Relkin, E., Johnson, S. K., & Bers, M. U. (2023). A Normative Analysis of the TechCheck Computational Thinking Assessment. *Educational Technology & Society*, *26*(2), 118-130. https://doi.org/10.30191/ETS.202304_26(2).0009

A Normative Analysis of the TechCheck Computational Thinking Assessment

Emily Relkin[1*], Sara K. Johnson[2] and Marina U. Bers[3]
[1]Center for Children and Technology, Education Development Center, USA // [2]Eliot-Pearson Department of Child Study and Human Development, Tufts University, USA // [3]Lynch School of Education and Human Development, Boston College, USA // erelkin@edc.org // s.johnson@tufts.edu // marina.bers@bc.edu
[*]Corresponding author

ABSTRACT: *TechCheck* is an assessment of Computational Thinking (CT) for early elementary school children consisting of fifteen developmentally appropriate unplugged challenges that probe six CT domains. The first version of *TechCheck* showed good psychometric properties as well as ease of administration and scoring in a validation cohort of 768 children between 5 and 9 years of age. To increase sensitivity and reduce possible ceiling and floor effects, grade-specific versions of *TechCheck* (K, 1, 2) were subsequently created. In the present study, we explored how CT skills could be compared across grades when grade-specific versions of *TechCheck* are administered. First, we examined *TechCheck* raw score distributions and responses within CT domains in a representative sample of students from the three grades. Grade-specific Z-scores and percentile rankings were then calculated. To show utility of this normalization system, we used percentiles to compare CT outcomes between first and second graders who participated in a ScratchJr coding educational intervention. While *TechCheck* change scores suggested an unexpected 42.74% difference in CT outcomes between first and second grade, application of the normative scoring system indicated a more plausible 5.17 percentile rank difference between grades. Normative analysis may provide a more meaningful way to compare results across grades when grade-specific versions of *TechCheck* are used. Implications for the future use of the *TechCheck* CT assessments are discussed.

Keywords: Assessment, Computer science, Early childhood, Coding

1. Introduction

Computer Science (CS) is an integral part of early childhood education around the world (Fraillon et al., 2018; Hubwieser et al., 2015; White House, 2016). Children as young as preschool age are capable of learning to code with developmentally appropriate platforms (Clements & Gullo, 1984; Papadakis, 2021). One of the most important goals of teaching computer science (CS) to young children is to promote the development of computational thinking (CT) skills that allow for framing and solving problems using computers and other technologies. Acquiring CT skills is not limited to increasing CS knowledge but also can promote skills relevant to other disciplines, problem-solving, and self-expression in everyday life (Barr & Stephenson, 2011; Chen et al., 2017; Wing, 2010, Wing, 2006). There has been increasing interest in CT with many attempts to further define the concept, implement educational initiatives, and to create novel forms of assessment (Bakala et al., 2021; Lye & Koh, 2014; Román-González et al., 2019; Tang et al., 2020; Zhang & Nouri, 2019). Despite these efforts, most CT definitions do not take into account the context of early childhood. To address this gap, Bers (2018) developed a framework consisting of seven powerful ideas from Computer Science that are developmentally appropriate for children ages 4-9. These include the following domains: hardware/software, algorithms, modularity, control structures, representation, debugging, and design process.

There is a recognized need for well-designed and validated CT assessments for young children that can be easily administered in classroom and online settings (Grover & Pea, 2013; Lee et al., 2011; Poulakis & Politis, 2021; Román-González et al., 2019). An ideal CT assessment can be used to monitor students' progress in learning CT and allow educators to gauge the effectiveness of their lessons and CS curricula. CT assessment can be used to identify students in need of extra support as well as those with exceptional talents (Relkin et al., 2021; Román-González et al., 2019). In the context of research, it can provide new insights into how children's CT abilities develop and can assist in the development of new curricula and best practices for CS education (Zhang & Nouri, 2019). Various CT assessments for early childhood education have been created but are not always well-characterized. Tang et al. (2020) reported that of the 96 CT assessment studies analyzed (including all ages), only 45% reported reliability measures and only 18% reported validity evidence. The majority of CT assessments with validity evidence were designed for older students. Prior work with older children has helped researchers and educators identify the elements of CT amenable to assessment in early childhood (e.g., Werner et al., 2012), establish the utility of unplugged CT challenges (e.g., Román-González et al., 2018) and demonstrate the applicability of item response theory for measuring the psychometric properties of CT assessments (e.g., Chen et al., 2017; Kong & Lai, 2022).

1.1. Assessments of CT for young children

Instruments for assessing CT in older students and adults have existed for some time (Chen et al., 2017; Fraillon et al., 2018; Werner et al., 2012). Many of these instruments are not developmentally appropriate for young children. A common assessment approach involves the use of coding exercises that are designed to elicit the same type of logic and reasoning that is involved in programming. However, coding-based assessments require prior knowledge of a coding language and can conflate coding ability with CT skills (Yadav et al., 2017). Assessments that require knowledge of coding cannot readily be used to assess baseline CT abilities in coding-naive students. In addition, research with older children has indicated that coding can become automatic and coding exercises may therefore not effectively probe CT (Werner et al., 2014).

There have been several attempts to create CT assessments for early childhood. Many CT instruments designed for early age groups utilize portfolio analysis, including interviews and/or observational methods (Bakala et al., 2021). For example, Mioduser and Levy (2010) used pre-programmed LEGO robotics construction tasks which they presented to kindergarten-age children. The children's CT level was qualitatively assessed by analyzing the terms that children used to describe the robot's actions as it navigated through a constructed environment. Children who attributed the robot's actions to magic or personification were given low CT skills ratings and those who provided mechanical explanations were considered more advanced. Wang et al. (2014) used a similar approach with 5-to-9-year-old children, who were asked open-ended questions about a tangible programming task that they created called "T-maze." "T-maze" uses TopCode to convert physical programs into digital code (Horn, 2012). The researchers identified elements of CT in the children's responses (e.g., abstraction, decomposition) as a basis for determining whether the children grasped these concepts. Bers et al., (2014) created a checklist to assess programs created by kindergarteners (ages 4.9 to 6.5 years old) exposed to a tangible and graphical programming language called CHERP (Creative Hybrid Environment for Robotics Programming). During one session, children were tasked with programming their robot to dance the "Hokey Pokey." The researchers then assessed four CT concepts by scoring children's projects on a Likert scale. Moore et al. (2020) used task and interview techniques to assess CT. Three participants were videotaped while they were interviewed and performed tasks using the Code and Go Robot Mouse Coding Activity (Learning Resources, Vernon Hills, IL). Researchers explored qualitatively how children use representations and translations to invent strategies for solving problems. Portelance and Bers (2015) conducted an exploratory study that assessed CT in young children by analyzing ScratchJr artifact-based video interviews of students in pairs. Researchers then analyzed videos of the dyads using holistic coding to identify categories.

Some effort has been put into creating activity-based CT assessments for young children. Marinus et al. (2018) created the Coding Development (CODE) Test 3–6 (for children between 3 and 6 years of age), which uses the robot Cubetto. CODE requires children to program the robot to go to a specified location on a mat by inserting wooden blocks into a "remote control." The task is to either build the program from scratch or debug an existing program. Children are given maximally three trials to complete each of the 13 items, with more points being awarded if fewer attempts are needed. Although the authors state that CODE is meant to measure CT, their assessment requires coding knowledge raising the possibility that their assessment conflates coding with CT skills. Clarke-Midura et al. (2021) are in the development stage of attempting to use evidence-centered design to develop a task-based assessment of CT for kindergarten-age children.

It is advantageous to be able to measure CT skills in children regardless of whether they have past knowledge or experience with computer programming (Grover et al., 2014). With this in mind, researchers began exploring the use of code-free instruments to assess CT skills in children. CT is exercised in the context of many "unplugged" activities (Bell & Vahrenhold, 2018; Zapata-Cáceres et al., 2020). Unplugged activities involve puzzles, games and exercises that exemplify CS concepts without requiring knowledge of coding or the use of computers. An unplugged activity typically involves a set of artifacts and procedures that are well-known to most school-age children. Unplugged activities have been used to teach CS concepts for over two decades (e.g., CSUnplugged.com; code.org), and in recent years have started to be used for the purposes of assessment. It has been argued that the unplugged assessments offer advantages because they do not rely on a particular computer language or curricula and are therefore purer reflections of CT abilities (Dagienė & Futschek, 2008).

Studies were published in 2018, 2020, 2021, and 2022 on five different unplugged CT assessments designed specifically for young children. The CTt for Beginners (BCTt) (Zapata-Cáceres et al., 2020), The Competent Computational Thinking Test (cCTt) (El-Hamamsy et al., 2022), *TechCheck* (Relkin et al., 2020), the

Computerized Adaptive Programming Concepts Test (CAPCT) (Hogenboom et al., 2021), and the Computational Thinking Assessment (CTA) (Tran, 2018). All four use unplugged challenges to probe CT domains and can be administered to children who lack prior coding experience. These instruments differ in the types of unplugged challenges they include, the CT domains assessed, the age ranges they cover, and the time required to complete and score the respective assessments (see Table 1). Some of the concepts probed by the BCTt, the CAPCT, and the CTA such as complex conditionals may be problematic for younger children on developmental grounds (Barrouillet & Lecas, 1999; Janveau-Brennan & Markovits, 1999; Muller et al., 2001). In addition, the CAPCT and the CTA require more advanced language and mathematical skills than typical K-2 students possess.

Table 1. A comparison of four unplugged CT measures for young children

	The CTt for Beginners (BCTt)	The Competent Computational Thinking Test (cCTt)	*TechCheck*	Computerized Adaptive Programming Concepts Test (CAPCT)	Computational Thinking Assessment (CTA)
CT Concepts	Sequences, Loops (Simple, Nested), Conditionals (If-Then, If-Then-Else, While)	Sequences, Loops (Simple, Nested), Conditionals (If-Then, If-Then-Else, While)	Algorithms, Modularity, Debugging, Hardware/Software, Control Structures, Representation	Basic Sequences, Loops, Conditions (If & If-Else Statements), Debugging, Multiple Agents, Procedures, Generalization	Sequences, Algorithms, Loops, Debugging, Conditionals
Format Type	Pen and paper Multiple choice	Pen and paper Multiple choice	Pen and paper Online Multiple choice	Online Adaptive	Pen and paper Yes/No Prose responses
Items	25 items	25 items	15 items	4486 items (utilizes alternative forms of the same items)	10 items
Administrator Needed	Yes	Yes	Yes	No	No
Average Testing Time	40 minutes	30-35 minutes	13 minutes	Children play for as long as they want	6-10 minutes
Sample	299 students	1519 students	1844 students	93,341 students	183 students
Age Range	5-12 (1- 6th grade)	7-9 (3-4th grade)	3-9 (PreK – 2nd grade)	6-13 (1– 7th grade)	N/A (3rd grade) Not yet validated in younger children

Note. The CTt for Beginners (BCTt) (Zapata-Cáceres et al., 2020), The Competent Computational Thinking test (cCTt) (El-Hamamsy et al., 2022); *TechCheck* (Relkin et al., 2020), the Computerized Adaptive Programming Concepts Test (CAPCT) (Hogenboom et al., 2021), the Computational Thinking Assessment (CTA) (Tran, 2018).

1.2. Design, validation, and implementation of the original *TechCheck* assessment

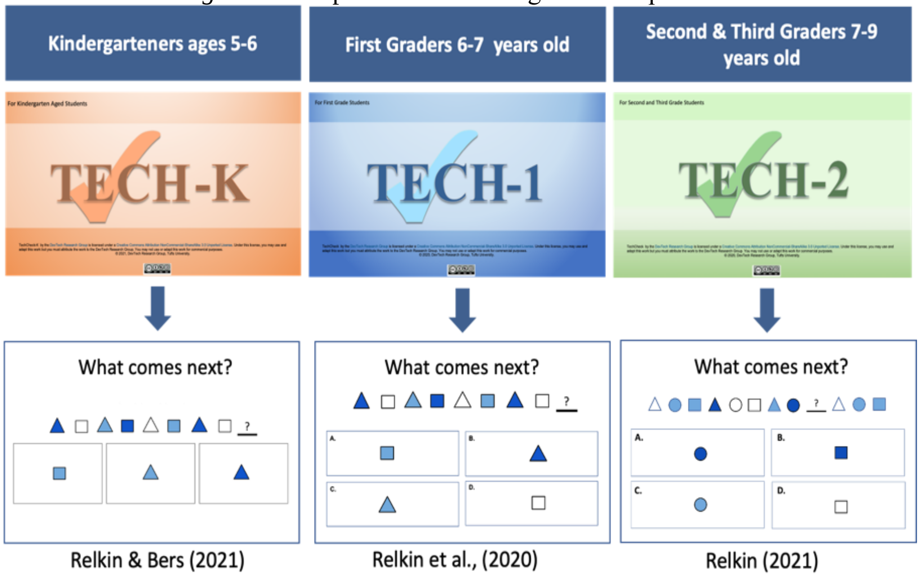
TechCheck, an unplugged CT assessment for young children, was developed based on six of the seven powerful ideas of CS put forth by Bers (2018) (Relkin et al., 2020). The excluded powerful idea, Design Process, is an iterative and open-ended process that does not lend itself to a short multiple-choice assessment. *TechCheck* was initially tested in a cohort of 768 first and second graders (ages 5-9) participating in a research study involving the CAL-KIBO curriculum. *TechCheck* showed good reliability and validity according to classical test theory

(CTT) and item response theory (IRT), models that are commonly used to better understand the relationship of assessment items to the underlying concepts being measured (Kingsbury & Weiss, 1983). The mean difficulty index of all items was − 1.25 (range = − 2.63, .7), the mean discrimination index was 1.03 (range = 0.65, 1.41). The coefficient alpha indicated a moderate reliability ($\alpha = 0.68$) (Hinton et al., 2004). The assessment scores were normally distributed, and the assessment readily distinguished among young children with different CT abilities. *TechCheck* scores correlated moderately ($r = .53$, $p < .001$) with a previously validated CT assessment tool called TACTIC-KIBO (Relkin & Bers, 2019).

1.3. Grade-specific *TechCheck* versions

There are now three grade specific versions of *TechCheck TechCheck-K*, *TechCheck-1*, and *TechCheck-2* that are optimal for kindergarten, first and second/ third graders respectively (See Figure 1). When *TechCheck-1* was administered to kindergarten students, it became apparent that certain modifications were required. Previous research has shown that the working memory of children of kindergarten age (~5 years old) limits them to hold an average of three items in immediate memory, compared to children in first and second grade (~6-9 years old) who can hold an average of four items (Cowan, 2016; Simmering, 2012). This limit can potentially impact kindergartener's performance on multiple-choice assessments. Consequently, the number of response options was reduced from four to three in *TechCheck-K* (the kindergarten version). *TechCheck-K* was administered to $N = 89$ 5-6-year-old students and the percentage of correct responses for each item on *TechCheck-K* closely paralleled that observed with *TechCheck-1*. We also noted a strong and significant correlation between the percentages correct on the two versions ($r = 0.76$, $p < .001$) (Relkin & Bers, 2021).

Figure 1. Sample *TechCheck* "algorithms" questions



To create a version of *TechCheck* with improved psychometric properties for second graders, an item analysis of all the *TechCheck-1* questions was conducted. Questions that had low difficulty, discrimination, and/or point biserial correlations were modified (Relkin, 2021). *TechCheck-2* was administered to $N = 63$ second graders. The level of difficulty was increased in this version to mitigate a previously observed ceiling effect found when second grade students took *TechCheck-1*. *TechCheck*-2 readily distinguished among young children with different CT abilities. Item equivalency to the original version of *TechCheck* was confirmed and the coefficient alpha was slightly higher ($a = 0.74$) than with the original assessment, (*TechCheck-1*). A paired sample *t*-test between baseline *TechCheck* and endpoint *TechCheck* was significant $t = 4.01$, $df = 62$, $p <. 0001$.

1.4. The present study

Among existing CT measures, there has been relatively little attention paid to methods for comparing CT skills across grades. The ability to perform cross-grade comparisons of CT skills is essential for assessing the applicability of CS curricula and coding platforms to specific age groups. The present study examines baseline performance on three versions of *TechCheck* (*TechCheck-K, TechCheck-1*, and *TechCheck-2* respectively) and applies item analysis to identify differences between these three grade-specific versions. Normalization using Z-scores and percentile ranks was then introduced to allow comparison of performance on *TechCheck* across three grades (K,1,2). This study was conducted to answer the research question:

How can CT skills be compared across grades K, 1, and 2 when using three grade-specific versions of *TechCheck*?

2. Method

2.1. Participants

To examine the distribution of *TechCheck* scores across grades, we collected data from children in grades K-2 located in six different states across the USA. All assessments were administered prior to initiation of any formal coding instruction. Table 2 summarizes the demographic information for participants by their grade. Altogether, 1948 students were included in this analysis. A total of *n* = 395 kindergarteners, *n* = 935 first graders, and *n* = 618 second graders participated. The average student age was 6.64, *SD* = .84 with a minimum age of 4 and a maximum age of 9. There were also similar numbers of males (*n* = 725) and females (*n* = 728) in the three grades. Of the *n* = 1399 students from which we obtained race/ethnicity information, the most common race/ethnicity was White (58.89%) followed in frequency by Hispanic/ Latino (15.08%), Black (14.87%), Biracial/Multiracial (5.72%), Asian/Pacific Islander (4.15%%), and other (1.30%) respectively. The group characterized as "other" consisted of children identified as American Indian, Alaskan Native, Pacific Islander or Native Hawaiian.

To explore the utility of the normalized scoring technique for comparing CT performance across grades we analyzed data we collected in a longitudinal study carried out in the states of California, Minnesota, and Arkansas involving administration of a coding curriculum called CAL-ScratchJr to a total of *n* = 163 students in kindergarten, first, and second grade (Bers et al., in press).

Table 2. Demographics of pilot study participants by grade

	Kindergarten	First Grade	Second Grade
Number of students	395	935	618
Mean Age (SD)	5.86 (.42)	6.50 (.56)	7.81 (.35)
Missing data	55	185	299
Gender			
Male	164	399	162
Female	171	400	157
Missing data	60	136	299
Race			
Black/African American	34	154	20
Hispanic/ Latino	42	113	56
Biracial/Multiracial	23	42	15
White	220	399	206
Asian	12	30	16
Other	4	8	6
Missing data	60	190	299

Note. There is missing demographic data because some schools only shared limited information.

2.2. Procedure

This study was initiated prior to the onset of the COVID-19 pandemic but was completed while the pandemic was in progress. As a consequence, different formats of administration of the *TechCheck* assessments were used over the course of this study. Some children were assessed in person while others participated virtually. Some assessments were carried out in group settings and others were conducted one-on-one. Some participants provided responses on paper while others used an online survey platform.

Regardless of the format, administrators were trained and certified to administer the assessment in a consistent fashion. Across all formats of administration, each question was read out loud to the students by an administrator who asked them to provide a single answer from a set of multiple-choice responses. There were two practice questions that were included at the beginning of the assessment to ensure that children felt comfortable with the format of administration and knew how to indicate their answers. Students were allowed to take breaks for up to 5 minutes during the assessment. Students were instructed to guess if they did not know the answer. Administrators were instructed to indicate any abnormal issues that occurred during testing in an error log. That information was later used to clean the data.

2.3. Data analysis

All statistical analyses and plots were conducted and created using R Studio version 1.2 (R Core Team, 2019) and Microsoft Excel version 16.23. Only students' baseline scores prior to receiving the CAL-ScratchJr coding curriculum were used in this analysis (Bers et al., in press). Descriptive statistics as well as data screening was conducted to examine assumptions for normality and linearity. A one-way ANOVA was conducted to explore differences between the three versions of the *TechCheck* assessments. Crossed random effects multi-level models were estimated to examine the relationship between domain specific scores and grade. Lastly, normalization was applied using Z-scores and percentile ranks.

3. Results

3.1. Descriptive statistics

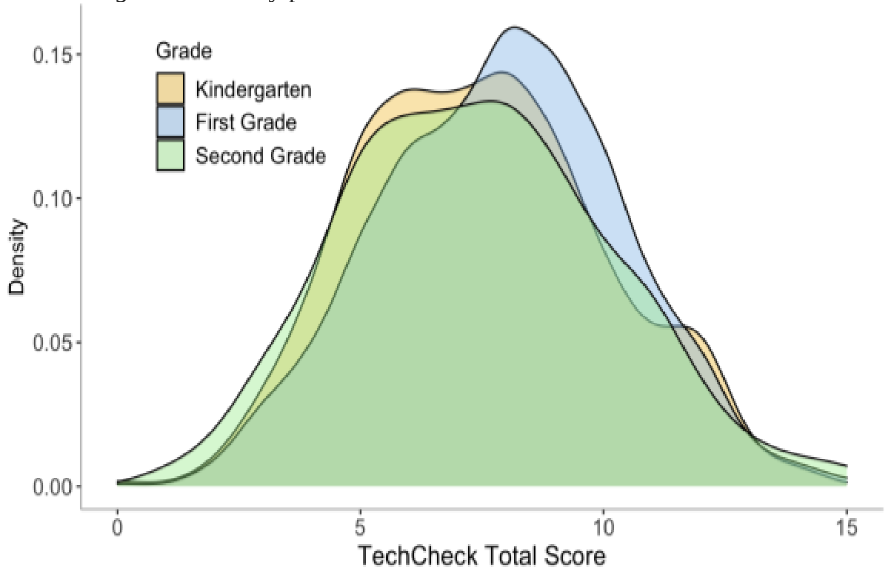
The average total scores were *M* = 7.48 (*SD* = 2.52) on *TechCheck-K*, *M* = 7.98 (*SD* = 2.46) on *TechCheck-1*, and *M* = 9.29 (*SD* = 2.75) on *TechCheck-2* out of a possible 15 points correct. Skewness and kurtosis values were within |2| ranging from -0.41 to 0.20 indicating the distributions were approximately normal for all three versions of the assessment (See Table 3). Density plots of each grade/assessment type also showed normal distributions with the second grade's cohort appearing to have a slightly more rightward skew than the other two grades (see Figure 2). Examination of Z- scores for kindergarten, first, and second grade revealed no extreme outliers of |3| or greater (Iglewicz & Hoaglin, 1993).

Table 3. Descriptive statistics for continuous variables

	N	Mean (*SD*)	Min	Max	Skewness	Kurtosis
TechCheck-K	395	7.48 (2.52)	0	14	0.19	-0.41
TechCheck-1	935	7.98(2.46)	0	15	0.02	-0.25
TechCheck-2	618	7.48(2.75)	0	15	0.20	-0.23

Note. Data in this table reflect scores on each version of *TechCheck* when administered to students prior to coding instruction.

Figure 2. Density plots of the three *TechCheck* assessments



Note. Density plots of total scores for *TechCheck-K* (orange, Kindergarten), *TechCheck-1* (blue, First grade), and *TechCheck-2* (green, Second grade).

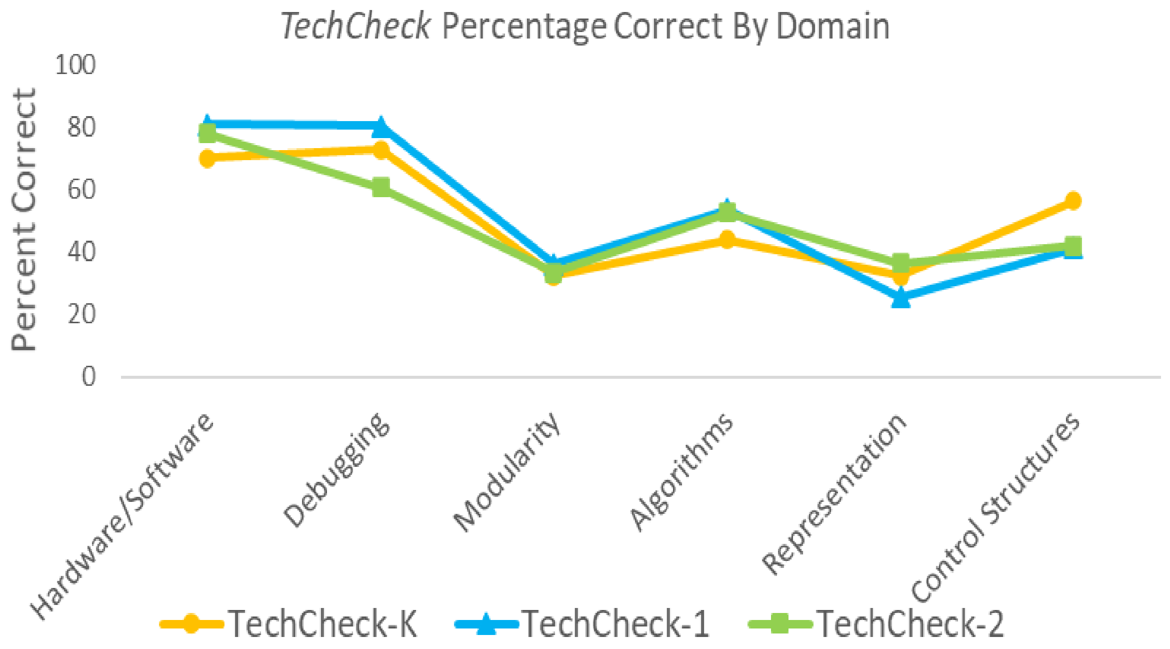
3.2. Differences between versions

To explore if the three grades' scores were significantly different from one another, a one-way ANOVA was conducted. This analysis showed baseline *TechCheck* scores were significantly different across grades $F(2, 1841) = 7.71$, $p < .001$. Tukey's HSD post-hoc test revealed a significant difference between first grade and kindergarteners ($d = .42$, $p = .02$) as well as between first grade and second grade ($d = 0.49$, $p < .001$).

To examine the possible basis for the observed differences between grades, item analysis was carried out by calculating percent correct responses within groups of questions corresponding to the six CT domains measured by *TechCheck*. When comparing the percentage correct at baseline for kindergarten, first, and second grade, the pattern of response is similar across the majority of domains (see Figure 3).

To establish whether the scores were statistically different within and across domains, we conducted a crossed random-effects multilevel model using REML estimation. In this model, CT domain and child were crossed and grade was a fixed effect predicting the CT domain score. First, we used an empty model with percent of questions correct within each CT domain as the outcome variable and a random effect of CT domain. The Intra-Class Correlation (ICC) was .37, which indicates that about 37% of the variation in CT domain percent correct was between domains (with the remaining percentage being differences between students across domains). A random effect of the intercept for the student variable was subsequently added to the model. The deviance significantly decreased, and the likelihood ratio test was significant, indicating the model with a random effect for both domain and student had a better fit ($\Delta\chi^2(1) = 90.68$, $p < .0001$) (in other words, there were overall differences between students when considering all domains together). Lastly, we added the predictor of grade (type of assessment administered). This addition significantly decreased deviance in the model ($\Delta\chi^2(1) = 16.59$, $p < .0001$), indicating that there is a difference across domains by grade. Upon examining the random effects, between-subjects variance attributable to the student and domain was .01 and .23 respectively. This indicates CT domain had a moderate contribution to the total variance (see Table 12). ICC of the final model was .36 suggesting approximately 36% of the variation is between domains.

Figure 3. Three versions of *TechCheck* percentage correct by CT domain



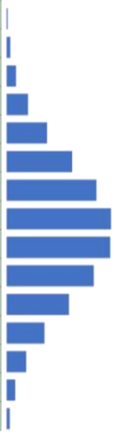
Note. Figure 3 shows the pattern of baseline responses between *TechCheck-K* (orange circle), *TechCheck-1* (blue triangle), *and TechCheck-2* (green square)

3.3. Normalization of scores

To permit comparison of *TechCheck* scores across the three grades, normalization techniques were carried out using Z-scores and percentile ranks. Results are shown in Figure 4. Consistent with the findings presented above, Z-scores and percentile scores appear similar in kindergarten and first grade but differ in second grade.

Figure 4. Z-scores and percentile rank norming for each of the three assessments/grades

	Z-score			Percentile Rank		
***TechCheck* Raw Score**	**Kindergarten**	**First**	**Second**	**Kindergarten**	**First**	**Second**
1	-2.56	-2.82	-2.59	0.52	0.24	0.48
2	-2.17	-2.4	-2.2	1.5	0.82	1.39
3	-1.77	-2	-1.81	3.84	2.28	3.51
4	-1.38	-1.59	-1.42	8.38	5.59	7.78
5	-0.98	-1.18	-1.03	16.35	11.9	15.15
6	-0.59	-0.78	-0.64	27.76	21.77	26.11
7	-0.19	-0.36	-0.25	42.46	35.94	40.13
8	0.21	0.04	0.14	58.32	51.59	55.57
9	0.6	0.45	0.53	72.57	67.36	70.19
10	1	0.86	0.92	84.13	80.51	82.12
11	1.39	1.27	1.31	91.77	89.8	90.49
12	1.79	1.67	1.69	96.41	95.25	95.45
13	2.18	2.08	2.08	98.54	98.12	98.12
14	2.58	2.49	2.47	99.51	99.36	99.32
15	2.97	2.9	2.86	99.85	99.81	99.79

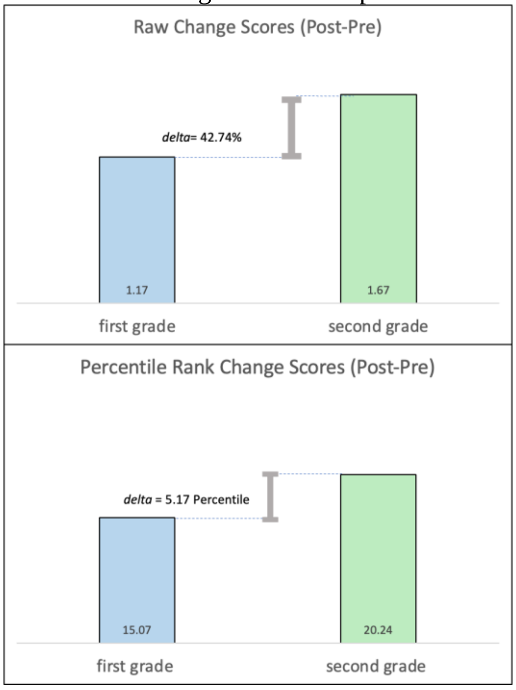


Note. Z scores and Percentiles were calculated for each possible score (0-15 points) on *TechCheck-K*, *TechCheck-1*, and *TechCheck-2*. These scores can be used to compare scores from those who took the different versions of *TechCheck*.

3.4. Field test of normalized scoring system

In a ScratchJr pilot longitudinal study (Bers et al., in press), students were observed to improve significantly on *TechCheck* after exposure to a coding curriculum. First graders' average percentile at baseline was improved by 1.17 points on *TechCheck-1* (baseline score = 7.81, end point score = 8.98 raw points) while second graders' scores increased by 1.67 points on *TechCheck*-2 (baseline score = 9.29, end post score = 10.95 raw points). Direct comparison of these mean change score results could be interpreted as showing a 42.74% greater change in second versus first graders. However, when expressed in terms of percentile changes using the normative scoring system first graders scored at the 43.40 percentile at baseline and at the 49.48 percentile after engagement in the CAL-ScratchJr curriculum (*delta* = 15.07). Second grade students scored at the 59.47 percentile at baseline and 73.02 percentile at the endpoint (*delta* = 20.24). Thus, first graders and second graders differed in outcome by only 5.17 percentile ranks when *TechCheck* score distributions were taken into account (See Figure 5).

Figure 5. Raw change score vs normalized change scores for a pilot CAL-ScratchJr pilot longitudinal study



4. Discussion

TechCheck was originally developed to fulfill the need for a well-characterized, developmentally appropriate CT assessment for early elementary school children. The original version (*TechCheck-1*) has shown considerable promise in children between the ages of 5-9. However, experience using the instrument revealed a possible ceiling effect in second graders and evidence of literacy/working memory limitations in kindergarteners. To address this, two modified versions were created (*TechCheck-K* and *TechCheck*-2) to supplement the original *TechCheck-1* which is most suitable for first grade students. In the current study, we compared baseline performance across grades using these three versions of *TechCheck*. We found that means and distributions differed across the three grade levels. Kindergarteners, first graders and second graders performed similarly by CT domain on the three versions of *TechCheck*.

Item analysis also showed small distinctions in responses across domains for the three grades-specific versions of *TechCheck*. The relative consistency of the pattern of responses across domains suggests these three versions of TechCheck are equivalent and developmentally appropriate for students across grades. The crossed random effects model provided evidence of differences in response patterns across domains but did not implicate particular domains as the basis for differences (Figure 3). To permit more meaningful cross-grade comparisons, we calculated Z-scores and percentile ranks for each grade from baseline data obtained from a large group of students. The creation of Z-score and percentile rank tables for the three versions of *TechCheck* offers certain advantages compared to raw scores in terms of understanding and communicating CT results. Raw scores can be difficult to interpret, particularly when score distributions differ across grades. When designing an assessment with the intention of comparing multiple grades/ages of students or following children longitudinally, many different techniques can be applied. One technique is to give children of different ages the same set of questions. This was our original plan for *TechCheck*. However, this resulted in ceiling effects in second graders and possible floor effects in kindergarten students. Another method is to create an adaptive assessment such as the approach taken by Hogenboom et al. (2021) with the CAPCT assessment. However, while this approach offers certain advantages, it does require a more complex system of administration, scoring, and interpretation. Typically, adaptive assessments employ larger numbers of questions and a broader scope of difficulty than is the case with *TechCheck*.

Standardizing scores is a common practice in large-scale educational assessment (Weiss, 2016). Although the present study does not utilize samples that are adequately representative of the populations in which they may be used, we chose to take an initial step towards normalization of the assessments for multiple reasons. The *TechCheck* data obtained to date has been normally distributed which facilitates the calculation of Z-scores. Percentile ranks based on Z-scores are familiar to many educators, parents, and administrators, making it easier for them to understand student performance and progress. Standardized scores can be used to compare students' performance to that of their peers. Percentile ranks can be conveyed to parents in a way that is easily understood. By providing a metric that can be used to evaluate students' progress from grade to grade, standardized scores can help schools evaluate the effectiveness of their programs. Standardized score benchmarks can be established to identify whose performance is significantly above or below expectations for grade. Growth norms can also be calculated, so that teachers can compare how much their students improved relative to other students (Set, 2018). Norms for different populations and cultures can be created to help researchers and practitioners compare performance cross-culturally. While the present results are a meaningful step towards normalization, data from larger, more representative populations of students will be required before the results can be considered fully standardized.

Percentile ranks take into account differences in the distributions of scores in ways that using raw score means alone cannot. This can be helpful when comparing performance across grades in longitudinal studies. For example, there was an unexpected 42.74% difference between the *TechCheck* change scores of first and second graders in the CAL- ScratchJr longitudinal study. However, when expressed in terms of percentile rank changes, this represented a relatively modest 5.17 percentile rank difference in outcome between grades. Percentile rank changes may therefore provide a more meaningful way to compare results from two or more grades when different versions of *TechCheck* are used for assessment.

5. Limitations and future directions

TechCheck has been successfully administered in a variety of formats including in-person or remotely, online and on paper, to groups of students and individuals in many countries. The instrument has been translated into several languages in addition to English (i.e., Spanish, Turkish, Chinese, Dutch) for use in a variety of educational and research settings around the world. Feedback from students, parents, teachers, and administrators have been remarkably positive. It is apparent that the assessment is easy to administer and score and that children enjoy taking it.

Although the multiple-choice format makes the assessment easy to administer and score, it also does not lend itself to creative self-expression and open-ended problem solving which is a significant part of CT. Thus, one of Bers' seven powerful ideas, Design Process, could not be probed in *TechCheck*. In addition, the possibility of guessing the correct answer is something that must be taken into account when interpreting multiple-choice results. Future studies should use item response theory statistical techniques such as 3pl models that take into account guessing. Román-González et al. (2019) pointed out that CT assessments often focus on "concepts "rather than "practices and perspectives," and as a consequence become "static and decontextualized" (p. 91). Other testing formats such as collection of CT telemetry data applied to real-time programming and/or individualized adaptive assessments may address these concerns.

The normalization carried out in this study is based upon data from cohorts of children in six US states constituting a relatively diverse sample. Nevertheless, our findings are subject to potential biases inherent in cohort studies including the sample not being fully representative of all children in the target age groups. While the cohorts were relatively balanced in terms of variables such as gender and race/ethnicity, other potential covariates such as socioeconomic status were not examined.

A new version of the assessment suitable for preschool age children ages 3-5 called *TechCheck-PreK* was recently validated. Future studies should explore whether the assessment can be used with neuro-diverse children and in other contexts. The goal of these efforts is to establish *TechCheck* as an assessment that can be used in a wide variety of research and real-world educational settings and assist in identifying the best CS educational practices to enhance the acquisition of CT in children.

Acknowledgement

This work was supported by grants from the Department of Defense Education Activity (DoDEA) "Operation: Break the Code for College and Career Readiness". Unique Entity Identifier: "WORLDCL10" awarded to the Norfolk, Virginia Public Schools, the US Department of Education PR/Award Number: U411C190006 awarded to Dr. Marina Bers of Tufts University, and the Institute of Education Sciences, U.S. Department of Education, through Grant Award Number: R305A190433 awarded to the University of California, Los Angeles. We would like to thank the Norfolk Public School ITRTs, The Shaffer Evaluation Group, and the DevTech assessment team, and the participating children and teachers who made this work possible.

References

Bakala, E., Gerosa, A., Hourcade, J. P., & Tejera, G. (2021). Preschool children, robots, and computational thinking: A Systematic review. *International Journal of Child-Computer Interaction, 29*, 100337. https://doi.org/10.1016/j.ijcci.2021.100337

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48-54. https://doi.org/10.1145/1929887.1929905

Barrouillet, P., & Lecas, J. (1999). Mental models in conditional reasoning and working memory. *Thinking & Reasoning, 5*(4), 289–302.

Bell, T., & Vahrenhold, J. (2018). CS Unplugged—How Is It Used, and Does It Work? In H.-J. Böckenhauer, D. Komm, & W. Unger (Eds.), *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday* (pp. 497–521). Springer International Publishing. https://doi.org/10.1007/978-3-319-98355-4_29

Bers, M. U. (2018). *Coding as a playground: Programming and computational thinking in the early childhood classroom*. Routledge Press.

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: exploration of an early childhood robotics curriculum. *Computers in Education*, *72*, 145–157. https://doi.org/10.1016/j.compedu.2013.10.020

Bers, M. U., Blake-West, J., Govind, M., Levinson, T., Relkin, E., Unahalekhaka, A., & Yang, Z. (in press). *Coding as another language: Research-based curriculum for early childhood computer science.*

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 in Education, 109*, 162–175. https://doi.org/10.1016/j.compedu.2017.03.001

Clarke-Midura, J., Silvis, D., Shumway, J. F., Lee, V. R., & Kozlowski, J. S. (2021). Developing a kindergarten computational thinking assessment using evidence-centered design: The Case of algorithmic thinking. *Computer Science Education, 31*(2), 1–24. https://doi.org/10.1080/08993408.2021.1877988

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology, 76*, 1051–1058. https://doi.org/10.1037/0022-0663.76.6.1051

Cowan, N. (2016). Working memory maturation: Can we get at the essence of cognitive growth? *Perspectives on psychological science: A Journal of the Association for Psychological Science*, *11*(2), 239–264. https://doi.org/10.1177/1745691615621279

Dagienė, V., & Futschek, G. (2008). Bebras international contest on informatics and computer literacy: Criteria for good tasks. In R. T. Mittermeir & M. M. Sysło (Eds.), *Informatics Education—Supporting Computational Thinking* (pp. 19–30). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-69924-8_2

El-Hamamsy, L., Zapata-Cáceres, M., Barroso, E. M., Mondada, F., Zufferey, J. D., & Bruno, B. (2022). The Competent computational thinking test: Development and validation of an unplugged computational thinking test for upper primary school. *Journal of Educational Computing Research*, 07356331221081753. https://doi.org/10.1177/07356331221081753

Fraillon, J., Ainley, J., Schulz, W., Duckworth, D., & Friedman, T. (2018). *International Computer and Information Literacy Study: ICILS 2018: Technical Report.* https://www.iea.nl/sites/default/files/2020-05/ICILS%202018%20Technical%20Report-FINAL_0.pdf

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

Grover, S., Cooper, R., & Pea, R. (2014). Assessing computational learning in K-12. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 57-62). ACM. https://doi.org/10.1145/2591708.2591713

Hinton, P., Brownlow, C., McMurray, I., & Cozens, B. (2004). *SPSS explained*. Taylor & Francis. https://doi.org/10.4324/9780203642597

Hogenboom, S. A. M., Hermans, F. F. J., & van der Maas, H. L. J. (2021). Computerized adaptive assessment of understanding of programming concepts in primary school children. *Computer Science Education*. https://doi.org/10.1080/08993408.2021.1914461

Horn, M. (2012). *TopCode: Tangible object placement codes*. http://users.eecs.northwestern.edu/~mhorn/topcodes

Hubwieser, P., Giannakos, M. N., Berges, M., Brinda, T., Diethelm, I., Magenheim, J., Pal, Y., Jacova J., & Jasute, E. (2015). A Global snapshot of computer science education in K-12 schools. In *Proceedings of the 2015 ITiCSE on working group reports* (pp. 65-83). https://doi.org/10.1145/2858796.2858799

Iglewicz, B., & Hoaglin, D. C. (1993). *How to detect and handle outliers*. American Society for Quality Control.

Janveau-Brennan, G., & Markovits, H. (1999). The Development of reasoning with causal conditionals. *Developmental Psychology, 35*(4), 904–911. https://doi.org/10.1037/0012-1649.35.4.904

Kingsbury, G. G., & Weiss, D. J. (1983). A Comparison of IRT-based adaptive mastery testing and a sequential mastery testing procedure. In *New horizons in testing* (pp. 257-283). Academic Press. https://doi.org/10.1016/B978-0-12-742780-5.50024-X

Kong, S.-C., & Lai, M. (2022). Validating a computational thinking concepts test for primary education using item response theory: An Analysis of students' responses. *Computers & Education*, *187*, 104562. https://doi.org/10.1016/j.compedu.2022.104562

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads, 2*(1), 32–37. https://doi.org/10.1145/1929887.1929902

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

Marinus, E., Powell, Z., Thornton, R., McArthur, G., & Crain, S. (2018). Unravelling the cognition of coding in 3-to-6-year olds: The Development of an assessment tool and the relation between coding ability and cognitive compiling of syntax in natural language. In *Proceedings of the 2018 ACM Conference on International Computing Education Research - ICER '18* (pp. 133–141). https://doi.org/10.1145/3230977.3230984

Mioduser, D., & Levy, S. T. (2010). Making sense by building sense: Kindergarten children's construction and understanding of adaptive robot behaviors. *International Journal of Computers for Mathematical Learning, 15*(2), 99–127. https://doi.org/10.1007/s10758-010-9163-9

Moore, T. J., Brophy, S. P., Tank, K. M., Lopez, R. D., Johnston, A. C., Hynes, M. M., & Gajdzik, E. (2020). Multiple representations in computational thinking tasks: A Clinical study of second-grade students. *Journal of Science Education and Technology*, *29*(1), 19–34. https://doi.org/10.1007/s10956-020-09812-0

Muller, U., Overton, W. F., & Reene, K. (2001). Development of conditional reasoning: A Longitudinal study. *Journal of Cognition and Development*, *2*(1), 27–49. https://doi.org/10.1207/S15327647JCD0201_2

Papadakis, S. (2021). The Impact of coding apps to support young children in computational thinking and computational fluency. A Literature review. *Frontiers in Education, 6*. https://www.frontiersin.org/articles/10.3389/feduc.2021.657895

Portelance, D. J., & Bers, M. U. (2015). Code and tell: Assessing young children's learning of computational thinking using peer video interviews with scratchjr. In *Proceedings of the 14th International Conference on Interaction Design and Children - IDC '15* (pp. 271–274). https://doi.org/10.1145/2771839.2771894

Poulakis, E., & Politis, P. (2021). Computational thinking assessment: Literature review. In T. Tsiatsos, S. Demetriadis, A. Mikropoulos & V. Dagdilelis (Eds.), *Research on E-Learning and ICT in Education* (pp. 111–128). Springer, Cham. https://doi.org/10.1007/978-3-030-64363-8_7

R Core Team. (2019). *R: A Language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Relkin, E., & Bers, M. (2021). TechCheck-K: A Measure of computational thinking for kindergarten children. In *2021 IEEE Global Engineering Education Conference (EDUCON)*. IEEE. https://sites.tufts.edu/devtech/files/2021/05/1487.pdf

Relkin, E. (2021). TechCheck: Creation of an unplugged computational thinking assessment for young children. In M. U. Bers (Ed.), *Teaching Computational Thinking and Coding to Young Children* (pp. 250-264). IGI Global. https://doi.org/10.4018/978-1-7998-7308-2

Relkin, E., & Bers, M. U. (2019). Designing an assessment of computational thinking abilities for young children. In L.E. Cohen & S. Waite-Stupiansky (Eds.), *STEM for Early Childhood Learners: How Science, Technology, Engineering and Mathematics Strengthen Learning* (pp. 85-98). Routledge.

Relkin, E., de Ruiter, L., & Bers, M. U. (2020). TechCheck: Development and validation of an unplugged assessment of computational thinking in early childhood education. *Journal of Science Education and Technology, 29,* 482–498. https://doi.org/10.1007/s10956-020-09831-x

Relkin, E., de Ruiter, L., & Bers, M. U. (2021). Learning to code and the acquisition of computational thinking by young children. *Computers & Education, 169*, 104222. https://doi.org/10.1016/j.compedu.2021.104222

Román-González, M., Moreno-León, J., & Robles, G. (2019). Combining assessment tools for a comprehensive evaluation of computational thinking interventions. In S. C. Kong, & H. Abelson (Eds.), *Computational Thinking Education* (pp. 79-98). Springer. https://doi.org/10.1007/978-981-13-6528-7_6

Román-González, M., Pérez-González, J. C., Moreno-León, J., & Robles, G. (2018). Can computational talent be detected? Predictive validity of the Computational Thinking Test. *International Journal of Child-Computer Interaction*, *18*, 47-58. https://doi.org/https://doi.org/10.1016/j.ijcci.2018.06.004

Set, A. (2018). Making assessment more meaningful with norms. *NWEA*. https://www.nwea.org/blog/2018/making-assessment-more-meaningful-with-norms/

Simmering, V. R. (2012). The Development of visual working memory capacity during early childhood. *Journal of Experimental Child Psychology*, *111*(4), 695–707. https://doi.org/10.1016/j.jecp.2011.10.007

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: a systematic review of empirical studies. *Computers in Education, 148*, 103798. https://doi.org/10.1016/j.compedu.2019.103798

Tran, Y. (2018). Computational thinking equity in elementary classrooms: What third-grade students know and can do. *Journal of Educational Computing Research*, 57(1), 3-31. https://doi.org/10.1177/0735633117743918

Wang, D., Wang, T., & Liu, Z. (2014). A Tangible programming tool for children to cultivate computational thinking. *The Scientific World Journal*, *2014*, 428080. https://doi.org/10.1155/2014/428080

Weiss, L. G. (2016). *Standardized assessment for clinical practitioners: A Primer*. https://www.pearsonassessments.com/content/dam/school/global/clinical/us/assets/featured-topics/assessment-primer-whitepaper.pdf

Werner, L., Denner, J., & Campe, S. (2014). Using computer game programming to teach computational thinking skills. In K. Schrier (Ed.), *Learning, education and games: Volume 1, curricular and design considerations* (pp. 37–53). ETC Press. https://dl.acm.org/doi/10.5555/2811147.2811150

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). https://doi.org/10.1145/2157136.2157200

White House. (2016). *Educate to innovate*. https:// www.whitehouse.gov/issues/education/k-12/educate-innovate

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3),33–35. https://doi.org/10.1145/1118178.1118215

Wing, J. (2010). *Computational thinking: A Definition* (Unpublished manuscript). https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Yadav, A., Good, J., Voogt, J., & Fisser, P. (2017). Computational thinking as an emerging competence domain. In *Technical and vocational education and training* (Vol. 23, pp. 1051–1067). https://doi.org/10.1007/978-3-319-41713-4_49

Zapata-Cáceres, M., Martín-Barroso, E., & Román-González, M. (2020). Computational thinking test for beginners: Design and content validation. In *2020 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1905-1914). IEEE. https://doi.org/10.1109/EDUCON45650.2020.9125368

Zhang, L., & Nouri, J. (2019). A Systematic review of learning computational thinking through Scratch in K-9. *Computers in Education, 141*, 103607. https://doi.org/10.1016/j.compedu.2019.103607