







# Improve University Humanities Students' Problem-Solving Ability Through Computational Thinking Training

Jim-Min Lin<sup>1</sup> (✉) , Zeng-Wei Hong<sup>1</sup> , Zong-Kun Song<sup>1</sup>, Wei-Wei Shen<sup>2</sup> ,  
and Wai-Khuen Cheng<sup>3</sup> 

<sup>1</sup> Department of Information Engineering and Computer Science, Feng Chia University,  
Taichung City, Taiwan

jimmy@fcu.edu.tw

<sup>2</sup> Department of Foreign Language and Literature, Feng Chia University, Taichung City, Taiwan

<sup>3</sup> Faculty of Information and Communication Technology, Universiti Tunku Abdul Rahman  
(UTAR), Kampar, Malaysia

**Abstract.** Humanities students are often considered to have lower problem solving ability and information application skills. This may be due to the fact that many humanities students receive basic computer education that only teaches word processing or other information software. In this paper, we want to investigate whether university humanities students' problem-solving ability could be improved through the computational thinking (CT) training. The experiment was divided into 2 phases. The first phase was to investigate whether the students' problem solving abilities could be affected by the CT education comparing with the training of MS-Office operating skills adopted in the traditional "Introduction to Computers" course. The result of this investigation showed that the experimental group received the CT training had significantly better achievement than the control group in both problem-solving ability and IT application ability.

**Keywords:** Computational Thinking (CT) · Problem-solving ability · Information application ability · Bebras challenge · Scratch programming

## 1 Introduction

In the information age, everyone should have a certain degree of information ability, including the ability to use computational tools to solve practical problems. In Taiwan, university humanities students are generally considered to be inadequate in Information Technology (IT) skills. The reasons may be: lack of interest, lack of IT skill or confidence; lack of logical thinking training for a long time in their school education; current computer-related courses do not attract students' interest; and few contact with the latest IT; and so on. It has always been a trouble for teachers who conduct computer-related courses, such as the "Introduction to Computers", in the liberal arts departments. Moreover, many humanities students have already given up the study of computer programming, or even the study of information-related knowledge. But these large groups

of humanities and social science students actually have good cultural literacy in language, literature, society, history and so on. If these students can be guided to integrate into the modern information society and give full play to their cultural abilities, they can enhance not only their own employability, but also their information skills. It can be helpful to the quality and connotation of the information industry. Therefore, how to improve these humanities students' IT concepts/mindset to solve their real problems in life, and then give full play to their ability to apply IT, will be a very meaningful task. In order to achieve the above-mentioned goals, "Computational Thinking (CT)" [1], which has become a trend in recent years, may be a feasible method and strategy.

Professor Jeannette M. Wing, a former Chair of the Department of Computer Science at Carnegie Mellon University (CMU), first proposed the term Computational Thinking (CT) in 2006. She proposed "CT is the use of basic computer science concepts to solve problems, design systems, and understand the thinking mode of human behavior." She stated that CT is also the basic qualities that everyone in modern society should have, as well as the computer science skills and knowledge necessary for future life [1]. She emphasized that the ability of computational thinking is as important as 3R (Reading, wRiting, and aRithmetics). It is not only a necessary ability for computer scientists, computer workers, or programmers, it is also the basic skill of every modern person. Computational thinking means a computerized problem-solving ability. This ability has four aspects [2]:

1. Decomposition: Disassemble data, workflow, or actual problems into small, operationally manageable parts.
2. Pattern Recognition: Observe phenomena such as the characteristic pattern, trend and overall pattern presented by the data.
3. Abstraction: Identify and establish the general principles of these patterns.
4. Algorithm Design: Design the implementation methods and steps to solve a problem or similar problems.

The computer introduction courses of the humanities department in Taiwan have always been teaching the operation and use of software tools, such as MS-Office, Web page creation, or simple APP programming design. The content is mostly boring input and operation following the teacher's examples. For students, there is only practice on skills, and no training on their problem-solving ability. Such a course does not motivate students on the one hand, and on the other hand, the effect of improving students' problem-solving ability is limited, and the expected results are not obtained. In general, students' course satisfaction is therefore not high. Therefore, the teachers are all trying their best to find a more suitable computer introduction course content for the humanities students. Therefore, if this study can confirm that the training of CT can achieve some help and effects for university humanities students, then one can consider, in the future, incorporating the content of the computer introduction course of the humanities department into the CT training and applications. On the other hand, the same CT training model can be transplanted to students of other disciplines, as well as the college students with low information learning achievements, so as to enhance the IT ability of college students; therefore, the results of this research can be of reference value for future IT education.

Therefore, for humanities students who are relatively inadequately trained in sciences such as mathematics, physics, and logic, the purpose of this paper is to study whether they can, by cultivating CT concepts, not only improve their ability to think, analyze, and find solutions to problems in life, but also improve their ability to think, analyze, and find solutions by using IT. This research hopes to take the social robot application system, a TA robot, as an example, to produce their own information tools designed and implemented by ourselves. In this research, we will take students of foreign language department as an example.

This paper is organized as the following: some related works are listed in Sect. 2; An experiment is reported in Sect. 3; The experimental results and discussion are addressed in Sect. 4. Finally, a brief conclusion is made in Sect. 5.

## 2 Related Works

The Computer Science Teachers Association (CSTA) and the International Institute of Educational Technology (ISTE) have defined a series of concepts encompassed by computational thinking and practice. These concepts are subdivided into data collection, data analysis, data representation and analysis, abstraction, analysis and model verification, automation, testing and verification, algorithm & process, problem decomposition, control structure, parallelization and analogy. Other scholars and organizations have also put forward their own definitions and opinions [3–5]. In addition, computational thinking is also a thinking process involved in formulating problems, so it is necessary to find computational steps and algorithms from its design and analysis to solve useful and more complex problems [3]. Korkmaz et al. further clarified that computational thinking can be defined as the necessary knowledge and skills, as well as the effective use of computers to illustrate the attitude of solving life problems [6].

Table 1 is based on the connotation of computational thinking proposed in the literature of many scholars [3–5, 7, 8], and the connotation of computational thinking that is currently generally accepted in private information technology education, such as: Google for Education Exploring Computational Thinking website [9] and Bitesize which is a website under the British Broadcasting Corporation BBC that provides free online learning resources for students [10].

It can be seen from Table 1 that the computational thinking proposed by scholars and the computational thinking proposed by private IT education resources have a high degree of overlap in Abstraction, Algorithm, and Decomposition, which means that These connotations are the consensus on the connotation of computational thinking in current research and education. Also, because the Ministry of Education (MOE) in Taiwan is also promoting learning computational thinking in education, the connotation of computational thinking that was used in this study should be as close as possible to the educational policy of Taiwan MOE.

**Table 1.** Connotation of computational thinking

	Wing [3]	Barr and Stephenson [4]	Selby and Woollard [7]	Google [9]	Bitesize [10]
Abstraction	✓	✓	✓	✓	✓
Algorithm		✓	✓	✓	✓
Automation	✓	✓			
Data		✓			
Decomposition		✓	✓	✓	✓
Evaluation			✓		
Generalization			✓		
Pattern recognition				✓	✓
Parallelization		✓			
Simulation		✓	✓		

### 3 Experiment

#### 3.1 Research Questions

We have two research questions raised in this research:

1. Can CT education improve university humanities students' problem solving abilities by comparing with the MS-Office operating skills training in the traditional "Introduction to Computers" course?
2. Can CT education improve university humanities students' information application ability by comparing with the MS-Office operating skills training in the traditional "Introduction to Computers" course?

#### 3.2 Participants

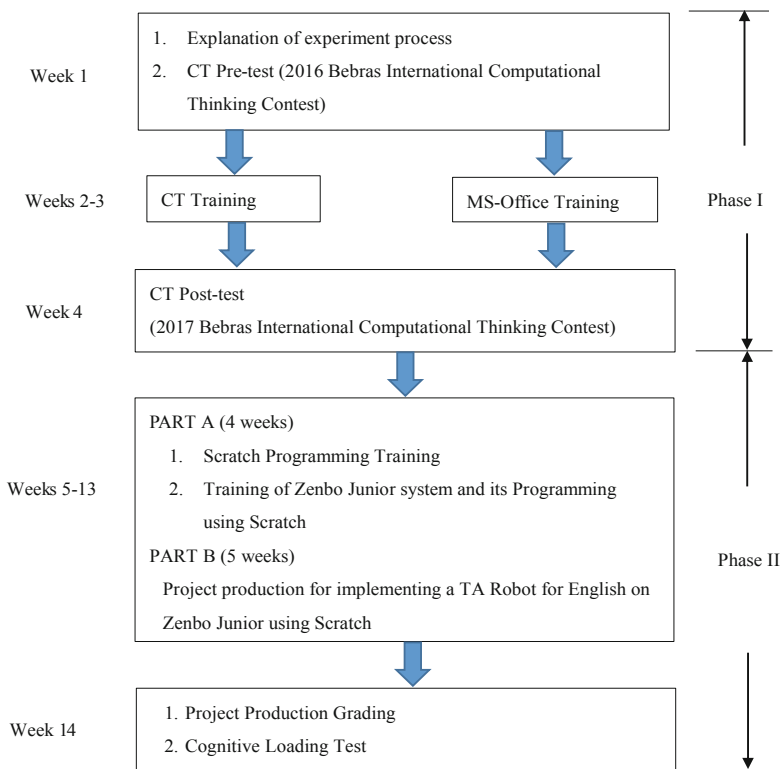
The experimental targets of this research were 20 students from the third and fourth grades (13 juniors and 7 seniors) foreign language department of a university in Taiwan. Before starting the experimental teaching, all participants took a pretest to assess their initial level of computing thinking ability, and then we used random grouping to divide them into the experimental group (EG) and the control group (CG). Levene's test was used to test the homogeneity of the variance. The test results showed no significant difference between the two groups' pretest, where  $p = .596$ . The equal variance was used, which means that the two groups were similar before the start of teaching, and the experiment can continue. Table 2 shows the Levene's test results of the two groups.

**Table 2.** Analysis confirms that there is no difference between the two groups before teaching

EG (N = 10)		CG (N = 10)		P
Mean	SD	Mean	SD	
139.50	42.911	100.56	25.427	.596

### 3.3 Procedure

Figure 1 shows the experimental procedure. The experiment was divided into two phases. The first phase is for the first research question, and while the second phase is for the second research question. The whole experiment lasted for 14 weeks, and one hour a week. The first week is the explanation of the experimental precautions, informing the students of their rights and obligations, and the CT pretest (and also a placement test). The CT pretest used the “2016 The Bebras International Computational Thinking Contest” (Bebras 2020) questions in the Senior Group (11th and 12th grade). In the training stage (Weeks 2–3), we had a CT training for experimental group compiled by



**Fig. 1.** Experimental process

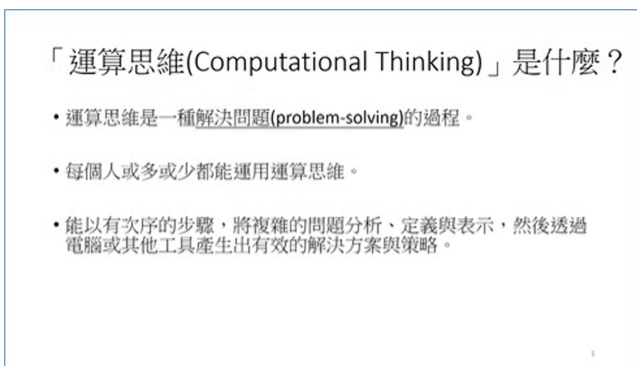
this research and implemented the MS-Office training for the control group. The MS-Office training is a common training in the course - “Introduction to Computer Science” for university humanity students in Taiwan. Week 4 was the CT posttest using the “2017 The Bebras International Computational Thinking Contest” (Bebras 2020) questions in the Senior Group (11th and 12th grade). Weeks 5–13 was for the test of CT concept application. We would like to know whether the CT concept is helpful for university humanity students in increasing their ICT application ability. It included four weeks of Scratch programming training and then 5 weeks of project production of implementing a Robot Teaching Assistant (TA) for English. In the last week, students were required to submit their project production demonstration, the reports, and presentations for scoring.

### 3.4 Materials

**Computational Thinking Training.** Since there is no existing material for university humanity students’ CT training, we decided to refer to documents in MOE, Taiwan, and then produce self-compiled CT materials, in forms of PowerPoint Slides [11, 12]. The contents of CT training materials included:

- CT basic concepts:
  - What is CT;
  - Why CT;
  - Introduction to CT components: Decomposition, Pattern Recognition, Abstraction, Algorithm Design;
- CT examples and applications that are related to humanity students’ life and knowledge;
- Basic programming concepts that are introduced in the Computational Thinking test (CTT): Sequence, Loops, Events, Parallelism, Conditionals, Data, and Operators [13].

Some material examples are shown in Figs. 2, 3 and 4.



**Fig. 2.** Example PPT slide of the teaching materials - Introduction to computational thinking

After introduction to CT, we explained the composition of CT to the students. The conceptual elements explained here were the decomposition, pattern recognition, abstraction, and algorithm design. Then we explained the details of each part one by one. In order to prevent students from having difficulty in understanding because they only listened to the explanation of the textual meaning of the lesson, we used examples close to daily life to help students to understand CT more easily.

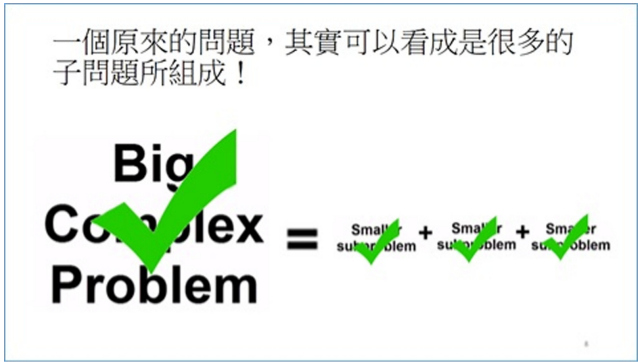


Fig. 3. Decomposition

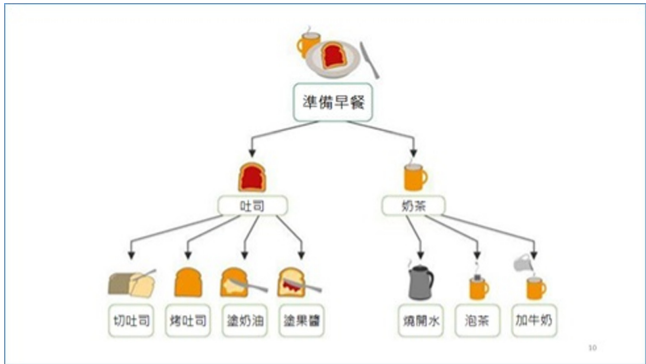


Fig. 4. Example of teaching slides of CT components

Figure 3 was used to introduce the concept of “decomposition” in CT. It explains that an original complicated problem could be divided into several smaller and easy-to-solve sub-problems such that a big and complicated problem could be conquered easily. To prevent students from just listening to the explanation and feeling that the concept is too abstract and difficult to understand, Fig. 4 was an example of decomposition presenting the preparation of breakfast in daily life. More supplementary examples close to daily life were given in the materials to help students understanding CT concepts.

**Scratch Training.** Scratch is regarded as a popular tool for cultivating elementary school students’ computational thinking. It uses visualization tools to realize program

construction through the simple building operation of building block instruction modules, and thus can effectively improve students' independent thinking ability. Since there have been lots of ready textbook for Scratch in Chinese, we choose one textbook that is appropriate for university humanity students.

### 3.5 Instruments

To know students' problem-solving ability in two groups before and after the CT training course, an assessment tool, the International Contest on Informatics and Computational Thinking (Bebras Contest), for measuring CT concepts was adopted. The Bebras competition originated in Lithuania. "Bebras" means "beaver" in Lithuanian, so the whole question type is based on beaver. What does Computational Thinking involve? From Bebras Web site: "The Bebras challenge promotes problem solving skills and Informatics concepts including the ability to break down complex tasks into simpler components, algorithm design, pattern recognition, pattern generalisation and abstraction. More about computational thinking" (Bebras 2020). The first Beblas Challenge was officially held in Lithuania in October 2004. There were 3470 students of different ages from 146 schools participated in the competition. From then to 2018, according to official statistics, 54 countries around the world have implemented this event. The competition accumulates nearly 3 million participants, and it is held only once a year, held by the locals of various countries, and more online competitions are used.

The difficulty age groups currently held in Taiwan are:

- Benjamin (fifth and sixth grade) (added since 2016)
- Cadet (seventh and eighth grade) (added since 2016)
- Junior (ninth and tenth grade)
- Senior (eleventh and twelfth grade)

For all age groups, there are 5 questions for each of the 3 difficulty levels: easy, medium and difficult, for a total of 15 questions. It is considered that the experimental students are college students and the highest level of the questions is only up to the high school stage, so a total of 10 questions with medium and difficult levels are selected to form test questions. The scoring method applied in this research was also based on the original contest scoring rules, but the part, difficult, was not included, and the full score was adjusted from 300 points to 225 points. This study used the 2016 and 2017 Taiwan Bebras Contest Senior Group questions as the Pretest and Posttest respectively to evaluate the computational thinking ability.

## 4 Results

### 4.1 Analysis of Computational Thinking Scores

In experiment phase 1, the experimental group implemented the computational thinking materials compiled by this research, while the control group implemented the general tradition of MS-Office training. After the experiment, checked by using Box plot [14],



there are 3 invalid samples outlier caused by the large difference with the values of other students in the experiment process. Therefore, there are only 17 valid samples: 8 students in experimental group and 9 students in control group. Due to the small number of students in the experiment, it is necessary to use the non-parametric test analysis. Two independent sample groups are statistically analyzed by the Mann-Whitney U test. The null hypothesis of this test is “the two groups have the same degree of CT performance”. The analysis results are shown in Table 3.

**Table 3.** Analysis of computational thinking scores

EG (N = 8)			CG (N = 9)			Mann-Whitney U test	
Mean	SD	Mean Rank	Mean	SD	Mean Rank	U	P [2*(single tail)]
141.25	36.228	11.75	100.56	25.427	6.56	14.000	.036*

Note: \*  $p < .05$

The results showed that the average rank (Mean Rank = 11.75) of the experimental group (N = 8) was greater than the average rank of the control group (N = 9) (Mean Rank = 6.56), with statistical significance ( $U = 14.000, p = .036 < .05$ ), that rejects the null hypothesis. This result shows that in the Bebras test performance, the experimental group was significantly better than the control group. It means that the problem-solving ability of experimental group with CT training was better than that of the control group with traditional MS-Office training.

#### 4.2 Analysis of Student Project Productions

The second phase of the experiment was to assess students' information application ability by observing how the students to apply their problem-solving skills with their majors in the Department of Foreign Languages to produce an IT-related project in which an English Teaching Assistant (TA) Robot should be designed and implemented by themselves. In the end, all students in both groups completed their robotics project. System demonstration and viewing of Scratch source programs will be performed to determine students' level of problem-solving and information application ability.

Scratch programming training has been commonly used to cultivating children's CT ability. In the research, we want to observe the performance difference in problem-solving between experimental group and the control group in which the former received CT training and Scratch programming training while the latter just received Scratch programming training only.

The grading standard is as shown in Table 4. The grading method is to directly observe the program codes submitted by the students, and award a score for each item corresponding to one of the four CT components. Each scoring item has a maximum of 5 points, and there are 20 points in total for four scoring items. Scoring was done by 3 IT experts separately, and then a final average score was obtained.

**Table 4.** Metrics for Scoring the Programming Part in Students' Projects

Evaluation item	Performance in corresponding programming	Example
Decomposition	Check the amount of functional blocks implemented in the program codes	Students can separate the small functions from the entire program instruction sequence, and then make these small functions into reusable functions
Pattern recognition	Check the amount of looping structures in the program codes	Students can streamline the sequential program/instructions into one or more looping blocks without redundant sequential process instructions
Abstraction	Check the amount of variables are defined and how complex the data structures are in the program codes	Check if the students can convert 5W: people (Who), events (What), times (When), places (Where), and things (Which) in real world into the corresponding program variables and data structures in the cyber world?
Algorithm design	Combination of the above three parts. Instructor can examine the core problem solving process in student's programs and check if the program codes matching with concepts shown above	In designing a 9 * 9 multiplication table, an instructor could check: <ul style="list-style-type: none"> <li>• If the data types of the multiplicand and the multiplier is properly defined?</li> <li>• If the operation of multiplying the multiplicand by the multiplier are put into the structure of a double-loop program codes</li> </ul>

The program codes were measured by mainly observing whether the students had used the program instructions learned in the Scratch training and practice. Using the instruction blocks that were not taught in training and practice didn't affect the score of this part, so as to prevent these additional performances from deviating from the scoring standard.

After all the students' programming works were scored, the scores of the two groups were then analyzed by Mann-Whitney U test. Table 5 shows the results of the analysis. The results showed that the average rank of the experimental group ( $N = 8$ ) (Mean Rank

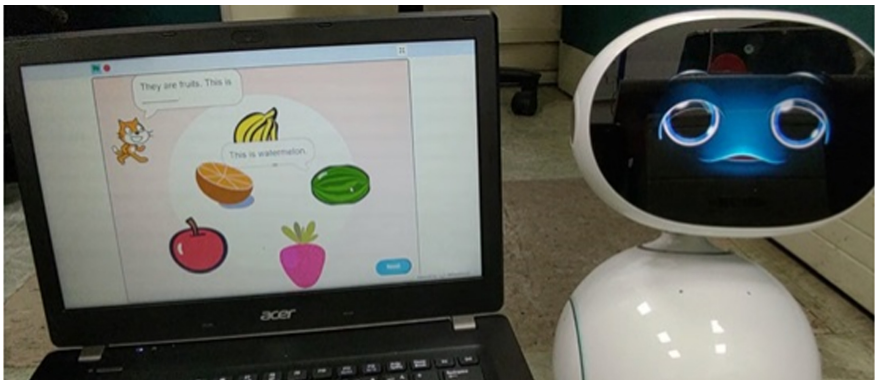
= 13.50) and the average rank of the control group ( $N = 9$ ) (Mean Rank = 5.00) showed a statistically significant difference, where  $U = .000$  and  $p < .001$ ). This means that the project production programs coded by the experimental group students who were trained with CT is very significantly better than the control group students who were trained MS-Office in Phase I.

**Table 5.** Analysis of the CT performance on students' projects production

EG ( $N = 8$ )			CG ( $N = 9$ )			Mann-Whitney U test	
Mean	SD	Mean Rank	Mean	SD	Mean Rank	$U$	$p$ [2*(single tail)]
17.875	0.856	13.50	13.780	3.113	5.00	.000	.000*

Note: \*  $p < .001$

Figure 5 is an example of the execution screen output of students' works: Robot Zenbo teaches English in elementary schools - thematic teaching.



**Fig. 5.** Robot Zenbo teaches English in elementary schools - thematic teaching

## 5 Discussion and Conclusion

This paper aimed to improve university humanity students' abilities of problem-solving and information application through CT training. Brief conclusions that reflect two research questions are made as the following:

1. Compared with the traditional computer introduction course MS-Office training, the humanities students can indeed improve their problem-solving ability after receiving CT training. The difference is significant ( $p < .05$ ).

2. Compared with the traditional computer introduction course MS-Office training, the humanities students can indeed improve their information application ability after receiving CT training. The difference is very significant ( $p < .001$ ).

Although university humanities students are often considered to have lower problem solving ability and information application skills, this research shows their problem-solving ability and information application ability could be significantly improved through the CT training. Therefore, we would like to suggest that CT education should be considered to be added into the “Introduction to Computers” course in humanity departments, such as Department of Foreign Language and Literal, Department of Chinese Studies, Department of History, and other similar departments, in order to promote humanities students’ problem solving ability in modern life.

In addition to the experimental results, we also collected feedback from students participating in the experimental courses. Students’ feedback combined with the experimental results are useful for our future related research, experimental methods, and improvement of teaching materials.

Since the number of students in the experiment is small, it is easy to be suspicious to extend the results obtained to a larger scope. Therefore, it is recommended that the number of samples in the experiment should be expanded as much as possible to obtain more representative data when conducting related research in the future.

Most of the participants found that the CT training and topic production were moderately difficult and interesting, and they had done exercises, and felt a sense of accomplishment. A few students felt that the content of the textbook was a bit complicated and the teaching process was a bit stressful. We will make appropriate adjustments to the teaching materials based on the opinions of students, and hope that in the future, it can be provided to interested teachers of computer courses in the humanities department to meet the needs of most humanities students.

## References

1. Wing, J.M.: Computational thinking. *Commun. ACM* **49**(3), 33–35 (2006)
2. Wing, J.M.: Computational thinking and thinking about computing. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **366**(1881), 3717–3725 (2008)
3. Aho, A.V.: Computation and computational thinking. *Comput. J.* **55**(7), 832–835 (2012)
4. Barr, V., Stephenson, C.: 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 (2011)
5. Sengupta, P., Kinnebrew, J.S., Basu, S., Biswas, G., Clark, D.: Integrating computational thinking with K-12 science education using agent-based computation: a theoretical framework. *Educ. Inf. Technol.* **18**(2), 351–380 (2013)
6. Korkmaz, O., Çakir, R., Ozden, M.Y.: A validity and reliability study of the computational thinking scales (CTS). *Comput. Hum. Behav.* **72**, 558–569 (2017)
7. Selby, C., Woollard, J.: Computational thinking: the developing definitions. In: 45th ACM Technical Symposium Proceedings on Computer Science Education. ACM, Canterbury (2013)
8. Zhang, L., Nouri, J.: A systematic review of learning computational thinking through Scratch in K-9. *Comput. Educ.* **141**, 103607 (2019)

9. Google Computational Thinking for Educators. <https://computationalthinkingcourse.withgoogle.com>. Accessed 10 Sept 2021
10. Bitesize Introduction to computational thinking. <https://www.bbc.co.uk/bitesize/guides/zp92mp3/revision/1>. Accessed 10 Sept 2021
11. Taiwan MOE Computational thinking in Taiwan. <http://compthinking.csie.ntnu.edu.tw/>. Accessed 10 Sept 2021
12. Computational thinking for K12. <https://ctfork12.ice.ntnu.edu.tw/conception.html>. Accessed 10 Sept 2021
13. Román-González, M., Pérez-González, J.C., Jiménez-Fernández, C.: Which cognitive abilities underlie computational thinking? Criterion validity of the computational thinking test. *Comput. Hum. Behav.* **72**, 678–691 (2017)
14. Tukey, J.W.: *Exploratory Data Analysis*. Addison Wesley, MA (1977)