Chapter 3

Computational Thinking: A Pedagogical approach for Constructive Classroom

Vidyanand Khandagale

INTRODUCTION

Computational thinking for solving problems is the advancement in thinking process and helps to enhance the thinking skills scientifically among an individual. Considering Computational Thinking as a pedagogical tool / technique led to a challenge and aroused interest and curiosity among the students and facilitators as it involves active participation.

The stages in the computational thinking process initiates from contextualization and decomposition, then patterns recognition and follows the abstraction and Algorithm. Every stage of the process in CT is linked meticulously and precisely to each other to arrive at the solution of the issue or a problem. CT pedagogy is student centric and gives an opportunity to the learners to apply various thinking skills to deal with a social issue or a problem with thorough analysis.

This chapter deals with the conceptual understanding of the CT and the guidelines to implement the CT in the classroom for effective learning in the fourth Industrial era.

The term 'Computational Thinking' can be defined as a study of the problem-solving skills and tactics involved in writing or debugging software programs and applications. It was first used in the year 1980 and later in 1996 by Seymour Papert in his article 'An exploration in the space of Mathematics Educations' which was published in the International Journal of Computers for Mathematical Learning. Computational thinking is very closely related to computer science. Jeannette Wing brought this term into limelight in an ACM Communications essay wherein she proposed that computational thinking is a basic skill. It is not limited just to computer scientists but can be learnt by everyone. This was when she suggested that it is very important to integrate computational ideas into other disciplines. Computational Thinking can be applied across a

large number of disciplines such as Maths, Science as well as Social Sciences, Languages and Arts. In the field of education, computational thinking refers to the set of means that are used to solve problems in a way that computers could solve.

The following elements are now widely accepted as comprising CT and form the basis of curricula that aim to support its learning as well as assess its development:

- Abstractions and pattern generalizations (including models and simulations)
- Systematic processing of information
- Symbol systems and representations
- Algorithmic notions of flow of control
- Structured problem decomposition (modularizing)
- Iterative, recursive, and parallel thinking
- Conditional logic
- Efficiency and performance constraints
- Debugging and systematic error detection.

THEORETICAL BACKGROUND BASED ON RESEARCH AND LITERATURE

Recently, Computational Thinking (CT) has been advocated as a twenty-first-century skill that students should possess in order to develop problem-solving skills using principles from computer science (Selby, 2015). Wing (2006) described computational thinking as "solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (p. 33).

Since then, researchers have suggested that computational thinking involves a number of subskills, including breaking down complex problems into familiar ones (problem decomposition), developing algorithmic solutions to the problems (algorithms), and capturing the fundamental simplicity of a problem to develop quick heuristics that might lead to a solution (abstraction) (Barr & Stephenson, 2011; Grover & Pea, 2013; Wing, 2008; Yadav et al., 2014). Furthermore, given that computation is a crucial driver of innovation and productivity in today's technology-rich society (Selby, 2015), it is imperative that students engage in computing ideas at the K-12 level (CSTA & ISTE, 2011). For computational thinking to become part of the K-12 curriculum, there is a critical need to prepare teachers who are well trained to integrate computational thinking in their everyday pedagogical activities (Lye & Koh, 2014).

While computational thinking has been suggested as a problem-solving approach using principles from computer science, many of the existing efforts use

programming tools and environments to expose students to computational thinking. Fletcher and Lu (2009) argued that this approach might continue the misconceptions about computer science as being equivalent to "programming". Instead, they suggested, "just as proficiency in basic language arts helps us to effectively communicate and proficiency in basic math helps us to successfully quantitate, proficiency in computational thinking helps us systematically and efficiently process information and tasks" (Fletcher & Lu, p. 23).

This effort to lay foundations of CT needs to start early on in students' K-12 experience before they learn programming languages (Fletcher & Lu). Hence, we need to develop ways to embed computational thinking concepts and practices across disciplines both with and without the programming context to benefit students with varied interests.

Barr and Stephenson (2011) proposed nine core computational thinking concepts and abilities to integrate CT concepts in K-12 classrooms across core content areas. These core computational thinking ideas include data collection, data analysis, data representation, problem decomposition, abstraction, algorithms and procedures, automation, parallelization, and simulation. These computational thinking concepts can be implemented in K-12 classrooms through digital storytelling, data collection and analysis, and scientific investigations (Lee, Martin & Apone, 2014), creating games (Howland & Good, 2015; Lee et al., 2014; Nickerson, Brand, & Repenning, 2015), educational robotics (Atmatzidou &Demetriadis, 2014), physics (Dwyer, Boe, Hill, Franklin, & Harlow, 2013), visual programming languages like Scratch or other interactive media (Brennan & Resnick, 2012; Calao, Moreno-Leon, Correa, & Robles, 2015), and even through maker movements (Rode et al., 2015).

While computational thinking is relatively a new concept, Mannila et al. (2014) found that a majority of K-9 teachers from various disciplines were already practicing and implementing CT concepts and practices in their own teaching. These implementations ranged from using data collection, analysis, and representation to algorithm design and writing (i.e., programming). Additionally, in a review of 27 empirical studies about programming in K-12 and higher education settings, Lye & Koh (2014) reported that visual programming languages were most often used in K-12 to create digital stories and games. They found that constructionism was a common instructional strategy used by teachers, involving students to create artifacts displaying their understanding of CT concepts.

Moreover, research has also exhibited that exposing students to computational thinking ideas also improves their problem-solving abilities and critical thinking skills (Akcaoglu & Koehler, 2014; Calao et al., 2015; Lishinski, Yadav, Enbody, & Good, 2016).

For example, Akcaoglu & Koehler (2014) used a Scratch-based curriculum to examine the influence of CT on middle school students' problem-solving

skills as measured by a PISA problem-solving test. When compared to the control group, the results suggested that students who participated in Scratch activities significantly increased their problem-solving skills, including system analysis and design, decision-making, and troubleshooting skills.

In another study, Calao et al. (2015) embedded computational thinking in a sixth-grade mathematics classroom. Their results suggested that the intervention significantly improved students' understanding of mathematical processes when compared to a control group that did not learn about computational thinking ideas in their math class.

Taken together, these policy-related and practical initiatives strongly highlight the significance of introducing students to computational thinking in K-12 classrooms. However, preparing teachers to embed these concepts in their teaching or in their specific subject areas can be a daunting task.

Barr and Stephenson (2011) highlighted that a systematic change regarding CT implementation in school could not be accomplished without educational policies that include teacher preparation to help educators understand and implement CT in their teaching. Even though most of the computational thinking initiatives we describe in this chapter underline the necessity to train teachers in all subject areas to embed CT, little has been done to examine the instructional, curricular, and pedagogical implications for teacher preparation, particularly for preservice teachers (Lye & Koh, 2014). While preparing Teachers for Computational Thinking Instruction, there is an increasing need for teachers to be prepared to integrate CT into their classroom practices (Prieto-Rodriguez & Berretta, 2014). Recent efforts to expose teachers to computational thinking have focused on both preservice teachers through modules in existing teacher education courses (Yadav et al., 2014) as well as in-service teachers through professional development (Prieto-Rodriguez & Berretta, 2014). At the in-service level, a majority of the work has involved working with teachers through short professional development opportunities to embed computational thinking. Blum and Cortina (2007) examined how a weekend-long workshop to introduce teachers to computational thinking and the role of computer science in relation to other disciplines influenced their perceptions of computer science (CS). Results from the study suggested that teachers' perceptions of computer science significantly changed from being focused on CS as programming to viewing CS as being applicable to other disciplines. Teachers reported that they not only changed their ideas about computer science, but the workshop also allowed them to present CS in a way that would make it relevant to their students' day-to-day lives. Similarly, in another study Prieto-Rodriguez and Berretta (2014) focused on in-service teachers' thinking about the nature of computer science and whether teachers' perceptions about computer science change after a workshop. Findings suggested that connecting teachers to the skills and resources needed to teach computer science and computational thinking concepts can have a positive im-

pact on their perceptions of computer science. While there has been a considerable focus on professional development for in-service teachers, there is limited work on how to prepare preservice teachers to embed computational thinking in their future classrooms. In one study, Yadav et al. (2014) introduced preservice teachers to computational thinking and how to embed computational thinking in the K-12 classroom through a one-week module in an introductory educational psychology course. The authors used a quasi-experimental design to examine the effectiveness of the module on preservice teacher's definition of computational thinking and their ability to embed CT in their future classrooms. Results from the study suggested that preservice teachers who were exposed to the modules were significantly more likely to accurately define computational thinking and were also more likely to agree that computational thinking could be implemented in the classroom by allowing students to problem-solve (and not just by using computers). The results from this study are promising; however, while a one-week module might be enough to develop preservice teachers' understanding of computational thinking, it might not provide them with enough knowledge to embed computational thinking in meaningful ways. We need to consider how to expose preservice teachers to computational thinking constructs within the context of the subject area they will teach in their future classrooms. Barr & Stephenson (2011) recommended that in order for computational thinking to be part of every student's education, all preservice teacher preparation programs need to include a class on computational thinking across the disciplines. We would argue that teacher preparation programs should go beyond one class and teach computational thinking in the subject matter context of methods courses. The majority of teacher education programs offer an introductory educational technology course, which could serve as a core class to introduce preservice teachers to CT ideas. The teaching methods courses could then be used to expand on preservice teachers' understanding of computational thinking within the context of their subject area and build upon that knowledge to embed CT in their future classes. Given the calls to expand the pool of teachers who "teach" computational thinking (Cuny, 2012; Yadav et al., 2014; Yadav, Hong, & Stephenson, 2016; Gretter & Yadav, 2016), teacher preparation programs are critical and provide an opportune setting to introduce future teachers to CT. However, before being able to guide preservice teachers' implementation of CT in their future classrooms, we need to better understand how these student teachers think about CT. Specifically, we need to examine how teachers view computational thinking and its role in their classrooms given that teachers' conceptions can significantly influence and even stereotype students' views about what computer scientists do. Guzdial (2008) explained how the field of computing education research can start looking at what non-computing students - here, the training of future teachers - understand about computing for formal education to enhance their knowledge of computing.

Basu, et.al., (2012) in their research entitled 'A Science Learning Environment using a Computational Thinking Approach' developed a cross-domain, visual programming and agent-based learning environment named Computational Thinking in Simulation and Modelling. Class sixth students learnt distance-speed-time relations from physics and ecological processes in a fish tank system from biology. Remarkable learning was noted in both the science units. Students created their own computational models of scientific phenomena. Using these models and simulation tools, they performed experiments and compared the simulation behaviour produced by their models and the simulation behaviour produced by expert models.

Grover, S. and Pea, R. (2013) have given the following elements which are now widely accepted as comprising Computational Thinking and form the basis of curricula that aim to support its learning as well as assess its development: Abstractions and pattern, Generalizations (including models and simulations), Systematic processing of information, Symbol systems and representations, Algorithmic notions of flow of control, Structured problem decomposition (modularizing), Iterative, recursive and parallel thinking, Conditional logic, Efficiency and Performance constraints along with Debugging and Systematic error detection.

Weintrop, D. et.al., (2015) in their research entitled 'Computational Thinking in the Science Classroom: Preliminary Findings from a Blended Curriculum' found that in the case of science and computational thinking, there is a remarkable gap in the attitudes and confidence of male and female students. Very few students have access to preparing themselves for using advanced technological developments such as computation. Female and minority students especially lack behind in this case. Therefore, they recommend computational thinking content should be blended with high school science coursework.

Lockwood, J. and Mooney, A. (2017) in their systematic literary review entitled 'Computational Thinking in Education: Where does it fit?' found that many countries all over the world have yet not introduced computational thinking in the mainstream education process. Work related to computational thinking needs to grow and develop to a great extent. There is a huge scope for the teachers who wish to incorporate computational thinking in their classes as they have an abundance of tools, resources, programmes, hands-on exercises and more. However, more detailed lesson plans and curriculum structure would be more beneficial to the teachers who aim at using computational thinking in teaching and for curriculum development.

Wing, J. (2017) in her research entitled 'Computational Thinking's Influence on Research and Education for All' has concluded that apart from extremely useful hardware and software computer science provides for the system of computational thinking skills which can be used extensively in the field of ed-

ucation and research. Computational thinking should be treated as a basic skill like reading, writing or arithmetic.

Donna Kotsopoulos, & et.al (2017) proposed a Computational Thinking Pedagogical Framework (CTPF), developed from constructionism and social-constructivism theories. CTPF includes four pedagogical experiences: (1) unplugged, (2) tinkering, (3) making, and (4) remixing. Unplugged experiences focus on activities implemented without the use of computers. Tinkering experiences primarily involve activities that take things apart and engage in changes and/or modifications to existing objects. Making experiences involve activities where constructing new objects is the primary focus. Remixing refers to those experiences that involve the appropriation of objects or components of objects for use in other objects or for other purposes. Objects can be digital, tangible, or even conceptual. These experiences reflect distinct yet overlapping CT experiences which are all proposed to be necessary for students to fully experience CT. In some cases, particularly for novices and depending on the concepts under exploration, a sequential approach to these experiences may be helpful.

SUMMARY REVIEW

The review of literature helps to conclude and design the essential steps for the CT as a Pedagogic approach for constructive learning.

The review of literature may be summarized by taking into consideration the research and literature with reference to computational thinking. The initiation was traced in the work of Wing (2006) described computational thinking as "solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science". Grover, S. and Pea, R. (2013) concluded the elements of Computational thinking as the Abstractions and pattern generalizations (including models and simulations) Systematic processing of information Symbol systems and representations Algorithmic notions of flow of control Structured problem decomposition (modularizing) Iterative, recursive, and parallel thinking Conditional logic Efficiency and performance constraints Debugging and systematic error detection.

Furthermore, given that computation is a crucial driver of innovation and productivity in today's technology-rich society (Selby, 2015), it is imperative that students engage in computing ideas at the K-12 level (CSTA & ISTE, 2011). For computational thinking to become part of the K-12 curriculum, there is a critical need to prepare teachers who are well trained to integrate computational thinking in their everyday pedagogical activities (Lye & Koh, 2014). Prieto-Rodriguez & Berretta, (2014) stated an increasing need for teachers to be prepared to integrate CT into their classroom practices. (Yadav et al., 2014) made an effort to expose teachers to computational thinking have focused on both preservice teachers through modules in existing teacher education courses as well as in-service teach-

ers through professional development. Akcaoglu & Koehler, 2014; Calao et al., 2015; Lishinski, Yadav, Enbody, & Good, 2016) research has exhibited that exposing students to computational thinking ideas also improves their problem-solving abilities and critical thinking skills.

Weintrop, D. et.al., (2015) in their research entitled 'Computational Thinking in the Science Classroom: Preliminary Findings from a Blended Curriculum' concluded that very few students have access to preparing themselves for using advanced technological developments such as computation. Female and minority students especially lack behind in this case. Therefore, they recommend computational thinking content should be blended with high school science coursework. Wing, J. (2017) Computational thinking should be treated as a basic skill like reading, writing or arithmetic. Donna Kotsopoulos, et al. (2017) proposed a Computational Thinking Pedagogical Framework (CTPF), based on constructionism and social-constructivism theories and it four pedagogical experiences: (1) unplugged, (2) tinkering, (3) making, and (4) remixing.The review of literature reveals that various effective attempts has been made by the researchers and teacher educators to conceptualize the concept of Computational Thinking and its implementation in teaching learning process. The RRL had broadened the comprehension OF Computational Thinking skills and gave insight to design the essential steps for the CT as a Pedagogic approach for constructive learning.

In the present chapter authors have attempted to propose the essential skills of Computational thinking as a pedagogical approach for Constructive learning.

EXPLANATION OF TOOL

The approach of Computational thinking for constructive learning is generic in nature. It may be applied as per the content of the subject. Each component along with the skills are equally important. The facilitator may adopt the skills as per the need of the content and learning outcomes.

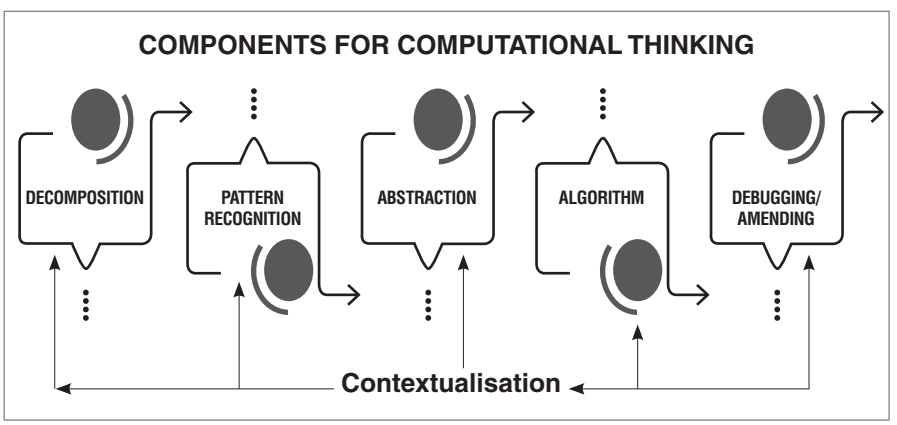

Figure 1 - Computational Thinking for Problem-solving

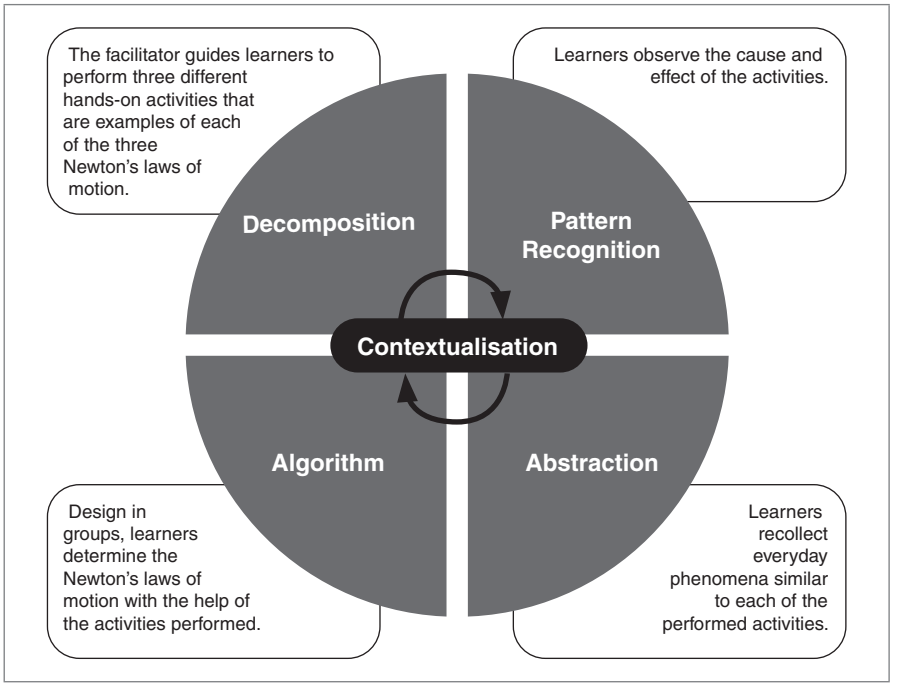

Figure 2 - Generic form of Computational Thinking

A. Aim of the Tool

- To equip the student teachers with the pedagogical approach for the computational thinking skills.

B. Expected Outcomes

The teacher will be able to

- Explain the computational thinking skills and their parameters.
- Create lesson plans/ sessions based on the parameters for computational thinking skills.
- Design evaluation stages/ procedure during the implementation of the computational thinking skills.

The learner will be able to

- analyze the topic / content / issue.
- Synthesis the topic / content / issue.
- Demonstrate analytical, critical thinking, and creative thinking to some extent.

C. Role of Teacher (Facilitator)

The role of teacher as a facilitator is to facilitate learning and hence has to be proactive to form and provide the in advance planning with resource and material need to make for the implementation. The computational thinking approach is analytical, creative, and critical in nature. The teacher has to work as catalysts where he/she finds that students are stuck up due cognitive dissonance and need to trigger the aspiration and motivation to complete the process of learning through a computational thinking approach.

D. Role of Students

The students have to work in collaboration hence need to understand the peers' strength and limitations. The group of students having different capacity and competency get involved and follow the instructions/ suggestions made by the teacher (facilitator).

They have to make a presentation at the end of the learning process. They have explained each stage clearly and their basis with references and substantial evidence (Objective or subjective).

E. Steps to Use the Tool (In few content /topics they are interchangeable).

The educator should follow the below-mentioned steps to ensure proper usage of the pedagogical tool:

1) Identify the topics based on content analysis for computational thinking.

For e.g., the topics can be from the subject Science, Social Science.

PILLARS OF COMPUTATIONAL THINKING:

Contextualization

It refers to the context as the teacher has to give an overarching view and need to educate about the topic / problem and concept as students would be acquainted with what has to be done in the process. Prior planning needs to be done by the teacher based on the topic / problem / concept.

Decomposition takes a complex problem and breaks it into more manageable sub-problems. By solving each potentially simpler sub-problem, we can put the solutions together to arrive at a solution to the original complex problem. You probably already do this in solving everyday problems like writing a paper by breaking it into sections that can be individually written and put together.

For instance, teachers have a topic / problem on online learning issues and resolution. How would you go about decomposing the problem to make it more manageable? First, you would need to get the information / knowledge. This would come from a variety of different sources. Teacher describes the concept of online learning and ask the students do the review and note the observation

with reference to the topic, students understand the problem / topic and initially first stage / pillar i.e. decomposition of the problem and subproblems with reference to student and teachers and identifies the problems and subproblems of the same.

A brief discussion is conducted by the teacher as a facilitator with students to understand their perception and understanding of the problems based on the same teacher asking to note subproblems. Students decompose the online learning issues and solutions into smaller or sub problems. How will you find a solution for effective online learning for the different strata of society like from villages, mountain areas or marginalized sections of society.

Teaching decomposition to young learners means that students are invited into problem-solving scenarios. Teachers share the complex, multi-step problem and facilitate conversations that help students to break it down. While students at their schooling ages are not always developmentally ready for multi-step directions or problems, they are ready to be exposed to models of adult thinking. In doing this, students begin to develop a framework of strategic, computational thinking.

Facilitators may try: Teachers might describe a scenario, such as planning a birthday party, that involves multiple steps. This type of task can quickly become overwhelming without an organized to-do list of smaller, more approachable challenges. Students can help to break down the larger task, and the teacher can help to draw or write a visual representation of their thinking, giving students a mental map of how to solve similar problems in the future.

PATTERN RECOGNITION

When the problem is decomposed, we frequently find patterns among the sub-problems, i.e., similarities or shared characteristics. Discovering these patterns make the complex problem easier to solve since we can use the same solution for each occurrence of the pattern recognition.

So, let's think about the problem related to network issues of village students. For example, students use to climb the tree to avail network while online teaching and learning. The sub problems were the network and data pack of the student's parents.

This entails finding similarities or shared characteristics within or between problems and allows us to use the same solution for each occurrence of the pattern.

Pattern recognition, as a cornerstone of computational thinking, begins with the basic ABAB pattern creation that is taught in the primary grades and extends to more complex layers of thinking. Pattern recognition invites students to analyze similar objects or experiences and identify commonalities. By finding what the objects or experiences have in common, young students can begin to develop an understanding of trends and are therefore able to make predictions.

Facilitators may try: To teach students to recognize patterns, you might begin by investigating trees. What do all trees have in common? They all have a trunk. They all have roots. They all have branches. While there are many differences between types of trees, these components are present in all trees.

ABSTRACTION

Data representation and abstraction involves determining what characteristics of the problem are important and filtering out those that are not. From this, we can create a representation of what we're trying to solve.

The important characteristic of the problem is the network issue of the student during the online teaching learning process. As it was found that most of the students are either from outskirt/village and mountains and network issues/ congestion. However, the only important characteristic for the purpose of online learning is to determine the network issue.

Abstraction is focusing on the information that is relevant and important. It involves separating core information from extraneous details.

Facilitator may try: In primary classrooms, teachers naturally teach kids the concept of abstraction with literature as they identify the main idea and key details. To take this one step further, teachers can encourage students to hunt for information, clues, or treasures by giving them a goal as they approach a book or even an experience.

As students listen to a speaker during a school presentation about dental hygiene, a kindergarten class might be hunting for details about brushing your teeth. By teaching abstraction to the students', they are able to sort through all of the information available to identify the specific information they need. This is an invaluable skill as students read larger texts and are presented with more and more complex information.

Information Literacy is the ability to think critically and make balanced judgements about any information we find and use.

Information Literacy skill is utmost important in today's context and needs to be trained to the students as most of the data and information is made available through the web resources hence the **Accuracy, Authority, Objectivity, Currency, and Coverage** are important to check the validation of the data and information.

ALGORITHM

An algorithm is a set of step-by-step instructions of how to solve a problem. It identifies what is to be done (the instructions), and the order in which they should be done.

For instance:

- Network issue for online learning

- At the first student will find the application with low band width
- Ask the student to download Jitsy app
- Orient the basic functions
- Apply and share

Algorithmic thinking involves developing solutions to a problem. Specifically, it creates sequential rules to follow in order to solve a problem. In the early grades, kids can learn that the order of how something is done can have an effect.

Facilitator may try: To present this idea to students, you might ask them to think about making a sandwich. What should we do first? Second? What if I put the cheese and lettuce on my sandwich before I add the mayonnaise? Conversations about sequence and order develop the foundations of algorithmic thinking.

To get students thinking in algorithms, invite them to design the path from their classroom to the gym by detailing a series of steps. Then, let them try it out! Additionally, invite students to think about their morning routine. What steps do they take to get ready for school each morning? How would the order impact the outcome? Asking students to consider how inputs change the outcome encourages them to be reflective in their thinking and to make changes to their plan to achieve the desired result.

DEBUGGING/ AMENDING

Refers to Detecting Errors by various methods and techniques. Reviewing the phases with reference to contextualization. The debugging / amending is an important component as it helps to review and detect the errors to find solutions through the algorithm. There is no perfect solution to the problem / issue as it may change as per the time and context hence detecting the errors is essential and after the detection relate to the contextualisation and find the right solution / answer.

F. Assessment

The assessment of a Computational thinking as a pedagogical approach depends on a predetermined set of rubrics, which incorporates the core aims of the technique. The rubrics entail the allocation of marks from 1 to 5, depending on the learner's performance.

- Allocated time: Overall 4 sessions in the classroom and home assignment (Subject to the content).
- Classroom Setting or place layout – A classroom can be set in groups of four to five students. The blended approach may be adopted based on the content selected for learning.
- Organization questions (if any).

- Necessary materials – the reference material of the resources to create contextualization and understanding of the problem and concepts.
- Number of participants – Four member for each group.

TABLE

Sample of the Rubrics

Rubrics	Excellent	Good	Satisfactory	Average	Need to motivate
The student engages in the contextualization process and exhibits analytical / critical thinking skills.					
Students display / exhibit analytical thinking skills during the decomposition of the topic / unit.					
Students display / exhibit analytical thinking skills during the pattern recognition of the topic / unit.					
Students display / exhibit analytical thinking skills / Critical thinking skills during the Abstraction of the topic / unit.					
Students display / exhibit analytical thinking skills during the Algorithm of the topic / unit.					
Students display / exhibit analytical thinking / Critical skills during the Debugging / Amending of the topic / unit.					
The learner understands Computational Thinking and was active during the entire process.					

CONCLUSION

The 4th Industrial Revolution is a fusion of advances in artificial intelligence, robotics, the Internet of Things, and more... We often are faced with a disjunctive differentiating technological thinking from human thinking. Nonetheless, this chapter teaches us the traits and applications of computational thinking and how it can be an adequate pedagogical tool to promote creativity, critical, and analytical thinking skills in secondary schools.

Computational thinking creates the adequate environment to solve a problem by thinking scientifically. Once the challenge is presented to the students, the teacher has to guide his pupils through the contextualisation of the problem, to later decompose its different parts in order to recognise a pattern. This organised thinking develops the analytical and critical thinking of the students as they are able not only to recognise common patterns in different scenarios and contexts, but they are also able to follow a set of instructions to reach a well-structured solution. In our current societies characterised by dynamism, interconnection, and promptness, being able to contextualise and recognise common trends and patterns is a required competence to thrive personally and professionally.

REFERENCES

Basu, Satabdi&Kinnebrew, John &Dickes, Amanda & Farris, Amy & Sengupta, Pratim&Winger, Jaymes& Biswas, Gautam. (2012). A Science Learning Environment using a Computational Thinking Approach. Proceedings of the 20th International Conference on Computers in Education, ICCE 2012. Retrieved from https://www.researchgate.net/publication/256065501_A_Science_Learning_ Environment_using_a_Computational_Thinking_Approach on 8th January, 2019

Grover, S. and Pea, R. (2013). Computational Thinking in K–12 : A Review of the State of the Field. Educational Researcher, 42 (1). 38-43. doi: 10.3102/0013189X12463051.

Donna Kotsopoulos& et.al. (2017) A Pedagogical Framework for Computational Thinking retrieved from https://link.springer.com/article/10.1007/s40751-017- 0031-2

Akcaoglu, M., & Koehler, M. J. (2014). Cognitive outcomes from the Game-Design and Learning (GDL) after-school program. Computers & Education, 75, 72–81.

Astrachan, O., Hambrusch, S., Peckham, J., & Settle, A. (2009). The present and future of computational thinking. ACM SIGCSE Bulletin, 41(1), 549–550.

Atmatzidou, S., &Demetriadis, S. (2014). How to support students' computational thinking skills in educational robotics activities. In Proceedings of 4th International Workshop Teaching Robotics, Teaching with Robotics & 5th International Conference Robotics in Education (pp. 43–50).

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

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? ACM Inroads, 2(1), 48–54.

Blum, L., & Cortina, T. J. (2007). CS4HS: An outreach program for high school CS teachers. ACM SIGCSE Bulletin, 39(1), 19–23.

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

Brovelli, D., Bölsterli, K., Rehm, M., & Wilhelm, M. (2014). Using vignette testing to Measure student science teachers' professional competencies. American Journal of Educational Research, 2(7), 555–558.

Bundy, A. (2007). Computational thinking is pervasive. Journal of Scientific and Practical Computing, 1(2), 67–69.

Calao, L. A., Moreno-León, J., Correa, H. E., & Robles, G. (2015). Developing Mathematical thinking with scratch. In Design for Teaching and Learning in a Networked World (pp. 17–27). Cham: Springer.

College Board. (2014). AP Computer Science Principles Draft Curriculum Framework. Retrieved 26 June 2015 https://advancesinap.collegeboard.org/stem/computer- science-Principles Computer Science Teachers Association, & International Society for Technology in Education.

(2011). Computational Thinking: Leadership Toolkit (1st ed..) Retrieved from http://www.csta.acm.org/Curriculum/sub/CurrFiles/471.11CTLeadershiptToolk it-SP-vF.pdf.

Creswell, J. W. (2002). Educational Research: Planning, Conducting, and Evaluating Quantitative. New Jersey, Upper Saddle River: Pearson.

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., &Woollard, J. (2015). Computational Thinking A Guide for Teachers. Cuny, J. (2012). Transforming high school computing: A call to action. ACM Inroads, 3(2), 32–36.

Denning, P. (2009). The profession of IT Beyond computational thinking. Communications of the ACM, 52, 28–30.

Dwyer, H., Boe, B., Hill, C., Franklin, D., & Harlow, D. (2013). Computational Thinking for Physics: Programming Models of Physics Phenomenon in Elementary School.

Fletcher, G. H., & Lu, J. J. (2009). Education human computing skills: rethinking the K- 12 experience. Communications of the ACM, 52(2), 23–25.

Good, J., Yadav, A., &Lishinski, A. (2016). Measuring computational thinking preconceptions: analysis of a survey for pre-service teacher's' conceptions of computational thinking. In Paper presented at Society for Information Technology and Teacher Education, Savannah, GA.

Gretter, S., & Yadav, A. (2016). Computational thinking and media & information literacy: An integrated approach to teaching twenty-first century skills. TechTrends, 60, 510–516. doi:10.1007/s11528-016-0098-4.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. Educational Researcher, 42(1), 38–43. doi: 10.3102/0013189X12463051.

Guzdial, M. (2008). Education paving the way for computational thinking. Communications of the ACM, 51(8), 25–27.

Howland, K., & Good, J. (2015). Learning to communicate computationally with Flip: A bi-modal programming language for game creation. Computers & Education, 80, 224–240.

Ingersoll, R., Merrill, L., & Stuckey, D. (2014). Seven Trends: The Transformation of the Teaching Force. Retrieved from: http://cpre.org/sites/default /files/workingpapers /1506_7trendsap ril2014.pdf

ISTE. (2011). Teacher Resources. Retrieved from https://www.iste.org/explore /articl detail ? articl eid=152

Lee, I., Martin, F., &Apone, K. (2014). Integrating computational thinking across the K-8 curriculum. ACM Inroads, 5(4), 64–71.

Lishinski, A., Yadav, A., Enbody, R., & Good, J. (2016). The influence of Problem-solving abilitieson students' Performance on Different Assessment Tasks in CS1. In Proceedings of the 47th ACM Technical Symposium on Computing Science Education (pp. 329–334). New York: ACM.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? Computers in Human Behavior, 41, 51–61.

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014). Computational thinking in k-9 education. In Proceedings of the Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference (pp. 1–29). New York: ACM.

Nickerson, H., Brand, C., &Repenning, A. (2015). Grounding computational thinking skill acquisition through contextualized instruction. In Proceedings of the Eleventh Annual International Conference on International Computing Education Research (pp. 207–216). New York.

Polly, D., Mims, C., Shepherd, C. E., &Inan, F. (2010). Evidence of impact: transforming teacher education with preparing tomorrow's teachers to teach with technology (PT3) grants. Teaching and Teacher Education, 26(4), 863–870.

Prieto-rodriguez, E., & Berretta, R. (2014). Digital technology teachers' perceptions of computer science: It is not all about programming. In IEEE Frontiers in Education Conference. doi:10.1109/FIE.2014.7044134.

Qualls, J. A., & Sherrell, L. B. (2010). Why computational thinking should be integrated into the curriculum. Journal of Computing Sciences in Colleges, 25(5), 66–71.

Rode, J. A., Weibert, A., Marshall, A., Aal, K., von Rekowski, T., el Mimoni, H., & Booker, J. (2015). From computational thinking to computational making. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (pp. 239–250).New York: ACM.

Selby, C. C. (2015). Relationships: computational thinking, pedagogy of programming, and bloom's taxonomy. In Proceedings of the Workshop in Primary and Secondary Computing Education on ZZZ (pp. 80–87). New York: ACM.

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

Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 366(1881), 3717–3725.

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding a 21st century Problem-solving in K-12 classrooms. TechTrends, 60, 565–568. doi:10.1007/s11528-016-0087-7.

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., &Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. ACM Transactions on Computing Education, 14(1), 1–16.

Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., &Korb, J. T. (2011). Introducing Computational Thinking in Education Courses, Proceedings of ACM Special Interest Group on Computer Science Education (pp. 465–470). Dallas, TX. doi:10.1145/1953163.1953297.

Web Resources

Aman Yadav, Sarah Gretter, Jon Good, and Tamika McLean Computational Thinking in Teacher Education Edited book by Peter J. Rich Charles B. Hodges (2017) Emerging Research, Practice, and Policy on Computational ThinkingSpringer International Publishing AG 2017 retrieved from:

https://link.springer.com/book/10.1007/978-3-319-52691-1

https://www.gettingsmart.com/2018/03/early-learning-strategies-for-developing-computational-thinking-skills/

Acknowledgement

https://www.coursera.org/learn/computational-thinking-problem-solving/lecture/qUro3/1-3-pattern-recognition