IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Computational Thinking in Classroom: An Innovative Pedagogy

Dr.Neepa Bharucha,

Associate Professor, M.B.Patel College of Education, Sardar Patel University, Vallabh Vidyanagar.

Abstract: In the ever-evolving field of education, Computational Thinking (CT) has emerged as a pivotal instructional pedagogy. Computational Thinking (CT) has become a key component of contemporary education, promoting problem-solving abilities that are essential for navigating in a technology-focused world. Computational thinking is a problem-solving approach that draws on techniques and strategies from computer science. It encompasses skills such as decomposition, pattern recognition, abstraction, and algorithmic thinking, allowing individuals to address complex problems by breaking them into smaller, more manageable parts, recognizing patterns, and creating step-by-step solutions. This way of thinking is not limited to programming; it is a core skill applicable across many fields, enhancing critical and analytical thinking.As technology becomes increasingly intertwined with all aspects of life, developing computational thinking skills from an early age prepares students to meet the demands of the modern workforce. By incorporating computational thinking into education, teachers can help students systematically and creatively handle problems, making them better equipped to handle the challenges of the digital world. In this line this theme paper introduces Computational Thinking (CT) as an Innovative Pedagogy along with its Origin, Theoretical framework, Components of Computational thinking and its application in the classroom for student community.

Index Terms - Computational thinking, Innovative pedagogy, decomposition, pattern recognition, abstraction, algorithmic thinking.

INTRODUCTION

1.0 Introduction and Theoretical Background

In today's rapidly evolving digital world, computational thinking has grown increasingly vital. It equips individuals with the tools to navigate and succeed in a society where technology influences nearly every aspect of life. By integrating computational thinking into educational practices, teachers can equip students with the essential skills to analyse challenges, develop innovative solutions, and adapt to an ever-changing environment.

Papert (1996) originally mentioned computational thinking as "procedural thinking." in the process of studying how computers and software are used to solve geometric problems, Papert, who was a member of the mathematics department at MIT, asserted that computational thinking may be used to define the link between a problem and its solution as well as to organize data. In the 1960s, Papert and his associates created the LOGO programming language. This language was designed primarily to support students' logical and mathematical reasoning. Fundamentally, LOGO was a constructivist language, acknowledging that learning is an essentially individual endeavour and elucidating it using Piagetian concepts. Papert(1991:1)
It is hardly unexpected that Papert adopted this perspective given his collaboration with Piaget at the Centre of Genetic Epistemology in Geneva from 1958 to 1963. Therefore, LOGO was created to provide a setting that would encourage and assist Piagetian learning (Logo, 2015).

Here are some definitions of Computational thinking:

Computational thinking is the thought processes used to formulate a problem and express its solutions in terms a computer can apply effectively. Wing (2014)

The mental process for abstraction of problems and the creation of automatable solutions. Yadav et al. (2014)

Computational thinking is the process of recognising 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.(Furber :2012)

Individuals' cognitive performances and processes are frequently the subject of these definitions. As a result, we may draw the conclusion that computational thinking-based activities primarily aim to enhance cognitive abilities and facilitate teaching and learning processes in the impacted persons. Computational thinking is the process of determining a clear, step-by-step solution to a complex problem. It starts with breaking the problem into smaller parts, recognizing patterns, removing unnecessary elements, and then developing a step-by-step solution that can be replicated.

This method is called "computational" because it directly draws from the core principles of computer science. It involves structuring problems in a way that allows a computer or algorithm to help solve them, using concepts like algorithms, data structures, and automation. The emphasis is on using precise, logical steps that can be repeated and executed by a machine, making problem-solving more efficient and scalable. By applying computational thinking, individuals learn to break down complex tasks into manageable sub-tasks, identify patterns and trends, and create generalizable, automated solutions—key components in computing and programming. This problem-solving approach is not limited to computer science but is also applied in areas like language, history, science, math, and art. While "unplugged" computational thinking exists, modern computational thinking often involves using technology, such as computers, to carry out the algorithm.

2.0 Literature Reviewed

In the article "Computational Thinking," Wing (2006) defined computational thinking and argued that it is an essential future skill that will be needed by everyone and that it ought to be included in curricula for students at all educational levels. But the essay itself in Wing (2006) was only four pages long, lacking in-depth study of many of the themes it covered, and not grounded in independent research. Although numerous scholars have utilized the essay as a starting point for their research, it has also received a great deal of criticism. In particular, Hemmendinger (2010) asserted that Wing (2006)'s presentation of the elements of computational thinking is not exclusive to computational thinking. As per Hemmendiger's (2010) assertion:

- Reformulating difficult problems is a common practice across all problem-solving domains;
- philosophers have long considered thinking recursively;
- mathematics undoubtedly uses abstraction, as do all disciplines that construct models;
- problem-solving generally involves separating concerns and applying heuristics.

Hemmendinger (2010) also makes the argument that it is irrational to try training people from other fields how to think like computer scientists. Instead of allowing one discipline to dictate the way that all other disciplines think, physicists and economists should use computational thinking and computational processing technologies to think like scientists and identify new questions that, when solved, will lead to new and efficient methods in their respective fields. Another criticism of Wing is made by Denning (2016). Denning (2016) claims that the paper gives algorithms and algorithmic thinking a too large weight. Denning (2016) argues that an algorithmically-controlled computational thinking model is a viable alternative to valuing algorithms above their contribution.

> "The ultimate goal should not be to teach everyone to think like a computer scientist, but rather to teach them to apply these common elements to solve problems and discover new questions that can be explored within and across all disciplines (Barr and Stephenson, 2011: p.113)."

Understanding computer science and computational thinking are two different things. However, in casual speech, these two terms are interchangeable. This purported equality is false because the latter is primarily intended to instruct students in the application and study of mathematical calculation principles. The original assertion made by Wing (2006) that "computational thinking is thinking like a computer scientist" may be one factor contributing to the widespread acceptance of this viewpoint. Denning (2009) and Hemmendinger (2010) dispute this assertion, primarily because they argue that a definition of computational thinking like this could lead prospective computational thinking learners to believe that computational thinking is limited to the field of computer science and is not very applicable to real-world scenarios.

2. Components of Computational Thinking as a Pedagogy in Classroom

2.1Decomposition

The first step in computational thinking is decomposition. Though it may go by different names depending on the context, the fundamental idea remains the same: to solve a complex problem, you must first break it down into smaller, more manageable components.Decomposition plays a crucial role in computational thinking because it makes complex problems more manageable (as the saying goes, "the best way to eat an elephant is

one bite at a time"). This approach allows problem solvers to better define and understand the issue at hand, while also simplifying the problem through techniques like pattern recognition and abstraction.

Decomposition involves analyzingartifacts by considering the individual components that make them up. Each part can be independently understood, addressed, developed, and assessed. In class, this can be seen in the following ways:

• Splitting artifacts into their fundamental parts to make them more manageable.

• Reducing a problem into simpler versions of itself, which can be solved using similar methods

2.2 Pattern Recognition

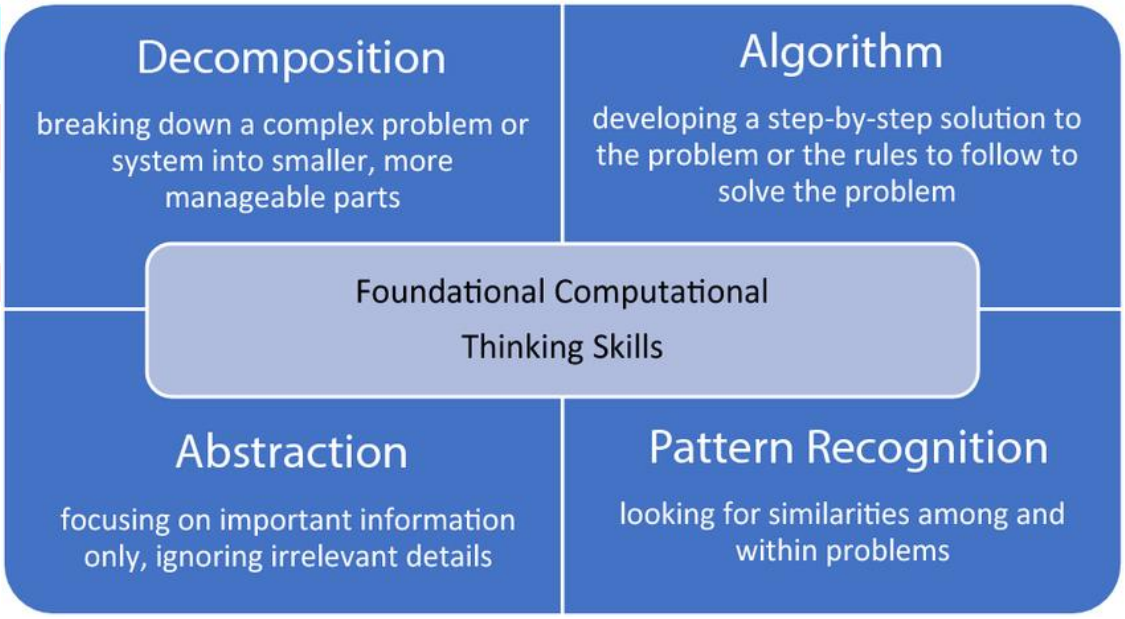
Pattern recognition is another key aspect of computational thinking. This involves identifying patterns or connections between different components of a larger problem. The goal of pattern recognition is to further simplify the problem by pinpointing similarities or differences in the details, which helps build a deeper understanding of the complex issue at hand.

Generalization involves solving new problems by leveraging solutions from previous ones. It focuses on recognizing and utilizing patterns or models. In class, this can be observed through:

• Recognizing patterns and shared features in artifacts.

• Modifying solutions, or components of them, to apply to a broader range of similar problems.

• Transferring ideas and solutions from one problem domain to another.



Computational thinking foundational skills, Adapted from ISTE (2016)

2.3 Abstraction

Abstraction is the process of pulling out the most important information from each part of a decomposed problem. This helps define or generalize what needs to be done to solve the problem as a whole. In this stage of computational thinking, students learn to identify how these key details can be applied to solve other aspects of the same problem.

Abstraction is the process of enhancing the understanding of an artifact by omitting details. In class, this can be observed through:

• Simplifying complexity by removing irrelevant details.

• Selecting a representation of artifacts that allows for practical manipulation.

• Concealing the functional complexities of artifacts.

• Hiding complexity within data, such as through the use of data structures.

• Identifying connections between different abstractions.

• Filtering information during the development of solutions.

2.4 Algorithmic Thinking

The last element of computational thinking is algorithmic thinking. This involves outlining a systematic solution to a problem, which can be consistently replicated to yield a predictable and dependable result. In the contemporary context of computer science, this solution is typically a sequence of steps that a computer will execute. Nevertheless, this procedure can also be partially or entirely performed by humans.

Algorithmic thinking refers to the ability to devise problem-solving methods that yield consistent results across various fields, beyond just science, mathematics, and logic (Mezak&Papak, 2018). An algorithm can be compared to a recipe, where each step is clearly outlined and must be followed in a particular sequence (Peel & Friedrichsen, 2018). It involves identifying the key steps to solve a problem in a way that suits its nature, while considering both typical and exceptional scenarios to ensure the optimal performance of the algorithm (Hromkovic et al., 2017).

3.0 Computational Thinking in Classroom

In the classroom, computational thinking can be included into a number of topics, like:

- Literature / Language Learning: Create word or phrase cards about a story or series of events for younger children or those learning a new language. Ask them to use logic and their general awareness to place the jumbled story in the right order. Older kids can make a Choose Your Own Adventure game by "programming" a story with multiple endings.
- Mathematics: By detecting and recognizing patterns after examining data, use decomposition to answer word problems and express generalizations (as algebraic representations).
- Business studies and economics: Creating financial models, business plans, and exit strategy scenarios; creating decision trees for business decisions and options; or having students
- Social Sciences: Study data and Identify patterns/trends in wars & other historical events, then create visualizations of these patterns and trends.

In order to prepare students for the rapidly changing and ever-evolving world of work, educators must integrate Computational Thinking into the curriculum, given the increasing prevalence of computing in all spheres of life, industries, and organizations.

4.0 Characteristics of Computational Thinking

Computational thinking is defined by the following characteristics:

- Conceptualization Over Programming: Computational thinking transcends simple programming by involving various levels of abstraction, a key aspect of computer science.
- Fundamental Skill, Not Rote Knowledge: Computational thinking is an essential skill for modern life, distinguished from mechanical routines. It includes problem-solving strategies that are unique to humans.

- Human-Centric Problem Solving: Computational thinking focuses on human-driven problem-solving, using creativity and imagination. Rather than training humans to think like machines, it enables us to leverage computing tools to solve complex problems and create innovative systems.
- Integration of Mathematical and Engineering Thinking: Computational thinking merges mathematical and engineering principles, relying on mathematics for formalization and using engineering concepts to design systems that engage with the real world.
- Emphasis on Ideas, Not Just Artifacts: Computational thinking goes beyond physical software or hardware. It includes the conceptual tools used to solve problems, manage everyday tasks, and enhance communication and interaction.
- Universal Accessibility: The goal is for computational thinking to become a universally applied skill, seamlessly integrated into everyday activities. As it becomes a natural part of daily life, it will evolve beyond an explicit methodology and become an intrinsic
  aspect of human existence.

5.0 Techniques associated with computational thinking

Several techniques are used to demonstrate and evaluate computational thinking, often referred to as 'computational doing'. These techniques serve as the means by which computational thinking is applied in settings of classrooms.

4.1Reflecting

Reflection involves the ability to make fair and honest evaluations in complex, non-neutral situations. In computer science, this process of evaluation is guided by the criteria that define the product, as well as heuristics (rules of thumb) and user needs, to inform sound judgments.

4.2 Coding

A key aspect of developing any computer system is transforming the design into code and evaluating it to ensure that it operates correctly across all expected conditions. Debugging plays a critical role in this, involving the methodical use of analysis and evaluation through skills like testing, tracing, and logical reasoning to predict and confirm outcomes.

4.3 Designing

Designing entails determining the structure, appearance, and functionality of artifacts. It includes creating representations of the design, such as flowcharts, storyboards, pseudo-code, and system diagrams. Additionally, it involves processes like decomposition, abstraction, and algorithm design.

4.4 Analysing

Analysing consists of breaking down components (decomposition), simplifying unnecessary complexity (abstraction), identifying processes (algorithms), and recognizing patterns or commonalities (generalisation). It relies on logical thinking to gain a deeper understanding and to assess if something is fit for its intended purpose.

4.5 Applying

Applying involves using existing solutions to address the needs of a different context. It is a form of generalisation, identifying patterns, similarities, and connections to leverage structural or functional aspects of artifacts. An example would be reusing a subprogram or algorithm developed for one context in another.

6.0 Application of Computational Thinking in Classroom

It is imperative that schools implement a computational thinking curriculum. This guarantees that students are not only users of technology but also producers and problem solvers, making them more equipped to handle obstacles down the road. For pupils, computational thinking is crucial for the following seven reasons:

- Enhances Problem-Solving Skills: Students who use computational thinking are better equipped to tackle challenging problems. In addition to encouraging critical thinking and computation, it teaches students how to break complicated issues down into smaller, more manageable portions.
- Boosts Critical Thinking: This method helps students develop their analytical abilities, which is important for 21st-century learning. It motivates students to approach issues from several perspectives and come up with several answers.
- Promotes International Collaboration: Computational thinking emphasizes technology education for schoolchildren by giving them the tools to collaborate across cultural and geographic borders in our globally interconnected environment.
  Encourages Creativity and Innovation: By incorporating digital thinking into the classroom, teachers can inspire students to think creatively and unconventionally, which can result in original solutions and inventive applications in a variety of sectors.
- Develops Resilience and Confidence: Equipping kids with the fundamental 21st-century abilities boosts their self-assurance. It also instils in them a persistent mind-set toward obstacles as they learn to approach difficult issues.
- Improves Digital Literacy: Computational thinking and digital literacy go hand in hand in the digital age. These abilities are critical for comprehending and utilizing technology, which is a necessary skill for kids in the modern world.

7.0 Conclusion

From above discussion it is concluded that computational thinking is a holistic problem-solving strategy that involves breaking down complex issues into smaller, more manageable parts. It focuses on identifying patterns, understanding relationships, and creating efficient solutions to tackle problems effectively. This skill extends beyond computer science, impacting teaching methods and research across various disciplines.

The NEP 2020 is a progressive policy designed to prepare students with the essential skills for thriving in the 21st century. Emphasizing critical thinking, creativity, digital literacy, and collaboration, it establishes the foundation for a well-rounded and multidisciplinary education system.For today's students, computational thinking is essential since it promotes creativity, critical analysis, and problem-solving. As a result, educational boards are emphasizing its importance by incorporating it into curricula. With initiatives like Tech

Tinkerer and Skillful Minds, which are in line with NEP 2020 and NCF 2023 and improve AI and robotics education, STEM pedia promotes this change. Apart from this, teachers are equipped to facilitate dynamic learning. The robotics and AI laboratories work with more than 100 schools to prepare kids for the future. These programs offer a future filled with creativity and superior problem-solving abilities, in addition to providing students with the necessary technical and digital literacy skills. For students, this means a strong basis for employment in technology and other fields, guaranteeing a future of exceptional creativity and problem-solving skills.

References:

- 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.
- Denning, P. J. (2009). The profession of IT Beyond computational thinking. Communications of the ACM, 52(6), 28-30.
- Furber S (2012) Shut down or restart? The way forward for computing in UK schools. Technical report, The Royal Society, London.
- Hemmendinger, D. (2010). A plea for modesty. Acm Inroads, 1(2), 4-7.
- Hromkovic, J., Kohn, T., Komm, D., &Serafini, G. (2017). Algorithmic thinking from the start. Bulletin of EATCS, 1(121).
- Logo Foundation (2015). Logo and Learning, retrieved 24.12.2017 from: http://el.media.mit.edu/logofoundation/what_is_logo/logo_and_learning.html.
- Mezak, J., & Papak, P. P. (2018, May). Learning scenarios and encouraging algorithmic thinking. 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) (pp. 0760-0765). IEEE.
- Papert, S., &Harel, I. (1991). Situating constructionism. Constructionism, 36(2), 1-11.
- Peel, A., & Friedrichsen, P. (2018). Algorithms, abstractions, and iterations: Teaching computational thinking using protein synthesis translation. The American Biology Teacher, 80(1), 21-28.
- Selby, C. &Woollard, J. 2013. Computational thinking: the developing definition. Definition Available: http://eprints.soton.ac.uk/356481/ [Accessed 01-04-2014].
- Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.
- Wing, J. (2014). Computational thinking benefits society. 40th Anniversary Blog of Social Issues in Computing, 2014.
- 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 (TOCE), 14(1), 5.
- https://www.learning.com/blog/defining-computational-thinking/
- chromeextension://efaidnbmnnnibpcajpcglclefindmkaj/https://files.eric.ed.gov/fulltext/EJ1214682.pdf
- https://thestempedia.com/blog/what-is-the-importance-of-computational-thinking-for-school-students/?srsltid=AfmBOoqnInwWAtAYxUZwWyU8RoA_A91UtedkUWQbXI0gvLGqP0mLtioy
- https://www.jaroeducation.com/blog/computational-thinking-a-21st-century-skill/