

Computational Thinking: Need Of The Hour In Engineering Programs In India

Dr.S. Sudhakara Reddy, Principal, Dr. K J Sarma, SMIEEE; Professor in mathematics,
MALLA REDDY ENGINEERING COLLEGE, MAISAMMGUDA, SECUNDERABAD – 500 100, India.

Abstract: For Indians to forge ahead with “Make in India & Made in India” greater technical skill endeavors are required. By developing systematic and focused thinking it is possible to streamline and develop systematic thinking in the student. Computational thinking is one which can develop algorithmic and systematic thinking. An attempt is made to understand, why computational thinking is imperative for every such endeavor.

Keywords: *solving problems, algorithmic thinking, cognitive skills, imagination, collaboration, pattern recognition, algorithmic design.*

1. INTRODUCTION

Thanks to technology and facilities, our lives are more comfortable. Advanced Technology has improvised and upgraded peoples’ **knowledge tools and skills**. Indian education system also is trying to reform and keep abreast with the western system. Facilities are growing and people have been contributing to the advancement of science and technology and making innovations. Indian graduates working in hardware and software industries are unable to produce good results. Only a small percentage of technologies developed by Indians are unique in the world due to various reasons.

Our professionals are unable to make a mark in respect of creating innovative designs and epoch making developments. The performance of the Indian students is comparatively less and lacking deep understanding of the fundamental concepts. India has a history of nearly 5000 years unlike USA which is about 400 years old. Indian professionals are unable to keep pace with Science & Technology and utilization of technology. The Indian education system in general is developing, run of the mill graduates. Many instant education experts and educationists are also responsible for the damage to the education per se.

The education system in India needs serious reforms. Youngsters lack consistent constructive thinking, critical thinking, creativity and good imagination. There is a need for the perfectionism in all directions and dimensionally. We need to explore the specific ways through which we can find remedies.

Every one possesses different viewpoints of addressing every unique problem. There are many remedies, but one singular and effective answer is the introduction of the paper on COMPUTATIONAL THINKING. There is a dire necessity to introduce the course on “Computational Thinking “in the curriculum of every engineering programme, so that we can

generate thinking potentials. In fact it is now the buzz word in the professional and technical applications for conclusions from data.

Computational Thinking (CT) developed by CMU [1,13, 14, 17] which can create thinking recursively. CT is an approach to solving problems, designing systems, understanding human behavior that draws fundamental concepts from Computer Science.

Thinking computationally is fundamental for everyone, for it is important for integrating computational ideas into other disciplines. CT is a fundamental skill that everyone must learn. This is something like a general skill that any human being must learn like reading, writing, speaking and arithmetic. Truly this will contribute to the child's analytical ability [c.f.4, 5, 11].

CT suggests an alternative approach to traditional teaching methodologies for the problems in the areas of science, mathematics, engineering and technology. But this has to be an interactive one.

2. A DETAILED PICTURE OF CT:

The principles of CT is a way of solving problems , designing systems , developing algorithms and understanding human behavior that draws on concepts fundamental to computer science. We explore different levels of abstraction, decomposition, modularization and recursion to understand and solve problems more effectively utilizing methodology of CT. Mathematical concepts. such as induction develop algorithms and producing more

efficient, fair and elegant and secure solutions are part of CT.

The characteristics of CT being : 1) analyzing and logically organizing data, 2) Data modeling, data abstractions, and simulations, 3) Formulating problems such that computers may assist,4) Identifying, testing and implementing possible solutions, 5) Automating solutions via algorithmic thinking, 6) Generalizing and applying this process to other problems.

3. DEPTH AND WIDTH OF CT:

In 21 st century any issue must be based on some quantitative studies, based on data analysis, modeling, or simulation. The applications range spans the areas of applied mathematics, biology, chemistry, design economics, linguistics, mechanics, statistical neuroscience, photography, physics, and analysis. Obviously CT is one of the mental tools that reflect the breadth of the field of computer science.

CT is considered to be good answer to many problems in computer aided design, computer aided language learning, Computer music, Electronic commerce, Entertainment, Technology, Human computer interaction, Language technologies, and in Robotics which uses the tools of Computer Science , electrical, computer engineering and mechanical engineering. CT suggests a deep understanding of the underlying structure and relationships

This is important not only to a computer scientist who would be eager to learn and use CT, rather it a universally applicable attitude

and skill set that everyone builds on the power of computational processes. Modeling is responsible for understanding and designing engineering systems, which need a greater team effort. Constructive imagination is used every time when faced with unique computing problems.

CT needs a combined effort to answer the questions -- What is computable? How difficult is it to solve? What is the best way to solve it? The difficulty of the problem lies in the underlying power of the computing device also. CT facilitates working with systems which generate lot of data with several positive and negative aspects attached to it. We need to understand the problematic situations clearly to solve the problems efficiently. CT confronts the riddle of the machine intelligence. What can humans do better than computers? What can computers do better than humans? This is a fundamental skill for every one. CT involves solving problems, designing systems and understanding human behavior by taking concepts from CS.

4. CT CONFRONTS SITUATIONS:

Sometimes we are confronted with whether approximate solution is good enough, when globally optimal solution is difficult, whether randomization gives any advantage, to a particular problem. CT helps in reformulating the difficult problems into one we know how to solve, by reduction, embedding, transformation or simulation.

CT is thinking recursively, it is a parallel processing. It interprets code as data and data

as code. It is a generalization of dimensional analysis. It checks not only correctness and efficiency but the aesthetics, systems design for simplicity and elegance. It uses abstraction and decomposition when attacking a large complex task or designing a large complex system.

CT is for choosing properly fitting representation and modeling. It helps in describing a systems behavior succinctly and declaratively, it does not need complete thoroughness. It is modularizing something in anticipation of multiple users. CT is thinking in terms of prevention, protection, and recovery from worst case scenarios through redundancy, damage containment and error correction. CT also uses heuristics to discover the solution. It is planning, learning, scheduling in the presence of uncertainty. CT uses massive data and speeds up the computation.

5. CMU IN ITS PPT ON CT SUGGESTS:

The article goes on to expand on this definition and offer examples. Wing [13, 14] says, “Computational Thinking overlaps with logical thinking and systems thinking. It includes algorithmic thinking and parallel thinking, which in turn engage other kinds of thought processes, such as compositional reasoning, pattern matching, procedure thinking, and recursive thinking.” Jeannette Wing [14] pointed to a section of the paper on “Benefits of Computational Thinking” as being key:

Computational thinking enables you to bend computation to your needs. It is becoming the new literacy of the 21st century. Why should everyone learn a little computational thinking?

Cuny Snyder and Wing [] advocate the benefits. Computational thinking for everyone means being able to: Understand which aspects of a problem are amenable to computation, Evaluate the match between computational tools and techniques and a problem, Understand the limitations and power of computational tools and techniques, Apply or adapt a computational tool or technique to a new use,

Recognize an opportunity to use computation in a new way, and Apply computational strategies such divide and conquer in any domain.

6. WHAT CT MEANS FOR PROFESSIONALS:

Computational thinking for scientists, engineers, and other professional further means being able to:

Apply new computational methods to their problems, Reformulate problems to be amenable to computational strategies, Discover new science through analysis of large data,

Ask new questions that were not thought of or dared to ask because of scale, but which are easily addressed computationally, and Explain problems and solutions in computational terms.

This definition is still pretty high-level, but is still *much* better than having no definition. It's a broad definition that encompasses a lot of powerful cognitive skills. We can move away from trying to draw lines between what is and what isn't computational thinking, and instead focus on implications. What parts of this are appropriate to

see at the middle school level? How do we teach these abilities? How would we measure them?

Jeannette Wing [13] says, “Computational thinking overlaps with logical thinking and systems thinking. It includes algorithmic thinking and parallel thinking, which in turn engage other kinds of thought processes, such as compositional reasoning, pattern matching, procedure thinking, and recursive thinking”. Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can effectively be carried out by an information-processing agent. Computational thinking enables you to bend computation to your needs. It is becoming the new literacy of the 21st century.

7. THE EDUCATIONAL PROCESS:

The primary goal of Computational Thinking at IMSA [2] is to provide an introduction to the fundamental concepts found throughout the field of computer science. As an overview of the discipline, the course covers a breadth of topics like algorithmic foundations of computer science, hardware issues such as number systems and computer architectures, and software issues such as operating systems, programming languages, compilers, networks, and human-computer interaction. More than just teach students how to program, this course will teach them how to think more methodically and how to solve problems more effectively. This course will aim to provide students with an understanding of the role computation can play in solving problems.

There are some schools where in CT was already introduced in K-12 itself. The methodology consists of problem solving strategies, teaching algorithms, motivating CT, applying CT to any content area and the big picture. While discussing about describing the CT in education courses Aman Yadav et al [3] outlines the CT module suggests that CT is a must for in 21 st century problem solving. The concepts being abstraction, logical thinking, algorithms, debugging. They further gave the survey questions to be made after implementing the CT in the curriculum.

8. SYLLABUS OF CT [9] :

A systematic formulation of the content of syllabus and the outcomes follows.

8.1 Objectives: To introduce students to fundamental concepts from Computer Science. To give an awareness of the importance of computation and computational thinking in the modern world and the impact it has on areas not immediately associated with Computer Science. The Learning Objectives for “computational thinking” are

1) *Decomposition* , which is the ability to break down a task into minute details so that we can clearly explain a process to another person or to a computer, or even to just write notes for ourselves. Decomposing a problem frequently leads to pattern recognition and generalization, and thus the ability to design an algorithm.

2) *Pattern Recognition*, which is the ability to notice similarities or common differences that will help us make predictions or lead us to shortcuts. Pattern recognition is frequently the basis for solving problems and designing algorithms.

3) *Pattern Generalization and Abstraction* is the ability to filter out information that is not necessary to solve a certain type of problem and generalize the information that is necessary. Pattern generalization and abstraction allows us to represent an idea or a process in general terms (e.g., variables) so that we can use it to solve other problems that are similar in nature.

4) *Algorithm Design*, consisting of developing a step by step strategy for solving a problem. Algorithm design is often based on the decomposition of a problem and the identification of patterns that help to solve the problem. In computer science as well as in mathematics, algorithms are often written abstractly, utilizing variables in place of specific numbers. And

5) *Data analysis and visualization*, consisting of Analogical Reasoning, Hypothesis Testing (Causal Inference), Spaced Repetition which is the most robust findings in memory literature, the generation effect and Multiple exemplars (comparison and contrast) in the category and word learning

8.2 Content: The nature of computational problems: Solution methods for computational problems and notion of an algorithm, Notions of computer science and its formalism, measurement, efficiency of solutions, and intrinsic complexity barriers Examples on computational thinking in variety settings.

8.3 The educational process consists of Lectures enable the students to learn new material relevant to computational thinking. Practical classes enable the students to put into practice learning from lectures and strengthen their understanding through application (by implementing and applying algorithms in a general-purpose, high-level programming language). Students are assessed by

formative and summative assessment and examinations.

8.4 Learning Outcomes: The knowledge gained will be useful. On completion of the module, students will be able to demonstrate an understanding of the fundamental notions relating to problems and their solution in Computer Science, an appreciation of the role of Computer Science and computational thinking in the modern world, an understanding of several approaches to solving computational problems. On completion of the module, students will be able to demonstrate: an ability to recognize and analyze computational problems in a variety of settings, an ability to apply methods and techniques relating to algorithms and computation in order to solve problems, an ability to reason about the quality of a solution or an algorithm. an ability to construct basic algorithms in a general-purpose high-level programming language. The attitudes, skills, problem solving and design skills students learn is given in the figure [c.f.. 17].

An introductory course in the Computational Thinking (CT I), should have Function models, Iterative methods (solving 1-variable algebraic equations), Linear and non-linear least-squares regression (single variable), Descriptive statistics (measures of center and spread, visualization), Probability (discrete and continuous distributions), Probabilistic simulation (coins, dice, cards, continuous distributions that are uniform and normal), Hypothesis testing (Z-test for ratios, t-tests for means) and Finite difference models. These fundamentals must be given to students in addition to the introductory course in mathematics

In order to be effective each group must be limited to 20 students. There must be at least 110 mts of exposure per week. One of the most successful parts of the course involved is the emphasis on modeling and simulation. In the

traditional introductory statistics course, students are taught the central limit theorem also.

Again in Computational Thinking II, the course must be piloted with principles of computer science and computing, while the focus on developing more advanced data analysis techniques (like ANOVA, data science etc). There are more statistical methods which have to be covered in the course with the topics like Nonparametric statistical tests, Analysis of variance (single/multi-factor, single/multivariate), Linear regression and correlation, Multivariate regression and correlation, Basic data mining skills, Basic idea of how computers work. It is advised to select a smaller group of students

In offering the course on Computational Thinking II the idea is to keep in mind the limited time in students' schedules and their ability to fulfill the requirements. The material used can be from statistics. There must be a computer lab at least once in a week. Students must be allowed to present reviews on the current developments. These reviews must have a focus on the hypotheses being tested, the statistical analysis techniques used, and the resulting decisions based on that analysis. In the beginning students may get confused, but as time gets advanced they will find it more exhilarating.

9. WOMEN CAN EXCEL IN CT:

The first computer programmers were women and even now they make up about 25% of the computing industry [c.f.8]. The demand for talented computer professionals is growing in both academics and industry. Women are most likely to succeed in computer science. So if they are introduced to the concepts of computing at the very early stages, they can demonstrate and build on the ideas. This is a feeling as they have rich corpus-collosum compared to men. Thus they can create

wonders and offer solutions to problems related multilevel, multi-objective and multi-criteria systems of life.

10. CONCLUSIONS:

Computational thinking is a way of solving problems, designing systems, and understanding human behavior that draws on concepts fundamental to computer science. To flourish in today's world, computational thinking has to be a fundamental part of the way people think and understand the world.

Computational thinking means creating and making use of different levels of abstraction, to understand and solve problems more effectively.

Computational thinking means thinking algorithmically and with the ability to apply mathematical concepts such as induction to develop more efficient, fair, and secure solutions.

Computational thinking means understanding the consequences of scale, not only for reasons of efficiency but also for economic and social reasons.

Computational thinking for everyone means being able to: Understand which aspects of a problem are amenable to computation, Evaluate the match between computational tools and techniques and a problem, Understand the limitations and power of computational tools and techniques, Apply or adapt a computational tool or technique to a new use, Recognize an opportunity to use computation in a new way, and Apply computational strategies such as divide and conquer in any domain.

Computational thinking for scientists, engineers, and other professionals' further means being able to:--

Apply new computational methods to their problems, Reformulate problems to be amenable to computational strategies, Discover new science through analysis of large data, Ask new questions that were not thought of or dared to ask because of scale, but which are easily addressed computationally, and Explain problems and solutions in computational terms.

It encompasses a lot of powerful cognitive skills. Experts suggest that it is appropriate to introduce CT at the middle school level. Then the questions are, how do we teach these abilities? How would we measure them? If CT is combined with data science consisting data analysis, data representation, problem decomposition, abstraction, algorithms and procedures, automation, parallelization and simulation. One can create wonders and can gain a very closer picture of the working system and can make a better design. The Details of the structure of computational training curriculum is enclosed, at the end of the paper [cf. CMU, 9].

11. REFERENCES:

1. Jeannette Wing, Computational Thinking, in Communications of the ACM, March 2006, Vol.49, no.3.
2. IMSA, Comprehensive Computer Science Syllabus; https://www.imsa.edu/sites/default/files/upload/apcs_ccs_s14.pdf
3. Aman Yadav et. al., Introducing Computational thinking in Education Courses - 10 fundamental problems cs4edu.cs.purdue.edu/_media/sigcse11-final.pdf.

4. Valerie and Chris Stephenson; Bringing computational thinking to K-12: what is the role of the computer science education Community in ACM in roads, March 2011, vol.2 (1).
5. Vannevar Bush ; As we may think in Atlantic, The Atlantic Monthly magazine:
6. Computational Thinking and writing research toolbox, <http://www.ilt.mdh.se/kurser/computing>.
7. Paul Curzon,; Computational Thinking searching to speak, In this cs4fn special:
8. Christie Lee Lilli Prottzman; Computational Thinking and women in computer science, , thesis Presented to the Department of Computer and Information Science, and the Graduate School of the University of Oregon.
9. Computational Thinking, Learning Modules. http://www.dur.ac.uk/faculty.handbook/module_description/?module_code=COMP1051
10. David Barr, John Harrison, and Leslie Conery; Computational Thinking: A Digital Age Skill for Everyone Learning & Leading with Technology March/April 2011.
11. Dave Moursund; "Computational Thinking and Math Maturity" ; Jun, 2010 , ghr Maths forum, Drexel
12. Christopher Kuster, John Symms, Christopher May, Chenglie Hu ; Developing Computational Thinking Skills across the ,Undergraduate Curriculum, Departments of Math, Computer Science, and Psychology, Carroll University; <http://www.csci.psu.edu/background.html>.
13. J.M. Wing; "Computational Thinking will be a fundamental skill used by everyone in the world in the middle of the 21st century" ; www.cs.cmu.edu/afs/cs/usr/wing/www/talks/ct-and-tc-long.pdf.
14. Computational Thinking, Communications of the ACM, Viewpoint, Mar 2006, pp. 33-35.
15. David Weintrop, Elham Beheshti, Michael Horn, Kai Orton, Kemi Jona, Laura Trouille, Uri Wilensky; Defining Computational Thinking for Science, Technology, Engineering, and Mathematics ccl.northwestern.edu/papers/2014/CT-STEM_AERA_2014.pdf.
16. Michael Gr. Voskoglou s, Sheryl Buckley; 'Problem Solving and Computers in a Learning Environment' Egyptian Computer Science Journal ,ECS ,Vol.36 No.4, September 2012,ISSN-1110-2586 -28.
17. Briggs, J. Benefits of Programming (2013), Computational thinking : Attitudes and Skills,
18. Computational Thinking - Naked in the Sunlight - Privacy in the Digital World.
19. Prof. Paul Curzon , Developing computational thinking in the classroom: a framework Working group Queen Mary University of London, June 2014.
20. Steve Easterbrook, From Computational Thinking to Systems Thinking: A conceptual toolkit for sustainability computing; Proceedings of the 2 nd international conference on 'Information and Communication Technologies for Sustainability, Sweden, August , 2014.

21. Dr. Alfredo J. Perez, Dr. Ivan Lopez Hurtado, Dr. Jorge Crichigno, Mr. Raul R. Peralta Dr. David Torres ; Enhancing Computational Thinking Skills for New Mexico Schools 121 st , 2014 ASEE Annual Conference, INDIANA POLIS, June, 2014.

22. Kyu Han Koh, Ashok Basawapatna, Vicki Bennett, Alexander Repenning; Towards the Automatic Recognition of Computational Thinking for Adaptive Visual Language Learning.

23. Amir Rubinstein and Benny Chor, Computational Thinking in Life Science Education PLoS Comput Biol. 2014 Nov; 10(11).

24. Computational Thinking vs. Systems Thinking, Serendipity Applying systems thinking to computing, climate and sustainability; 08. March 2009;

25. C. Mohtadi, M. Kim, J. Schlosser, Why integrate computational thinking into a 21st century engineering curriculum?; 41st SEFI Conference, 16-20 September 2013, Leuven, Belgium MathWorks Ltd, Cambridge, UK..

26. Computational Thinking and Computer Science in Schools, ‘What the Research Says’ Briefing 2 27th April, 2012.

27. Xihua Long,, Jiehui Zhang, Zhanli Li , The Design of Teaching Structure based on Competency Training of Computational Thinking; International Conference on Education Technology and Information System (ICETIS 2013).

<p>It is a way of thinking It is making the impossible possible It is creating solutions to problems in everyday life</p>	<p>ATTITUDES :</p>	<p>SKILLS</p>	<p>It is not thinking like a computer It is not always using a computer as the solution It is not limiting creativity</p>
<p>1. MAKING MISTAKES: I can enjoy things that go wrong and learn from them. I see mistakes as a normal part of solving problems.</p>	<p>THE COMPUTATIONAL THINKER :</p>	<p>SKILLS AND ATTITUDES</p>	<p>2. PATTERN RECOGNITION: Is this similar to a problem I’ve already solved? How is it different? How are the parts of the problem connected?</p>

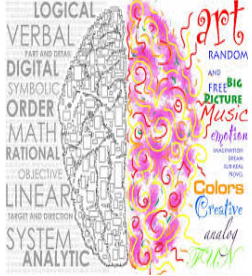

<p>3. PERSEVERANCE: I don't give up. I'm prepared to keep having a go to see what happens. I keep going, even when things seem confusing. I'm determined to find soluti about something else.</p>			<p>4. DECOMPOSITION: Can I break this problem up into smaller parts? Can I explain the different parts of this problem and solution?</p>
<p>5. IMAGINATION I can look at things in unusual ways. I'm ready to consider the impossible. Sometimes I leave a problem for a while. A solution might come to me when I'm thinking about something else.</p>	<p>Problem Solving Designing systems &</p>	<p>Under- standing behavior</p>	<p>6. ALGORITHM DESIGN: What do I need to think about to make this happen? What are the steps I will need to do to solve this problem?</p>
<p>7. COLLABORATION: I can use other people's ideas. I can share my ideas. We can talk together to solve a problem. I can teach my peers and they can teach me.</p>	<p>COURTESY: Briggs, J. Benefits of Programming (2013) https://slp.someerset.gov.uk/cypd/elim/somerset/Computing_Priary/Planning/MA_JBriggs_Oct2013.pdf,</p>	<p>Google, Exploring Computational Thinking http://www.google.com/edu/computational-thinking/index.html, Wing, J. Computation Thinking (2006) http://www.cs.cmu.edu/afs/cs/usr/wing/www/publications/Wing06.p</p>	<p>8. ABSTRACTION AND GENERALISATION: Which is the information I actually need? What don't I need to know? Have I made this more complicated than I need to? Will this work for other things?</p>

Figure 1