

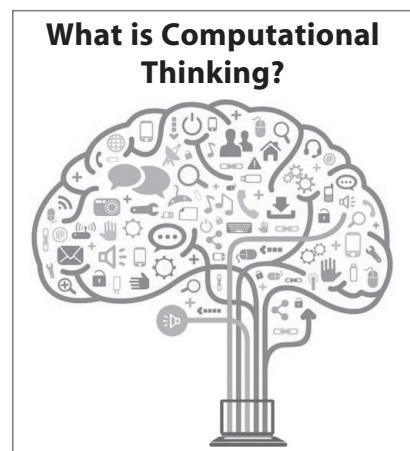
Computational Thinking

(New Chapter)

INTRODUCTION

Computers can be used to help us solve problems. However, before a problem can be tackled, the problem itself and the ways in which it could be solved need to be understood.

When we write a program, we are writing a set of instructions to solve a problem involving data processing. These instructions have to be executed by the computer to solve the problem. If the instructions are correct and given in correct sequence, then the problem will be solved properly by the computer, and if the instructions are incorrect or in wrong sequence, then the problem will not be solved correctly or will not be solved at all.



Therefore, writing correct instructions in correct sequence is extremely important. It means that before we start writing a program, we should be thoroughly clear in our mind about the steps for solving the problem. If we start writing the program without knowing how to solve the problem, then we can never write the correct program. This leads us to the concept of Computational Thinking. Wikipedia defines Computational Thinking as the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer—human or machine—can effectively carry out. So, to write any program, first we have to think computationally—about the problem and its solution—and then we have to write the program in a programming language.

But every day, in all aspects of our lives, we perform computational thinking. Think of the following actions and decisions that go into them:

- You want to make yourself a cup of tea or coffee.
- You want to buy a car.
- You want to change career.
- You want to move to another city.
- You want to buy a house.
- You want to write a book.
- You want to create an app.

All the above situations require thought processing for finding the best solution and, thus, computational thinking becomes a crucial aspect.

WHAT IS COMPUTATIONAL THINKING?

Computational Thinking allows us to take a complex problem, understand what the problem is, and develop possible solutions. We can then present these solutions in a way that a computer, a human, or both, can understand.

The process of planning and reaching the goals involves computational thinking of some kind. Computational thinking involves taking that complex problem and breaking it down into a series of small, more manageable problems (**decomposition**). Each of these smaller problems can then be looked at individually, considering how similar problems have been solved previously (**pattern recognition**) and focusing only on the important details, while ignoring irrelevant information (**abstraction**). Next, simple steps or rules to solve each of the smaller problems can be designed (**algorithms**).

Let us discuss the components of Computational Thinking in detail.

COMPONENTS OF COMPUTATIONAL THINKING

Computational Thinking is a technique used to solve problems, logically, and consists of four key components or pillars described as follows:

- ☞ Decomposition
- ☞ Pattern Recognition
- ☞ Abstraction
- ☞ Algorithms

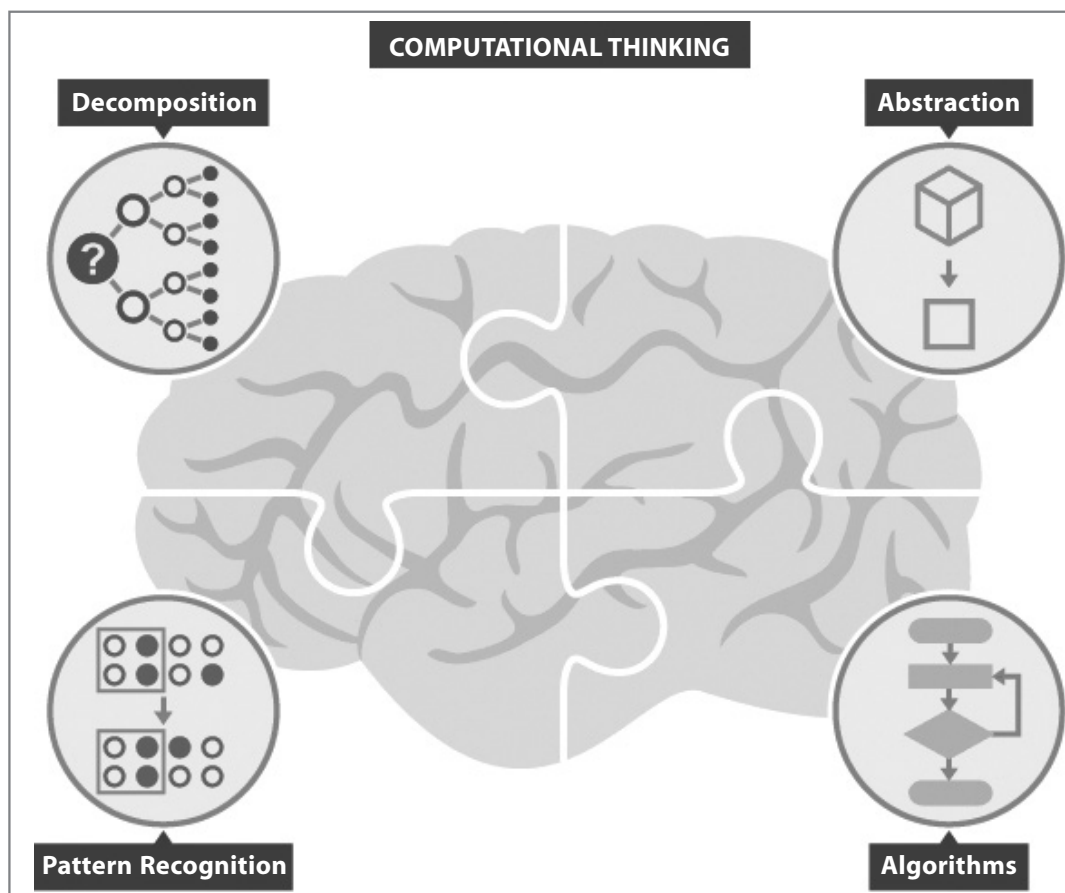


Fig. 1: Pillars of Computational Thinking

Decomposition

Decomposition is the process of breaking down a complex problem or system into smaller, more easily solved parts. These smaller problems are solved one after another until the bigger complex problem is solved.

This stage involves breaking down the problem into smaller components so that they can be tackled easily. The more you can break down a problem, the easier it is to solve.

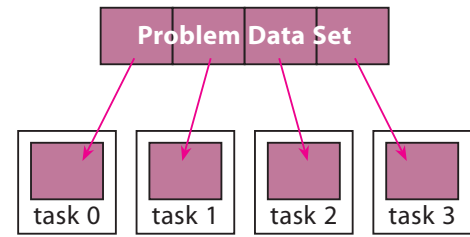


Fig. 2: Decomposition Process

Some real-life examples of decomposition process are:

- ☞ When we taste an unfamiliar dish and identify several ingredients based on the flavour, we are decomposing that dish into its individual ingredients.
- ☞ When we give someone directions to our house, we are *decomposing* the process of getting from one place to another (e.g., city, state, etc.).
- ☞ When we break a course project into several steps, we are decomposing the task into smaller, more manageable sub-tasks.
- ☞ In mathematics, we can *decompose* a number such as **256.37** as follows:
 $2*10^2 + 5*10^1 + 6*10^0 + 3*10^{-1} + 7*10^{-2}$

Let's look at a more specific example, the equation to work out the roots of a quadratic equation.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

On first look it might appear a little scary, but if we decompose it, we should stand a better chance of solving it:

- | | |
|---|--|
| 1. b^2 | 2. $4ac$ |
| 3. $b^2 - 4ac$ | 4. $\sqrt{b^2 - 4ac}$ |
| 5. $-b + \sqrt{b^2 - 4ac}$ | 6. $2a$ |
| 7. $x = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$ | 8. repeat for $\frac{-b - \sqrt{b^2 - 4ac}}{2a}$ |

By noting the steps down to solve a problem, we can often recognise patterns, and by giving a list of steps, we are one step closer to creating an algorithm.

This stage also allows you to develop a better understanding of the problem you face by identifying all the components in detail.

Pattern Recognition

Pattern Recognition is the next process in succession that helps in identifying patterns or trends within a problem. Once you've decomposed the complex problem into smaller problems, the next step is to look at similarities they share. We look for *patterns* when we play games to decide when to do certain things. Based on experience, we develop shortcuts for mapping problem characteristics to solution.

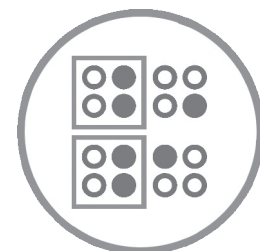


Fig. 3: Pattern Recognition



What are patterns?

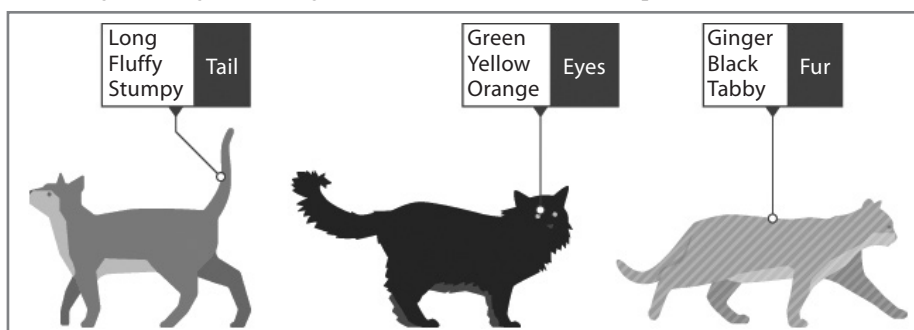
Imagine that we want to draw a series of cats.

All cats share common characteristics. Among other things **they all have eyes, tails and fur**. They also like to eat fish and make meowing sounds.

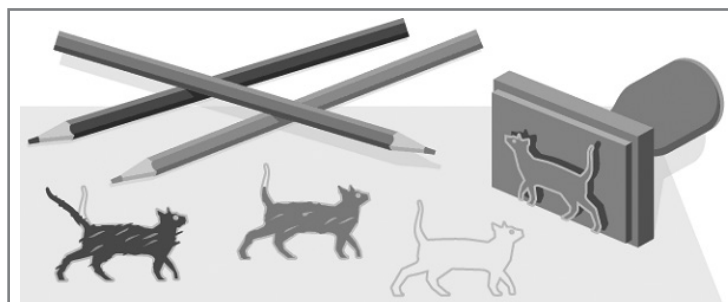
Because we know that all cats have eyes, tails and fur, we can make a good attempt at drawing a cat, simply by including these common characteristics.

In computational thinking, these characteristics are known as patterns. **Once we know how to describe one cat, we can describe others simply by following this pattern.** The only things that are different are the specifics:

- one cat may have green **eyes**, a long **tail** and black **fur**
- another cat may have yellow **eyes**, a short **tail** and striped **fur**



- If we want to draw a number of cats, finding a pattern to describe cats in general, *for example*, they all have eyes, tails and fur, makes this task quicker and easier.
- We know that all cats follow this pattern, so we don't have to stop each time we start to draw a new cat to work this out. From the patterns we know cats follow, we can quickly draw several cats.



We experience pattern recognition in everyday situations like:

- ☞ Drivers look for patterns in traffic to decide whether and when to switch lanes.
- ☞ People look for patterns in stock prices to decide when to buy and sell.
- ☞ Scientists look for patterns in data to derive theories and Models.
- ☞ We look for patterns and learn from them to avoid repeating the same mistake.

Patterns are shared characteristics that occur in each individual problem. What similarities do you observe? Finding these similarities in small decomposed problems can help us solve complex problems more efficiently.



Abstraction

“Abstraction” refers to focusing on the important information only, ignoring irrelevant detail. To reach a solution, we need to ignore unnecessary characteristics in order to focus on those that we do. It helps to identify specific similarities and differences among similar problems to work towards a solution.

So what is this important information that we need to focus on? In abstraction, the focus is mainly on general characteristics that are common to each element, instead of specific details. Common examples of abstraction process can be:

- ☞ A world map is an *abstraction* of the earth in terms of longitude and latitude, helping us describe the location and geography of a place.
- ☞ A sign of an aisle in a store—*e.g.*, Walmart—is an *abstraction* of the items available in that aisle.
- ☞ When we write a book report, we summarize and discuss only the theme or key aspects of the book, it is abstraction.

Once you have the general characteristics, you can create a “model” of the problem; a model being the general idea of the problem we are trying to solve.

Once we have a model, we can design an algorithm.

Algorithm Writing

You’ve broken down the big problem into smaller, easily manageable problems. You’ve identified similarities between these problems. You’ve focused on relevant details and left behind anything irrelevant.

Now it’s time to develop step-by-step instructions to solve each of the smaller problems, or the rules to follow when solving the problem. These simple steps or rules are used to program a computer to help solve a complex problem in the best possible way. They are also called “algorithms”.

Definition: An algorithm is a plan, a set of step-by-step instructions used to solve a problem.

Writing an algorithm requires extensive planning for it to work correctly. The solution your computer offers is as good as the algorithm you write. If the algorithm is not good, then your solution will not be good either.

INTEGRATING COMPUTATIONAL THINKING WITH ACADEMICS

Computational thinking is extremely important not only while working with computers but in other applications also. Computational thinking even works well in the classroom teaching as well. The implementation of computational thinking in a classroom can be compared with various important subjects.

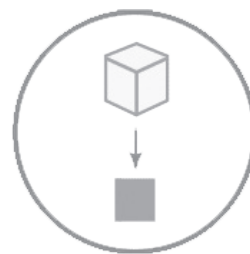


Fig. 4: Abstraction



Fig. 5: Algorithm Writing



Computational Thinking Concept	Subject Area Application
Break down a problem into parts or steps	Literature: Break down the analysis of a poem into analysis of metre, rhyme, imagery, structure, tone, diction and meaning.
Recognize and find patterns or trends	Economics: Find cycle patterns in the rise and fall in the country's economy.
Develop instructions to solve a problem or steps for a task	Culinary Arts: Write a recipe for others to use.
Generalize patterns and trends into rules, principles, or insights	Mathematics: Figure out the rules for factoring 2nd-order polynomials
	Chemistry: Determine the rules for chemical bonding and interactions.

In the left column, notice that all of the skills are computational thinking skills or concepts. However, in the right column, those skills are being used in literature, economics, the culinary arts, and music. The basic skills of computer scientists and the way they think are computational thinking. The area in which you apply computational thinking can be any subject area or topic, even the subject area or topic you teach. These ways of thinking can be used anytime you want to develop a process or algorithm to solve a problem.



MEMORY BYTES

- Computational Thinking allows us to take a complex problem, understand what the problem is, and develop possible solutions.
- Decomposition, pattern matching, abstraction and algorithms are the four pillars of Computational Thinking.
- Decomposition is the process of breaking down a complex problem or system into smaller, more manageable parts.
- Pattern matching refers to looking for similarities among and within problems.
- Algorithm refers to the rules to follow to solve a problem.
- Computational Thinking enables us to work out exactly what to tell the computer to do.

SOLVED QUESTIONS

1. What are the four main parts of computational thinking?

Ans.

- Decomposition
- Pattern Recognition
- Pattern generalization and abstraction
- Algorithm design

2. Can you spot a pattern in this set of numbers?

- 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

Ans. In programming terms, the sequence F_n of Fibonacci numbers, i.e., the third number, is the sum of two previous consecutive numbers and is defined by:

$$F_n = F_{n-1} + F_{n-2};$$

$$\text{where } F_1 = 0, F_2 = 1$$

with seed (initial or starting) values



3. Can you come up with an algorithm to make a cup of tea?

- Ans.** (i) put water into kettle (ii) turn kettle on
(iii) get a cup and saucer (iv) put tea bag into cup
(v) when water gets boiled, add to cup (vi) stir with spoon
(vii) remove tea bag (viii) add milk, if needed

4. Define the term computational thinking.

Ans. Computational Thinking allows us to take a complex problem, understand what the problem is, and develop possible solutions. We can then present these solutions in a way that a computer, a human, or both, can understand.

5. What is decomposition?

Ans. Decomposition is one of the four cornerstones of Computer Science. It involves breaking down a complex problem or system into smaller parts that are more manageable and easy to understand. The smaller parts can then be examined and solved, or designed individually, as they are simpler to work with.

6. Why is decomposition important?

Ans. If a problem is not decomposed, it is much harder to solve. Dealing with many different stages all at once is much more difficult than breaking down a problem into a number of smaller problems and solving each problem one at a time. Breaking down the problem into smaller parts means that each smaller problem can be examined in more detail.

7. Why do we need pattern recognition?

Ans. Pattern recognition is one of the four cornerstones of CT. It involves finding the similarities or patterns among small, decomposed problems that can help us solve more complex problems efficiently.

8. Why is abstraction important?

Ans. Abstraction allows us to create a general idea of what the problem is and how to solve it. The process instructs us to remove all specific details, and any patterns that will not help us solve our problem. This helps us form our idea of the problem. This idea is known as a 'model'.

9. Explain the characteristics of computational thinking.

Ans. The salient features of computational thinking are:

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

UNSOLVED QUESTIONS

1. What are some computational thinking skills?
2. How are computational thinking skills applied in finding solutions that can be interpreted into software applications?
3. What is an example of computational thinking?
4. How can I use computational thinking for problem-solving?
5. What is the difference between computational thinking and algorithmic thinking?
6. Explain the components of computational thinking.
7. What is pattern matching?
8. "Decomposition leads to simplicity." How?
9. Explain the significance of planning in computational thinking.
10. Write an algorithm to find a factorial of an inputted number.