Current Psychology (2023) 42:8039–8050
https://doi.org/10.1007/s12144-021-02085-9

Measuring coding ability in young children: relations to computational thinking, creative thinking, and working memory

Lin Wang [1] · Fengji Geng [1,2] · Xiaoxin Hao [1] · Donglin Shi [1] · Tengfei Wang [3] · Yan Li [1]

Accepted: 6 July 2021 / Published online: 29 July 2021

Abstract

Coding ability has become an important literacy in the twenty-first century. Coding education starts from early childhood in many countries. However, it lacks of tools with good psychometric properties to assess the changes in the coding ability of young children (i.e., preschoolers or kindergarteners) after learning coding. To fill this gap, the current study aimed to develop a tool by using card-based and age-appropriate games to measure the coding ability of young children aged 5–6 years. In the tool, coding ability was evaluated on the dimensions of Variable, Control, Modularity, and Algorithm. The first three dimensions of the tool included the skills of Assignment, Type, Conditional, Loop, Decomposition, and Function. We tested the psychometric properties of the tool by assessing its reliability and validity. The results indicated that the tool not only had good internal consistency, inter-raters reliability, and test-retest reliability, but also showed good content validity, construct validity, and item discrimination. Additionally, the coding ability measured by the tool was significantly related to creative thinking and computational thinking, suggesting good criterion validity. To conclude, this study developed an age-appropriate and game-based tool with good psychometric properties to assess the coding ability of young children. This tool can be also used to evaluate the effectiveness of educational coding programs for young children.

Keywords Coding ability · Computational thinking · Creative thinking · Working memory · Young children

Introduction

Many countries around the world have incorporated coding education to school curriculum (Heintz, Mannila, & Färnqvist, 2016; Hsu et al., 2019). Recently, increasing attention has been paid to coding education for preschool children and kindergarteners (Arfé et al., 2019, 2020). Young children have been found capable of learning coding skills (e.g., Lye & Koh, 2014; Popat & Starkey, 2019). Additionally, coding learning has positive impact on the development of many other cognitive abilities, such as planning and inhibition, the abilities that help young children maintain and retrieve information in goal-directed ways (Arfé et al., 2019, 2020; Scherer et al., 2019). However, there is a lack of tools to measure coding learning outcomes in early childhood. To fill this gap, the current study focused on developing an age-appropriate tool with good psychometric properties to measure the coding ability of young children aged 5–6 years.

Definition of Coding Ability

For primary school children or older, coding ability has been defined as the skills to create, modify, and evaluate codes and the knowledge about programming concepts and procedures (Lye & Koh, 2014). In contrast, it is difficult for younger children to manipulate abstract codes on computer screen. Therefore, it is important to define coding ability in developmentally appropriate ways (Bers, 2018; Menon et al., 2019). Specifically, as young children in kindergarten or preschool usually manipulate real objects (e.g., robotics or cards) to code, we defined the coding ability of young children as the ability in using programming-related concepts and procedures to solve problems by manipulating real objects. Such

✉ Fengji Geng
gengf@zju.edu.cn

[1] Department of Curriculum and Learning Sciences, Zhejiang University, 866 Yuhangtang Road, Zijingang Campus, Hangzhou 310058, China

[2] National Clinical Research Center for Child Health, Children's Hospital, Zhejiang University School of Medicine, Hangzhou 310052, China

[3] Department of Psychology and Behavioral Sciences, Zhejiang University, 148 Tianmushan Road, Xixi Campus, Hangzhou 310028, China

problem-solving processes also involve a great amount of creation, modification, and evaluation for young children.

Measuring Coding Ability in Young Children

Researchers have taught and measured young children's coding ability using different approaches and tools, such as Scratch (e.g., Rodríguez-Martínez, González-Calero, and Sáez-López 2020), Code.org (e.g., Arfé et al. 2020), educational coding robotics (KIBO; Bers et al. 2019), or tablet applications (e.g., Pila et al. 2019). These previous studies have commonly recognized that coding ability engages multiple dimensions. For instance, Bers et al. (2019) used KIBO to measure the coding ability of young children in terms of sequence, debugging, repeats, and conditionals. Additionally, by using tablet applications, researchers have measured young children's coding ability by tapping into the dimensions of sequence, condition, and loops (Pila et al., 2019). Therefore, the coding ability of young children has been measured in terms of multiple dimensions.

These measurement approaches are actually consistent with the theoretical frameworks proposed in previous studies (K-12 Computer Science Framework Steering Committee, 2016; Bers et al., 2019). For example, seven powerful ideas have been proposed to underlie the coding ability of young children, including algorithm, modularity, control structure, representation, hardware/software, design process, and debugging (Bers et al., 2019). Additionally, children are expected to have developed several concepts related to coding, such as the dimensions of Variable, Control, Modularity, Algorithm, and Program Development, by the end of second grade (K-12 Computer Science Framework Steering Committee, 2016).

Based on the measurement approaches and theoretical frameworks outlined in aforementioned studies, we propose that the coding ability of young children can be measured in terms of four dimensions: Variables, Control, Modularity, and Algorithms. Moreover, the dimension of Variables includes the skills of Assignment and Type; the dimension of Control includes the skills of Conditional and Loop; the dimension of Modularity includes the skills of Decomposition and Function (see the definition for each coding skill in Table 1).

Relations of Coding Ability to Other Cognitive Functions

Computational Thinking

Computational thinking refers to the process of solving problems, designing systems, and understanding human behaviors based on the principles and methods of computer science (Wing, 2006, 2008). As the coding process involves a large amount of knowledge about computer science, many educators claim that learning coding provides an important context and a set of opportunities to develop computational thinking for K-12 students (Popat & Starkey, 2019). Thus, coding ability and computational thinking are thought to be closely related. Some researchers have even directly measured children's computational thinking by testing their coding ability (Korkmaz et al., 2017; Román-González et al., 2017). The current study would test the criterion validity of the tool that we developed by examining the relations between coding ability and computational thinking.

Creative Thinking

Creative thinking is comprised of the skills needed to exhibit creative behaviors, such as originality, fluency, flexibility, and elaboration (Pardamean et al., 2011; Scherer et al., 2019). Creativity is thought to play a critical role in coding by encouraging children to solve problems in novel ways (Clements, 1995; Grover & Pea, 2013). As a result, learning coding has been suggested to exert positive influence on creativity, which has been supported by empirical evidence (Kim et al., 2013; Pardamean et al., 2011; Scherer et al., 2019). Additionally, a meta-analysis study reported that the positive effect of learning coding can be transferred to creative skills with large effect sizes (Scherer et al., 2019). Moreover,

Table 1 Definition for each dimension and related coding skills

Dimension	Skill	Definition	Game
Variable	Assignment	A statement in computer programing that is used to set a value to a variable name	Game 1
	Type	A classification identifying one of various types of data	Game 2
Control	Conditional	A statement in a computer program that is executed only when a certain condition is satisfied	Game 3
	Loop	A sequence of instructions that is continuously repeated until a certain condition is reached	Game 4
Modularity	Decomposition	Break down a problem into smaller parts to understand and manage the problems better	Game 5
	Function	A body of computer instructions designed to be used by other routines to accomplish some particular purpose	Game 6
Algorithm	–	A set of instructions designed to accomplish a specific task	Game 7–9

creativity has been treated as a critical aspect of computational thinking that is closely related to coding ability (Korkmaz et al., 2017). To summarize, previous studies have focused on the effect of coding learning on creativity, but the present study aimed to test whether the coding ability inherently possessed by young children prior to intensive coding instruction was related to creative thinking.

Working Memory

Learning coding has been reported to benefit multiple cognitive abilities. For example, coding activity was reported to improve first graders' planning and inhibition skills significantly more than standard STEM activities did (Arfé et al., 2019; Çiftci & Bildiren, 2020). Additionally, a pilot study using small sample size found that a 6-week educational robotics intervention, during which children played with a toy robot by designing and inputting instructions, had positive effect on the executive functions of preschool children (Di Lieto et al., 2017). Specifically, working memory was a component of executive functions that was positively affected by the educational robotics intervention (Di Lieto et al., 2017). The positive effect may arise from the interaction with educational robotics as it takes much effort for young children to maintain and manipulate the information stored in their working memory systems. The current study would further test whether the inherent coding ability possessed by young children is related to their working memory.

Goals of Current Study

Based on the definition of coding ability for young children and its underlying dimensions, we aimed to develop a tool using card-based and age-appropriate games to measure the coding ability of children aged 5–6 years. The reason for focusing on this age range is that it is only recently that coding activities have been integrated into the education toward children at this young age and no tool with good psychometric properties has been developed to measure the coding ability of children as young as 5–6 years. To fill this gap, we aimed to develop a tool to measure the coding ability of young children, report its psychometric properties, and test its criterion validity by examining the concurrent relations of coding ability to computational thinking, creative thinking, and working memory.

Methods

Study Group

Sixty children from the senior classes of a kindergarten at Hangzhou, China participated in the study (mean age = 5.70 years, range from 5.23 to 6.21 years, SD = 0.29, 35 boys). The kindergarten was selected due to its reasonable distance from the authors' university and the representativeness of the sample. For example, all but two parents of the children reported to have received high school education or above. Additionally, 18 children were reported to have more or less coding experience and the others had never learnt coding before the testing. All children were native mandarin Chinese speakers.

According to parents' report, all children were not diagnosed with any physical or mental disease. Children attended the study only if their parents agreed and signed the consent forms that they received at the Kindergarten. Additionally, children were verbally assented before attending the study due to their limited reading and writing abilities. Three children did not finish the coding task and one child did not finish the tests of computational thinking and working memory.

Development of the Tool Measuring Coding Ability of Young Children

Design of the Tool

According to the definition of coding ability and its underlying dimensions abovementioned in the Introduction, we developed a card-based game corresponding to each skill involved in the Variables, Control, and Modularity dimensions (Game 1–6, Fig. 1 and Fig. 2). For Algorithm dimension, three games were developed (Game 7–9, Fig. 2), and each of them also involved three skills selected from the other three dimensions. Children were given five minutes to play each of Game 1–6, and seven minutes to play each of Game 7–9. If children did not finish each game within the time limit, experimenters helped children complete the game. We only rated children's coding behaviors that happened within time limits.

In Game 1–9, children were instructed to help the cartoon character move from the Start to the End, during which children had to use their coding skills to fulfill the designed tasks. In Fig. 1 and Fig. 2, the original maps of games presented to children are labeled as A, the marked maps labeled as B illustrate the design of the games; the maps labeled as C are the correct commands. In each game, the involved coding skills were different.

In Game 1 (Assignment), children were instructed to help the cartoon character move from the Start to the End by following the specified path. Additionally, instead of listing all forward arrows, children were asked to put a number underneath the forward arrows to indicate how many steps to be moved, reflecting the use of Assignment skill.

In Game 2 (Type), on the way from the Start to the End, children would meet various pictures, representing different types of data in coding language. Children were required to use different commands when encountering different pictures,

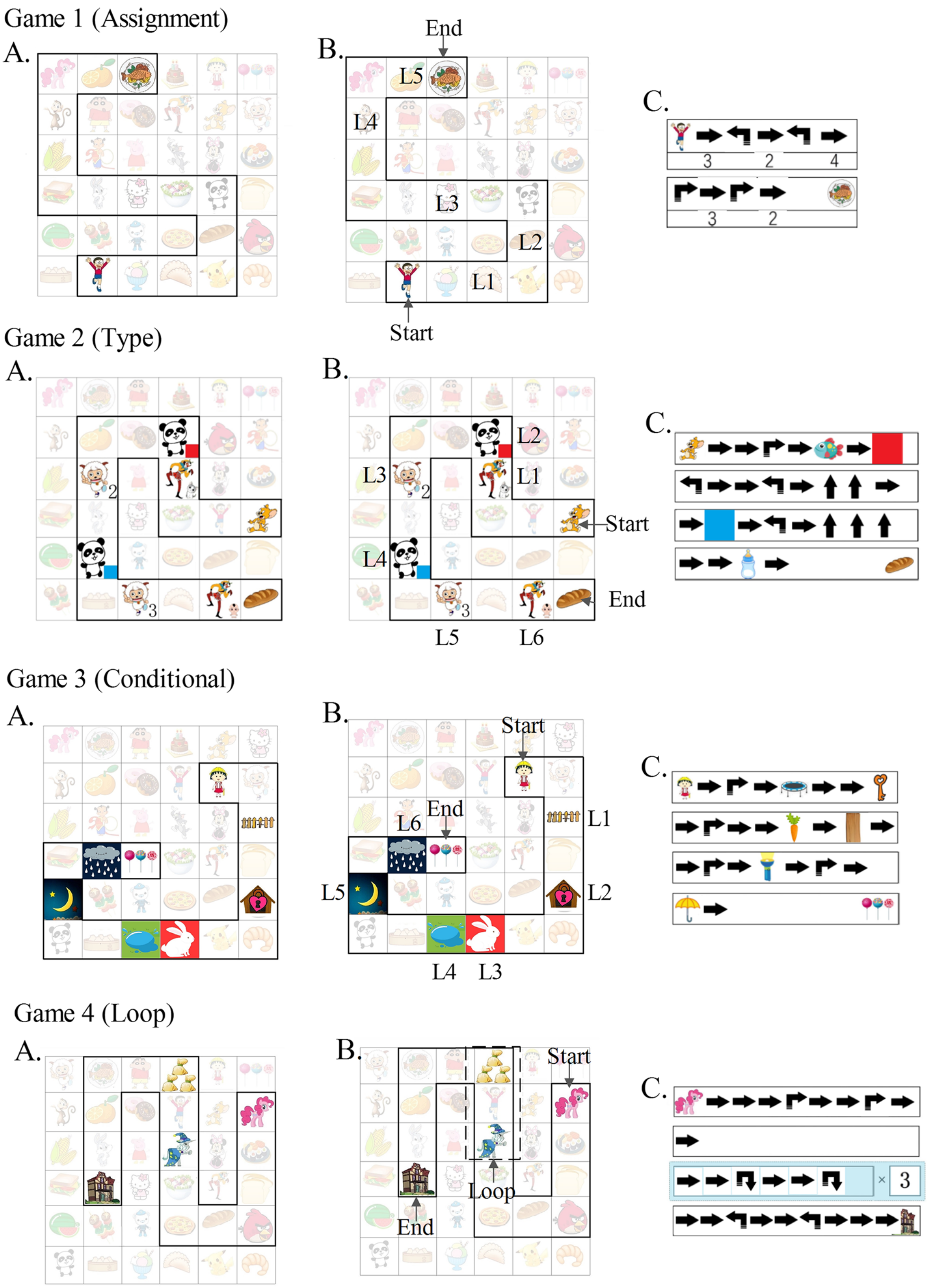


Fig. 1 Game design for Variable and Control dimensions

reflecting the use of the Type skill. This skill had to be correctly used at six locations (L1-L6) before reaching the End.

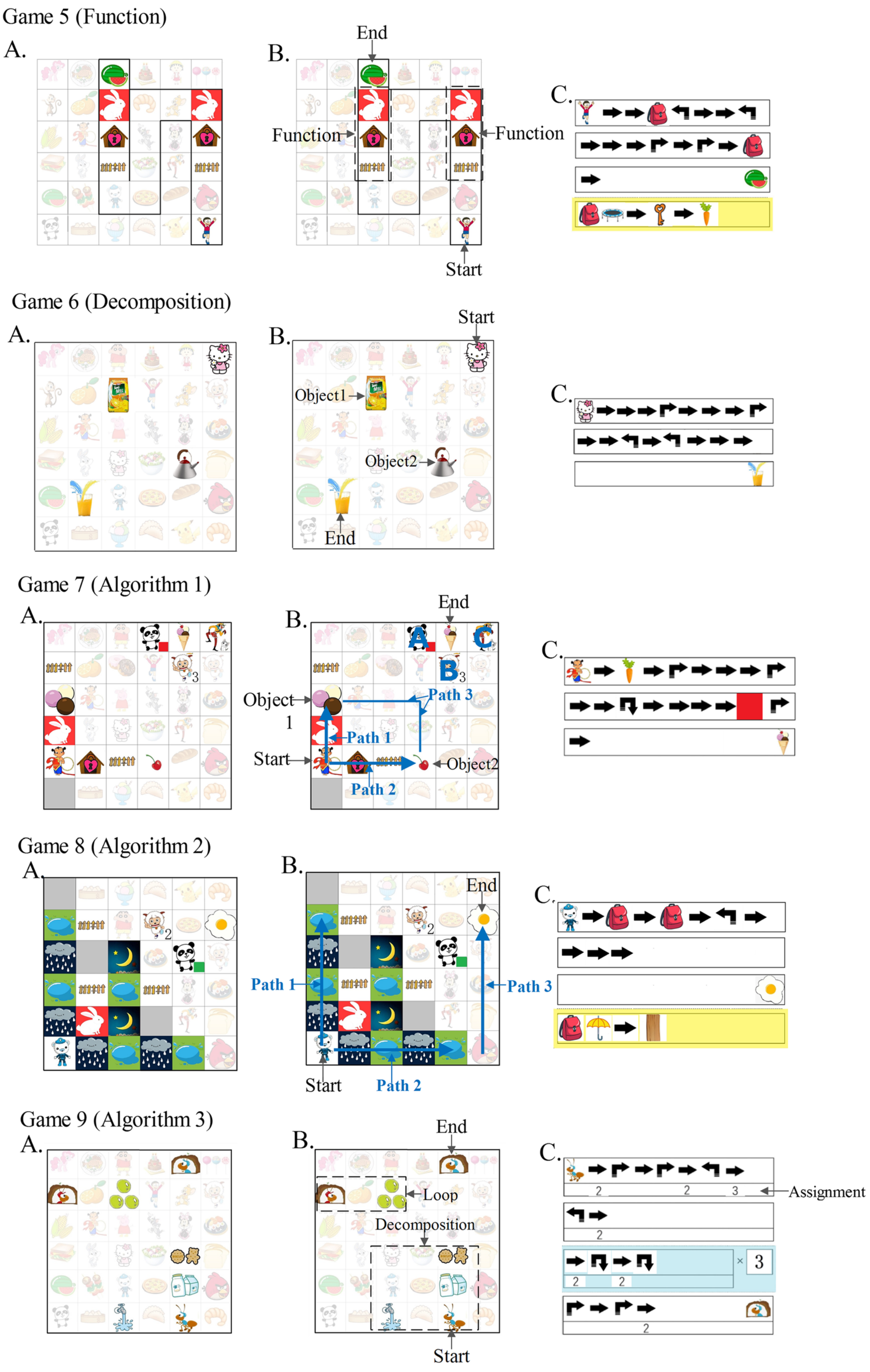
In Game 3 (Conditional), there were six locations (L1-L6) in each of which a specific picture was presented to create an "if" condition. A specific command had to be executed on each "if" condition, reflecting the use of the Conditional skill.

In Game 4 (Loop), a young pink horse helped an old blue horse retrieve three bags of rice on the way to the end. Each time, only one bag of rice could be retrieved. Therefore, the pink horse had to be back and forth between the locations of the blue horse and the rice for three times. Instead of listing each command, the Loop skill could be used by putting number three in the box beside one cycle of commands, indicating the cycle would be repeated for three times.

In Game 5 (Decomposition), Object 1 and 2 had to be retrieved before reaching the end, reflecting the use of the Decomposition skill.

In Game 6 (Function), a sequence of three different pictures were presented twice. In response, a set of commands in a certain order had to be executed repeatedly. These commands were packed in a bag. Children only needed to put the bag in the command list whenever the set of commands had to be used sequentially.

Fig. 2 Game design for Modularity and Algorithm dimensions



In Game 7–9 (Algorithm), the path from the Start to the End was not specified. The six coding skills mentioned previously could be used in the games to solve problems. Children decided which path and skills to use in order to reach the end efficiently and effectively.

Testing Procedure of the Tool

Before children played these nine games, there were two baseline games (Game 0) to familiarize children with the basic rules, including how to move forward and how to turn left or right. Only after experimenters rated that children had mastered the rules, children were allowed to perform the nine formal games. Right before playing Game 1–6, experimenters used a simple demonstration game to show how to execute the specific command engaged in each test. For the three games tapping Algorithm component, we asked children to use the commands that they used in the previous six games to solve the problems in the current games as efficiently as possible. The whole process that children played these games was videotaped.

Rating Strategies of the Tool

Children's performance in each game was coded in two steps. As suggested by Bers et al. (2019), experimenters first rated children's performance in terms of goal attainment on a 5-point Likert scale during game play (1 = not at all, 2 = almost not, 3 = partially, 4 = mostly, 5 = completely). We then rated children's final commands in each game in a more detailed way based on the videos taped during the test. If scores of the first and second steps were significantly related to each other, the two types of scores would be standardized and added up to index the coding ability of each child. Additionally, the experimenters also rated children's emotion status and engagement level on a 5-point Likert scale, respectively (emotion: 1 = very unhappy, 2 = somewhat unhappy, 3 = neutral, 4 = somewhat happy, 5 = very happy; engagement: 1 = very low, 2 = somewhat low, 3 = ordinary, 4 = somewhat high, 5 = very high). These ratings were expected to provide additional insight into whether the games were appropriate for young children.

Game 1 (Assignment): (1) Correct moves that children had made were counted. One point was granted for each correct move. (2) The Assignment skill should be used in five locations (L1-L5). Two points were granted for each correct use. One point was given for the incorrect use of the Assignment skill that resulted in any error.

Game 2 (Type): (1) Correct moves that children had made were counted. One point was given for each correct move. (2) Children received two points for each correct use of the Type skill and one point for each incorrect use.

Game 3 (Conditional): (1) Raters counted the number of correct moves that children had made. One point was granted for each correct move. (2) Children received two points for each correct use of the Conditional skill and one point for each incorrect use.

Game 4 (Loop): (1) Raters counted the number of correct moves that children had made. Children received one point for each correct move. (2) If all moves were correct within the Loop, two points would be granted. One point would be granted if there was any wrong move within the loop. (3) If children had defined the right cycles of the loop, two points would be granted. If incorrect cycles had been defined, one point would be given.

Game 5 (Decomposition): Children received two points for retrieving each Object successfully. Additional two points would be granted if children reached the End.

Game 6 (Function): (1) Correct moves that children had made were counted. One point was granted for each correct move. (2) If the Function command had been defined correctly in the yellow box, two points were graded. If the Function command had been defined incorrectly, one point was granted. (3) The Function command should be called twice in this game. Additional two points were granted for each correct use of the Function command; one point was granted for each incorrect use of the command.

Game 7 (Algorithm 1): (1) If children chose to retrieve Object 1 by Path 1 as the beginning step, two points were granted; if children chose to retrieve Object 2 by Path 2 as the beginning step, one point was granted. (2) If children chose Path 3 to retrieve another Object after retrieving the first Object successfully, two points were granted. One point was granted if children chose paths other than Path 3 to retrieve the second Object. (3) If children chose to reach the End through Character A by using the Type skill, two points were granted; if any of the other two options were selected, one point was granted.

Game 8 (Algorithm 2): (1) If children chose Path 2 as the first step, two points were granted; if children chose Path 1, one point was granted. (2) In Path 1 or 2, if children defined and executed the Function command correctly, two additional points were granted; one additional point was granted if there was any error. (3) If children chose Path 3 and reached the End successfully, two points were granted.

Game 9 (Algorithm 3): (1) For the involved Decomposition, two points were granted if children chose to wash hands, get the milk, and get biscuits sequentially; one point was given if children chose to wash hands, get the biscuits, and get the milk sequentially; no point was given for the other orders. (2) Two points were granted if the Loop command was defined and executed correctly; one point was granted if there was any error in the commands. (3) The use of the Assignment skill was rated up to seven times. Two points were granted for each correct use of the Assignment skill. One point was granted each time when the Assignment skill was used but resulted in an error.

Computational Thinking

We used the Bebras challenges (for age 6 to 8 years) to measure the computational thinking of children (UK Bebras, 2018). The Bebras challenges have been operating for more than 15 years with about 70 countries and eight million students participating (Dagienė & Sentance, 2016). The problems in the challenges were constructed at three levels of difficulty: A, B, and C. We only used the problems at levels A and B since children were relatively young (5–6 years) in the current study. There were six problems with three at each difficulty level. Considering the limited computer skills possessed by children, we converted the online test into a paper/pencil test. For each problem, children were asked to circle the correct answer using a pencil. Two versions of the test were designed (versions A and B), differing only in the order of the presented problems. Each child was randomly presented one version and asked to complete it within 27 min.

We scored children's performance in terms of the rules suggested by the Bebras challenges. For problems at level A, zero point was granted for no response or wrong responses, and six points were granted for each correct response. For problems at level B, zero point was granted for no response, minus two points were granted for wrong responses, and nine points were granted for each correct response. The scores a child earned on all problems were added up to represent his or her ability in computational thinking.

Creative Thinking

We measured children's creative thinking by using the Torrance Tests of Creative Thinking (TTCT; Torrance, 1966). The TTCT includes figural and verbal subtests. We only administered the Figural subtest in this study because 5- to 6-year old children had limited ability in verbalization. The figural subtest had two versions (A and B). The Version we used included three types of activity: Picture Construction, Picture Completion, and Circles. During the activity of Picture Construction, children were asked to make a picture using a jelly-bean shape as a stimulus on the page. Such shape must be an integral part of the composition. The activity of Picture Completion required children to use 10 incomplete figures to make an object or picture. The activity of Circles asked children to make objects or pictures using the circles. Additionally, children were asked to name each object or picture they made.

According to the scoring manual of TTCT, children's creative thinking was graded in terms of fluency, originality, elaboration, abstractness of titles, and resistance to premature closure. Raters were trained to grade children's performance on TTCT following four steps: 1) an expert trained raters on a group meeting; 2) raters graded five children's data together, and then discussed and resolved the differences between their ratings; 3) raters independently graded the data of 15 children and the inter-rater reliability was calculated; 4) after reaching good inter-rater reliability (Cronbach Alpha reliability coefficient, fluency = .998, originality = .983, elaboration = .981, abstractness of titles = .984, and resistance to premature closure = .977), the data of whole sample were graded by raters.

Working Memory

We measured children's working memory by using the backward digital span task and the backward Corsi block tapping subtest of the Wechsler Memory Scale (Wechsler, 1987). In the digital task, the child listened to a sequence of numbers and repeated them in the reversed order. In contrast, the Corsi block tapping subtest was administered on a standard plastic board, on which nine blocks of the same color, shape, and material were placed. The examiner tapped some of the nine blocks in a certain sequence and children were asked to tap them in the reversed order.

For both tests, the length (n) of the sequence ranged between two and seven. Each sequence was given twice in the ascending order of length. A sequence of n items was given only when children responded correctly to at least one trial of the sequence of n – 1 items. The test stopped when children failed to make correct response to both trials of a certain sequence. Half a point was granted for each correct response. The total points were added up in each task to represent numerical and visuospatial working memory, respectively.

Data Collection and Analyses

The testing lasted about three weeks and was conducted in a quiet room in the kindergarten. Coding ability and working memory were measured individually. The testing of coding ability lasted about 90 min with breaks between games, and was videotaped. Creative thinking and computational thinking were tested in groups on a separate day. Each group had 10 children with six to eight experimenters. One experimenter gave the general instructions, while the other experimenters supervised one to two children simultaneously just in case of any further question. Two weeks later, the coding ability of 15 children was tested again to examine test-retest reliability.

We used expert evaluation method to examine the content validity of the coding tool. First, we contacted experts who were independent of the team that developed the testing tool; they had expertise in related research fields such as Computer Sciences and Computational Education. If an expert agreed to participate, we asked them to rate the importance and appropriateness of each item (i.e., Game) on a 4-point Likert scale (1: extremely not important/appropriate; 2: not important/appropriate; 3: important/appropriate; 4: extremely important/appropriate), respectively. The ratings were used to calculate the Item Content Validity Index (I-CVI) and the Scale Content Validity Index (S-CVI). For each Game, I-CVI was calculated by dividing the number of experts who rated the importance/appropriateness as 3 or 4 by the total number of experts participating in the rating. In contrast, for the whole scale, S-CVI was calculated by dividing the number of items rated as high as 3 or 4 by the total number of items.

Additionally, the Cronbach Alpha reliability coefficient and Intra Class Correlation coefficient were calculated to measure the tool's internal consistency level and raters' reliability, respectively. To test the construct validity, we applied exploratory factor analysis to extract the factors involved in the coding tool. Specifically, we used principle components analysis for the EFA extraction and selected Varimax as the rotation method. To calculate the discrimination power of each item, a lower and a higher group were created to include 15 children (27%) with the lowest scores on the nine games and 15 children (27%) with the highest scores on these games. Independent sample *t*-test was used to compare the differences in each item between the lower and higher groups. Finally, correlational coefficients between test scores over two time points were computed to measure test-retest reliability. The correlations between coding ability and creative thinking, computational thinking, and working memory were computed to confirm the criterion validity of the tool.

Results

Emotional and Engagement Status

In order to examine whether our testing tool was appropriate for young children, the emotional status and engagement level rated by experimenters for all children were analyzed (Table 2, rated scores >3). The results indicated that the emotional status of young children was between neutral and somewhat happy, and the engagement level was between ordinary and somewhat high on all games, suggesting that these games were appropriate for young children in terms of emotional status and motivational level.

Reliability of the Tool

The scores of each game rated in step 1 were significantly related to the ones rated in step 2 ($rs \geq .33$). Therefore, the two types of scores for each game were standardized and averaged. Additionally, the scores of Game 7–9 were combined to represent the Algorithm skill.

For all the coding skills involved in the assessment tool, the Cronbach Alpha reliability coefficient was .90, suggesting good internal consistency. Additionally, two raters coded ten children's data independently. The IntraClass correlation coefficients were greater than .82 for all nine games (Table 2), indicating good inter-rater reliability. Finally, the coding ability of 15 of the 60 children was tested again two weeks after the initial test. We calculated the correlation coefficient for each game, which ranged between .45 and .94 (Table 2). Except that the coefficient of the Function game was only marginally significant ($p = .089$), all the other coefficients were significant ($ps < .05$). To summarize, the coefficients reported above suggested that the tool can measure the coding ability of young children with good consistency and reliability.

Validity of the Tool

Content Validity

The I-CVI for each Game and the S-CVI for the whole scale were greater than 0.80, indicating good content validity.

Construct Validity

The Kaiser-Meyer-Olkin measure of sampling adequacy was 0.842, well above the minimum threshold of 0.5 (Kaiser, 1974). Bartlett's test of sphericity yielded a Chi square of

Table 2 Rated scores and reliability coefficients for each game

Game	Emotion		Engagement		Rated scores (step 1)		Rated scores (step 2)		Raters' reliability	Test-retest reliability
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	*r*	*r*
1	3.83	.59	3.92	.67	2.30	.89	6.84	5.13	.99**	.85**
2	3.85	.63	3.90	.63	3.08	1.21	12.75	7.96	.99**	.87**
3	3.92	.74	4.03	.78	3.15	1.16	13.05	7.36	.99**	.89**
4	3.87	.89	3.92	.91	2.52	1.08	6.98	3.99	.98**	.56*
5	3.88	.87	3.87	.89	2.17	.76	5.88	3.86	.98**	.45
6	3.88	.92	3.92	.96	3.17	1.33	1.96	2.28	.95**	.66*
7	3.88	.96	3.93	1.01	2.73	1.09	1.81	1.69	.93**	.56*
8	3.92	1.05	3.97	1.07	2.95	1.21	1.37	1.40	.82**	.94**
9	3.87	1.10	3.87	1.13	2.23	.93	2.33	3.18	.99**	.79**

*indicates $p < .05$; ** indicates $p < .001$

265.26 ($p < 0.001$), indicating that the intercorrelation matrix contained variables with sufficient collinearity for analysis (Bartlett, 1950). Only one factor was extracted from the nine items, with Eigenvalue being 4.94. This indicated that all items fitted onto a single theoretical construct, which was coding ability.

Item Discrimination

Independent *t* tests yielded significant differences between the lower an higher 27% groups (i.e., the bottom 27% and the top 27%) in the scores of each item as well as the total scores (Table 3). These results indicated that the discrimination power of the scale was good.

Relations of Coding Ability to Computational Thinking, Creative Thinking, and Working Memory

To test the criterion validity of the coding ability assessment tool, we measured whether coding ability was significantly related to computational thinking, creativity, and working memory with age as covariate. Significant positive correlation was observed between coding ability and computational thinking (Fig. 3A). Additionally, TTCT generated five creativity indicators (fluency, originality, elaboration, abstractness of titles, and resistance to premature closures). Scores of these five indicators were also added up to represent the general creative thinking ability. Coding ability was significantly related to fluency (Fig. 3B), originality (Fig. 3C), and the general creative thinking ability (Fig. 3D). Coding ability was not related to numerical and visuospatial working memory (*ps* > .395).

Discussion

The current study aimed to develop a tool to measure the coding ability of young children aged 5–6 years. To reach this goal, we designed a set of card-based games. The emotional status and motivational level rated by experimenters indicate that this game-based coding ability assessment tool is age-appropriate. The tool also has good internal consistency, inter-raters reliability, test-retest reliability, as well as good content validity, construct validity, and item discrimination. Furthermore, as expected, coding ability measured by this tool was significantly related to computational thinking and creative thinking, suggesting good criterion validity.

Reliability and Validity of the Tool

Quality is a major concern in designing assessment tools (Tang et al., 2020). In the present study, we assessed the reliability of the tool in terms of internal consistency level, raters' reliability, and test-retest reliability. Most indicators suggested that the tool had good reliability. The only exception was that the correlational coefficient for measuring the test-retest reliability of the game measuring Function skill was only marginally significant. We proposed two possible reasons for this result. First, it might result from the small sample size. We only re-tested 15 children's coding ability after the first run of test. Such small sample size could reduce the power to get significant findings and cause the correlational coefficient to be easily affected by extreme values. Second, the game measuring Function skill might be quite challenging for young children; the manipulation of commands in the game was probably more abstract or complicated than the other games. Future study is desired to examine whether reducing the abstraction level can improve the test-retest reliability of the Function game in young children.

The validity of the tool was measured by content validity, construct validity, and item discrimination. The expert evaluation method suggested that the tool had good content validity both at the item and scale levels. As expected, exploratory factor analysis suggested that the nine items in the tool fitted onto a single theoretical construct, which was defined as coding ability. Additionally, the discrimination power of the tool was good both at the item and scale levels.

Relations between Coding Ability and Computational Thinking

Significant positive correlation was observed between coding ability and computational thinking, supporting the claim that these two constructs were related to each other (Popat &

Table 3 Statistical details for item distinctiveness

Item	*T*	*P*	Item	*t*	*p*
Item 1 (Assignment)	−4.262	< .001	Item 6 (Decomposition)	−7.386	< .001
Item2 (Type)	−10.799	< .001	Item 7 (Algorithm 1)	−7.748	< .001
Item3 (Condition)	−11.047	< .001	Item 8 (Algorithm 2)	−5.940	< .001
Item4 (Loop)	−10.185	< .001	Item 9 (Algorithm 3)	−5.269	< .001
Item5 (Function)	−8.757	< .001	Total	−17.892	< .001

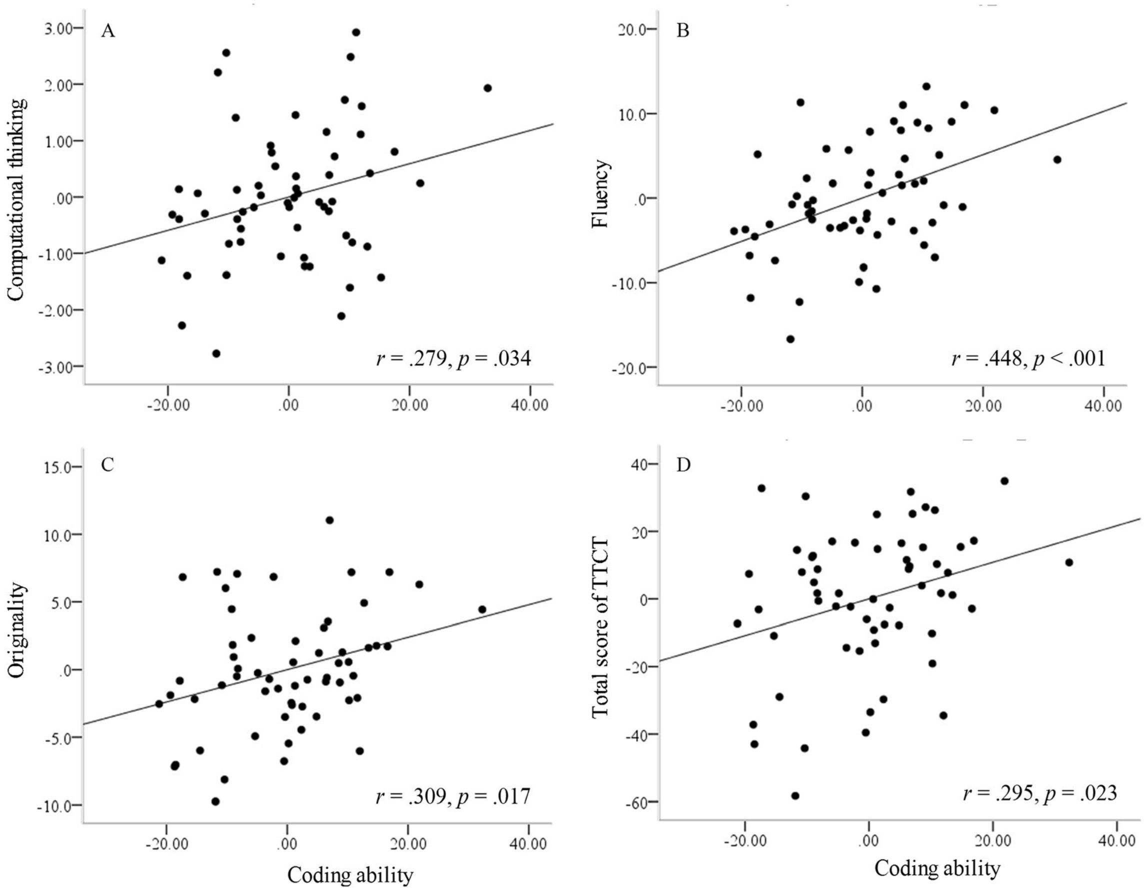


Fig. 3 Relations between coding ability and computational thinking (A), fluency (B), originality (C), and the total score of TTCT (D)

Starkey, 2019). However, the effect size for the correlation was only medium ($r = .28$), suggesting that they were not completely the same construct (Barr & Stephenson, 2011; Scherer et al., 2019). As mentioned above, researchers have suggested that when problem situation is not involved during coding, computational thinking would not be developed (Menon et al., 2019). Additionally, it has also been suggested that compared to coding, computational thinking includes computational perspective which is defined as expressing, connecting, and questioning. The computational perspective has been viewed as a critical component of computational participation (Kafai & Burke, 2014; Scherer et al., 2019). Therefore, instead of viewing coding and computational thinking as completely the same, it is critical to measure them independently in order to reflect their commonality and differences.

Relations between Coding Ability and Creative Thinking

Coding ability measured by our tool was related to the dimension of fluency and originality as well as the general creative thinking, but not related to the dimensions of elaboration, abstractness of titles, and resistance to premature closures. The findings suggested that coding ability were inherently related to creative thinking. Such inherent relations might be due to the engagement of creativity in the processes of coding (Clements, 1995; Grover & Pea, 2013). Additionally, although the current study did not test how learning coding affected creativity in young children, our findings were consistent with previous findings that learning coding exerted positive effects on the dimensions of fluency and originality in creative thinking (Clements, 1995; Kim et al., 2013; Pardamean et al., 2011).

Relations between Coding Ability and Working Memory

Coding ability was not related to digital and visuospatial working memory in the present study. This finding was not consistent with a previous study, in which a 6-week educational robotics intervention had positive effect on the visuospatial working memory of 12 children aged 5–6 years (Di Lieto et al., 2017). The inconsistent finding could be interpreted in different ways. First, the study above yielded a positive effect of learning coding on working memory, but the sample size was quite small, with only 12 children. Research with appropriate sample size is needed to further establish whether there are true relations between learning coding and working memory as well as how they are related. Second, our study focused on the relation between the inherent coding ability and working memory, while the study above only tested the effect of educational robotics intervention on working memory. Future research needs to compare the differences between coding learning and the inherent coding ability possessed by children in their relations to working memory. The lack of correlation might also be due to the games in the coding assessment tool posing few challenges to children's working memory.

Strengths and Limitations

The strengths of this study include: 1) we proposed a definition of young children's coding ability; 2) we designed a game-based assessment tool that can quantify the coding ability of young children with or without coding learning experience. The tool is not only appropriate for young children based on emotional status and motivation level, but also has good psychometric properties to measure the coding ability of young children aged 5–6 years.

The current study also has limitations. First, it does not allow to examine the causal relations between coding learning and other cognitive abilities. Second, as this study focused on children aged 5–6 years, further psychometric property tests are needed to see if the tool can also be used to measure the coding ability of other age groups. Finally, as the sample size of the current study was small ($n = 60$), the present findings need to be verified with bigger samples. However, despite the small sample size, our study has its own value because it is the first study, to the best of our knowledge, to test the reliability and validity of an assessment tool that is specifically developed to measure the coding ability of young children (i.e., 5–6 years). For children at this age range, it is a great challenge to conduct group testing without causing too much noise, especially when we intend to observe each child's emotional and behavioral response during testing. Meanwhile, unlike questionnaires or scales, our tool allowed children to finish each test after they had reached time limit, in order to improve user experience. Therefore, although the sample size in the current study is small, it provides important information regarding the design and feasibility for testing the coding ability of young children.

Conclusions

Improving coding ability is not the sole purpose of early coding education. A more important aim is to provide contexts where children can use the coding skills to solve problems in effective, efficient, and creative ways, through which we can promote the cognitive development of young children. To reach these goals, we developed a game-based and age-appropriate tool with good psychometric properties to assess the coding ability of young children aged 5–6 years. Moreover, as this tool does not rely upon any electronic media and does not pose challenges to the computer skills of young children, it has the potentiality to be widely applied to evaluate the inherent coding ability of young children as well as the learning outcome of educational coding programs.

Author's Contributions Lin Wang and Fengji Geng designed and contributed to data collection, analysis, and interpretation. They drafted and critically revised the manuscript. Xiaoin Hao and Donglin Shi contributed to data collection and analysis. Tengfei Wang and Yan Li contributed to study design, data interpretation, and manuscript revision.

Funding This work was supported by the MOE (Ministry of Education in China) Project of Humanities and Social Sciences (20YJA190002) and, the Fundamental Research Funds for the Central Universities, the Zhejiang University Educational Foundation Global Partnership Fund, and the National Natural Science Foundation of China (81773440).

Data Availability The data that support the findings of this study are available from the corresponding author, FG, upon reasonable request.

Declarations

Conflict of Interest and On behalf of all authors, the corresponding author states that there is no conflict of interest.

Ethnic Statement This study was approved by the IRB of Zhejiang University.

References

Arfé, B., Vardanega, T., Montuori, C., & Lavanga, M. (2019). Coding in primary grades boosts children's executive functions. *Frontiers in Psychology, 10*(2713). https://doi.org/10.3389/fpsyg.2019.02713.

Arfé, B., Vardanega, T., & Ronconi, L. (2020). The effects of coding on children's planning and inhibition skills. *Computers & Education, 148*, 103807. https://doi.org/10.1016/j.compedu.2020.103807.

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.

Bartlett, M. S. (1950). Tests of significance in factor analysis. *British Journal of Statistical Psychology, 3*(2), 77–85. https://doi.org/10.1111/j.2044-8317.1950.tb00285.x.

Bers, M. U. (2018). Coding, playgrounds and literacy in early childhood education: The development of KIBO robotics and ScratchJr. *IEEE Global Engineering Education Conference (EDUCON), 2018*, 2094–2102. https://doi.org/10.1109/EDUCON.2018.8363498.

Bers, M. U, González-González, C., & Armas-Torres, M. B. (2019). Coding as a playground: Promoting positive learning experiences in childhood classrooms. Computers & Education, 138, 130–145. https://doi.org/10.1016/j.compedu.2019.04.013.

Çiftci, S., & Bildiren, A. (2020). The effect of coding courses on the cognitive abilities and problem-solving skills of preschool children. *Computer Science Education, 30*(1), 3–21. https://doi.org/10.1080/08993408.2019.1696169.

Clements, D. H. (1995). Teaching creativity with computers. *Educational Psychology Review, 7*(2), 141–161. https://doi.org/10.1007/BF02212491.

Dagienė, V., & Sentance, S. (2016). It's computational thinking! Bebras tasks in the curriculum. In A. Brodnik & F. Tort (Eds.), *9th international conference on informatics in schools: Situation, evolution, and perspectives* (pp. 28–39). Springer International Publishing.

Di Lieto, M. C., Inguaggiato, E., Castro, E., Cecchi, F., Cioni, G., Dell'Omo, M., Laschi, C., Pecini, C., Santerini, G., Sgandurra, G., & Dario, P. (2017). Educational robotics intervention on executive functions in preschool children: A pilot study. *Computers in Human Behavior, 71*, 16–23. https://doi.org/10.1016/j.chb.2017.01.018.

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.

Heintz, F., Mannila, L., & Färnqvist, T. (2016). A review of models for introducing computational thinking, computer science and computing in K-12 education. *IEEE Frontiers in Education Conference (FIE), 2016*, 1–9. https://doi.org/10.1109/FIE.2016.7757410.

Hsu, Y. C., Irie, N. R., & Ching, Y. H. (2019). Computational thinking educational policy initiatives (CTEPI) across the globe. *TechTrends, 63*(3), 260–270. https://doi.org/10.1007/s11528-019-00384-4.

K-12 Computer Science Framework Steering Committee. (2016). *K-12 Computer Science Framework*. http://www.k12cs.org

Kafai, Y. B., & Burke, Q. (2014). Connected code: Why children need to learn programming. MIT press.

Kaiser, H. F. (1974). An index of factorial simplicity. *Psychometrika, 39*(1), 31–36. https://doi.org/10.1007/BF02291575.

Kim, S., Chung, K., & Yu, H. (2013). Enhancing digital fluency through a training program for creative problem solving using computer programming. *The Journal of Creative Behavior, 47*(3), 171–199. https://doi.org/10.1002/jocb.30.

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior, 72*, 558–569. https://doi.org/10.1016/j.chb.2017.01.005.

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.

Menon, D., Bp, S., Romero, M., & Viéville, T. (2019). Going beyond digital literacy to develop computational thinking in K-12 education. In D. Linda (Ed.), *smart pedagogy of digital learning*. Taylor&Francis (Routledge). https://hal.inria.fr/hal-02281037

Pardamean, B., Evelin, E., & Honni, H. (2011). The effect of logo programming language for creativity and problem solving. In *proceedings of the 10th WSEAS international conference on E-activities* (pp. 151–156). World scientific and engineering academy and society (WSEAS).

Pila, S., Aladé, F., Sheehan, K. J., Lauricella, A. R., & Wartella, E. A. (2019). Learning to code via tablet applications: An evaluation of daisy the dinosaur and Kodable as learning tools for young children. *Computers & Education, 128*, 52–62. https://doi.org/10.1016/j.compedu.2018.09.006.

Popat, S., & Starkey, L. (2019). Learning to code or coding to learn? A systematic review. *Computers & Education, 128*, 365–376. https://doi.org/10.1016/j.compedu.2018.10.005.

Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020). Computational thinking and mathematics using scratch: An experiment with sixth-grade students. *Interactive Learning Environments, 28*(3), 316–327. https://doi.org/10.1080/10494820.2019.1612448.

Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2017). 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.

Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2019). The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. *Journal of Educational Psychology, 111*(5), 764–792. https://doi.org/10.1037/edu0000314.

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

Torrance, E. P. (1966). *Torrance tests of creative thinking-norms-technical manual research edition-verbal tests, Forms A and B-figural tests, Forms A and B*. Personnel Press

UK Bebras. (2018). International challenge on informatics Computational Thinking. Retrieved October 1, 2019, from https://challenge.bebras.uk/index.php?action=anon_join&grp_id=2131

Wechsler D. 1987. *Weschler memory scale-Revised*. The Psychological Corporation, San Antonio

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 a: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717–3725. https://doi.org/10.1098/rsta.2008.0118.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.