for measuring the students' understanding of computational concepts after a new computing curriculum is implemented (e.g., see Zur-Bargury, Pârv, & Lanzberg, 2013).
- **CT formative–iterative tools:** They are aimed at providing feedback to the learner, usually in an automatic way, in order to develop and improve his/her CT skills. Strictly speaking, these tools do not assess the individuals, but their learning products, usually programming projects. Therefore, these tools are mainly used during the learning process, and they are specifically designed for a particular programming environment. Thus, we find *Dr. Scratch* (Moreno-León, Robles, & Román-González, 2015) or *Ninja Code Village* (Ota, Morimoto, & Kato, 2016) for Scratch; *Code Master*<sup>2</sup> for App Inventor; and the *Computational Thinking Patterns CTP-Graph* (Koh, Basawapatna, Bennett, & Repenning, 2010) or *REACT* (Koh, Basawapatna, Nickerson, & Repenning, 2014) for AgentSheets.
- **CT data-mining tools:** These tools, like the previous ones, are focused on the learning process. Nevertheless, while the formative–iterative tools statically analyze the source code of the programming projects, the data-mining tools retrieve and record the learner activity in real time. These latter tools provide valuable data and learning analytics from which cognitive processes of the subject can be inferred, and they are especially useful to detect gaps and misconceptions while acquiring computational concepts. It can be highlighted the research done by the team of Shuchi Grover (Grover et al., 2017; Grover, Bienkowski, Niekrasz, & Hauswirth, 2016) in the Blockly environment, and the one from Eguiluz, Guenaga, Garaizar, and Olivares-Rodriguez (2017) using Kodetu.
- **CT skill transfer tools:** Their objective is to assess to what extent the students are able to transfer their CT skills onto different kinds of problems, contexts, and situations. Hence, we find the *Bebras Tasks* (Dagiene & Futschek, 2008), which are focused on measuring CT skills' transfer to real-life problems. We also find the *CTP-Quiz* (Basawapatna, Koh, Repenning, Webb, & Marshall, 2011), which evaluates how CT skills are transferred to the context of scientific problems and simulations. Finally, the projection of CT skills onto kinesthetic tasks, and vice versa (i.e., embodied learning of CT), can be highlighted (Daily, Leonard, Jörg,

<sup>1</sup><http://appinventor.cs.trincoll.edu/csp/quizly/>.

<sup>2</sup><http://apps.computacaonaescola.ufsc.br:8080/>.

Babu, & Gundersen, 2014). This type of tools is especially suitable for assessing the degree of retention and transfer of CT, once a time has elapsed since the end of a CT educational intervention.

- **CT perceptions–attitudes scales:** They are aimed at assessing the perceptions (e.g., self-efficacy perceptions) and attitudes of the subjects not only about CT, but also about related issues such as computers, computer science, computer programming, or even digital literacy. Among scales targeted to students, we can name the *Computational Thinking Scales* (CTS) (Korkmaz, Çakir, & Özden, 2017), the *Computational Thinking Skills Scale* (CTSS) (Durak & Saritepeci, 2018), or the *Computer Programming Self-Efficacy Scale* (CPSES) (Kukul, Gökçearsan, & Günbatar, 2017). When we are interested in assessing the perceptions and attitudes of teachers, the research work of Yadav, Mayfield, Zhou, Hambrusch, and Korb (2014) can be highlighted. This kind of tools can be administered both before and after a CT educational intervention.
- **CT vocabulary assessment:** Finally, these tools intend to measure several elements and dimensions of CT, when they are verbally expressed by the subjects. These verbal expressions have been denominated as “computational thinking language” (e.g., see Grover, 2011).

It is worth noting that those different types of instruments have their own intrinsic characteristics, which lead each of them to approach CT assessment in a particular way. For example, while the diagnostic and the summative tools are based on student responses to predefined CT items or questions, the formative–iterative and the data-mining tools rely on the analysis of student programming creations and of student activity when developing CT, respectively. Thus, the information coming from each type of instruments has a different nature and all of them must be harmonized and triangulated to reach a complete CT assessment of the individual, as will be exemplified in the following empirical section.

Consequently, if only one from the aforementioned types of CT assessment tools is utilized, then it is very likely that an incomplete view of the students’ CT is obtained. This incomplete and biased view can lead us to misunderstand the CT development of our students, and to take wrong educational decisions. In the same vein, Brennan and Resnick (2012) have stated that assessing students’ computational competencies just looking at the programs created by the learners could be clearly insufficient, so they have emphasized the need of multiple means of assessment. Following this line of reasoning, Grover (2015) affirm that different types of complementary assessment tools must be systematically combined to reach a total and comprehensive understanding of the CT of our students. These combinations have been denominated as “systems of assessment,” which is the *leitmotiv* on which we want to contribute in the next two sections.

Therefore, in the next section, we investigate these “systems of assessments” from an empirical psychometric approach. It is supposed that a “system of assessment” will be composed of instruments that provide convergent measures. Specifically, in the next section, we study the convergence of the measurements provided by three of the aforementioned CT assessment tools. Later on, in the fourth section, we speculate

about some “systems of assessments” from a pedagogical research point of view, raising some questions like what assessment tools should be used according to the different moments of an educational CT intervention? What assessment tools should be used according to the different levels of cognitive complexity in our educational goals within a CT intervention? How to combine all the above to give answer to the most common research questions in the field of CT Education?

### 6.3 Convergent Validity Studies

In this section, we report two different convergent validity studies, which were carried out with two independent samples. The first study investigates the convergent validity of the Computational Thinking Test (CTt) with respect to a selection of Bebras Tasks. The second study does the same, but between the CTt and Dr. Scratch. Before reporting the results of both studies, some background and details about these three CT assessment tools are offered:

- **Computational Thinking Test (CTt):** The CTt<sup>3</sup> is a diagnostic assessment tool that consists of a multiple-choice instrument composed of 28 items, which are administered online in a maximum time of 45 min. Each of the items of the CTt is presented either in a “maze” or in a “canvas” interface, and is designed according to the following three dimensions:
  - **Computational concept(s) addressed:** Each item addresses one or more of the following computational concepts, which appear in increasing difficulty and which are progressively nested along the test: basic directions and sequences; loops (repeat times, repeat until); conditionals (if, if/else); while conditional loops; simple functions.
  - **Style of response options:** In each item, responses are depicted in any of the following two styles: “visual arrows” or “visual blocks”.
  - **Required cognitive task:** In order to be solved, each item demands to the subject one of the following cognitive tasks: to sequence an algorithm, to complete an incomplete algorithm, or to debug an incorrect algorithm.

The CTt has demonstrated to be reliable and valid for assessing CT in subjects between 10 and 16 years old (Román-González, 2015; Román-González et al., 2017b). We show some examples of the CTt items in Figs. 6.1, 6.2, and 6.3, whose specifications are detailed in the respective caption.

- **The Bebras Tasks:** These tasks consist of a set of activities designed within the context of the *Bebras International Contest*,<sup>4</sup> a competition created in Lithuania in 2003, which is aimed at promoting the interest and excellence of K-12 students around the world in the field of CS from a CT perspective (Dagiene & Futschek,

---

<sup>3</sup>Sample copy available at: <https://goo.gl/GqD6Wt>.

<sup>4</sup><http://bebras.org/>.

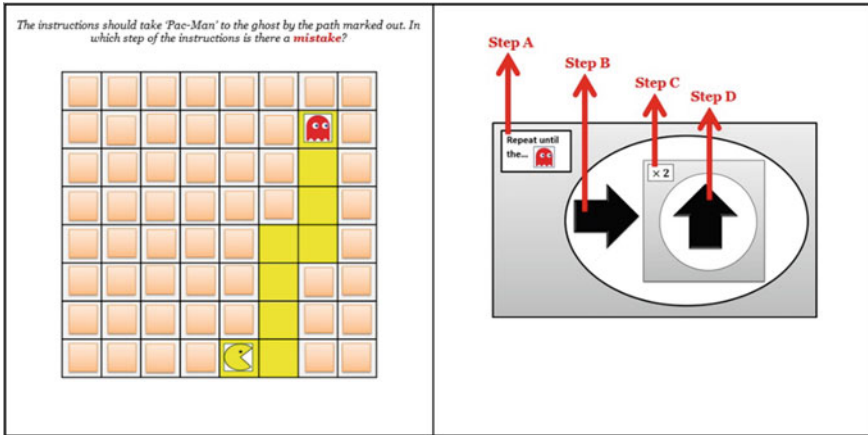


Fig. 6.1 CTt, item #11 (“maze”): loops “repeat until + repeat times” (nested); “visual arrows”; “debugging”

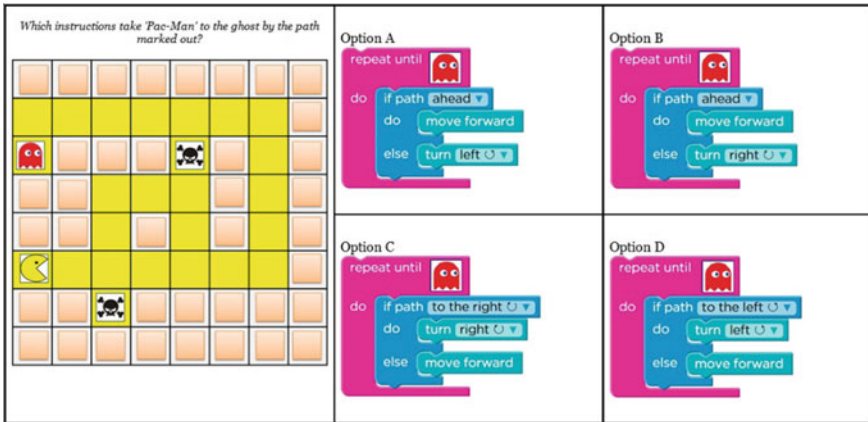


Fig. 6.2 CTt, item #18 (“maze”): loops “repeat until” + if/else conditional (nested); “visual blocks”; “sequencing”

2008). Every year, the contest launches a new set of Bebras Tasks, which require the students to transfer and project their CT skills in order to solve “real-life” problems. For this feature, in the previous sections, we have classified the Bebras Tasks as a CT skill transfer assessment tool. Moreover, another advantage of the Bebras Tasks is that they are independent from any particular software or hardware, and they can even be administered to individuals without any prior programming experience. The three Bebras Tasks used in our convergent validity study are shown in Figs. 6.4, 6.5, and 6.6.


<p>The following set of instructions is called 'my function', and draws one triangle of 50 pixels each side:</p> <pre> Function my function repeat 3 times do   move forward by 50 pixels   turn left by 120 degrees         </pre>	<p>Option A</p> <p>15</p>	<p>Option B</p> <p>5</p>
<p>The instructions below should make the artist draw the following design. Each side of each triangle measures 50 pixels. What is missing in the instructions?</p> <pre> repeat ??? times do   my function   jump forward by 50 pixels         </pre> 	<p>Option C</p> <p>4</p>	<p>Option D</p> <p>3</p>

Fig. 6.3 CTt, item #26 (“canvas”): loops “repeat times” + simple functions (nested); “visual blocks”; “completing”

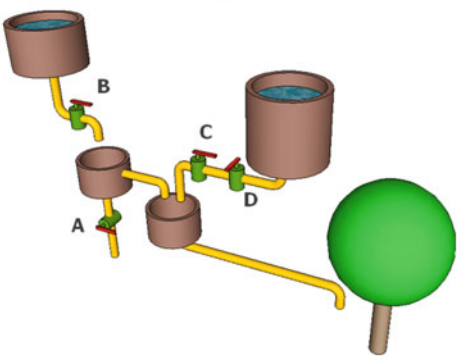
<p>Beaver has constructed a pipeline system to water his apple tree. The expressions contain variables A, B, C, D, which may be true or false. A variable has the value true, if the corresponding gate is open, and false, if it is closed.</p> <p><b>In which case the apple tree gets water?</b></p> 	<p><b>Option 1</b> A = false, B = true, C = false, D = false</p> <p><b>Option 2</b> A = true, B = true, C = false, D = false</p> <p><b>Option 3</b> A = true, B = false, C = false, D = true</p> <p><b>Option 4</b> A = false, B = false, C = false, D = true</p>
--	---

Fig. 6.4 Bebras Task #1: Water Supply (CT dimension involved: logic-binary structures) (reprinted by permission of <http://bebras.org/>)

- **Dr. Scratch**<sup>5</sup> (Moreno-León et al., 2015) is a free and open-source web application that analyzes, in an automated way, projects programmed with Scratch language. The score that Dr. Scratch assigns to a project is based on the degree of development of seven dimensions of CT competence: abstraction and problem decomposition, logical thinking, synchronization, parallelism, algorithmic notions of flow control,

<sup>5</sup><http://www.drscratch.org/>.



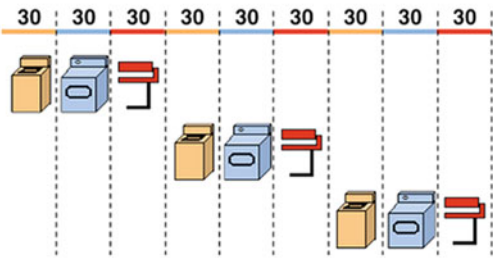
<p>Beaver Joe has started a new laundry business. He has got three machines: a washer, a dryer and a pressing iron. Every machine is connected through its own timer which provides for half an hour of electricity.</p> <p>So, when a client arrives, he needs 90 minutes for all of the three procedures. And three clients using the machinery consequently need 270 minutes.</p>  <p>But now, there are three beavers arriving which are really busy. Each one of them has enough clothes for a load of its own. But they agree that they want to finish as quickly as possible.</p> <p><b>How many minutes does it take for all three of them to finish their laundry?</b></p>	<p>Option A 90 minutes</p>
	<p>Option B 120 minutes</p>
	<p>Option C 150 minutes</p>
	<p>Option D 270 minutes</p>

Fig. 6.5 Bebras Task #2: Fast Laundry (CT dimensions involved: parallelism, algorithms) (reprinted by permission of <http://bebras.org/>)

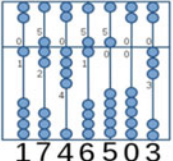
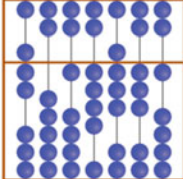
<p>A number is represented on a Chinese abacus by the position of its beads. The value of a bead on the top part is 5; the value of a bead on the bottom part is 1. The abacus is reset to zero by pushing the beads away from the centre.</p> <p>To represent the number 1 746 503 the appropriate beads are moved towards the centre of the abacus:</p>				
<p>What number does the following abacus represent?</p>				
				
<p>Option A 3014431</p>	<p>Option B 7514831</p>	<p>Option C 3514431</p>	<p>Option D 7014831</p>	

Fig. 6.6 Bebras Task #3: Abacus (CT dimension involved: abstraction, decomposition, algorithms) (reprinted by permission of <http://bebras.org/>)

user interactivity, and data representation. These dimensions are statically evaluated by inspecting the source code of the analyzed project and given punctuation from 0 to 3, resulting in a total evaluation (“mastery score”) that ranges from 0 to 21 when all seven dimensions are aggregated. In addition, Dr. Scratch generates a feedback report that is displayed to learners, which includes ideas and proposals to enhance the students’ CT skills. The feedback report also encourages learners to try new Scratch blocks and structures, in order to improve the “mastery score” of their next projects (see Fig. 6.7). Because of this feature, in the previous sections, we have classified Dr. Scratch as a CT formative–iterative assessment tool.

The ecological validity of Dr. Scratch, for being implemented with positive results in school settings, has been demonstrated (Moreno-León et al., 2015). Furthermore, the convergent validity of Dr. Scratch with respect to the grades provided by CS educators (Moreno-León, Román-González, Hartevelde, & Robles, 2017), and with respect to several software engineering complexity metrics (Moreno-León, Robles, & Román-González, 2016), has been already reported. Finally, Dr. Scratch has also proved discriminant validity to distinguish between different types of Scratch projects, such as animations, art projects, music projects, stories, and games (Moreno-León, Robles, & Román-González, 2017).

Given that the CTt, the Bebras Tasks, and Dr. Scratch are aimed at assessing the same construct (i.e., CT), but they approach this goal from different perspectives, a total convergence ( $r > 0.7$ ) is not expected among them, but a partial one ( $0.4 < r < 0.7$ ) (Carlson & Herdman, 2012).

Thus, the convergent validity of these three instruments was investigated through two different correlational studies, with two independent samples. In the first study, the CTt and the aforementioned selection of Bebras Tasks were concurrently administered to a sample of Spanish middle-school students ( $n = 179$ ), in pure pretest condition (i.e., students without prior formal experience in programming or similar). A positive, moderate, and statistically significant correlation was found ( $r = 0.52$ ). As depicted in Fig. 6.8, the higher the CT ability of the subject (as measured by the CTt) is, the more likely it is that the subject correctly transfers CT to real-life problems (as measured by the Bebras Tasks).

Regarding the second study, it was performed in the context of an 8-week programming course in the Scratch platform, following the *Creative Computing* (Brennan, Balch, & Chung, 2014) curriculum and involving Spanish middle-school students ( $n = 71$ ). Before starting with the course, the CTt was administered to the students in pure pretest condition. After the programming course, students took a posttest with the CTt and teachers selected the most advanced project of each student, which was analyzed with Dr. Scratch. These measures offered us the possibility to analyze the convergent validity of the CTt and Dr. Scratch in predictive terms ( $CTt_{pre-test} * Dr. Scratch$ ) and in concurrent terms ( $CTt_{post-test} * Dr. Scratch$ ). As in the first study, positive, moderate, and statistically significant correlations were found again, both in predictive ( $r = 0.44$ ) and in concurrent terms ( $r = 0.53$ ) (Fig. 6.9).

As we expected, the CTt, the Bebras Tasks, and Dr. Scratch are just *partially convergent* ( $0.4 < r < 0.7$ ). This result is consistent with their different assessment approaches: *diagnostic-aptitudinal* (CTt), *skill transfer* (Bebras Tasks), and *forma-*

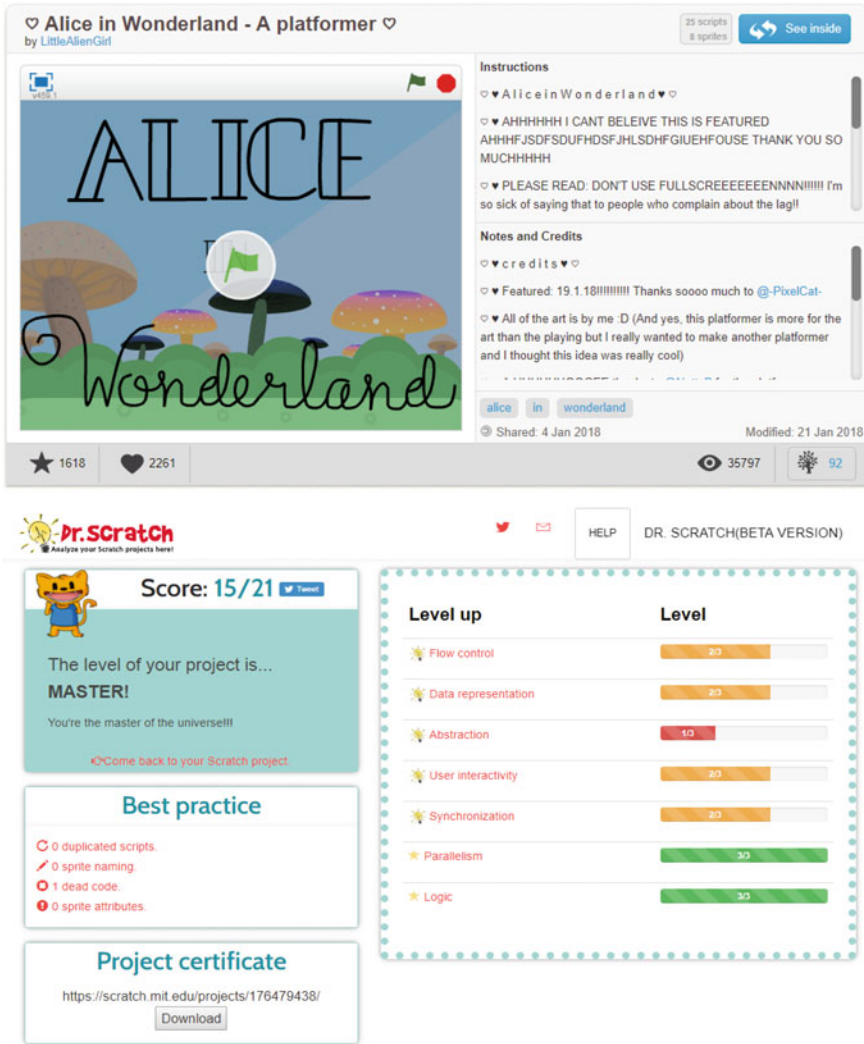


Fig. 6.7 Dr. Scratch assessment results and feedback report (bottom), for the Scratch project “Alice in Wonderland—a platformer” (<https://scratch.mit.edu/projects/176479438/>) (top)

tive-iterative (Dr. Scratch). Precisely, because of that *partial convergence*, we can extract very relevant pedagogical information, especially from the cases in which the expected correlation is not met (i.e., the cases that are especially deviated from the regression line). For example, if we find students with high CT aptitude (as measured by the CTt), but with low or medium scores in their programming projects (as measured by Dr. Scratch), we should probably review how programming is being taught in the classroom, since that teaching could be limiting CT high-ability students by

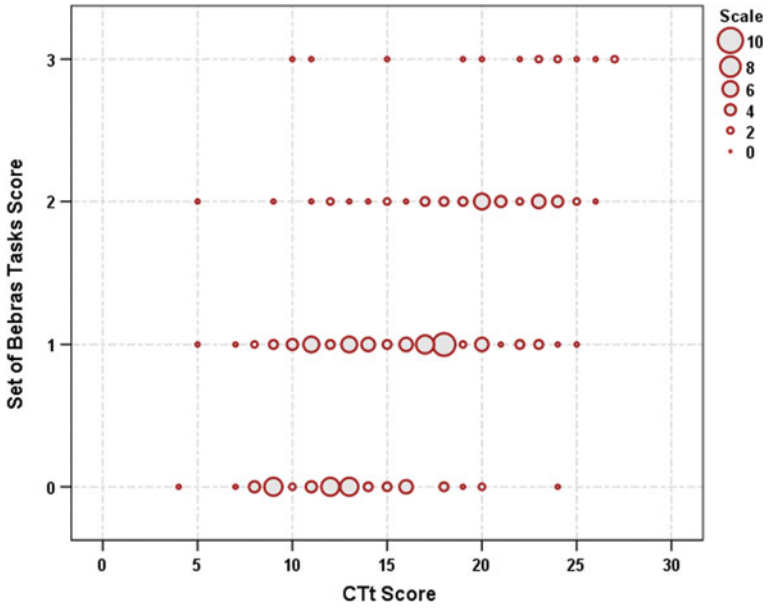


Fig. 6.8 Scatterplot CTt \* set of Bebras Tasks

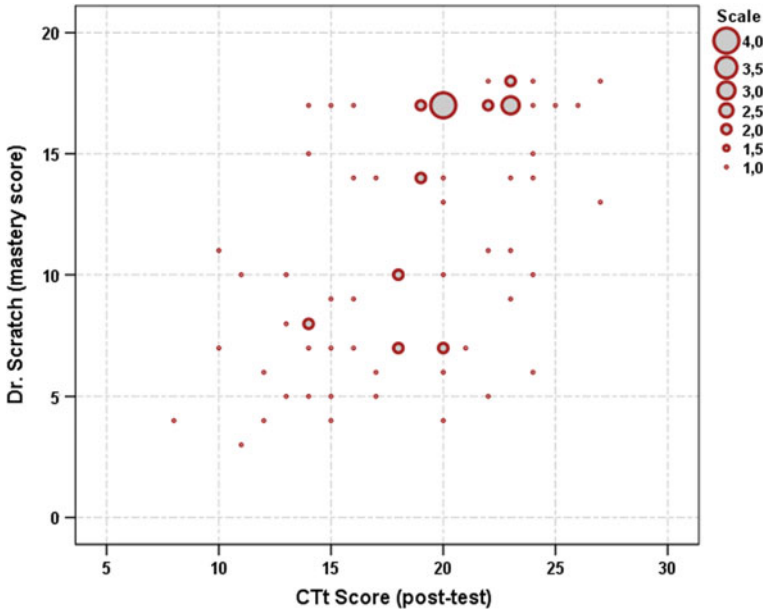


Fig. 6.9 Scatterplot  $CTt_{\text{post-test}}$  \* Dr. Scratch

means of a “*low ceiling*” methodology. Analogous reasoning can be followed for other partial convergences between any considered couple of CT assessment tools. Moreover, our results are one of the first empirical evidences that can contribute to depict a map with the convergence values between the main CT assessment tools all around the world, which ultimately would lead CT to be seriously considered as a psychological construct with its own entity. Further details of the aforementioned convergent validity studies can be found in Román-González, Moreno-León, and Robles (2017a).

## 6.4 A Comprehensive Evaluation of Computational Thinking Interventions

The empirical findings presented in the previous section have some implications. On the one hand, the just partial convergence found implies that none of the CT assessment tools considered in our studies should be used instead of any of the others. In other words, since the scores coming from these different instruments are only moderately correlated, none of the measures can replace or be reduced to any of the others; otherwise, the three tools might be combined in school contexts in order to achieve a more sensitive portrait of the students’ CT skills. On the other hand, from a pedagogical point of view, the three assessment tools empirically studied seem to be theoretically complementary, as the weaknesses of the ones are the strengths of the others.

Thus, the CTt has some strengths, which are common to other diagnostic tools, such as it can be collectively administered in pure pretest conditions, so it could be used in massive screenings and early detection of computationally talented subjects or individuals with special needs in CT education. Moreover, the diagnostic tools (e.g., the CTt), and also the summative ones, can be utilized for collecting quantitative data in pre–post-evaluations aimed at developing CT. However, the CTt and other tools of the same type have some obvious weaknesses too, since they just provide static and decontextualized assessments, which are usually focused on computational “concepts” (Brennan & Resnick, 2012), ignoring “practices” and “perspectives” (see Table 6.1).

As a counterbalance of the above, the Bebras Tasks and other skill transfer tools provide naturalistic assessments, which are contextualized in significant “real-life” problems. Thus, this kind of tools can be used not only as measuring instruments, but also as tasks for teaching and learning CT in a meaningful way. When used strictly for assessment, the skill transfer tools are especially powerful if they are administered after certain time has elapsed from the end of the CT educational intervention, in order to check the degree of retention and transfer of the acquired CT skills. Nevertheless, the psychometric properties of this kind of tasks are still far of being demonstrated, and some of them are at risk of being too tangential to the core of CT.

**Table 6.1** Adequacy of different types of CT assessment tools regarding CT dimensions

	Computational concepts	Computational practices	Computational perspectives
Diagnostic tools	c	a	—
Summative tools	c	a	—
Formative–iterative tools	b	c	a
Data-mining tools	b	c	a
Skill transfer tools	b	a	b
Perceptions–attitudes scales	—	a	c
Vocabulary assessments	—	a	c

<sup>a</sup>Little adequacy, <sup>b</sup>Moderate adequacy, <sup>c</sup>Excellent adequacy, — No adequacy at all

Finally, Dr. Scratch complements the CTt and the Bebras Tasks, given that the former involves “computational practices” (Brennan & Resnick, 2012) that the other two do not, such as iterating, testing, remixing, or modularizing. However, Dr. Scratch, like other formative–iterative or data-mining tools, cannot be used in pure pretest conditions, since it is applied onto programming projects after the student has learnt at least some computer programming for a certain time. In other words, these formative–iterative and data-mining tools are affected by the kind of programming instruction delivered by the teacher, and by the kind of programming projects that are proposed to the students.

In this vein, Table 6.1 shows a tentative proposal on the degree of adequacy between the different types of CT assessment tools with respect to the three CT dimensions stated by Brennan and Resnick (2012) framework: “*computational concepts*” (sequences, loops, events, parallelism, conditionals, operators, and data); “*computational practices*” (experimenting and iterating, testing and debugging, reusing and remixing, abstracting, and modularizing); and “*computational perspectives*” (expressing, connecting, and questioning).

All of the above leads us to affirm the complementarity of the different types of CT assessment tools in educational scenarios, and to raise the clear possibility of building up powerful “systems of assessments” through a proper combination of them. Hence, from a chronological point of view, the degree of adequacy in the use of each type of assessment tools at the different phases of CT educational interventions and evaluations is presented in Table 6.2.

Furthermore, when reflecting about the characteristics of the various and diverse CT assessment tools, it is possible to state what levels of the Bloom’s (revised) taxonomy of cognitive processes are being addressed by each type (Fig. 6.10). Thus, the diagnostic tools provide information about how the students “remember” and “understand” some CT concepts; the skill transfer tools inform about the students’ competence to “analyze” and “apply” their CT skills to different contexts; and the formative–iterative tools allow the students to “evaluate” their own and others’ projects, as well as to “create” better and more complex ones. In addition, the summative tools

**Table 6.2** Chronological uses of the different types of CT assessment tools

	Before the intervention (pretest)	Along the intervention (progression)	Just after the intervention (posttest)	Sometime after the end of the intervention (retention and transfer)
Diagnostic tools	c	–	b	a
Summative tools	a	–	c	b
Formative–iterative tools	–	c	b	–
Data-mining tools	–	c	b	–
Skill transfer tools	a	b	b	c
Perceptions–attitudes scales	c	–	c	a
Vocabulary assessments	a	–	c	b

<sup>a</sup>Little adequacy, <sup>b</sup>Moderate adequacy, <sup>c</sup>Excellent adequacy, – No adequacy at all

serve as a transition between the lower and the intermediate levels of the taxonomy, while the data-mining tools do so between the intermediate and the upper ones. Nonetheless, there might be still some CT issues, such as usability, originality, interface design, etc., which could not be assessed by tools, otherwise by human sensitivity. Finally, since the perceptions–attitudes scales do not assess cognitive processes, but noncognitive ones, these tools are not placed along the pyramidal taxonomy, but they surround and frame it.

As a final contribution, we raise some research designs (RD) that could compose together a comprehensive evaluation model of CT interventions. For each research design, we specify what kind of research question is addressed, and what CT assessment tools should be used:

- **Correlational–predictive RD:** Through this RD, we investigate to what extent the current level of the subject’s CT can explain–predict his/her future performance in related tasks (e.g., his/her academic achievement in STEM subjects, his/her computer programming competence, etc.). The right instruments to be utilized in this kind of RD are the diagnostic tools. Once the predictive power of a CT diagnostic tool has been established, then it can be used subsequently to detect subjects who may need preventive CT educational interventions.
- **Quasi-experimental RD:** This type of RD is aimed at verifying if a certain intervention has been effective to develop the CT of the students. A quasi-experimental RD needs pretest and posttest measures, administered on treatment and control groups, in order to monitor the causes of the changes in those measures, if any. The proper instruments to be used in this RD are the summative tools (and some of the diagnostic tools too). Moreover, it is imperative that the intervention under

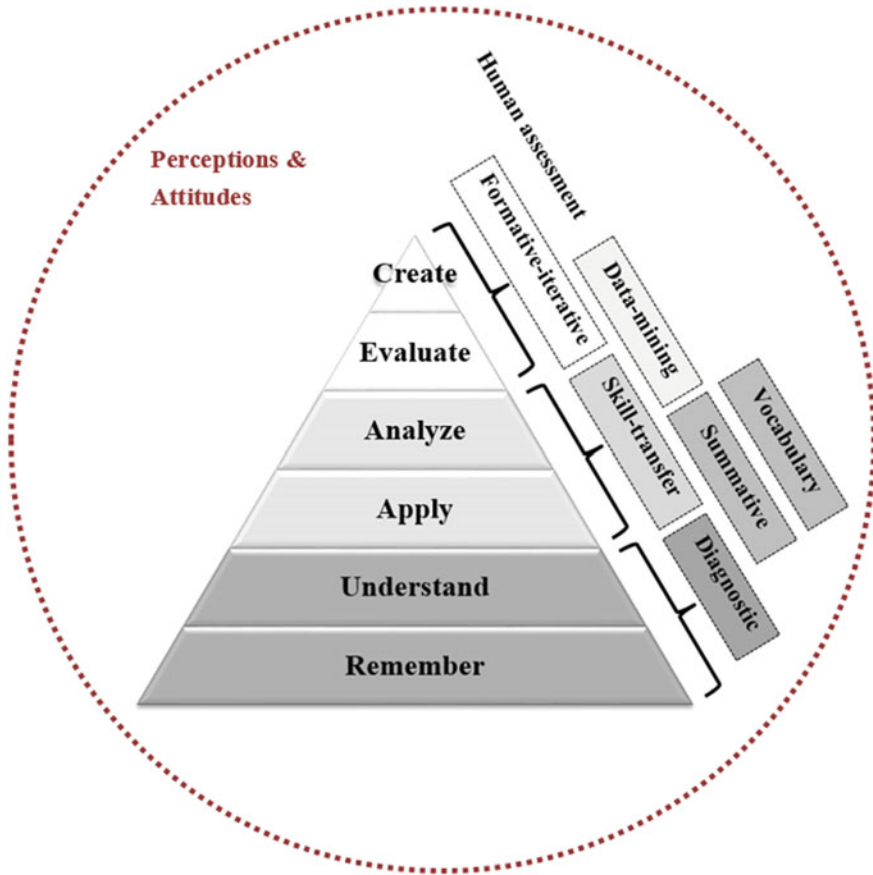


Fig. 6.10 Bloom’s taxonomy and CT assessment tools

evaluation is well-documented, so that the quasi-experiment could be replicated in other contexts, and the effect sizes could be later compared.

- **Causal-comparative RD:** If some modulating variables are introduced in the previous RD, then we can also address the question about what factors affect the correlations and/or differences found. Some of the classical modulating variables are gender, age, programming experience, teacher experience, etc. In addition, it would be extremely relevant to include students and teachers’ perceptions and attitudes as complementary modulating variables, since it is becoming more evident that noncognitive aspects influence the progression and the results of CT interventions (Román-González et al., 2018). Thus, perceptions and attitudes scales should be incorporated in this kind of RD.
- **Longitudinal RD:** Longitudinal studies involve repeated measures of the same variables, coming from the same subjects, and over a certain period of time. Although this kind of RD is quite difficult to perform, due to its high cost and



sample mortality among other things, it provides answer to extremely relevant questions such as how the CT learning paths are, or what barriers and misconceptions tend to appear along them. If the longitudinal RD is performed along a CT intervention, then the formative–iterative and/or the data-mining tools should be used. Else, when this kind of RD is chronologically extended beyond the end of the CT intervention, then skill transfer tools should be included in the research too.

Furthermore, when combining several CT assessment tools in a single RD, then powerful multivariable analysis techniques can be applied to shed some light on some relevant issues: for example, cluster analysis could be performed to state and describe some profiles of students according to their different CT scores; multiple regression or discriminant analysis could be used to explain what factors better explain different CT levels, etc.

In summary, everything exposed in this section may help scholars and policy-makers to decide and perform accurate evaluation designs of CT according to their needs and inquiry goals.

## 6.5 Conclusions and Further Research

Computational thinking (CT) is entering adulthood. Along this process, CT is being demystified and it is being demanded to become a more solid psychological variable in order to assure its survival. In this vein, CT assessment might be a key aspect within the whole CT study field. Because of its own nature, CT assessment can decisively contribute to consolidate CT as a mature construct. Pushing in this direction, in this chapter, we have reviewed the state-of-the-art CT assessment tools; we have presented two convergent validity studies between three of those instruments, which could be a starting point for a map with the convergence values between the main CT assessment tools all around the world; and we have raised a comprehensive evaluation model that could combine properly the different types of instruments according to the inquiry and research goals, and which could guide future actions of scholars and policy-makers.

Further research should focus on completing the “convergence map,” and on performing and replicating the aforementioned prototypical research designs in order to compare their effectiveness in different countries, populations, and educational contexts.

## References

- Basawapatna, A., Koh, K. H., Repenning, A., Webb, D. C., & Marshall, K. S. (2011). Recognizing computational thinking patterns. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education* (pp. 245–250). <https://doi.org/10.1145/1953163.1953241>.
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., Engelhardt, K., et al. (2016). *Developing computational thinking in compulsory education-implications for policy and practice*. Seville: Join Research Center (European Commission). Retrieved from [http://publications.jrc.ec.europa.eu/repository/bitstream/JRC104188/jrc104188\\_computhinkreport.pdf](http://publications.jrc.ec.europa.eu/repository/bitstream/JRC104188/jrc104188_computhinkreport.pdf).
- Brennan, K., Balch, C., & Chung, M. (2014). *Creative computing*. Harvard Graduate School of Education.
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada* (pp. 1–25). Retrieved from <http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf>.
- Carlson, K. D., & Herdman, A. O. (2012). Understanding the impact of convergent validity on research results. *Organizational Research Methods*, 15(1), 17–32.
- Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., & Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computers & Education*, 109, 162–175. <https://doi.org/10.1016/j.compedu.2017.03.001>.
- Dagiene, V., & Futschek, G. (2008). Bebras international contest on informatics and computer literacy: Criteria for good tasks. In *International Conference on Informatics in Secondary Schools-Evolution and Perspectives* (pp. 19–30). Berlin, Germany: Springer. [https://doi.org/10.1007/978-3-540-69924-8\\_2](https://doi.org/10.1007/978-3-540-69924-8_2).
- Daily, S. B., Leonard, A. E., Jörg, S., Babu, S., & Gundersen, K. (2014). Dancing alice: Exploring embodied pedagogical strategies for learning computational thinking. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (pp. 91–96). <https://doi.org/10.1145/2538862.2538917>.
- Durak, H. Y., & Saritepeci, M. (2018). Analysis of the relation between computational thinking skills and various variables with the structural equation model. *Computers & Education*, 116, 191–202. <https://doi.org/10.1016/j.compedu.2017.09.004>.
- Eguiluz, A., Guenaga, M., Garaizar, P., & Olivares-Rodríguez, C. (2017). Exploring the progression of early programmers in a set of computational thinking challenges via clickstream analysis. *IEEE Transactions on Emerging Topics in Computing* (in press). <https://doi.org/10.1109/TETC.2017.2768550>.
- Grover, S. (2011). Robotics and engineering for middle and high school students to develop computational thinking. In *Annual Meeting of the American Educational Research Association, New Orleans, LA*. Retrieved from <https://pdfs.semanticscholar.org/69a7/c5909726eed5bd66719aad69565ce46bbdce.pdf>.
- Grover, S. (2015). “Systems of assessments” for deeper learning of computational thinking in K-12. In *Proceedings of the 2015 Annual Meeting of the American Educational Research Association* (pp. 15–20). Retrieved from [https://www.sri.com/sites/default/files/publications/aera2015\\_-\\_systems\\_of\\_assessments\\_for\\_deeper\\_learning\\_of\\_computational\\_thinking\\_in\\_k-12.pdf](https://www.sri.com/sites/default/files/publications/aera2015_-_systems_of_assessments_for_deeper_learning_of_computational_thinking_in_k-12.pdf).
- Grover, S., Basu, S., Bienkowski, M., Eagle, M., Diana, N., & Stamper, J. (2017). A framework for using hypothesis-driven approaches to support data-driven learning analytics in measuring computational thinking in block-based programming environments. *ACM Transactions on Computing Education (TOCE)*, 17(3), 14:1–14:25. <https://doi.org/10.1145/3105910>.
- Grover, S., Bienkowski, M., Niekrasz, J., & Hauswirth, M. (2016). Assessing problem-solving process at scale. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale* (pp. 245–248).
- Grover, S., & Pea, R. (2013). Computational thinking in K–12 a review of the state of the field. *Educational Researcher*, 42(1), 38–43. <https://doi.org/10.3102/0013189X12463051>.

- Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, 4(3), 583–596. Retrieved from [http://www.bjmc.lu.lv/fileadmin/user\\_upload/lu\\_portal/projekti/bjmc/Contents/4\\_3\\_15\\_Kalelioglu.pdf](http://www.bjmc.lu.lv/fileadmin/user_upload/lu_portal/projekti/bjmc/Contents/4_3_15_Kalelioglu.pdf).
- Koh, K. H., Basawapatna, A., Bennett, V., & Repenning, A. (2010). Towards the automatic recognition of computational thinking for adaptive visual language learning. In *2010 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 59–66). <https://doi.org/10.1109/VLHCC.2010.17>.
- Koh, K. H., Basawapatna, A., Nickerson, H., & Repenning, A. (2014). Real time assessment of computational thinking. In *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 49–52). <https://doi.org/10.1109/VLHCC.2014.6883021>.
- Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior*. <https://doi.org/10.1016/j.chb.2017.01.005>.
- Kukul, V., Gökçearslan, S., & Günbatır, M. S. (2017). Computer programming self-efficacy scale (CPSES) for secondary school students: Development, validation and reliability. *Eğitim Teknolojisi Kuram ve Uygulama*, 7(1). Retrieved from <http://dergipark.ulakbim.gov.tr/etku/article/view/5000195912>.
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61. <https://doi.org/10.1016/j.chb.2014.09.012>.
- Maiorana, F., Giordano, D., & Morelli, R. (2015). Quizly: A live coding assessment platform for App Inventor. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)* (pp. 25–30). <https://doi.org/10.1109/BLOCKS.2015.7368995>.
- Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with scratch. *Computer Science Education*, 23(3), 239–264.
- 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*, 15(46), 1–23. Retrieved from [http://www.um.es/ead/red/46/moreno\\_robles.pdf](http://www.um.es/ead/red/46/moreno_robles.pdf).
- Moreno-León, J., Robles, G., & Román-González, M. (2016). Comparing computational thinking development assessment scores with software complexity metrics. In *2016 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1040–1045). <https://doi.org/10.1109/EDUCON.2016.7474681>.
- Moreno-León, J., Robles, G., & Román-González, M. (2017). Towards data-driven learning paths to develop computational thinking with scratch. *IEEE Transactions on Emerging Topics in Computing*. <https://doi.org/10.1109/TETC.2017.2734818>.
- Moreno-León, J., Román-González, M., Hartevelde, C., & Robles, G. (2017). On the automatic assessment of computational thinking skills: A comparison with human experts. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (pp. 2788–2795). New York, NY, USA: ACM. <https://doi.org/10.1145/3027063.3053216>.
- Mühling, A., Ruf, A., & Hubwieser, P. (2015). Design and first results of a psychometric test for measuring basic programming abilities. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (pp. 2–10). <https://doi.org/10.1145/2818314.2818320>.
- Ota, G., Morimoto, Y., & Kato, H. (2016). Ninja code village for scratch: Function samples/function analyser and automatic assessment of computational thinking concepts. In *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 238–239).
- Román-González, M. (2015). Computational thinking test: Design guidelines and content validation. In *Proceedings of the 7th Annual International Conference on Education and New Learning Technologies (EDULEARN 2015)* (pp. 2436–2444). <https://doi.org/10.13140/RG.2.1.4203.4329>.
- Román-González, M., Moreno-León, J., & Robles, G. (2017a). Complementary tools for computational thinking assessment. In S. C. Kong, J. Sheldon, & K. Y. Li (Eds.), *Proceedings of International Conference on Computational Thinking Education (CTE 2017)* (pp. 154–159).

- The Education University of Hong Kong. Retrieved from <http://www.eduhk.hk/cte2017/doc/CTE2017Proceedings.pdf>.
- Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017b). Which cognitive abilities underlie computational thinking? Criterion validity of the computational thinking test. *Computers in Human Behavior*, 72, 678–691. <https://doi.org/10.1016/j.chb.2016.08.047>.
- Román-González, M., Pérez-González, J.-C., Moreno-León, J., & Robles, G. (2018). Extending the nomological network of computational thinking with non-cognitive factors. *Computers in Human Behavior*, 80, 441–459. <https://doi.org/10.1016/j.chb.2017.09.030>.
- Rushkoff, D. (2010). *Program or be programmed: Ten commands for a digital age*. New York: OR Books.
- Sadin, É. (2015). *La vie algorithmique: Critique de la raison numérique [The Algorithmic Life: A Critique of Digital Rationality]*. Paris: L'Echappée.
- Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142–158. <https://doi.org/10.1016/j.edurev.2017.09.003>.
- Weintrop, D., & Wilensky, U. (2015). Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. In *ICER* (Vol. 15, pp. 101–110).
- Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012). The fairy performance assessment: Measuring computational thinking in middle school. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (pp. 215–220).
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35. <https://doi.org/10.1145/1118178.1118215>.
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717–3725. <https://doi.org/10.1098/rsta.2008.0118>.
- Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education*, 14(1). <https://doi.org/10.1145/2576872>.
- Zur-Bargury, I., Párv, B., & Lanzberg, D. (2013). A nationwide exam as a tool for improving a new curriculum. In *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education* (pp. 267–272).

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

