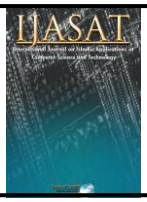

Implementing Computational Thinking Methods & Models in Computer Science Education

Abdulkarim Alsaleh
Ministry of Education – Hail administration
abdulkarimalsaleh@gmail.com

Abstract
In alignment with Saudi Arabia's Vision 2030, this project investigates the implementation of Computational Thinking (CT) methods and models within the computer science curriculum at the middle and high school levels. With a growing need to equip students with innovative problem-solving skills relevant to the digital age, CT emerges as a foundational approach that transcends programming, fostering skills such as decomposition, pattern recognition, abstraction, and algorithmic thinking. This project proposes a structured professional development workshop to introduce CT concepts to computer science teachers, enabling them to embed these methods into their daily instruction. A preliminary survey conducted among Saudi computer science teachers revealed a limited familiarity with CT, reinforcing the need for targeted training. The proposed five-day workshop includes interactive sessions with practical examples across disciplines, curriculum integration strategies, and lesson development guided by CT principles. An evaluation plan is designed to assess teachers' understanding and instructional application of CT before and after the workshop. Ultimately, this initiative aims to transform teaching practices, promote computational literacy across subjects, and prepare students to think like computer scientists, an essential step in building a knowledge-based, innovation-driven society.

Keywords: Computational Thinking - Educational Research - Curriculum Design - Workshop Evaluation - Digital Education.

1. Introduction:

As a part of the Saudi Vision 2030, to build the best future for the country, one goal is to elevate the outcomes of the education system to meet the needs of the current market and provide the workforce and entrepreneurs with sets of innovative, creative skills. The computer science discipline in the education system plays a significant part in doing that as it is important to improve computing literacy throughout education. Computational Thinking (CT) is one of the recent problem-solving methods that has been widely considered for introduction in K-12 schools. Many education systems in other countries have incorporated CT concepts in their teaching methods. This project suggests training middle school and high school computer science teachers in a workshop to be able to implement CT concepts and methods in their computer science classes in Saudi Arabia. These methods and models will help students to think like computer scientists. For the future prospects of the Saudi Vision, it is a crucial moment to begin developing CT skills.

Implementing CT Instruction is Possible; Change is Achievable!

Since the Saudi Ministry of Education has already implemented a Problem-Solving (PS) method in the computer science curricula, the change toward CT is a reasonable next step to elevate students' approach to solve problems. Voskoglou and Buckley (2012) have discussed similarities and differences between PS and CT. They suggested, "CT is a prerequisite to PS when we face real complex technological problems" (p. 33). This project recommends training computer science teachers in middle and high schools to use and implement CT methods and models in their teaching. The project will encourage them to help students to think computationally rather than just solve a technological problem. I will conduct a workshop for teachers by taking a topic or two from different school grades to show how CT can be incorporated.

The CT workshop can help teachers to better understand these methods and models and how facilitate them, with the knowledge on how CT can be implemented across the curriculum. In the future, the workshop could be broadened to support CT implementation across the curriculum, in different subjects such as mathematics, science, engineering, and even humanitarian subjects.

2. Review of Literature:

Computer Science curriculum in Saudi Arabia:

The Ministry of Education in Saudi Arabia built the required computer science curriculum focused on several objectives to accommodate the country's workforce needs. These objectives are mentioned in the curriculum description page of each computer science book for all levels of students, required in Saudi Arabia, as follows:

- Qualifying the student with the skills and practical abilities that facilitate entering the labor market immediately after the secondary stage.
- Building the cognitive aspects of specialized computer science and engineering, using the systems and software prevalent in the world so that the high school students can keep pace with scientific progress and successfully complete the university.
- Acquiring skills to employ computer technology and information in self-learning and project construction for the scientific and humanitarian fields in the secondary stage.

This leads us to a very important question: How could the problem-solving approach of CT help to meet these objectives?

CT Characteristics: The continued demand for deeper understanding of technology to serve other disciplines is increasing over time, and CT can be accommodated to serve these disciplines through these characteristics introduced by Wing (2006):

- "Conceptualizing, not programming" (p. 35). As a computer scientist, you are required to think beyond programming to form a profound concept of thinking. Thus, computational thinkers can solve problems with conceptual abstraction.
- "Fundamental, not rote skill" (p. 35). It is a skill that can be acquired to maintain functionality of solving real-life problems.
- "A way that humans, not computers, think" (p. 35). It is humans who make computers work, not vice versa. CT will make everyone able to develop solutions to solve problems and can programmatically translate these solutions to work in accordance with computer systems.
- "Complements and combines mathematical and engineering thinking" (p. 35). Computer science is fundamentally based on mathematical thinking and engineering thinking, as it involves building systems that interact with real life. Computer

limitations compel computer scientists to not only think mathematically but also computationally. Moreover, the virtual reality systems help engineers to think and build beyond the real world.
- "Ideas, not artifacts. It's not just the software and hardware artifacts we produce that will be physically present everywhere and touch our lives all the time, it will be the computational concepts we use to approach and solve problems, manage our daily lives, and communicate and interact with other people" (p. 35);
- "For everyone, everywhere" (p. 35). CT will be a continual human pursuit, with every person able to develop this kind of thinking.

Computational Thinking (CT) Definition

Researchers have worked to develop a definition of CT and identify relevant skills and concepts. The Royal Society (2012) defined the Computational Thinking (CT) as "the process of recognizing aspects of computation in the world that surrounds us and applying tools and techniques from Computer Science to understand and reason about both natural and artificial systems and processes" (p. 29).

One of the most prominent definitions of CT is the procedural definition presented by the American Association of Computer Science Teachers (CSTA) in cooperation with the International Association for Technology in Education (ISTE). This definition describes computer thinking as a problem-solving process, and includes the following elements:

- Formulating problems in a way that enables us to use a computer and other tools to help solve them.
- Logically organizing and analyzing data.
- Representing data through abstractions such as models and simulations.
- Automating solutions through algorithmic thinking (a series of ordered steps).
- Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources.
- Generalizing and transferring this problem-solving process to a wide variety of problems.

Furthermore, CSTA and ISTE mentioned attributes to assist the CT concepts and they include:

- Confidence in dealing with complexity
- Persistence in working with difficult problems
- Tolerance for ambiguity
- The ability to deal with open-ended problems
- The ability to communicate and work with others to achieve a common goal or solution.

Computational Thinking (CT) Concepts: BBC Bitesize, a school-based support for UK students introduced by BBC network, approached implementing CT when teaching computer science through four keys techniques:

- Decomposition- breaking down a complex problem or system into smaller, more manageable parts
- Pattern recognition– looking for similarities among and within problems
- Abstraction– focusing on the important information only, ignoring irrelevant detail
- Algorithms- developing a step-by-step solution to the problem, or the rules to follow to solve the problem

These cornerstone techniques are the concepts of CT that this research suggests using in computer science. The final step consists of one attribute of CT used mostly at the final stage of introducing a solution, and sometimes alongside algorithm writing, which is debugging to validate the produced solution.

Why Computational Thinking (CT)? Incorporating CT through the computer science subject in school can facilitate students' minds to think computationally when solving problems on other subjects. Yadav, Stephenson and Hong (2017) said that CT can be considered an "analytical thinking skill" that is critical for everyone (p. 56). It can also make it easier for students to better understand the subject-matter and level up their computing and programming skills. It is important for students to think more as computer scientists and not like computers. Computer science learners need to expand beyond learning how computers can help them to solve a problem and learn more about how they solve a problem computationally.

Other disciplines are relying on the involvement of computers to raise efficiency and productivity in their fields, which is what CT is about. CT concepts can be applied to several fields and subjects. CT teaching has been implemented in some schools in the U.S. its use has been encouraged by the Next Generation Science Standards (NGSS Lead States, 2013), which identify CT as key practice for science and engineering studies. The National Research Council (2012) also has highlighted the importance of exposing K-12 students to CT concepts to enhance their skills. The need for computing skills is growing and more jobs in the industry require computational thinkers; therefore, it is crucial for students to learn how to think computationally. Barr and Stephenson (2011) argued, "The computer science education community can play an important role in highlighting algorithmic problem-solving practices and applications of computing across disciplines and help integrate the application of computational methods and tools across diverse areas of learning" (p. 112).

CT across various disciplines introduced new solutions to many new problems, for example, problems associated with enormous amounts of information (Bundy, 2007). It helps in fields such as physics, biology, linguistics, and even law. For instance, "Computational thinking is transforming biology, first with the shotgun sequencing algorithm accelerating our ability to sequence the human genome, and now with our abstractions representing dynamic processes found in nature, from the cell cycle to protein folding" (Wing, 2008, p. 3719). The TangibleK Robotics Program is trying to teach CT to kindergarten level students through a robotics curriculum, and a study shows that children in this program developed CT skills (Bers, Flannery, Kazakoff, & Sullivan, 2014). Children were assessed on several CT key ideas such as debugging, similarities, patterns, and algorithmic thinking. The result of this study showed the success of the TangibleK Robotics Program in developing students' skills in the concepts mentioned. Teachers found it very beneficial for students, and they relatively achieved a high score towered the targeted goal in properly selecting and sequencing instructions. Lower achievement is shown in students' results in other aspects such as control flow but there is still a slight increase (Bers et al., 2014, p. 153).

Yadav, Zhou, Mayfield, Hambrusch, and Korb (2011) conducted a survey to assess future teachers' understanding of CT in a required course. The researchers asked students about their views of computing and CT before and after a CT lecture mid-semester. The results show a great difference between pre- and post-lecture surveys. Most participants improved their answers about their CT understanding from "to use computers and/or technology to solve a problem or make tasks easier" to "the process of solving problems" (p. 3).

3. Research Objectives:

This research project aims to support the integration of Computational Thinking (CT) into Saudi Arabia's computer science education by addressing current gaps in teacher preparation and instructional design. The specific objectives of the study are as follows:

1. To assess the current level of awareness and understanding of Computational Thinking among middle and high school computer science teachers in Saudi Arabia. Preliminary survey results revealed that only 43.4% of participating teachers were familiar with the term "Computational Thinking," indicating a substantial need for professional development.
2. To design and implement a structured professional development workshop aimed at introducing CT methods, models, and instructional strategies to teachers. Supported by: Yadav, Zhou, Mayfield, Hambrusch, & Korb, (2011) emphasized the effectiveness of CT workshops in enhancing educators' understanding and pedagogical skills in computing education.
3. To evaluate the impact of the workshop on teachers' ability to incorporate CT concepts into their lesson planning and classroom practices. Supported by: Barr and Stephenson (2011) highlighted the role of teacher training in effectively translating CT into classroom environments.
4. To demonstrate how CT can be applied across various academic disciplines, thereby promoting interdisciplinary problem-solving and computational literacy. Supported by: Wing (2006, 2008) and the Royal Society (2012) advocate for CT as a universal problem-solving framework applicable beyond computer science.
5. To align CT instruction with the goals of Saudi Vision 2030 by equipping students with innovative, future-ready thinking skills essential for national economic and technological development.

Contextualized by: Saudi Arabia's Vision 2030 prioritizes educational reform to develop a knowledge-based economy with emphasis on creativity, digital skills, and problem-solving (Ministry of Education, KSA, 2018).

4. Dissemination Plan and Presentation:

This paper recommends introducing CT in computer science classes. It is very important, implementing CT concepts and methods in the curriculum that we make sure computer science teachers are familiar with the term CT and its meaning.

A survey was sent to random computer teachers in Saudi Arabia to ask them about the term and what they consider the core purpose of teaching computer science. With 99 computer science teachers completing this questionnaire, only 43.4% of the teachers declared they have heard of the term before (Figure 1). Even more problematic, only 19% of the teachers defined CT as a process of solving a problem (Figure 2).

Thus, there is a great need to teach teachers about CT. We will begin by recruiting teachers who are willing to improve their teaching techniques to support their students to develop important problem-solving skills related to CT. Teachers will attend a five-day workshop comprised of two 55-minute sessions per day, with a break in between them.

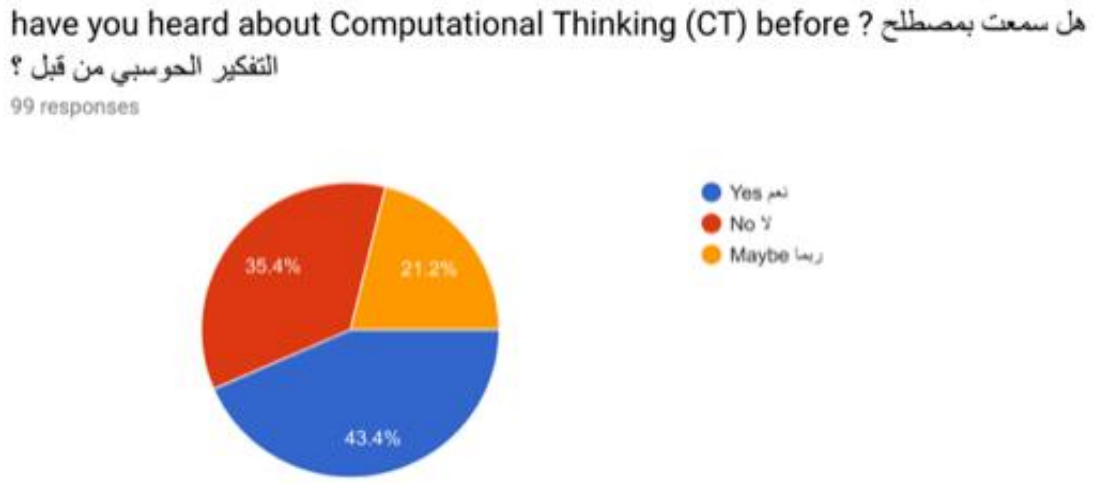

Figure 1: Teaches response of question 1.

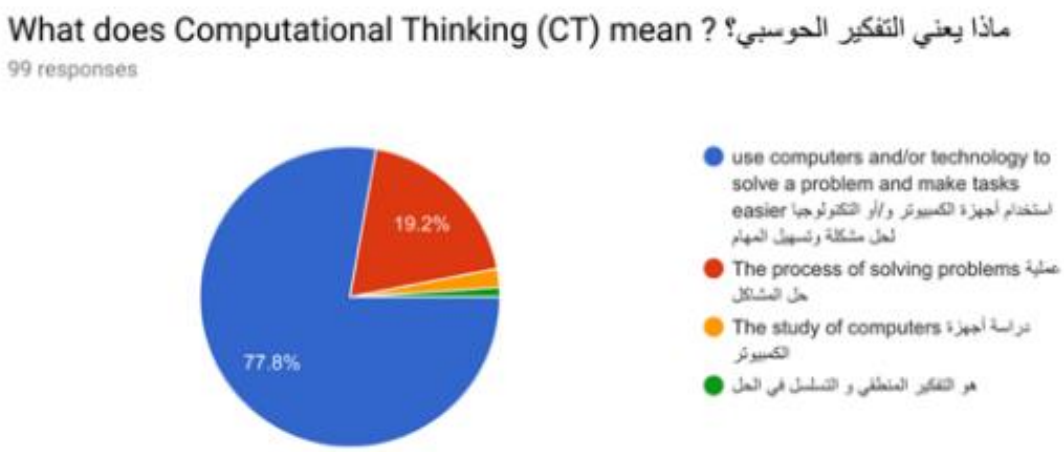

Figure 2: Teaches response of question 2.

4.1 Workshop Plan:

- **Day 1:**

Session 1:

- Teachers complete a pre-workshop survey and then talk about their ideas and questions about computer science and CT (10 minutes)
- Pre-workshop survey results will be shared (5 minutes)
- What is CT? Definition and Vision (20 minutes)
- In this session, we will go through the progression of the term definition. I will give some examples of how CT applies in computer science as well as a couple of other disciplines.
- Why is CT important? (20 minutes)
- Giving some case studies on how to apply CT to math, science, literature, medicine, and other subjects will give trainees a notion of CT's main purpose.

Session 2:

- CT Lesson Examples

A. Recipe example (15 minutes).

In this part we will give some ideas on how computationally work through some problems. One of the easiest problems that can be solved computationally is a food recipe and how to create it. The solution will start with the decomposition to know the ingredients, identify the patterns and looking for similarities to other more familiar dishes, then look at the abstraction of how the final dish will look, and finally go algorithmically through the steps of making it. Teachers will be asked to select a dish to do a recipe for and to do all stages of computational thinking. They have to decompose the ingredients of the dish, abstract and leave out the unnecessary things, look at similar steps have been done in other dishes, and to come up with an algorithm to cook it.

We will discuss connections to CT by looking at the problem and the steps we did through the example given. They need to identify every stage of the steps and connect it to CT concepts.

B. Math example (20 minutes).

Then we will move on to consider the math problem (1+2+3+...+n) as another example of how to use CT to solve this problem. It is a perfect example of using mathematical skills to describe how CT works. We will start at a very simple question like (what is the result of adding numbers between 1 & 200?) and after that we will generalize the solution for any n numbers.

We will discuss connections to CT by taking a different yet more complicated example and see how they can come up with a formula to solve it. They have to identify the stages while they are solving the problem. Starting with simple of 200 numbers to as hard as any range of numbers.

C. Password example (20 minutes)

One other example we will discuss is the password generator where trainees will try to come up with a formula to generate a unique password for every website, they sign up in. We will discuss connections to CT to see how they can do the formula by decomposing it (several passwords but same method of choosing it) , abstracting it (leave unnecessary details out), then do an algorithm to generate it.

- **Day 2:**

Session 1:

- CT Concepts – Decomposition (55 minutes)

  We will take very big problems and break them down into small manageable pieces that we can solve. One example can be presented to them is how companies can program a game and what stages of CT they go through to produce it. We might start with a simple game like PACMAN and see how it can be programmed with the help of CT concepts, then move to a very difficult one. This is how CT works with every problem. It is about reframing a difficult task and divide it into easier problems that we can solve.
- We will go back to these and Day 1 examples to discuss how decomposition applied to them. We will ask questions like: how the math problem can be more difficult? how can we generalize the password problem to make it work for everyone? (10 minutes)

Session 2:

- Computational Thinking Concepts – Abstraction (55 minutes)
  Problems, that are big and hard to solve, look very complicated and overwhelm the solver when they look at them. Abstraction is about losing all unnecessary details and look at the problem differently. It helps when we recognize parts that can be found in - or similar to - other solved problems. We will ask some questions to find patterns of abstractions in different situations and we will look at the given examples to see how they can be abstracted. They have to work in groups to brainstorm problems and situations then swap their methods of abstraction between groups to see how they did this.

- **Day 3:**

Session 1:

- CT Concepts - Pattern recognition & logical thinking (55 minutes)
  Finding patterns is extremely important. Patterns make the problem we are solving easier because we can use the same problem-solving solution wherever the pattern exists. We will use Scratch programming to find a simpler pattern to move an object across the screen. Typically to move the object we program it to move from the corner of the screen horizontally few steps then vertically another few steps, then we will do this again and again until the object reaches the opposite corner. Do you see the pattern here?

Session 2:

- Computational Thinking Concepts - Algorithmic thinking and design (55 minutes)
  One of the cornerstones of CT is developing a set of steps and a walk through the solution of the problem. These steps are called algorithms. In CT we use the parts we decomposed, see how the work recognize the patterns, then set a plan of instructions or algorithms to solve the problem. We will ask the teachers to do an algorithm for one of the problems we presented in day 1. We also will present how these algorithms can be implemented in Scratch programming.

- **Day 4:**

Session 1:

- CT in Computer Science curriculum (55 minutes)
  In this session, we will look at some lessons in the computer science curriculum that could be improved to embed the CT approach. We will go across various school levels of computer science subject and see how these lessons can be altered to develop students' computational skills. Lessons such as Scratch programming in 9th grade and Algorithm design in 10th grade can be a good start for teachers to implement CT concepts.

Session 2:

- Possible changes: (55 minutes)
  After being familiar with what CT means and knowing its concepts, we will recommend more effective tools to teach it using some online services and showing unplugged activities that can develop students' computational skills. We will suggest presenting the term of CT (Computational Thinking) more often to the student so they will be familiar with it and use its concepts as parts of key problem-solving skills.

- **Day 5:**

Session 1:

- Introduce resources to help (55 minutes).
  There are some tools that will help teachers to plan lessons and will let students acquire key problem-solving skills such as Computational Thinking. Websites like Code.org and Codehs.com are very easy to use by students and will help develop algorithmic thinking skills. LEGO education can provide schools with great products that will help develop an understanding of decomposition and abstraction.

Session 2:

- Develop a lesson with CT concepts integrated (55 minutes)
  We will ask computer science teachers to develop a lesson of their choice and try to teach CT concepts through it. We will provide teachers with a lesson plan template with key words indicated such as problem idea – decomposition – abstraction – pattern recognition – algorithm. We will include in the templates the tools and resources needed to solve the problem. Teachers by the last day will hopefully get a clear understanding of CT and how to implement it in their classes and this step is crucial to examine their comprehension of the workshop.

4.2 Evaluation Plan

In preparation for assessment, it is required to study the targeted teachers beforehand. A pre-professional development survey must be given to teachers – similar to the one that has been done for this research - to learn about their understanding of CT. Qualitative questions like: "Have you heard about CT before?" and "What does CT mean?" must be presented to targeted teachers to know what they know about the term. Additional questions will include: "What unplugged activities have you used in your class?" and "What in-class materials do you provide your students with to develop their problem-solving skills?" should also be asked.

These questions will examine if teachers are willing to improve their methods to elevate their students' CT. Quantitatively, teachers will also have to rank their comfort of using the methods we presented and scale how much they benefit from this workshop. Yadav, Zhou, Mayfield, Hambrusch, & Korb, (2011) have done a similar survey on teachers who had workshop about CT and their result showed an increase of teachers understanding of CT. we will ask similar questions on our teachers.

After the workshop, teachers' understanding of CT will be assessed through the implementation of CT in lesson planning. The same survey will be completed by teachers after the workshop to see to what extent they developed an understanding of key concepts related to CT. This research suggests drawing the post-workshop survey's questions from a study on teachers who have recently introduced to CT shows that their understanding of its concepts and methods have grown significantly (Yadav, Zhou, Mayfield, Hambrusch, & Korb, 2011). Furthermore, lesson plans that have been developed during the workshop should be assessed based on a template provided. Inclusion of CT concepts and applying its methods will be measured to demonstrate teachers' ability to implement CT. Application of this workshop to teach students with CT Concepts implemented can be examined by measuring CT skills in students' minds.

5. Conclusion:

This research underscores the pivotal role of Computational Thinking (CT) in advancing computer science education within Saudi Arabia's K–12 system. In light of the national emphasis on innovation and digital transformation, CT emerges as a core competency that empowers students to approach problems methodically, think analytically, and engage in creative solution design.

The proposed professional development workshop for computer science teachers offers a practical pathway to integrate CT concepts into daily instruction, fostering a shift from rote technological use toward deep, transferable problem-solving skills.

Survey data highlighted a considerable gap in CT awareness among educators, affirming the necessity for targeted interventions and continuous professional growth. By bridging this gap through hands-on training and curriculum alignment, this initiative contributes meaningfully to the broader objectives of Saudi Vision 2030. Ultimately, embedding CT across educational stages serves not only to enhance computational literacy, but also to cultivate a generation of learners prepared to think like computer scientists and adapt to the evolving demands of a global, technology-driven economy.

6. Future Prospects:

Building upon the foundation laid by this research, future efforts should aim to expand the scope of CT implementation across interdisciplinary domains. Embedding CT principles into subjects such as mathematics, science, engineering, and the humanities will enrich learners' cognitive flexibility and promote a unified framework for analytical thinking across curricula.

Further empirical studies may investigate the longitudinal impact of CT training on both teaching practices and student learning outcomes. Research could also explore the development of national standards and assessment tools to measure CT proficiency at various educational levels. In addition, strategic partnerships between educational institutions, government entities, and technology providers may support the creation of scalable CT programs, digital platforms, and certification pathways for educators.

As education systems continue to evolve in response to technological advancement, prioritizing CT will be instrumental in preparing students for emerging careers, complex global challenges, and lifelong learning. In doing so, Saudi Arabia positions itself at the forefront of educational innovation, equipped with the intellectual capital required to achieve a sustainable, knowledge-based future.

7. References

[1]. Anton, G., & Barany, A. (2013). Power of play: Exploring computational thinking through game design. Velvet Light Trap: A Critical Journal of Film & Television, (72), 74-75. doi:10.7560/VLT7207

[2]. Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. Learning & Leading with Technology, 38(6), 20-23.

[3]. Barr, V. and 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 (Mar. 2011), 48–54.

[4]. BBC Bitesize. (2018). BBC Bitesize - KS3 Computer Science - Introduction to computational thinking - Revision 1. [online] Available at: https://www.bbc.com/bitesize/guides/zp92mp3/revision/1 [Accessed 11 Dec. 2018].

[5]. 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. (Bers, Flannery, Kazakoff, & Sullivan, 2014)
[6]. International Society for Technology in Education (ISTE) & the Computer Science Teachers Association (CSTA) (2011). Operational Definition of Computational Thinking for K-12 Education.
[7]. Matsumoto, P. S., & Cao, J. (2017). The development of computational thinking in a high school chemistry course. Journal of Chemical Education, (9), 1217.
[8]. Royal Society. (2012). Shut down or restart: The way forward for computing in UK schools. Retrieved from http://royalsociety.org/education/policy/computing-in-schools/report/
[9]. Sneider, C., Stephenson, C., Schafer, B., & Flick, L. (2014). Computational Thinking in High School Science Classrooms: Exploring the Science" Framework" and" NGSS". Science Teacher, 81(5), 53-59.
[10]. Voskoglou, M. G., & Buckley, S. (2012). Problem solving and computational thinking in a learning environment. arXiv preprint arXiv:1212.0750.
[11]. Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.
[12]. Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366(1881), 3717–3725. https://doi.org/10.1098/rsta.2008.0118
[13]. Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. Communications of the ACM, 60(4), 55-62.
[14]. Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T. (2011). Introducing computational thinking in education courses. In Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE ’11 (p. 465). Dallas, TX, USA: ACM Press. https://doi.org/10.1145/1953163.1953297