




Computational Thinking: Constructing the Perceptions of Pre-service Teachers from Various Disciplines

Ragonis Noa^{1,2} 

¹ Beit Berl College, Kfar Saba, Israel
noarag@beitberl.ac.il

² Technion Israel Institute of Technology, Haifa, Israel

Abstract. In the last two decades, educators have been following the terminology of Computational Thinking first posed by Wing. Different viewpoints and commentaries have been adopted, and accordingly course syllabi and learning materials were developed, particularly for K-12. The field has become a mandatory part of the curriculum in various countries, even for preschool age. The paper presents an academic course for pre-service teachers with the main aim to facilitate and instruct students in the process of building their understanding and interpretation of Computational Thinking, in the context of teaching their own discipline. The course pedagogical approach emphasizes the adoption of Computational Thinking while identifying significant, non-trivial, computational processes in different disciplines. The course model was implemented with three pre-service teacher populations studying for their teaching certificate in: (1) sciences for high school; (2) humanities and social sciences for high school; and (3) various disciplines for elementary school. The course allows future teachers to experience for themselves learning activities that are recommended for implementation with their future students. The course pedagogical approach and rationale are presented, followed by detailed course structure and learning assignments. The teaching, learning, and assessment approach yielded impressive achievements, although not without obstacles and difficulties. The details of the course presentation enable its implementation with different populations of pre-service and in-service teachers, and can also be implemented in schools.

Keywords: Computational Thinking (CT) · Teachers preparation
Active learning · Simulation of computational process

1 Introduction

The concept of computational thinking has many definitions, the common definition stated by Wing [21] is that “Computational thinking is the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer—human or machine—can effectively carry out”. The development of computational thinking skill of 21st-century students is accepted as central and important. To achieve this, consideration must be given to teacher training, which is always the most

significant link in the process. Although different definitions have been given to the concept there is a significant core acceptable to all. It is important to allow teachers to be exposed to the different approaches and to formulate their understanding and meaning. The paper describes our interpretation of connecting computational thinking with computational processes and algorithmic computational thinking. Emphasis is placed on the importance of imparting Computational Thinking (CT) to learners of different ages and in different school subjects as a thinking skill for life. The presented pedagogical approach enables the development of algorithmic CT that links to computerized systems, since understanding and control of those systems is also a necessary skill for graduates of today's education system. The course development maintained a constant link between the conceptual and the applied levels. The central pedagogical approach is that a school student can identify and define a computational process in a particular discipline, and develop a script (in *Scratch* for example, a well-established creative and community environment) that simulates the process. In so doing, the students use their CT skills in two related areas: first, in the discipline while identifying the process, and second, when developing the script. Students will deepen their understanding of the computational process in the discipline since it should be very precise in order to develop a computerized simulation. At the same time, students acquire skills in developing algorithms, and acquire knowledge, skills, and control of scripts environments.

To clarify the main pedagogical idea, we give an example of one of the teams' projects. The computational process identified and defined by the team was movement problems in mathematics. Most school students face obstacles when confronting a problem such as: "A car drove from town A to town B at a speed of V_1 , and a truck drove from town B to town A at a speed of V_2 . At what point did the vehicles pass each other?" To develop a visual simulation of that process, a student must understand precisely what factors have influence, how the relative movement of the vehicles looks, the impact of different speeds of the vehicles, and of different distances between the towns. It enables the student to move from an abstract technical question to a concrete process that allows better understanding of it. All while investigating and developing the algorithmic CT skill as well as a colorful and creative animation product.

In what follows we present a literature review and full details of the course structure and assignments, and also share some key impressions from running the course three times with three different populations.

2 Background

2.1 Computational Thinking

The concept of CT is recognized as first proposed by J. Wing in 2006 [19] and her successive publications [20, 21]. The definition of CT as: "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" has been adopted by many educators [7]. In the last two decades, various viewpoints and commentaries have been offered, many that connect CT to algorithms and computers

[1, 5, 8, 13, 17] and others that emphasize the need to disconnect it from technology and computers [12, 22]. With or without computers, some key skills and processes commonly mentioned regarding CT are: formulating problems, logically organizing and analyzing data, representing data through abstraction such as models and simulations, suggesting and evaluating several solutions, implementing a possible solution, and generalizing and transferring the solution to variety of problems [10, 12, 21]. Following that, many educators deal with the concept and its acquisition. The greater percentage of the interpretations linking CT to computerized systems is probably because the CSE community dealt with it in the beginning [12], but now there are highly diverse voices [22].

2.2 Implementation in Schools

Over time, the global teaching community has focused on the development of teaching and learning materials. The focus is particularly on developing suitable curricula for different ages, accompanied by developing extensive and varied learning materials [3, 4, 6, 9]. Less attention is paid to the teachers. There are teachers' guides, but a specially-guided training process is needed, particularly when addressing a broad audience of teachers. The computer science education research community falls short regarding assessment of the success of the programs. Most of the papers present descriptions of implementing the various curricula but offer less formal research to evaluate their implementation. A valuable approach for assessing CT dimensions, based on using *Scratch*, was presented by Brennan and Resnick [2], and results of competitions such *CodeMonkey* or *Bebras*, that are based on algorithms and use computerized systems, can serve as measures of success in developing students' CT skills.

2.3 Teachers Preparation

It is clear that the implementation of the different curricula is based on teachers' backgrounds, knowledge, mindsets, and attitudes. However, attention to planned, structured teachers preparation programs has appeared mostly in the last two years [11, 14, 22]. These programs were actually activated in parallel to the academic course for pre-service teachers presented in this paper. The course is an approach to change the state of mind of pre-service teachers in all teaching subjects. We wish them to leave the perception of CT as a mathematical or at least a scientific concept, for a broader concept, which actually has different facets even in the traditional scientific disciplines.

3 The Course

The course for pre-service teachers was developed based on our recognition of the importance to develop school students' CT skills, which should be led by teachers of all teaching subjects. A key aspect is that we are not aiming to develop a new discipline to be added to the school curricula, but rather relate to CT as a skill that students must

acquire throughout their various learning processes, similar to the development of “critical thinking” in different contexts.

In this section, we present the prescribed course development, assumptions, rationale, aims, and learning outcomes. Subsequently we present the course structure and learning assignments.

3.1 Thoughts Towards Development

The implementation relies on three main pedagogical issues. First, we wish to develop teachers’ Pedagogical Content Knowledge (PCK) as presented by Shulman [18], with its expansion to Technological Pedagogical Content Knowledge (TPACK) presented by Mishra and Koehler [15]. Second, we adopt the interpretation that CT can be implemented in any discipline, and hence assume that it is possible to define a computational process in any discipline. And third, we wish the computational process to be accurate and concrete, and being such, to be implemented in *Scratch* simulation.

The decision to concentrate on a computational process that has importance to be developed as a simulation, is significant. In relation to teachers this requires to lower their concerns and to disconnect the immediate and inaccurate association of CT with “computation” (or calculation). For teacher students who intend to teach Humanities, Social Sciences or Languages, there is need to decrease their concern about any aspects of math or technology. Teacher students who intend to teach Math or Sciences also need to change their accustomed conceptions. For example, computing the coordinates of a point on a graph in relation to a given function is an expression of a computational process. But, we wish to emphasize and expand to other approaches of computational processes, which go beyond obtaining a numerical value resulting from a direct calculation, as demonstrated in the introduction to traffic problems.

The development of simulation as a concrete product of a defined computational process is of great value. The development of a computerized script in a digital environment, requires a precise refined definition of the selected process and enables the development of algorithmic CT skills as well. *Scratch* was chosen as the environment because of its known advantages [16].

Regarding the learning process, students should be given space to build their own interpretation of computational thinking and computational process, since there are indeed multiple viewpoints, with no single, precise definition. Our entire teaching-learning approach is in light of constructivism, students must employ active learning, as reflected in both the course structure and the assignments.

3.2 Course Rationale, Aims, and LO

Rationale. The rationale of the course is to develop the students’ CT, along with developing their understanding and confidence regarding its implementation in their teaching subject. Teachers should recognize the importance of the acquisition of CT by school students, to be able to teach it. They have to experience it in a supportive environment that encourages them to think about how to implement this “mindset” in schools. The course has two main interwoven foundations. The first is the definition of

a computational process in the discipline – applying CT skills. The second is the development of a simulation (to some extent depending on the course population) by applying algorithmic CT skills.

Aims. The course’s main aims are:

- (a) expose students to the concept of CT and raise awareness of its various definitions;
- (b) enable learners to build their own interpretation based on experience, literature reading, class discussions, and hands-on positive learning experiences;
- (c) make learning concrete and relevant through assignments that relate to their future teaching and teaching subject;
- (d) develop students’ CT skills in two integrated aspects: (i) develop their ability to identify and define a Computational Process (CP) in their discipline; (ii) develop algorithmic CT skills by developing a project that simulates the process.

Learning Outcomes (LO). At the end of the course, we expect students to be able to:

- (1) identify a relevant interpretation of their own view of the CT concept;
- (2) define a clear and systematic CP in their discipline;
- (3) develop a precise script using *Scratch* that simulates a CP in their discipline;
- (4) analyze given scenarios, define which expresses a CP, and explain their choice;
- (5) appreciate the importance of developing CT skills in students at any age, to equip them with essential skills in our era.

3.3 Course Structure

The course consists of 14 90-min meetings (28 academic hours), comprising five sections. Week 1: Explore the content and the environment; Weeks 2–3: Learning and working with *Scratch*; Weeks 4–5: Introduction to CT as a necessary 21st-century skill; Weeks 6–10: Project development; deepening the conceptual and the application levels; Weeks 11–14 Teaching CT. All content and activities were covered in each course launched; flexibility was essential to meet students’ conceptual needs and progress. Table 1 presents the course schedule.

3.4 Learning Assignments

The course had seven assignments requiring different degrees of depth in thinking, in the time needed to accomplish them, and in type of performance – individual, in pair or team work. The project is an anchor task intended to support students’ understanding and interpretation of the CT concept, and to scaffold their views and visions about how they might implement it in their future field work. Each assignment is presented below with a brief explanation of its intentions and goals.

Assignment #1: Preliminary Questions to Check Background and Pre-conceptions – Individually. Students are asked to complete a Google form questionnaire. Part A includes background: age, teaching disciplines, study track, previous experience with programming, and previous exposure to *Scratch*. Part B includes questions about

Table 1. Course schedule.

Week and assignments	Content
Week 1 Classwork: assignment #1 Homework: assignment #2	<ul style="list-style-type: none"> – Opening and completing a pre-questionnaire – Light exposure to <i>Scratch</i> – <i>Guidelines</i>: no definition of CT is presented; emphasis on the main course target - each participant will develop his/her own interpretation; we are learning together: they are the expert in their discipline and the lecturer will try to integrate his own knowledge with theirs
Weeks 2–3 Class and home work: assignment #3	<ul style="list-style-type: none"> – Demonstrating the <i>Scratch</i> environment and its main principles – Free personal or peer practice based on a suggested basic list of exercises or self-experience – Present solutions to some of the obstacles students faced – <i>Guidelines</i>: the main target is to let students enjoy and experience the environment mostly based on trial-and-error, thereby acquiring basic algorithmic CT without giving it a title. Reduce students' concerns and previous attitudes (against) technology/environments/computers
Week 4 Homework: assignment #4	<ul style="list-style-type: none"> – First discussion on the CT concept based on HW assignment #2 – Listen to students' comprehension and make some key remarks – Watch some of the videos students found for the task – <i>Guidelines</i>: Delineate students' current definitions; do not disqualify positions; lead students to see other aspects; enable different opinions
Week 5 Class and home work: assignment #5	<ul style="list-style-type: none"> – Continue discussion if needed – Present the main course project holistic perception – Students teams start to raise preliminary ideas for CPs in their discipline – <i>Guidelines</i>: Most teams will suggest inappropriate topics. Discuss the principles to evaluate the ideas
Weeks 6–10	<ul style="list-style-type: none"> – Main project development – <i>Guidelines</i>: The project development is challenging as described in assignment #5. Lecturer accompaniment and support is crucial to gain a positive learning process, while students construct their knowledge. Bring up leading questions and ideas, and suggest references
Week 11 Homework: assignment #6	<ul style="list-style-type: none"> – Present different curricula that implement CT ideas relevant to the course population. For example: <i>Code Monkey</i> activities and competition, <i>Beaver</i> activities and competition, <i>CS-unplugged</i> activities, the national schools curricula if they exist – <i>Guidelines</i>: The main aim is to expose students to existing worldwide varied approaches for implementation at schools
Weeks 12–13	<ul style="list-style-type: none"> – Project presentations and discussion – <i>Guidelines</i>: Teams present the CP they identified and characterized in their discipline and presents the process simulation on <i>Scratch</i>. The discussion enable reframing the central concepts and the main messages

(continued)

Table 1. (continued)

Week and assignments	Content
Week 14 Class and home work: assignment #7	<ul style="list-style-type: none"> – Demonstrating “fun” activities expressing CT in well-known problems – Course summary – Guidelines: The colorful activities enable ending the course with a good atmosphere, and at the same time exploring additional applications that facilitate further analysis and conceptualization. Examples: the <i>Hanoi Towers</i> problem or the <i>Konigsberg Bridges</i> problem addressed by Euler

concepts. It was emphasized that there are no incorrect answers; it is just to have the opportunity to stop and think about the concepts before studying them. The questions are: (1) Define briefly the concept “Computation”; (2) Define briefly the concept “Computational Process”; (3) Write the words you think are related to “Computational Process”; (4) Give an example of a Computational Process in your discipline; (5) Add any additional comments.

Intentions and Goals: Part A is for the lecturer, to be aware of students’ backgrounds and options for building teams. Part B, students’ pre-conceptions were considered and addressed in the first class discussion.

Assignment #2: Exploring Different Internet Resources - First Phase of Conceptualization – Individually. Students instructions: Locate a video on YouTube explaining what computational thinking is. Write a paragraph describing the video content and attach the video link.

Intentions and Goals: While choosing one video students usually explore more, and become aware of the extent and variety of existing implementations and interpretations. Students’ outcomes are uploaded to a shared forum so they can learn from their peers.

Assignment #3: Basic Exercises Using Scratch – Individually or in Pairs. The course website offered students extensive web learning resources, among them guided learning with demonstrations and developing exercises. Students can choose, according to their preference and confidence, whether to practice using exercises that develop slowly and gradually, choose exercises from the different guides, or develop their own new idea script. The exercise guidelines relate to what knowledge students have to demonstrate in their scripts. For example: use at least three characters – that move and relate to each other or to the frame; include scheduled conversation or sounds.

Intentions and Goals: It is students’ first targeted practice. Students can progress at a personal pace that suits them. The deliverables will be at different levels but the essence is on understanding the meaning of developing a script that “does something”. This practice is the first step in coping with the concept and skill of algorithmic CT.

Assignment #4: Learning About Computational Thinking and Taking a Stand - Second Phase of Conceptualization – Individually. Students instructions: Read at least one mandatory paper and one elective, use some web sources and videos from assignment #2, and write a 1–2 page position paper referring to: (a) Your understanding

of the CT concept; (b) Your personal attention to any difficulty/challenge the concept poses for you at this stage of learning; (c) Indicate what was interesting/surprising/confusing/raised objections/aroused your curiosity; (d) What do you think about developing CT skills among learners in your teaching subject? Explain your opinion. Any supportive or opposing position is relevant.

Intentions and Goals: After uncovering some different aspects of CT and CP throughout videos, discussions, and building scripts with *Scratch*, it is time to go deeper and read academic papers. Students are asked to conceptualize the knowledge, insights, and skills they have acquired, and reflect on their obstacles and impressions.

Assignment #5: Development of a Simulation Project - Concretizing Concepts – In Teams of 2–3 Students. Students instructions: After experiencing, reading, and learning about CT and experimenting with the *Scratch* environment, the goal of the task is to deepen the thinking about CT in the context of your teaching discipline. Based on the point of view that CT is a prerequisite thinking skill for citizens of the future, how do we develop and implement it in our discipline? The product will be a script in *Scratch*, which is a small simulation of a CP relevant to your discipline. Follow the next steps: (a) Find and define a CP in your discipline - consult with the lecturer about its suitability; (b) Consider a possible visual implementation of the process you introduced, i.e., develop a simulation that is a model for the behavior of the process in the real world; (c) Outline a script; (d) Create the simulation of the process. For the final submission, each student should address a personal reflective page relating to the change he/she experienced in relation to understanding the concepts of CT and CP.

Intentions and Goals: This assignment is at the heart of the course. It reflects the integration of the concepts and skills that the course wishes to impart. Students are active and develop a significant product that enables a deeper understanding of the concepts along with skills development. The project development consistently links abstract thinking around the computational process with refining the arising ideas with script-building skills and coping with the environment.

The main and unique pedagogical approach of the course is our desire for pre-service teachers to see the opportunity their own future school students can benefit from by doing the same assignment, as emphasized in the next text box.

A school student who develops a script in the learned subject through the lens of CT, will deepen his understanding of the CP that takes place in the discipline, through the construction of a computerized simulation of the process. In doing so, he/she refines the understanding of the computational process in the discipline since it is translated into an appropriate and precise visual script; and at the same time develops the skills of algorithmic CT in relation to developing the script in the *Scratch* environment.

Assignment #6: Position Paper: Implementation in Teaching – In Pairs. Students instructions: Write a 2–3-page position paper that present your position on the integration of CT in school instruction. Your personal learning and thinking processes are valuable. There is room to express independent, supportive and/or critical, positions. The position paper should be based on the contents of the course, on the methods of learning in the course, and on at least three bibliography sources.

Intentions and Goals: After accomplishing the project development and being exposed to different school curricula used around the world, students should state their opinions in a reasoned and justified position paper, while looking at the application of CT in their future classes.

Assignment #7: Analysis of Projects, Clarification of Concepts, and Reflection - Third Phase of Conceptualization – Individually. Students are asked to complete a Google form questionnaire. Part A relates to the projects developed in teams. Some represented projects are selected by the lecturer and each student is required to evaluate his or her opinion in relation to the question: “To what extent do you believe the project reflects the principles of the computational process as you understand it?”. The evaluation was ranked as insufficient application, limited application, reasonable application, good application, or excellent application. The goal is to enable students to apply their own conceptions and principles on peer projects. Part B includes three statements about concepts: (1) Today I understand that CT is: ____; (2) Today I understand that CP is: ____; (3) Rank your faith (on a 1–5 scale) that CT is a skill that can be applied in school and in your teaching subject, and explain your position.

Intentions and Goals: Bringing students to a high level of analysis while evaluating given application of CP in light of their own conceptual mental model, and explaining their judgments and conceptions.

4 Course Execution

The course was implemented similarly with three different populations. Changes and adjustments were made between each launch of the course following conclusions drawn from each previous course. The model shown here is the last approved one.

4.1 Populations

The course was conducted three times, each for different populations described in Table 2. All populations were studying for their teaching certificate.

4.2 Course Journeys: Difficulties, Successes, and What Lies in Between

In this section we elaborate on some key aspects that arose while running the courses. The course implementation was accompanied with research and the results will be published in the near future.

A Distant Starting Point. The vast majority of students stated the more obvious answers for “computation” and “computational process” and retained mathematical

Table 2. Description of the course populations.

Institution	Learning track	Teaching Subjects	Number of Students
A: Technion, Faculty of Education in Science and Technology. An obligatory course for the CS track, and elective for others.	Math, Sciences, and Technology for high school Year: 2015–2016	– CS (18) – Mathematics (11) – Chemistry (3) – Physics (5) – Electricity (1)	38
B: Beit Berl College, Faculty of Education. An obligatory course for the group of students.	Humanities and Social Sciences for high school Year: 2016–2017	– Social Sciences and Citizenship (28) – History (3) – English (1) – CS (2)	34
C: Beit Berl College, Faculty of Education. An elective course. Each student combines two disciplines in his/her track of learning.	Various disciplines for elementary school Year: 2017	– Literature – Language – Bible – History – Sciences – Mathematics	28

explanations. An exceptional and expected answer of CS students in course A related to computational theory, and in course B where students were adults with rich work experience. There, the frequent but not major answer related to building a model. However, looking farther to the following stages of the course, it was not easy to move most of the students from their starting position of “mathematical calculation”.

Insecurity Since No Formal Declaration of CT Is Presented. Most of the students constantly felt insecure at the conceptual level. Since no clear definition of CT was presented by the lecturer, students felt confused. Emphasis on the varying interpretations and implementation of the concept and encouraging students to develop their own view, was a new and challenging teaching approach for all populations.

What Should Have Been Mentioned Again and Again? It was necessary in almost each course meeting to remind students that the focus of the course was on the interpretation that a computational process is not built on calculations (even though it could include some). The process is composed of stages, which are developed to solve a problem or build a knowledge map. When developing their project, most students interpret their simulations as a tutoring tool for the school students. It was necessary to constantly emphasize the course’s pedagogical approach that school students can develop the simulation by themselves, and thereby deepen their understanding of the CP occurring in a particular discipline.

When the Idea Becomes Understandable. As expected, difficulties arose in two significant stages of the project development process: one, in the selection and definition of the computational process, and the second, when programming the scripts, since students have no previous background in the environment. The first is in the area of CT, and the second is in the area of algorithmic CT (beyond the mastery of the environment). Coping and passing those two critical stages clarifies the pedagogical idea of the main assignment (#5) and makes it more understandable. In that stage of the course, when the animated simulations appeared on the screens, the tension fell away, leaving room for satisfaction, pride, and smiles. In relation to the main message we wished to deliver – there were no meaningful differences between the populations. Some of the topics of CP that students addressed in their projects, is presented in the Appendix. We present one quote taken from the position paper (assignment #6) of a student from course B, as an example of the success of the process:

“I would like to point out that there has been a profound change in my understanding and approach of ‘There is nothing to do with computational thinking within the discipline of teaching citizenship’, in a completely opposite direction, which indicates that this should be especially involved in the field of citizenship I specialize in. In addition, I understand that I have gone through a personal process with a clear disagreement about the ability to integrate the CT topic, while taking a stubborn stance of not accepting the subject in any way - for understanding the ability to integrate CT in humanities and verbal fields. Thanks for the opportunity given me to undergo an internal change and to be given the tools to implement such an important process, which I believe will contribute greatly to the youth in the education system”

5 Reflective Summary

Following the process of designing and developing the course, the course achieved its aims and learning outcomes. The pre-service teachers expanded their Pedagogical Content Knowledge (PCK) and their Technological Pedagogical Content Knowledge (TPACK). They constructed their knowledge in many facets and were active in the entire learning process. They were able to define and elaborate on the main concepts: computational thinking, computational process, and algorithmic computational thinking. All in the relevant context and interpretation of their own teaching subjects.

The course was conceptually very challenging for the diverse student populations. As a lecturer, it took the skills of a magician and required great patience. The lecturer functions as a psychologist, a facilitator, a leader, a questions asker, and does not provide answers but only hints and reference to materials to follow. Although there was a need to cope with objections, the effort was fruitful. It was very satisfying to follow the students’ thought processes, to see their final projects, and particularly read their reflections about the deep process that they went through and appreciated. Presenting the course content and outcomes in the national leading CS teachers seminar aroused interest and curiosity.

Acknowledgments. We wish to acknowledge all the students who participated in the three courses. They are groundbreakers in being the first to experience that academic course. Students overcame their obstacles since they trusted the route before them, despite the “bumps” along the way.

Appendix

Examples of Students' Projects

Following some of the topics of CP that students addressed in their projects, is presented, to enable partial expression from students' products.

- *Mathematics*: The movement of ducks with and against the river current.
- *Physics*: Throwing an object on a slope.
- *Chemistry*: Simulation of splitting particles moving inside a container and colliding with the walls and themselves.
- *Computer Science*: Simulation of the Dijkstra algorithm in Graph Theory.
- *Citizenship*: Presenting alternatives to building a coalition based on the political platforms of parties.
- *Social Sciences*: Making decisions about the location of a new school according to socio-economic, social, and infrastructure considerations.
- *Language*: Root word recognition.
- *Literature*: Identifying versification in songs.
- *Mathematics for Elementary School*: Identifying common and diagnostic properties of the square family.

References

1. Barr, D., Harrison, J., Conery, L.: Computational Thinking: a digital age skill for everyone. *Learn. Lead. Technol.* **3–4**(2011), 2–23 (2011)
2. Brennan, K., Resnick, M.: New frameworks for studying and assessing the development of computational thinking. Paper presentation at AERA 2012 (2012)
3. Csizmadia, A., et al.: Computational thinking: a guide for teachers. *Comput. Sch. Community* 1–18 (2015). <https://community.computingschool.org.uk/files/6695/original.pdf>
4. Computing at schools web site. Material on Computational Thinking and related topics. <http://community.computingschool.org.uk/resources/252>
5. Computer Science Teachers Association (CSTA): Operational definition of Computational Thinking for K–12 education – Flyer (2011). <http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf?sfvrsn=2>
6. Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE): Computational Thinking in K–12 education leadership toolkit (2011). <https://c.ycdn.com/sites/www.csteachers.org/resource/resmgr/471.11CTLeadershipToolkit-S.pdf>
7. Cuny, J., Snyder, L., Wing, J.M.: Demystifying computational thinking for noncomputer scientists. Unpublished manuscript (2010). <http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.Pdf>. Key: citeulike:13256108
8. Denning, P.J.: The profession of IT beyond computational thinking. *Commun. ACM* **52**(6), 28–30 (2009)
9. Duncan, C., Bell, T., Atlas, J.: What do the teachers think?: introducing computational thinking in the primary school curriculum. In: Proceedings of the Nineteenth Australasian Computing Education Conference (ACE 2017), pp. 65–74 (2017)

10. Google: Computational Thinking for Educators course. https://computationalthinkingcourse.withgoogle.com/course?use_last_location=true
11. Hodhod, R., Khan, S., Kurt-Peker, Y., Ray, L.: Training teachers to integrate Computational Thinking into K-12 teaching. In Proceedings of the 47th ACM Technical Symposium on Computing Science Education (SIGCSE 2016), pp. 156–157 (2016)
12. Hu, C.: Computational thinking: what it might mean and what we might do about it. In: Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education (ITiCSE 2011), pp. 223–227 (2011)
13. Lye, S.Y., Ling Koh, J.H.: Review on teaching and learning of computational thinking through programming: what is next for K-12? *Comput. Hum. Behav.* **41**, 51–61 (2014)
14. Lodi, M.: Growth Mindset in Computational Thinking teaching and teacher training. In: Proceedings of the 2017 ACM Conference on International Computing Education Research (ICER 2017), pp. 281–282 (2017)
15. Mishra, P., Koehler, M.J.: Technological pedagogical content knowledge: a framework for teacher knowledge. *Teach. Coll. Rec.* **108**(6), 1017–1054 (2006)
16. MIT Media Lab: Scratch – a free visual programming language. <https://scratch.mit.edu/about/>
17. Sabitzer, B., Antonitsch, P.K., Pasterk, S.: Informatics concepts for primary education: Preparing children for computational thinking. In: Proceedings of the 9th Workshop in Primary and Secondary Computing Education (WiPSCE 2014), pp. 108–111 (2014)
18. Shulman, L.S.: Those who understand: Knowledge growth in teaching. *Educ. Res.* **15**(2), 4–14 (1986)
19. Wing, J.M.: Computational thinking. *Commun. ACM* **49**(3), 33–35 (2006)
20. Wing, J.M.: Computational Thinking: What and Why? The Magazine of Carnegie Mellon University’s School of Computer Science (2011). <https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why>
21. Wing, J.M.: Computational Thinking Benefits Society, Social Issues in Computing. Academic Press Blog, New York (2014). <http://socialissues.cs.toronto.edu/index.html%3Fp=279.html>
22. Yadav, A., Stephenson, C., Hong, H.: Computational thinking for teacher education. *Commun. ACM* **60**(4), 55–62 (2017)