An Exploration of Three-Dimensional Integrated Assessment for Computational Thinking

Abstract

Computational thinking (CT) is a fundamental skill for students, and assessment is a critical factor CT in education. However, there is a lack of effective approaches to CT assessment. Therefore, we designed the Three-Dimensional Integrated Assessment (TDIA) framework in this paper. The TDIA has two aims: one was to integrate three dimensions (directionality, openness, and process) into the design of effective assessment tasks; and the other was to assess comprehensively the three dimensions of CT including computational concepts, practices, and perspectives. Guided by the TDIA framework, we designed three pairs of tasks: closed forward tasks and closed reverse tasks, semi-open forward tasks and semi-open reverse tasks, and open tasks with a creative design report and open tasks without a creative design report. To further confirm each task's applicability and its advantages and disadvantages, we conducted a test experiment at the end of the autumn semester in 2014 in a primary school for three weeks. The results indicated: (a) the reverse tasks were not more superior than the forward tasks; (b) the semi-open tasks and the open tasks were more effective than the closed tasks, and the semi-open tasks had higher difficulty and discrimination than the others; (c) the self-reports provided a helpful function for learning diagnosis and guidance; (d) the scores had no significant difference between the schoolboys and the schoolgirls in all six tasks; and (e) the six tasks' difficulty and discrimination were all acceptable, the semi-open tasks had higher difficulty and discrimination than the others. To effectively apply them, the following suggestions for teachers to design computational tasks are proposed: motivaing students' interest and enthusiasm; incorporating semi-finished artifacts; involving learning diagnosis and guidance; and including multiple types of tasks.

Keywords

Primary education, Improving classroom teaching, Programming and programming languages, Computational thinking, Assessment

Introduction

Computational thinking (CT) is a fundamental skill for almost everyone, not just for computer scientists. In addition to reading, writing, and arithmetic, we should add CT to every child's analytical ability (Wing, 2006). Therefore, computer technologies should move towards being viewed as tools for thinking, learning, and creating (Burke & Kafai, 2014; Caperton, 2010). Since the publication of the influential article written by Wing (2006), there has been a growing focus on the concept of CT both within and outside of computer science. For examples, National Research Council (NRC) organized two workshops in 2009 and 2010; The International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA) collaborated with leaders from higher education, industry, and K–12 education to develop an operational definition of CT in 2011; British Department for Education issued a national curriculum: Computing Programmes of Study for key stages 1-4 in 2013.

Educators and researchers, meanwhile, are conscious of a rigorous problem, which is how to describe and assess students' CT in educational practices. Kolodner argued that "the CT community needs to identify exactly what is meant by CT to decide what learners should learn and to assess and evaluate what learners know, what they can do, and their attitudes and capabilities with respect to CT" (NRC, 2011, p.53). We can even say that CT has little hope of making its way successfully into any K–12 curriculum without paying attention to assessment (Grover, 2013). Therefore, more and more researchers highlight the importance of student evaluation for pedagogical purposes (NRC, 2011).

Some researchers have explored the assessment approaches for CT (Brennan & Resnick, 2012; Fields, Searle, Kafai, & Min, 2012; Webb, 2010; Werner, Denner, Campe, & Kawamoto, 2012). However, there is a lack of effective approaches to comprehensively assessing CT.

Literature Review

Two bodies of literature to make the assessment of CT effective are available: one is to seek out an operational definition of CT, and the other is to figure out an appropriate assessment approach. This section will firstly review some representative studies related to CT and its assessment, and then followed by clarifying the research questions and goals.

Definitions of Computational Thinking

CT has been defined in many ways and it has encompassed broad debates (Grover & Pea, 2013; NRC, 2000). In general, there are three different viewpoints about CT.

The first most popular viewpoint is that CT is a problem-solving process. Wing has never digressed from this viewpoint when she described CT in different occasions. The first time, Wing (2006) described that CT involved solving problems, designing systems, and understanding human behaviors, by drawing on the concepts fundamental to computer science. Two years later, Wing (2008) thought that the nuts and bolts in CT were defining abstractions, working with multiple layers of abstraction and understanding the relationships among the different layers. In 2011, Wing changed her description again. She said that CT was the thought processes involved in formulating problems and their solutions so that the solutions could be represented in a form that could be effectively carried out by an information-processing agent. Many researchers agreed with her core viewpoint and described CT in different ways. For example, Tinker argued that the core of CT was "the ability to break big problems into smaller ones until one can automate the solutions of those smaller problems for rapid response" (NRC, 2011, p.66). Wuff suggested that CT was primarily about processes. He noted that "other areas of science focus on physical objects, whereas CT focuses on processes and abstract phenomena that enable processes" (NRC, 2010, p.11). Bers, Flannery, Kazakoff, and

Sullivan (2014) developed a curriculum which specifically fostered CT skills including: problem representation; systematic ability in generating and implementing solutions; exploring multiple possible solutions; and problem-solving on multiple levels. Israel, Pearson, Tapia, Wherfel, and Reese (2015) utilized the term computational thinking to refer to students using computers to model their ideas and develop programs that enhance those programs. In the operational definition of CT for K-12 Education (ISTE & CSTA, 2011), CT was described as a problem-solving process. In brief, problem-solving represents the core function of CT, in which the key operations are abstracting and modeling, and automatically execute the result of abstracting and modeling.

The second viewpoint is that CT is an important form of expression. Although the first viewpoint is very popular, other researchers, however, believe that CT is more than solving problems. Wolz supported this view by claiming that CT was as essential a skill as reading, writing, and other basic language arts skills, pointing out that "programming is a language for expressing ideas. You have to learn how to read and write that language in order to be able to think in that language" (NRC, 2010, p.13). Resnick of the MIT Media Lab concurred, arguing that "CT is more than programming, but only in the same way that language literacy is more than writing" (NRC, 2010, p.13). Kolodner also believed that "for most people, CT means expressing oneself by utilizing computation fluently" (NRC, 2011, p.55). And Abrahamson saw CT as "the use of computation-related symbol systems (semiotic systems) to articulate explicit knowledge and to objectify tacit knowledge, to manifest such knowledge in concrete computational forms, and to manage the products emerging from such intellectual efforts" (NRC, 2010, p.13).

The third viewpoint is the three-dimension framework of CT proposed by Brennan and Resnick (2012). This framework included: (a) computational concepts (the concepts designers engage with as they program, such as sequences, loops, parallelism, events, conditionals, operators, and data); (b) computational practices (the practices designers develop as they engage with the concepts, such as being incremental and iterative, testing and debugging, reusing and remixing, and abstracting and modularizing); and (c) computational perspectives (the perspectives designers form about the world around them and about themselves, such as expressing, connecting, and questioning). This framework has attracted many researchers' attention and been cited frequently in literature in recent years.

Looking at the above viewpoints, we can find that each of them has unique perspectives. The first viewpoint addresses procedure analysis and process design which stemm from procedure oriented programming. Therefore, this viewpoint can be called as process thinking for short, since CT is regarded as the process of problem-solving. The second viewpoint stresses CT in the background of modern digital media. As information expression is mainly for visual media, we can call it visualization thinking. The last one builds an integrated framework based on visualization thinking and the process thinking, which not only emphasizes the social attribute of CT, but also has strong operability in education practices.

In view of its merits, the paper adopts the three-dimension framework of CT proposed by Brennan and Resnick (2012), and makes the following revisions:

- Computational concepts: objects, instructions, sequences, loops, parallelism, events, conditionals, operators, and data
- Computational practices: planning and designing, abstracting and modeling, modularizing and reusing, iterative and optimizing, and testing and debugging
- Computational perspectives: creative and expressing, communicating and collaborating, and understanding and questioning

Assessment of Computational Thinking

One important challenge in assessing learners' CT is the difficulty in evaluating the problem-solving ability with pencil and paper. The ability of identifying, debugging, and solving these problems is at the crux of being able to fulfill a computational task (Resnick, Berg, & Eisenberg, 2000). Therefore, many researchers seek for evidence-centered design (Mislevy, Almond, & Lukas, 2003; Mislevy & Haertel, 2006) for assessment tasks.

Schwarz, Reiser, Davis, Kenyon, Achér, Fortus, and Krajcik (2009) used a variety of methods to obtain such evidence including written pre-test and post-test items involving scientific modeling, reflective interviews about modeling practice with students, and in-person or videotaped observations of in-class student interactions. Webb (2010) designed an authentic assessment with five troubleshooting scenarios in Frogger game. Results suggested that troubleshooting scenarios could be an effective way for assessing student fluency in computer programming and computer-based problem solving. Fields et al. (2012) evaluated students' engineering and programming skills as they debugged prebuilt faulty e-textile projects with the LilyPad Arduino. Findings indicated that faulty e-textile projects were not only promising tools for evaluating students' learning of designing with e-textiles but also valuable learning tools, especially when peer collaboration was taken into account. Werner, Denner, Campe, and Kawamoto (2012) used predesigned semi-finished programming artifacts to evaluate students' CT concepts. They designed three closed tasks in an Alice scenario, which were proven effective. Brennan and Resnick (2012) described three ways to assess the development of CT in young people who were engaged in design activities with Scratch based on the three-dimension framework of CT. The first approach was project portfolio analysis,which was suitable for assessing computational concepts. The second approach was an artifact-based interview, which was proper for assessing computational concepts, computational practices and computational perspectives, but it was too time-consuming in practice. The third approach was design scenarios. They developed three sets of Scratch projects with increasing complexity. Within each set, there were two projects each including two tasks, one was to fix a bug, and the other was to remix the project by adding a feature. This approach was suitable for assessing computational concepts and computational practices.

Therefore, three conclusions can be summarized from the reviewed literature: (a) most researchers thought error correction or troubleshooting tasks were effective approaches to CT assessment; (b) tasks embedded in semi-finished projects could not only assess students' real level of CT, but also be easy to implement on a large scale;

and (c) it was not sufficient to assess students' CT depending on summative task evaluation only, but process evaluation was also needed to investigate and interview student's process of finishing a task. In brief, a constructionism-based problem-solving learning environment, with information processing, scaffolding and reflection activities, could be designed to foster and assess CT (Lye & Koh, 2014).

Our research aimed to produce an assessment model that integrates the above three notions. Some problems, however, need to be discussed and resolved further for task design:

First of all, there is a lack of a feasible combination of tasks to assess all of the three dimensions of CT, and most tasks could only assess computational concepts or a small fraction of the computational practices. However, K-12 intervention studies centering on computational practices and computational perspectives could be conducted in the regular classroom (Lye & Koh, 2014). Is there any combination of tasks to comprehensively assess CT?

Secondly, research results demonstrate that error correction or a troubleshooting task (the paper also names it as the reverse task) is an effective approach to assess CT. There is, however, still a lack of research data to support the claim that the reverse tasks are more effective than the same tasks without errors (the paper also names it as the forward task). That is to say, is the reverse task more effective than a forward task when these two tasks have the same target? For example, a task is to write a code segment to make one singer sing while she is moving in a semi-finished programming project; the reverse task is a semi-finished programming project, which has an incorrect codes to make the singer sing after she has moved, but the forward task is a semi-finished programming project that has no code to make the singer sing or move. It is worthy to investigate the superiority of the reverse task because problem-solving in CT always requires testing and debugging for reducing potential errors. Therefore, we need data to prove or falsify it.

Thirdly, research results have suggested that the semi-finished artifact provides a real task context to support the CT assessment. Many tasks designed by researchers look like different in different semi-finished artifacts, but actually, few differences existed among these tasks besides the difficulty and theme. In other words, these tasks were essentially homogeneous. Furthermore, these studies did not propose any clues for designing different tasks. How can we make the tasks really different and effective to assess different dimensional CT? Are there any clues that can be followed for task design?

Fourthly, research results show that students' thoughts in the process of finishing tasks are very important for assessing their CT. Therefore, artifact-based interviews with students become a possible approach, but conducting retrospective or real-time interviews is time consuming. Furthermore, students would have selective retention and ideally repeat at several points for a developmental portrait in the interviews. In addition, this kind of interviews could not provide learning diagnosis and guidance in time. Hence, are there any more effective and economic strategies for conducting interviews?

Conceptual Framework for Designing Tasks

Although existing studies did not describe clearly the clues to design assessment tasks, some implications were found in the tasks designed by various researchers.

At first, we could extend tasks from reverse tasks to forward tasks as mentioned above. Therefore, the first dimension is directionality (forward/reverse).

Secondly, we can easily find that most of the tasks designed by researchers are closed. That is to say, the tasks have a defined outcome and a defined process solely. The closed task is easy to carry out in assessment practices. It is, however, inadequate to assess CT. Are there different levels of openness in different tasks? Based on the taxonomy from task's target outcome and problem-solving process, we can find three types of tasks: (a) The closed task with a defined outcome and a defined process solely, which target is single and specific, and only a sole method can do it. This type of task is suitable for assessing computational concepts; (b) The semi-open task with a defined outcome solely and an undefined or open process, which target is single and specific, but there are different methods or combinations of methods to do it. This type of task is suitable for assessing computational concepts and computational practices; (c) The open task (or creative task) with an open outcome and open process, which is not infinitely open in order to assess easily, but students can freely add new functions or scenarios based on a semi-finished projects. Therefore, this type of task has more feasibility than that starting from zero. We believe this type of task fits to assess computational concepts, computational practices and computational perspectives in principle. These three types of tasks reflect different cognitive demands like the Chain of Cognitive Accomplishments in programming instruction proposed by Linn (1985) who thought that the chain included three links: (a) comprehension: students understand programs and are able to make small changes to single program instructions This is like the first type of task; (b) design: students build programs from collections of patterns and use procedural skills to combine these patterns to solve problems. This is like the second type of task; and (c) problem solving: students learn transferable problem-solving skills and use them in new and different formal systems. This is like the third type of task.

Finally, as mentioned above, the students' thoughts in the process of finishing tasks are very important for assessing their CT and provide aid for their learning. Therefore, we need an approach to get information from the process. We believe that student's self-report is an effective and efficient approach. Taking account of different types of tasks, we can divide a self-report into two types. One is a reflection report which describes the questions students encounter and methods they use, and every student should complete it after he/she finishes a task. The other is a creative design report involved in the open task only, which is used to describe one's design idea and flow chart before making an artifact. Because our research subjects are primary school students, we design the reflection report and the creative design report into semi-structured tables (refer to the next section). However, there is a question to discuss further for the creative design report. Does the creative design report have other functions except indicate student's CT in the process of finishing an open task? We suppose that it may provide the support to students to carry out a task like a scaffold. The creative design report is an initial and unfinished scaffold, but it may become a complete scaffold after a student fills it out by himself. Certainly, this scaffold is something different from a learning scaffold a teacher has made (Winnips, 2001). Consequently, is it really helpful to improve student's creative artifact? There is no answer to it in existing studies. In order to answer it, we also divide the open tasks into two types, one is to fill out a creative design report, and the other is not.

In summary, we focused on three dimensions in the design of assessment tasks: directionality (forward/reverse), openness (three types of openness), and process (self-report). Based on these dimensions, we design six tasks as follows:

(a) The closed forward task, which is an unfinished task with a defined outcome and a defined process solely;

(b) The semi-open forward task, which is an unfinished task with a defined outcome solely and an undefined or open process;

(c) The closed reverse task, which is a troubleshooting task with a defined outcome and a defined process solely;

(d) The semi-open reverse task, which is a troubleshooting task with a defined outcome solely and an undefined or open process;

(e) The open task with a creative design report, which is a creative task with an open outcome and open process; and

(f) The open task without a creative design report, which is a creative task with an open outcome and open process.

Research Goal and Questions

We hoped that the three dimensions as clues for designing tasks based on the semi-finished programming project can provide an evidence-based description of young children's learning trajectories. Therefore, for investigating each task's applicability, its advantages and disadvantages, five research questions were to be answered:

(a) From the dimension of directionality, is there any difference between the forward tasks and the reverse tasks?

(b) From the dimension of openness, is there any difference among the closed tasks, the semi-open tasks and the open tasks?

(c) From the dimension of process, can the reflection report reveal the students' thoughts in the process of finishing tasks? And is the creative design report helpful to improve student's creative artifact?

(d) Do these tasks lead to unfair assessment in different genders?

(e) Do these tasks have acceptable difficulty and discrimination?

Research Design

To answer the above questions, we developed a school-based curriculum "learning to storytelling by programming" based on the three-dimension framework of CT. This curriculum used a 3D programming language Alice2.4 developed at Carnegie Mellon. Alice (http://www.alice.org/) is an easy-to-learn environment which allows users to build 3D virtual worlds. Instead of creating traditional text-oriented programs which display meaningless messages such as "Hello World" to the screen, Alice allows the programmer to create and manipulate interesting objects (such as an ice skater). Those objects can be programmed to execute highly visual, exciting actions (such as skate and twirl). Users can create interesting environments in a short period of time, thereby increasing their satisfaction and motivation to continue (Bishop-Clark, Courte, Evans, & Howard, 2007). The curriculum took approximately a semester (18 weeks, 40 minutes per week) in the fall semester 2014 at a primary school on grade 6, and used the teaching model of storytelling (Kelleher & Pausch, 2007).

Participants

Participants were the sixth grade pupils coming from a primary school in……(it is called "school C" in this paper). We selected randomly 4 of 8f classes in the sixth grade and 144 pupils participated in the experiment. These students had taken a LOGO course and a scratch course each for a semester on grade five, and took the Alice curriculum for a semester in grade six taught by the same teacher.

Table 1. Sample characteristics

Classes	Class size	Effective participants▲	Boys	Girls

A	43	39	22	17
B	43	37	20	17
C	42	35	19	16
D	41	33	19	14
Total	169	144	80	64

Note: ▲ Some students were absent from part tests because sick leave etc.

Materials

The materials of the test experiment included six tasks in semi-finished programming projects, reflection reports, creative design reports, rubrics and coding Schemes.

Six Tasks in Semi-finished Programming Projects. We designed six tasks according to the conceptual framework for designing tasks (see Table 2):

Task 1 and task 3 were a pair of closed tasks. Both of them had the same outcome, but task 1 was the forward one and task 3 was the reverse one (troubleshooting task).

Task 2 and task 4 also were a pair of semi-open tasks. Both of them had the same outcome, but task 2 was the forward one, task 4 was the reverse one. Tasks 1 to 4 had the same story context.

Task 5 and task 6 were the same open tasks in a same story context, which difference just lied in whether a creative design report needed to be filled out.

Table 2. Introduction to tasks 1 to 6

Types of tasks	Goal/Outcome	CT	Assessment
Task1: (the closed forward task)	***Task:*** make the small rabbit eats off a green cauliflower. ***Specification:*** the small rabbit has the motion of stooping to eat, and after which a cauliflower will be disappeared, then the small rabbit will recover to stand erectly.	sequences, testing and debugging	Process evaluation：reflection report; Summative evaluation:

Task 3: (the closed reverse task)	***Task:*** make the small rabbit eats off a green cauliflower. ***Specification:*** ditto. ***Troubleshooting:*** the small rabbit does not stoop when it eats the cauliflower.		project evaluation.
Task 2: (the semi-open forward task)	***Task:*** make the big rabbit jumps to the front of red cauliflowers. ***Specification:*** the big rabbit's jump is fluent and coordinating, and both feet will swing when jumping.	sequences, loops, parallelism, modularizing, testing and debugging	ditto
Task 4: (the semi-open reverse task)	***Task:*** make the big rabbit jumps to the front of red cauliflowers. ***Specification:*** ditto. ***Troubleshooting:*** the big rabbit move to the front of red cauliflowers directly without jumping and swing.		ditto
Task 5: (the open task with a creative design report)	***Task:*** design a scenario to describe what is probably happed after the small rabbit became smaller, and need to fill out creative design report before working.	planning and designing, creative and expressing, abstracting and modeling, testing and debugging, iterative and	Process evaluation：reflection report, creative design report; Summative evaluation: project evaluation.

Task 6: (the open task without a creative design report)	***Task:*** design a scenario to describe what is probably happed after the small rabbit became smaller.	optimizing, modularizing and reusing, etc.	Process evaluation: reflection report; Summative evaluation: project evaluation.

In the practical test, we designed tasks 1 and 2 in a Alice project file (e.g., Project 1) since they were all forward tasks, tasks 3 and 4 in a Alice project file (e.g., Project 2) since they were all reverse tasks, and tasks 5 and 6 in a Alice project file (e.g., Project 3) since they were all open tasks.

Tasks 1 to 4 had the same story context: Two hungry rabbits, one is big, the other is small. They come to a beautiful garden. They see the cauliflowers in the garden...... (see Figure 1)

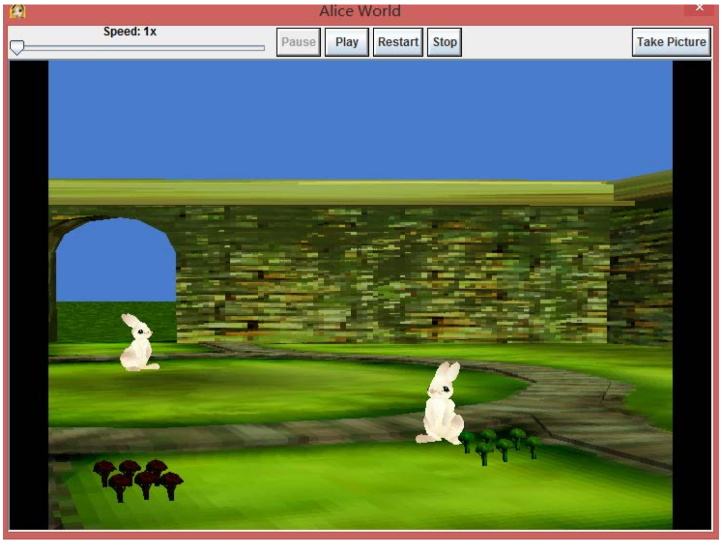

Figure 1. Scenario of rabbits' garden for tasks 1 to 4.

Tasks 5 and 6 had the same story context: Two hungry rabbits, one is big, the other is small. They come to a beautiful garden with cauliflowers. The small rabbit eats a cauliflower and becomes weaker suddenly because she is poisoned. She cries for help...... (see Figure 2 and Figure 3)

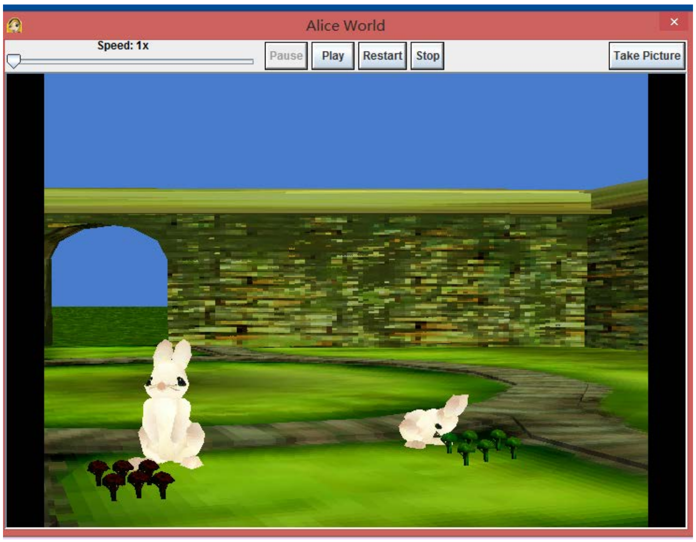

Figure 2. The small rabbit eats a cauliflower.

Figure 3. The small rabbit becomes smaller and cries for help.

Reflection Report. Every student must write a reflection report after he/she finished each task. All reflection reports were designed in a semi-structured table format so as to improve their operability. Different tasks, therefore, had different reflection report tables related to its task. For example, the reflection report for task 2 is shown in Table 3.

Table 3. Reflection report for task 2

Probable errors	Reasons	Your resolutions
Error 1. The rabbit moves to the cauliflowers directly		

without jumping and swing		
Error 2. The rabbit doesn't forward but jumping up and down		
Error 3. Both feet of the rabbit don't swing when jumping		
Error 4. Speed and distance of rabbits' jumping are not appropriate		
Error 5. The rabbit's jumping is not looping		
Error 6. The order of the rabbit's motion is wrong		

Note: Please fill out the errors which you have encountered.

Creative Design Report. Task 5 was an open task for students to fill out a creative design report before working on it. The creative design report included two parts: one was "My scenario" used to describe student's creative scenario to be developed, and the other was "My flow chart" used to describe the flow chart according to the scenario designed before.

Rubrics and Coding Schemes. In order to assess the students' CT comprehensively and find the obstacles in the process of problem-solving, we conducted process evaluation and summative evaluation for every task.

The process evaluation was used in the reflection reports and the creative design reports. All six tasks had the reflection reports, however, their codes were different because each task's reflection was different. Taking task 2 as an example, we coded it according to the coding scheme shown in Table 4. Only task 5 needed to fill out a creative design report, which was coded according to the scheme shown in Table 5.

Table 4. Process evaluation for task 2's reflection report

Probable errors	CT	Coding
Error 1. the rabbit moves to the cauliflowers directly without jumping and swing	objects, instructions, data, testing and debugging	from 1 to 5▲
Error 2. the rabbit doesn't forward but	Ditto	ditto

jumping up and down		
Error 3. both feet of the rabbit don't swing when jumping	Ditto	ditto
Error 4. speed and distance of rabbits' jumping are not appropriate	Ditto	ditto
Error 5. the rabbit's jumping is not looping	loops, testing and debugging	ditto
Error 6. order of the rabbit's motion is wrong	sequences, parallelism, testing and debugging	ditto
Error 7. additional code	Modularizing	from 1 to 5▲▲

Note:

▲ 1 for students who could not find the error in the project; 2 for students who could not figure out proper solution to correct the error; 3 for students who could not realize a solution; 4 for students who could resolve the error successfully; 5 for students who had no error in the project.

▲▲ 1 for students who did not define a method; 2 for students who defined an incorrect method; 3 for students who defined a correct method to make the rabbit jump, but called it unsuccessfully; 4 for students who defined a correct method to make the rabbit jump, but the effect was not very good, 5 for students who made the rabbit jump successfully by defining a method.

Table 5. Process evaluation for task 5's creative design report

Dimensions	CT	Coding
1.completeness of creative design report	planning and designing	from 1 to 5▲
2. rationality of scenario	planning and designing	ditto
3. creativity of scenario	creative and expressing	ditto
4. standardization of flow chart	abstracting and modeling	ditto

Note: ▲1 is very bad, 2 is relatively bad, 3 is ordinary, 4 is relative good; and 5 is very good.

In the summative evaluation, we gave each student a score for each task finished, which represented his problem-solving ability. A scale of 0 to 5 was applied for tasks 1 to 4, and a scale of 0 to 20 was applied for tasks 5 to 6. Taking task 2 as an example, we gave each student a score from 0 to 5 according to Table 6. And taking tasks 5 to 6 as example, we gave each student a score from 0 to 20 according to Table 7.

Table 6. Summative evaluation for task 2 (5 points)

Indicators & points	CT	Scores
1. the rabbit's jump is order(jumping up and jumping forwardly is paralleled, jumping down and jumping forwardly is paralleled) (1 point)	objects, instructions, parallelism, sequences, data	
2. both feet of the rabbit are swing when jumping (1 point)	objects, instructions, sequences	
3. the rabbit's jump and his feet's swing are paralleled (2 points)	parallelism, modularizing	
4. the rabbit is looping to jump forwardly (1 point)	loops	

Table 7. Summative evaluation for tasks 5 to 6 (20 points)

Indicators & points	Scores
1. Content (0 to 4 points): the story has a complete, clear, and healthy scenario.	
2. Creativity (0 to 4 points): the scenario is creative.	
3. Artistry (0 to 4 points): the layout of scene is reasonable, and the colors are harmony.	
4. Technology (0 to 8 points): the codes and program structure are correct and clear, and codes are reused by defining methods etc.	

Pilot Test and Measures

In order to ensure students can understand each task, and the feasibility of supporting materials, rubrics and the coding schemes, we carried out a pilot test. The subjects consisted of 38 students (16 girls) coming from a class of the sixth grade in school C. We revised some tasks, simplified the reflection report and the creative design report, and clarified the coding standards after the pilot test.

The formal test was scheduled at the end of the autumn semester in 2014, and lasted for three weeks in order to reduce the test-retest effect. Four classes of pupils participated in the formal test. Table 8 shows the schedule for the test experiment.

Table 8. Schedule for testing

Date	Class A and Class B	Class C and Class D
First week (40 minutes)	Project 1 (task 1 and task 2)	Project 2 (task 3 and task 4)
Second week (40 minutes)	Project 2 (task 3 and task 4)	Project 1 (task 1 and task 2)
Third week (40 minutes)	Project 3 (task 5 with a creative design report)	Project 3 (task 6 without a creative design report)

Results

Difference between the Forward Tasks and the Reverse Tasks

There were two forward tasks (task 1 & task 2), and two reverse tasks (task 3 & task 4) in the test experiment. These four tasks were actually two pairs of tasks (refer to the previous section). We gave each student a score for each task according to the summative evaluation mentioned above. Results indicated that the difference between the forward tasks and the reverse tasks was not statistically significant (see Table 9).

Table 9. Difference of scores between the forward tasks and the reverse tasks

	Pair tasks	N	Mean	Std. Dev.	t	Sig.
Pair 1	forward task: task 1	144	3.92	1.794	-1.104	0.272

	reverse task: task 3	144	4.08	1.784		
Pair 2	forward task: task 2	144	2.69	2.005	-0.224	0.823
	reverse task: task 4	144	2.72	1.967		

Difference among the Closed Tasks, the Semi-open Tasks and the Open Tasks

There were two closed tasks (pair1: task 1 & task 3), two semi-open tasks (pair 2: task 2 & task 4), and two open tasks (pair 3: task 5 & task 6) in our test experiment. For the purpose of comparison between pairs, we selected three groups of tasks which mean scores were closest in different pairs. We found that the mean score of the closed tasks was significantly higher than that of the semi-open tasks since the mean score of task 1 was significantly higher than that of task 4 (see Table 10). Similarly, we identified that the mean score of the closed tasks was significantly higher than that of the open tasks, and the mean score of the open tasks was significantly higher than that of the semi-open tasks. In other words, there was a significant difference of scores among the three pairs of tasks, with closed tasks＞open tasks＞semi-open tasks.

Table 10. Difference of scores in three groups of tasks

Groups		N	Mean	Std. Dev.	t	Sig.
Group 1	Task 1	144	3.92	1.794	7.087***	0.000
	Task 4	144	2.72	1.967		
Group 2	Task 1	144	3.92	1.794	2.378*	0.019
	Task 5&6	144	3.61▲	0.758		
Group 3	Task 4	144	2.72	1.967	-6.597***	0.000
	Task 5&6	144	3.61▲	0.758		

Note: * p < 0.05; ** p < 0.01; *** < 0.001.

▲ The students' scores in task 5 or task 6 had divided by 4, because each of tasks 5 to 6 had the full marks of 20, but each of tasks 1 to 4 had only 5 scores.

Functions of the Reflection Report and the Creative Design Report

In this study, the reflection report and the creative design report were used for the process evaluation. Most of the codes of the reflection report, however, could not be used to compare the difference among the six tasks because students possibly encountered different errors in different tasks. For example, the errors students encountered might be different between the forward tasks and the reverse tasks, and the number of errors students encountered was obviously unequal among the three pairs of tasks. Consequently, we could not say that some tasks were superior to the others in light of data from the process evaluation, but it really provided a good approach to help teachers find students' thinking barriers especially in the computational practices. Taking the reflection reports of task 1 and task 2 as an example, the students' thinking barriers focused on code 1 and code 2 in 4 errors of task 1 (see Figure 4), which indicated that these students could not discover errors or figure out the correct methods to solve problems; but in task 2, the students' thinking barriers were discrete (see Figure 5), which indicated that the teacher should provide personalized guidance for these problem students. Certainly, we could also count each class or each student's codes of the process evaluation, and provided evidence for the teacher to improve teaching.

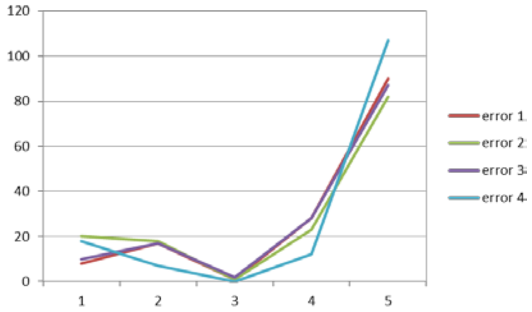

Figure 4. Errors in task 1

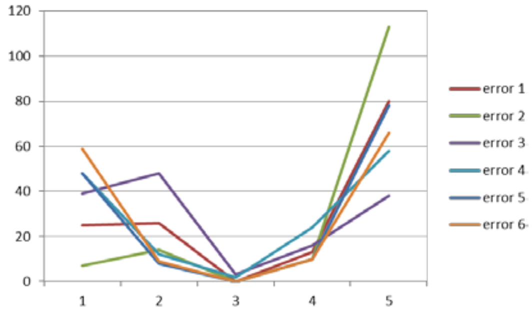

Figure 5. Errors in task 2

On the other hand, was the creative design report helpful for improving student's artifact? There was also no statistically significant difference between task 5 with a creative design report and task 6 without a creative design report, although the mean score of task 5 was slightly higher than that of task 6 (see Table 11). However, as students were assigned scores according to the rubric (see Table 7), we further compared students' scores in the light of the dimensions listed in the rubric. The result indicated that the mean score of task 5 in the technology dimension was significant higher than that of task 6 (see Table 12). Meanwhile, we found an obvious difference of students' program codes between task 5 and task 6, and the program codes were more complex and diverse in task 5 than in task 6. Actually, the quality of the creative design report and the scores of task 5 were positively correlated (see Table 13). Therefore, we could conclude that the creative design report as a task scaffold may provide effective technical support for students.

Table 11. Difference of scores in task 5 and task 6

	Pair tasks	N	Mean	Std. Dev.	t	Sig.
Pair 3	open task: task 5	74	14.86	2.844	-1.730	0.086
	open task: task 6	70	14.00	3.138		

Table 12. Difference of dimensions' scores between task 5 and task 6

Dimensions	Tasks	N	Mean	Std. Dev.	t	Sig.
Content	Task 5	74	3.74	0.621	-1.692	0.093
	Task 6	70	3.56	0.694		
Creativity	Task 5	74	2.53	0.687	-1.234	0.219
	Task 6	70	2.39	0.687		
Artistry	Task 5	74	3.18	0.783	-0.039	0.969
	Task 6	70	3.17	0.510		
Technology	Task 5	74	5.42	1.588	-2.088*	0.039
	Task 6	70	4.89	1.470		

Note: * $p < 0.05$; ** $p < 0.01$.

Table 13. Correlation between task 5 and the creative design report

Task	Dimensions of the creative design report	N	r	Sig.
Task 5	Completeness	74	0.667***	0.000
	Fluency	74	0.630***	0.000

Creativity	74	0.637***	0.000
Standardization	74	0.665***	0.000
Total	74	0.737***	0.000

Note: * $p < 0.05$; ** $p < 0.01$; *** $p < 0.001$.

Gender Difference in the Six Tasks

Injustice of evaluation would exist if gender difference was significant in scores of tasks. Therefore, we needed to know whether the six tasks had the potential risk or not. From the summative evaluation, there was no any significant difference of scores between the schoolboys and the schoolgirls in all six tasks (see Table 14); from the process evaluation, only two errors (Task 2, error 2; Task 5&6, error 1) in the six tasks had significant difference of frequency between the boys and the girls (see Table 15).

Table 14. Gender difference of scores in the six tasks

Tasks	Gender	N	Mean	Std. Dev.	t	Sig.
Task 1	M	80	4.00	1.779	0.570	0.570
	F	64	3.83	1.822		
Task 2	M	80	2.86	2.030	1.172	0.243
	F	64	2.47	1.968		
Task 3	M	80	4.19	1.707	0.782	0.435
	F	64	3.95	1.881		
Task 4	M	80	2.88	1.964	1.090	0.278
	F	64	2.52	1.968		
Task 5	M	41	14.37	3.520	-1.539	0.128
	F	33	15.48	2.502		
Task 6	M	39	14.41	2.953	1.362	0.178
	F	31	13.48	2.657		

Table 15. Gender difference in the process of finishing the six tasks

Probable errors	Gender	Options & Frequency	χ^2

		1	2	3	4	5	
Task 1, error 1	M	4	7	1	14	54	3.394
	F	4	10	0	14	36	
Task 1, error 2	M	11	8	1	13	47	1.814
	F	9	10	0	10	35	
Task 1, error 3	M	6	10	2	13	49	2.719
	F	4	7	0	15	38	
Task 1, error 4	M	8	3	0	5	64	3.080
	F	10	4	0	7	43	
Task 2, error 1	M	14	12	0	5	49	3.522
	F	11	14	0	8	31	
Task 2, error 2	M	1	10	0	1	68	15.640**
	F	6	4	0	9	45	
Task 2, error 3	M	17	28	2	7	26	6.012
	F	22	20	1	9	12	
Task 2, error 4	M	25	5	1	14	35	1.811
	F	23	7	1	10	23	
Task 2, error 5	M	26	4	0	7	43	0.988
	F	22	4	0	3	35	
Task 2, error 6	M	30	4	0	6	41	2.257
	F	29	5	0	5	25	
Task 3, error 1	M	0	11	1	68	0	1.095
	F	0	11	0	53	0	
Task 3, error 2	M	2	15	1	62	0	0.899
	F	2	13	0	49	0	
Task 3, error 3	M	10	2	0	0	68	0.360
	F	10	2	0	0	52	
Task 4, error 1	M	5	22	2	50	1	2.046
	F	3	23	1	37	0	

Task 4, error 2	M	1	2	0	0	77	0.967
	F	0	1	0	0	63	
Task 4, error 3	M	8	33	1	36	2	2.710
	F	5	30	2	27	0	
Task 4, error 4	M	25	0	0	0	55	1.366
	F	26	0	0	0	38	
Task 4, error 5	M	29	1	0	1	49	2.220
	F	25	1	1	0	37	
Task 4, error 6	M	37	0	0	1	42	2.129
	F	31	1	0	0	32	
Task 5&6, error 1	M	2	18	2	32	26	17.408**
	F	0	21	0	38	5	
Task 5&6, error 2	M	9	1	4	17	49	0.968
	F	5	1	2	64	43	

Note: * $p < 0.05$; ** $p < 0.01$.

Difficulty and Discrimination in the Six Tasks

We also calculated the difficulty and discrimination of each task (see Table 16), as the difficulty and the discrimination are two important quality indices for testing tasks.

On the difficulty, there was only a slight difference within the three pairs (task 1 and task 3, task 2 and task 4, and task 5 and task 6), but a significant difference among the three pairs. The most difficult tasks were the semi-open ones, and the easiest tasks were the closed ones. On the discrimination, the difference within and among the three pairs was similar to that reported on difficulty. The semi-open tasks had the highest discrimination, and the open tasks had the lowest discrimination.

Table 16. Six tasks' difficulty and discrimination

Pair	Tasks	Mean	Std. Dev.	Difficulty	Discrimination
Pair 1	Closed task: task 1	3.92	1.794	0.78	0.74
	Closed task: task 3	4.08	1.784	0.82	0.67

Pair 2	Semi-open task: task 2	2.69	2.005	0.54	0.99
	Semi-open task: task 4	2.72	1.967	0.54	0.96
Pair 3	Open task: task 5	14.86	2.844	0.74	0.34
	Open task: task 6	14.00	3.138	0.70	0.34

Discussion

Researchers commonly believe that reverse tasks (troubleshooting tasks) are more effective approaches to CT assessment (e.g., Brennan & Resnick, 2012; Fields et al., 2012; Webb, 2010; Werner et al., 2012). However, there was no significant difference between the forward tasks and the reverse tasks in this research. The result was consistent with our prejudgment that problem-solving in CT always requires testing and debugging for reducing potential errors. That is to say, no reverse tasks were superior to the forward tasks. The troubleshooting preference may be just a preconception and bias in researchers.

Most of the assessment tasks designed by researchers (e.g., Fields et al., 2012; Webb, 2010; Werner et al., 2012) were closed. In this paper, we extended the types of tasks in openness, and found that the semi-open tasks and the open tasks could assess more dimensions of CT than the closed tasks. In addition, they were equally easy to carry out in practices by following the rubrics designed in advance.

As an approach of process evaluation, the reflection report was easier to use than interviews in-person or videotaped observations. It also revealed that the students' thoughts in the process of finishing tasks could help teachers find students' thinking barriers especial in the computational practices. Certainly, the written reflection report had some limitations. For example, students tended to describe their result of thinking, but omitted the complicated process of thinking and solving problems.

The creative design report produced by student was different from the learning scaffold the teacher made. But the result indicated that it also could provide an effective technical support for students. Because the creative design report included "My scenario" and "My flow chart", filling out the creative design report was as essential as modeling. The modeling process is an important factor for a student to finish an open task. Blikstein (2010) compared two groups of students who were asked to sketch the relationship between atoms behavior and temperature change. The students with computational modeling experience drew and described a mechanism showing the behavior of the atoms as the temperature changed. However, students who had no experience instead drew a graphical curve showing the aggregate behavior of the atoms as the temperature changed. The result was consistent with this research. Nevertheless, the students' behavior of modeling happened in the teaching process in Blikstein's research, and happened in the test time in our research.

From the summative evaluation and process evaluation, we found that there was no significant difference between the schoolboys and the schoolgirls in all six tasks. In other words, there was no potential risk of gender discrimination in all six tasks.

According to the test standard accepted widely in education (Mitra, Nagaraja, Ponnudurai, & Judson, 2009), a task is considered difficult when the difficulty index value is less than 30% and the task is considered easy when the index value is greater than 80%; the task with 0.0–0.19 discrimination index is poor, 0.2–0.29 is acceptable, 0.3–0.39 is good, and >0.4 is excellent. Therefore, we concluded that the closed tasks were easy, the semi-open and the open tasks had medium difficulty; and the three pairs of tasks' discrimination were acceptable. Overall, the semi-open tasks had higher levels of difficulty and discrimination than the others. In fact, each of comprehensive assessment always had a combination of test items varied by content, difficulty and discrimination. In other words, the three types of tasks could be used in the same or different tests according to the actual requirement. Taking the closed tasks as an example, although this type of task was the easiest and just assessed the computational concepts, we mainly used it to encourage students to be self-confident.

Finally, the conceptual framework and the research results suggested that each task was applicable for assessing some aspects of CT (see Table 17). Actually, the learning of CT took place in different contexts, on different timescales, and with different structures and support – and these differences led to different tasks to assessment. Given that a single task was inadequate, we suggested that a combination of tasks could be more appropriate in the educational practice. In terms of combination, the Three-Dimensional Integrated Assessment (TDIA) framework had two meanings: one was to integrate the three dimensions for designing tasks, and the other was to assess comprehensively students' three dimensions of CT.

Table 17. Applicable tasks for assessing students' CT

	Computational concepts	Computational practices	Computational perspectives
The closed forward task	applicable	testing and debugging	N/A
The semi-open forward task	applicable	testing and debugging, modularizing and reusing	N/A
The closed reverse task	applicable	testing and debugging	N/A
The semi-open reverse task	applicable	testing and debugging, modularizing and reusing	N/A
The open task with a creative design report	N/A, lacking unified standard	applicable	creative and expressing , but theoretically for all
The open task without a creative design report	N/A, lacking unified standard	applicable, except planning and designing	creative and expressing , but theoretically for all

Conclusions

Our purpose of the research was to propose what types of tasks could be made accessible and meaningful for assessing students' CT. To that end, we presented the three dimensions and six types of tasks that could assess different dimensions of CT. The three dimensions for designing tasks included directionality (forward/reverse), openness (three types of openness), and process (self-report). These three dimensions provided an integrally conceptual framework to design assessment tasks. This framework made assessment tasks diversified, and extended the theoretical basis for designing assessment tasks.

In practice, we designed six types of tasks in a scenario of “rabbits’ garden” guided by the TDIA framework. The research results indicated:

(a) Students’ scores had no significant difference between the forward tasks and the reverse tasks, which mean no reverse tasks were superior to the forward tasks.

(b) Students’ scores had a significant difference among the closed tasks, the semi-open tasks and the open tasks, with closed tasks＞open tasks＞semi-open tasks. The semi-open tasks and the open tasks could assess more dimensions of CT than the closed tasks. In addition, they were equally easy to carry out in practice as the closed tasks.

(c) The reflection report and the creative design report were easy to carry out, and helped to provide learning diagnosis and guidance to help teachers find students’ thinking barriers especially in the computational practices. Moreover, the creative design report could provide an effective technical support for students to finish their project.

(d) The scores had no significant difference between the schoolboys and the schoolgirls in all six tasks.

(e) The six tasks’ difficulty and discrimination were all acceptable. Overall, the semi-open tasks had higher difficulty and discrimination than the others. The three types of tasks could be used in the same or different tests according to the actual requirement.

We also summarized the aspects of CT which each task was suitable for. To effectively apply them, the following suggestions for teachers to design computational tasks are proposed:

One is to motivate students’ interest and enthusiasm. We believe that the best tasks are those that are attractive to learners. The six tasks described in this paper were connected to the learners' interests and goals. The themes and the scenarios could be designed better by taking students' interest into account. This is a challenge for both the teacher and students.

Another is incorporating semi-finished artifacts. Assessment should involve the creation of meaningful projects, but it is difficult for students to create a new project starting from scratch and also difficult for the teacher to assess their CT. The previous studies and this research suggest that the semi-finished artifact can provide a real task context in which students' CT can be assessed. The different types of openness projects make assessment even richer, providing an opportunity for the teacher to see how students understand CT and finish semi-artifacts over time.

The third is to involve learning diagnosis and guidance because the best tasks must be useful to learners. CT is not a binary state at a single point of time, and any approach to assessment should strive to describe where a learner has been currently and might go (Brennan & Resnick, 2012). Adopting a formative assessment is necessary. It involves checking the three dimensions of CT at multiple points across the different types of tasks, and may also involve checking the process of evaluation like the self-reflection report and the creative design report. Given the limitation that the written self-reflection report cannot present conversation and dialogue, we should encourage students to reflect on their thinking and self-dialogue, which is a capacity important to develop as a self-regulating learner. Surely, we can encourage students to write the conversation with peers in the reflection report during the collaborative learning process.

The fourth is to include multiple types of tasks. As mentioned above, given that no single task is sufficient, we suggest a mixture of six types of tasks to create a new combination of assessment. This suggestion indicates that the TDIA framework has a two-fold meaning: integrating the three dimensions into task design, and comprehensively assessing students' three dimensions of CT.

Limitations and Future Research

In this research, the reflection report and the creative design report were used as approaches for process evaluation only. They have the potential to be converted into scores in summative evaluation in the future.

Open tasks could be used theoretically to assess all computational perspectives, but actually only the aspects of creativity and expression were assessed in this research because the students were asked to finish the open tasks independently. If they were asked to work in groups, their computational perspectives such as communicating and collaborating, understanding and questioning can be further assessed. We will further explore this possibility in future research.

Finally, we designed six tasks in a theme of “rabbits’ garden” only in this study. We will develop more themes to verify or extend our findings in future research.

Acknowledgments

The authors wish to acknowledge the helpful comments from anonymous reviewers.

References

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. Computers & Education, 72, 145-157.

Bishop-Clark, C., Courte, J., Evans, D., & Howard, E. V. (2007). A quantitative and qualitative investigation of using Alice programming to improve confidence, enjoyment and achievement among non-majors. Journal of Educational Computing Research, 37(2), 193-207.

Blikstein, P., & Wilensky, U. (2010). MaterialSim: A constructionist agent-based modeling approach to engineering education. In M. J. Jacobson, & P. Reimann (Eds.), *Designs for learning environments of the future* (pp.17-60). Springer US.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association*. Vancouver, Canada.

British Department for Education. (2013). Computing Programmes of Study for key stages 1-4. Retrieved from https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study.

Burke, Q., & Kafai, Y. B (2014). Decade of game making for learning: From tools to communities. In M. C. Angelides, & H. Agius (Eds.), *Handbook of digital games* (pp. 689-709). John Wiley & Sons.

Caperton, I. H. (2012). Toward a theory of game-media literacy: playing and building as reading and writing. In R. E. Ferdig, & S. de Freitas (Eds.), *Interdisciplinary advancements in gaming, simulations and virtual environments: Emerging trends* (pp. 1-16). PA: IGI Global.

Fields, D. A., Searle, K. A., Kafai, Y. B., & Min, H. S. (2012). Debuggems to assess student learning in e-textiles. In *Proceedings of the 43rd ACM technical symposium on computer science education* (pp. 699-699). NY: ACM Press.

Grover, S., & Pea, R. (2013). Computational Thinking in K–12: A review of the State of the Field. Educational Researcher, 42(1), 38-43.

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. Computers & Education, 82, 263-279.

ISTE&CSTA. (2011).Operational definition of computational thinking for K-12 education. Retrieved from http://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf.

Kelleher, C., & Pausch, R. (2007). Using storytelling to motivate programming.Communications of the ACM, 50(7), 58-64.

Linn, M. C. (1985). The cognitive consequences of programming instruction in classrooms. Educational Researcher, 14(5), 14-29.

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.

Mitra, N. K., Nagaraja, H. S., Ponnudurai, G., & Judson, J. P. (2009). The levels of difficulty and discrimination indices in type a multiple choice questions of pre-clinical semester 1, multidisciplinary summative tests. IeJSME, 3(1), 2-7.

Mislevy, R. J., Almond, R. G., & Lukas, J. F. (2003). A brief introduction to evidence-centered design. ETS Research Report Series, 1, 1-29.

Mislevy, R. J., & Haertel, G. D. (2006). Implications of evidence-centered design for educational testing. Educational Measurement: Issues and Practice, 25(4), 6-20.

NRC(2010). Report of a workshop on the scope and nature of computational thinking. Washington, D. C.: National academies press.

NRC(2011). Report of a workshop on the pedagogical aspects of computational thinking. Washington, D. C.: National academies press.

Resnick, M., Berg, R., & Eisenberg, M. (2000). Beyond black boxes: Bringing transparency and aesthetics back to scientific investigation. Journal of the Learning Sciences, 9(1), 7-30.

Schwarz, C. V., Reiser, B. J., Davis, E. A., Kenyon, L., Achér, A., Fortus, D., & Krajcik, J. (2009). Developing a learning progression for scientific modeling: Making scientific modeling accessible and meaningful for learners. Journal of Research in Science Teaching, 46(6), 632-654.

Webb, D. C. (2010). Troubleshooting assessment: an authentic problem solving activity for it education. Procedia-Social and Behavioral Sciences, 9, 903-907.

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). NY: ACM.

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.

Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society A, 366(1881), 3717-3725.

Wing, J. M. (2011). Research notebook: Computational thinking—What and why? The Link Magazine of Carnegie Mellon University. Retrieved from http://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why.

Winnips, J. C. (2001). Scaffolding by design: a model for www-based learner support. Universiteit Twente.