pens

2023

Introduction of Computational Thinking

Logic Algorithm

Tri Hadiah Muliawati

Politeknik Elektronika Negeri Surabaya
Department of Informatics and Computer Engineering

Objectives:

1. Students able to understand the importance of computational thinking
2. Students able to understand the pillars of computational thinking
3. Students able to implement computational thinking in everyday life

Outline:

1. Overview
2. Pillars of computational thinking
3. Examples

Overview

What actually is Computational Thinking?

Computational thinking (CT)'s definition has been a hot debate for years. Many people associate it with Computer Science (CS) or merely programming. However, it's much more than that. CT concepts are hardly exclusive to CS.

CT is 'a focused approach to **problem solving**, incorporating thought processes that utilize **abstraction, decomposition, algorithms, evaluation, and generalizations**' (Shelby, 2013). Therefore, it's applicable in a diverse array of fields.

Definition	Source
‘Computational thinking is 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**.’	Wing, 2014
‘The mental activity for **abstracting problems and formulating solutions** that can be **automated**.’	Yadav et al., 2014
‘A mental orientation to **formulating problems** as conversions of **some input to an output and looking for algorithms to perform** the conversions. Today the term has been expanded to include thinking with many levels of abstractions, use of mathematics to develop algorithms, and examining how well a solution scales across different sizes of problems.’	Denning, 2009
‘[Teaching CT is teaching] how to think like an economist, a physicist, an artist, and to understand how to **use computation to solve their problems**, to create, and to discover new questions that can fruitfully be explored.’	Hemmendinger, 2010

What actually is CT?

In relation to programming, CT teaches an **approach to problem-solving** where the ultimate **aim is to provide a solution** whose form means it is ready to be programmed into a computer.

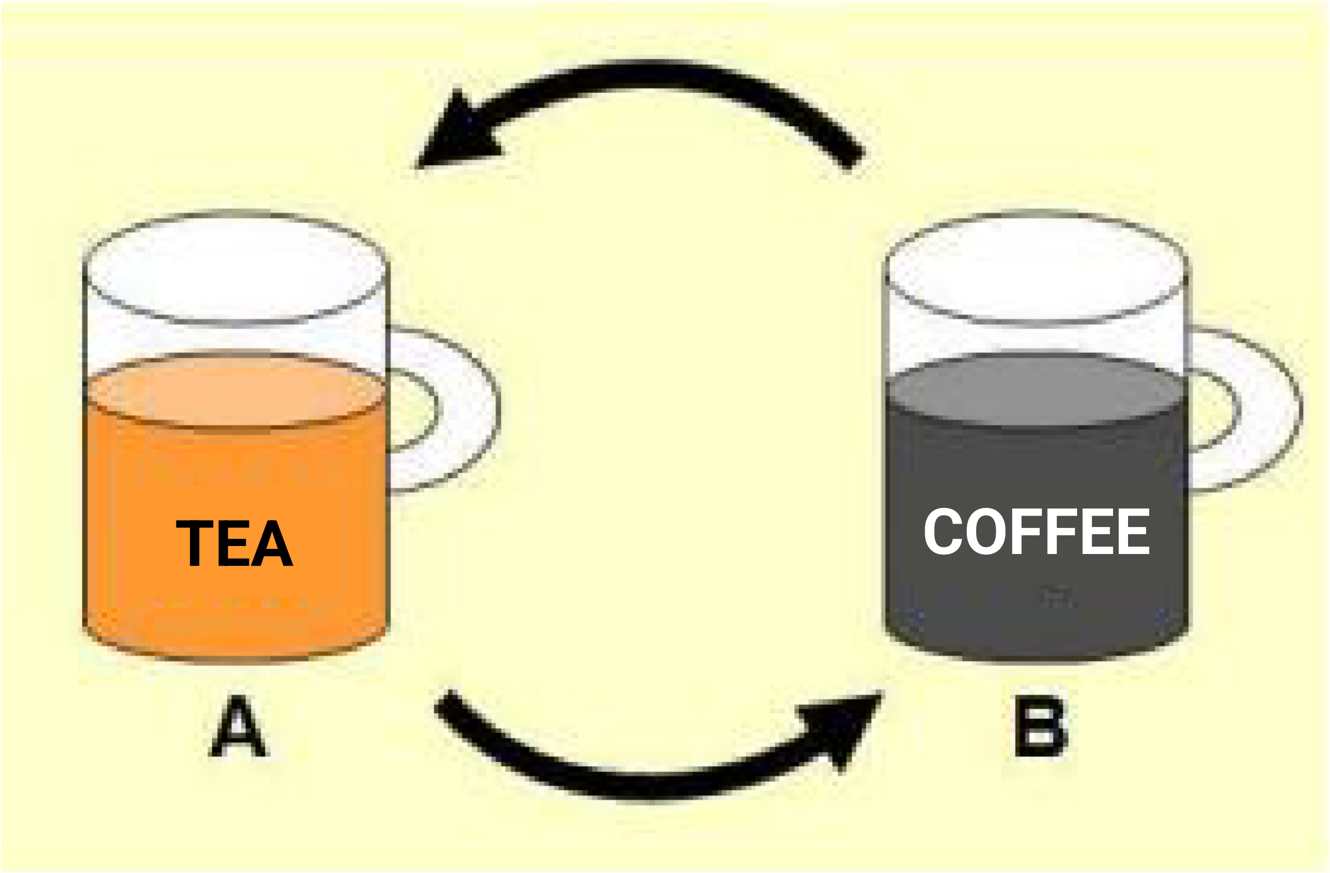
Let’s throwback to algorithm

Algorithm is a **way of specifying a multi-step task**, and is especially useful when we wish **to explain to a third party (be it human or machine) how to carry out steps with extreme precision**.

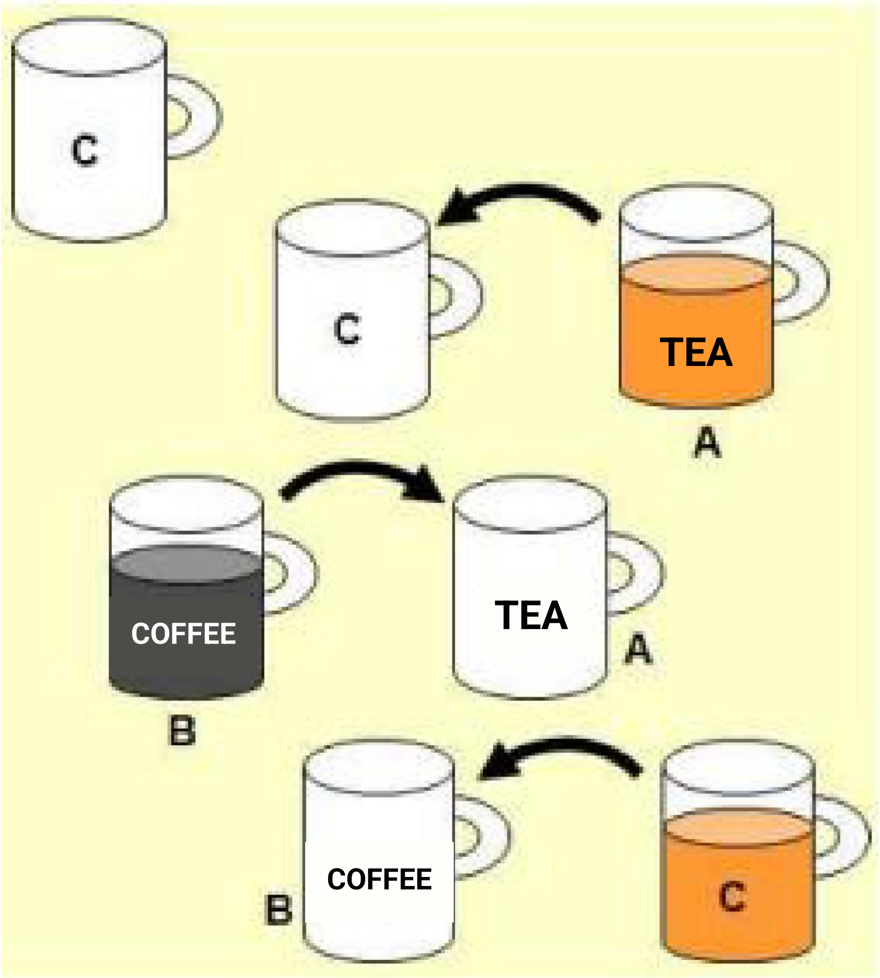
The properties of an algorithm are:

1. Finiteness
2. Definiteness
3. Input
4. Output
5. Effectiveness

Case Study: Swapping Tea and Coffee



Case Study: Swapping Tea and Coffee



1. Prepare glass C as a spare cup
2. Pour tea from glass A into glass C (glass A becomes empty).
3. Pour coffee water from glass B into glass A (glass B becomes empty).
4. Pour the tea from glass C into glass B.

Case Study: Planning a graduation ceremony

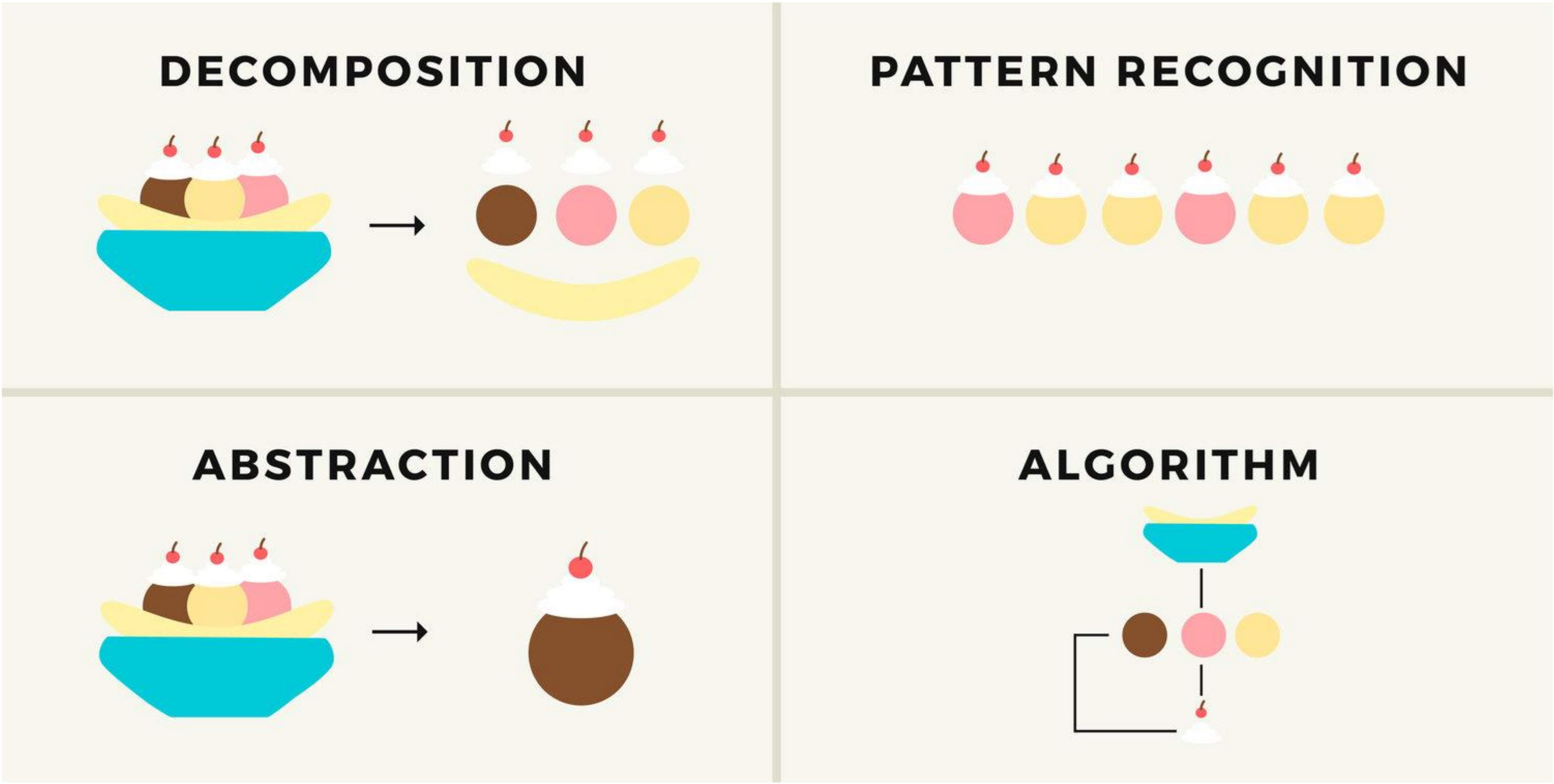
Why algorithm (alone) is not sufficient to solve a problem?

Sometimes we deal with **a complex problem** which may overwhelm us and makes it **difficult to directly form an algorithm to solve it**.

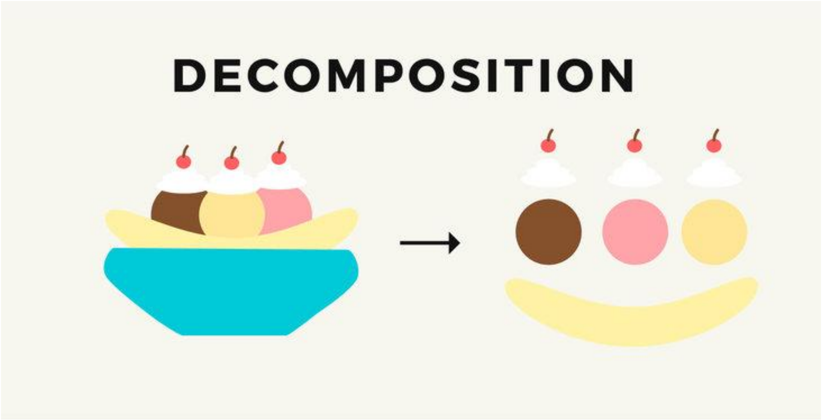
CT breaks down a complex problem into smaller, more comprehensible tasks. Therefore, CT helps us to recognize that some tasks that might **seem very difficult at first are actually very doable**.

Pillars of CT

Pillars of CT

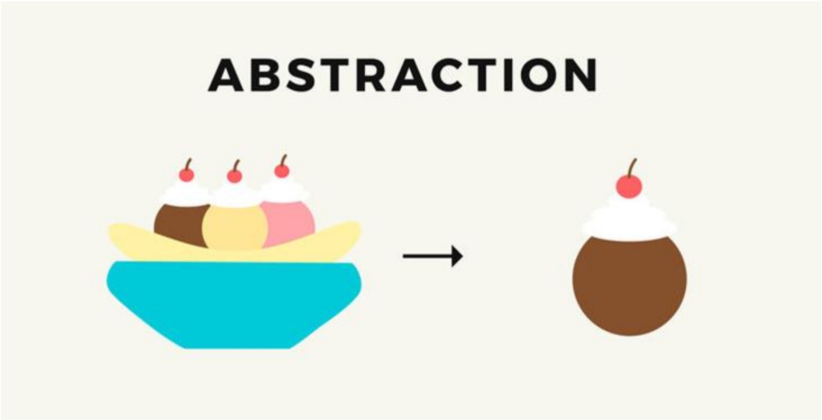


Pillars of CT (1/4)



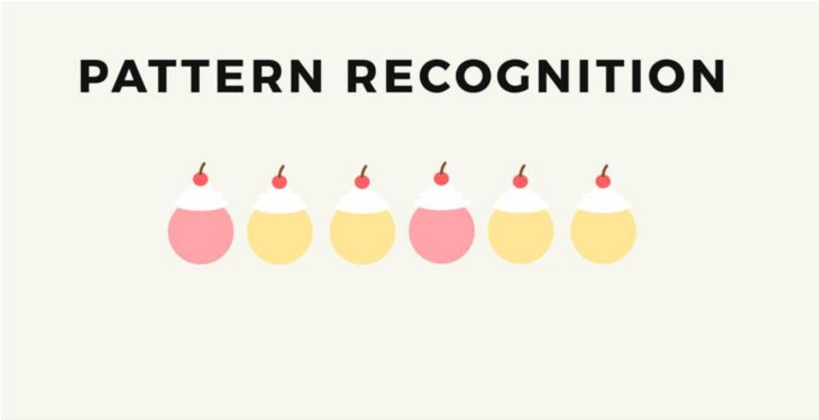
Decomposition refers to breaking down a big problem into small, manageable parts.

Pillars of CT (2/4)



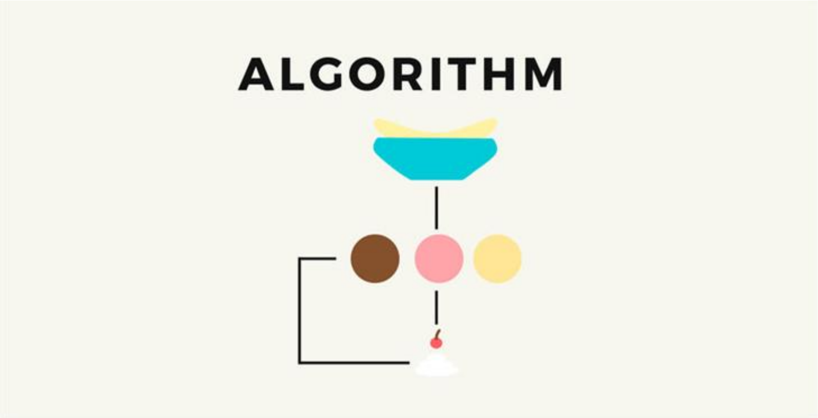
Abstraction refers to identifying and extracting the important parts of the problem.

Pillars of CT (3/4)



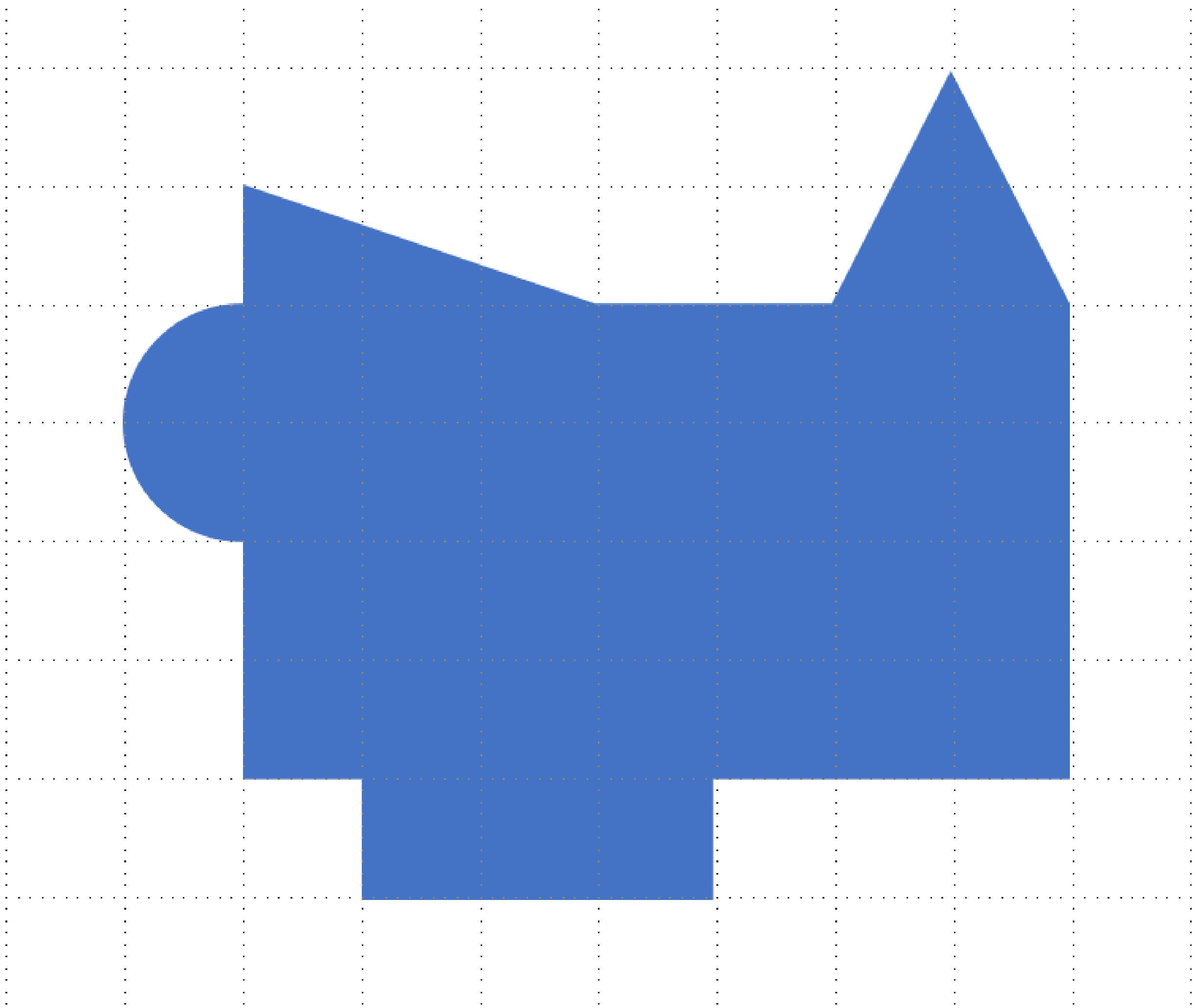
Pattern recognition refers to observing similarities and patterns in these smaller parts, to help us solve complex problems more efficiently.

Pillars of CT (4/4)



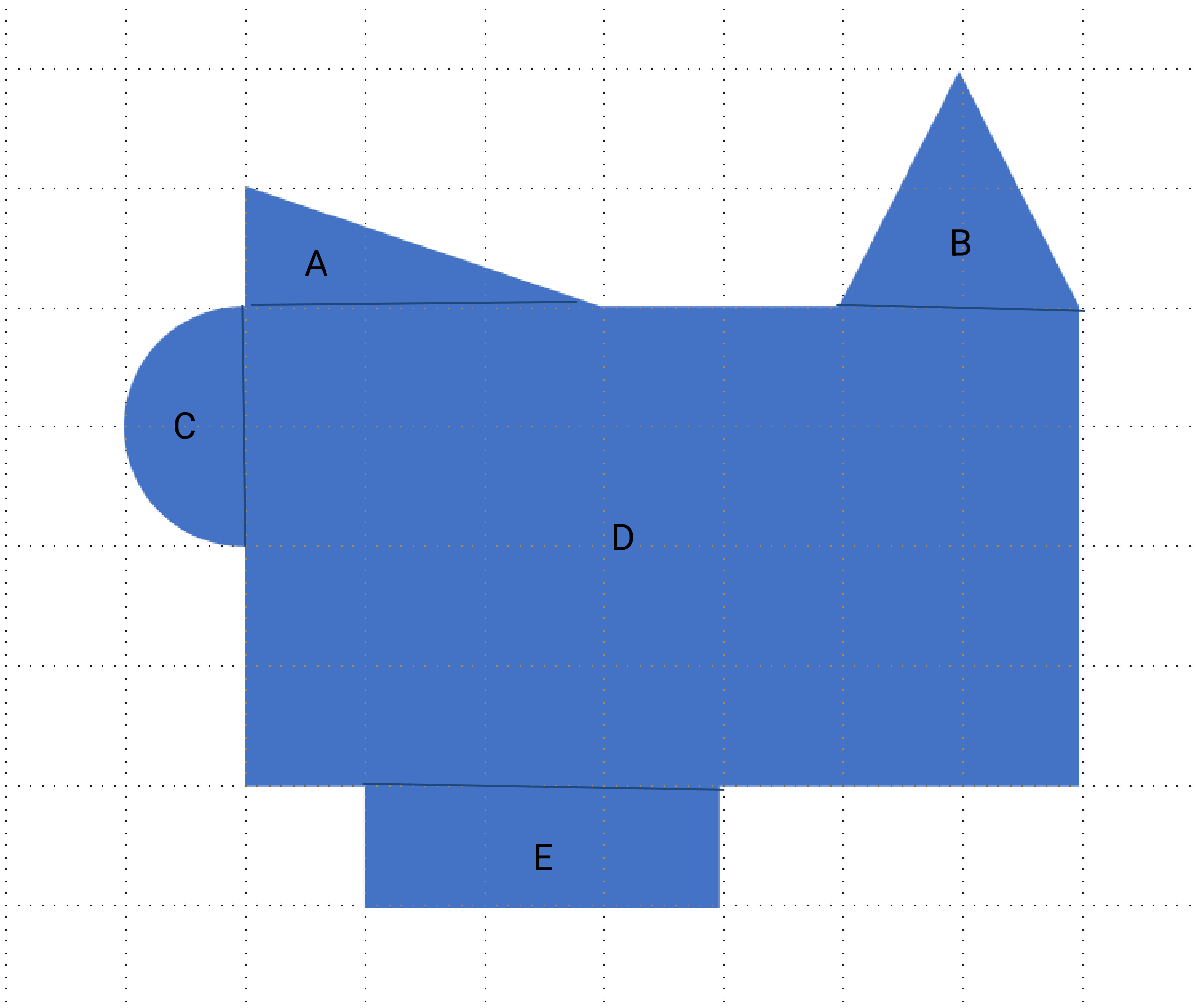
Algorithm refers to creating the step-by-step instructions of solving the problem.

Exercise



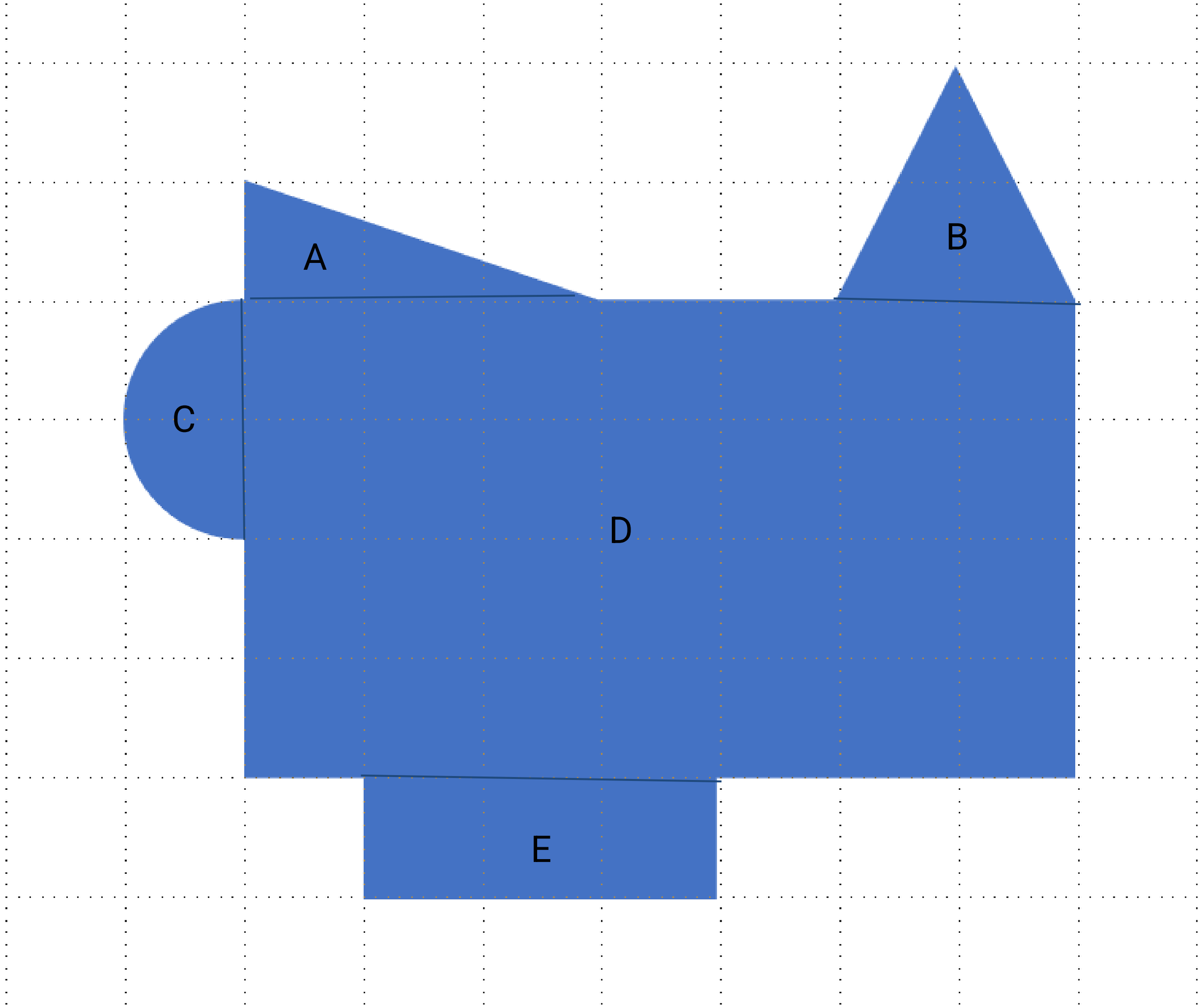
Try to implement CT to calculate the blue section area.

Answer



Decomposition:
Notice that the blue section can be split into several smaller sections to make it more manageable.

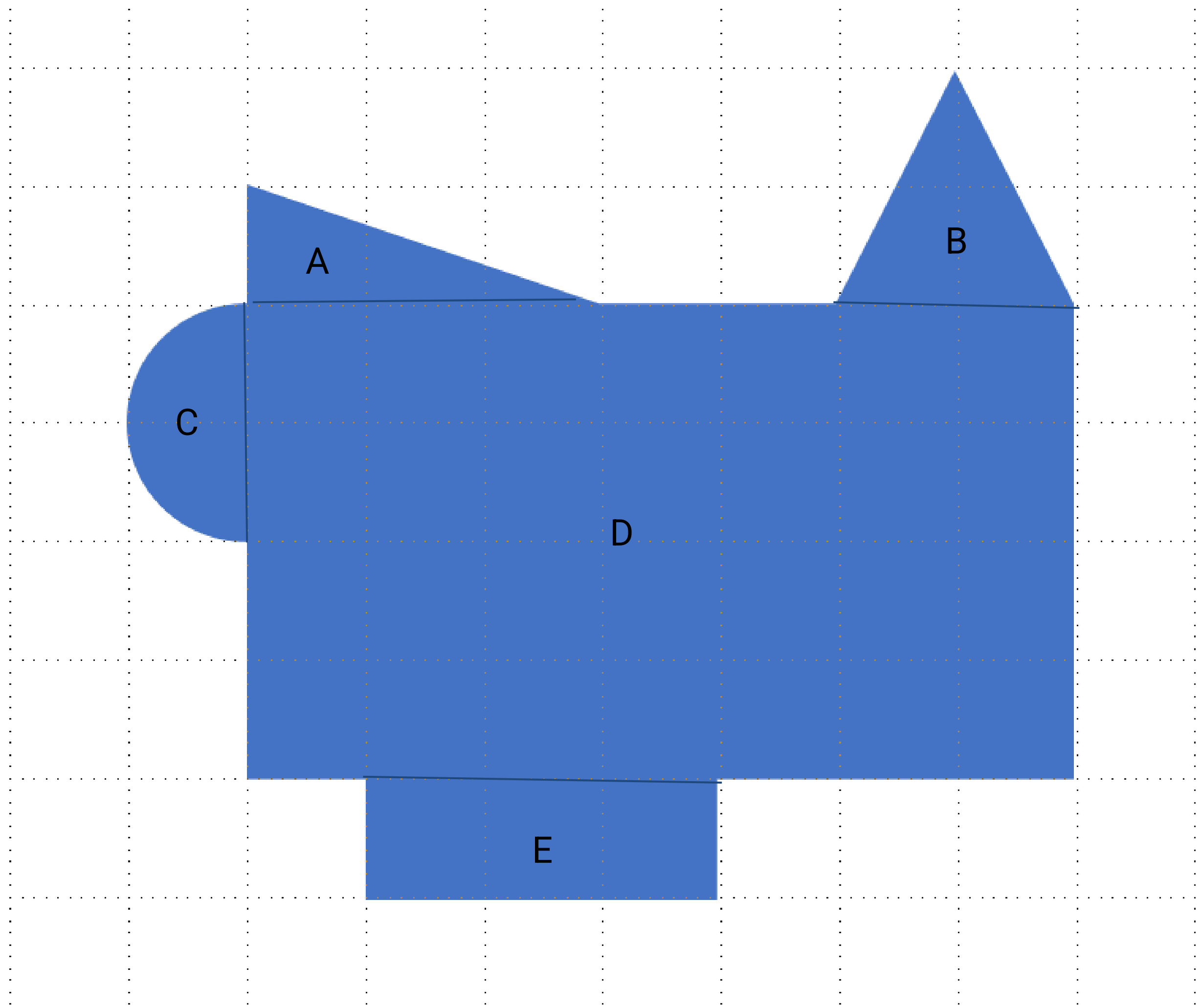
Answer



Abstraction:
Notice each section mostly resembles different geometric shapes. It means that they may need different equation to calculate each section's area.

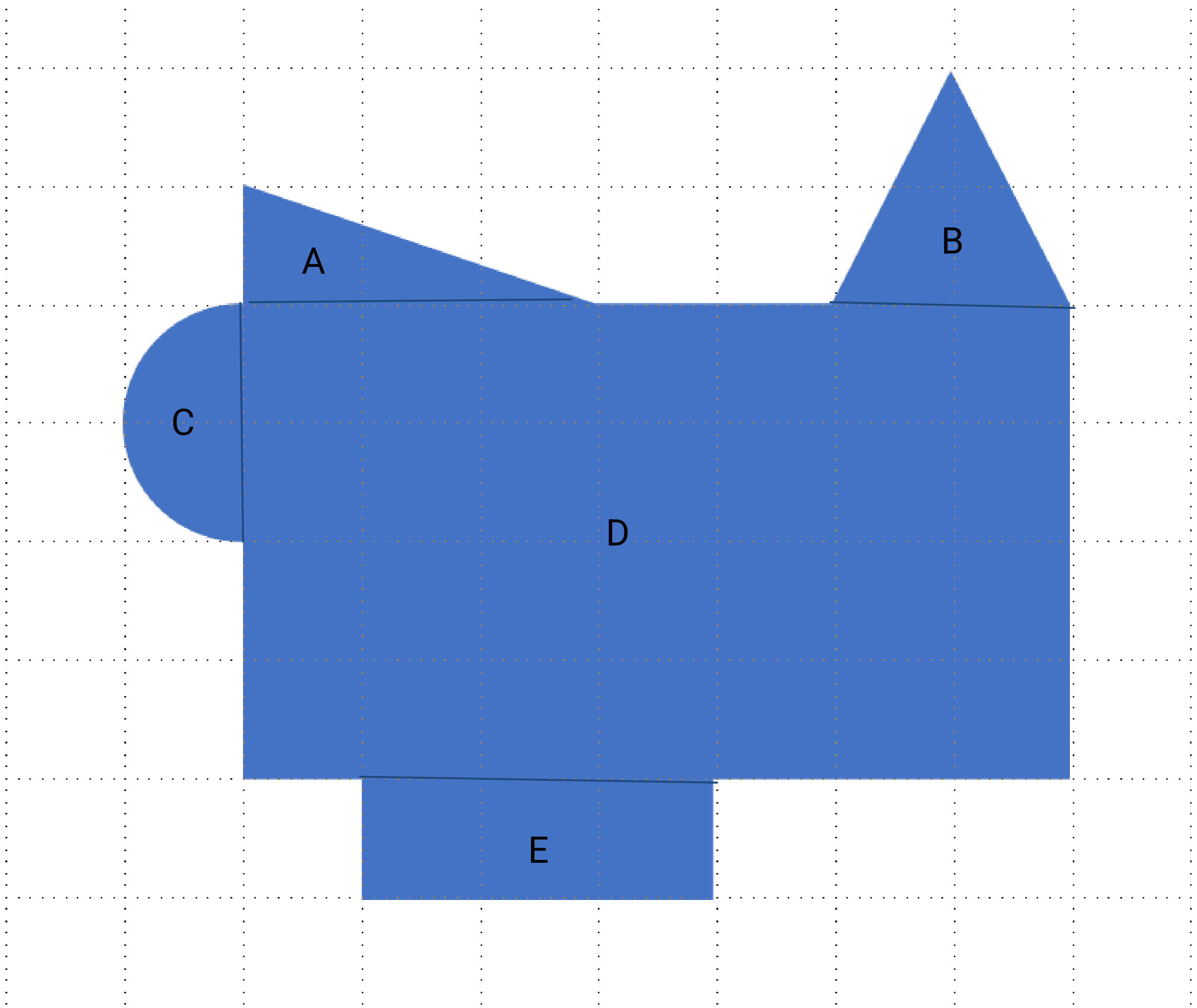
Thus, we need to know the important features to calculate each section's area.

Answer



Pattern Recognition:
Notice that some sections (Section A and B) have the same pattern, which is a triangle. Thus, we can use the same equation for both section's areas.

Answer



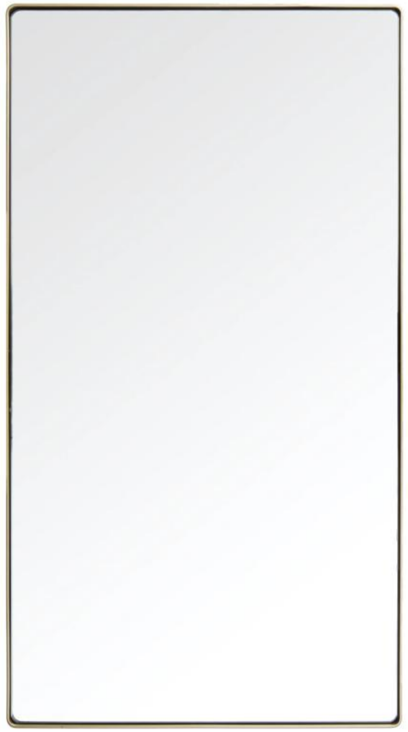
Algorithm:

Construct an algorithm to calculate the area of blue section by putting together the area of each section.

Pillars of CT

- Use them appropriately.
- It is not necessary to use all pillars when solving every problem.

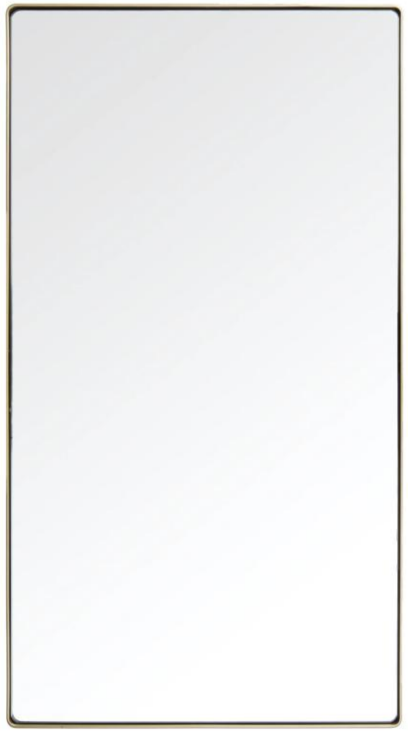
Example:
- You have a standing rectangle mirror.
- Try to calculate its mirror area.



Pillars of CT

Example:

- You have a standing rectangle mirror.
- Try to calculate its mirror area.



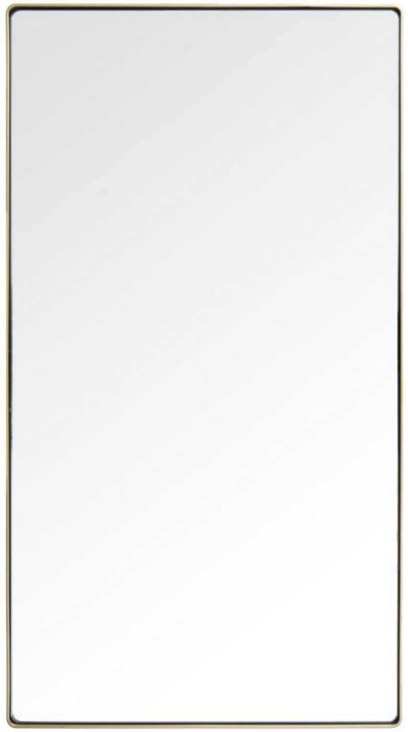
- Define which attribute is needed to calculate the area.
- Since the area of rectangle is calculated using length times width, we do not need to measure the perimeter (Abstraction).

Pillars of CT

Example:

- You have a standing rectangle mirror.
- Try to calculate its mirror area.

- We cannot calculate the area without knowing the value of its width and length.
- Thus, we need to measure its length and width before calculating the area. Then, we can calculate the area (Algorithm).



Examples

Pipelining a graduation ceremony

Dean Randy Bryant was pondering how to make the diploma ceremony at commencement go faster.

By careful placement of where individuals stood, he designed an efficient pipeline so that upon the reading of each graduate's name and honors by Assistant Dean Mark Stehlik, each person could receive his or her diploma, then get a handshake or hug from Mark, and then get his or her picture taken.

Developing an app to track calories

He breaks down the problem of building a calorie tracking app into smaller, more manageable pieces **(Decomposition)**. These problems include designing user interfaces, developing backend databases to store user data, and application programming to calculate and load daily calorie intake.

At this stage, he identifies and uses other commonly used application design patterns **(Pattern recognition)**. For example, he can adapt and modify application design patterns used by competitors.

Developing an app to track calories

He fixes the problem by ignoring unnecessary things **(Abstraction)**. For example, he can take the complexity out of building an application by leveraging an existing database management system instead of building it himself.

He then worked out how to complete the calorie tracking app **(Algorithm)**. For example, he can create a set of rules that allows an application to calculate how many calories are in different foods, or how many calories to consume each day.

Others

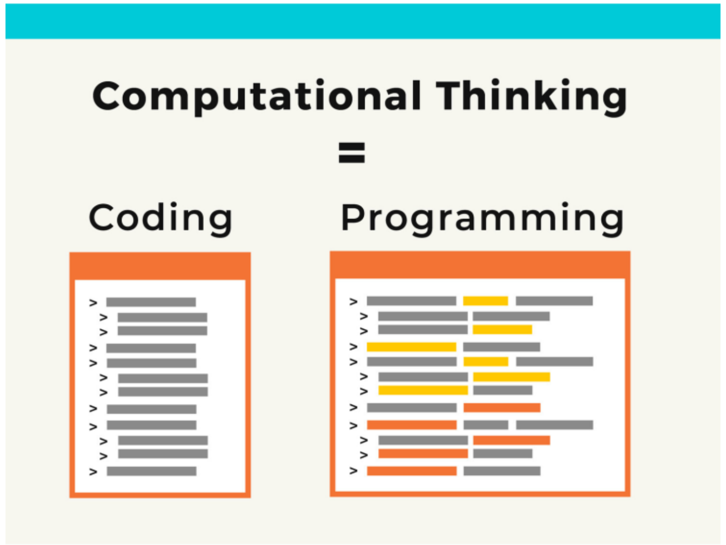
Benefit

According to McClelland and Lori Grata (2018), "Computational thinking provides students with **problem-solving skills that are applicable in numerous scenarios**."

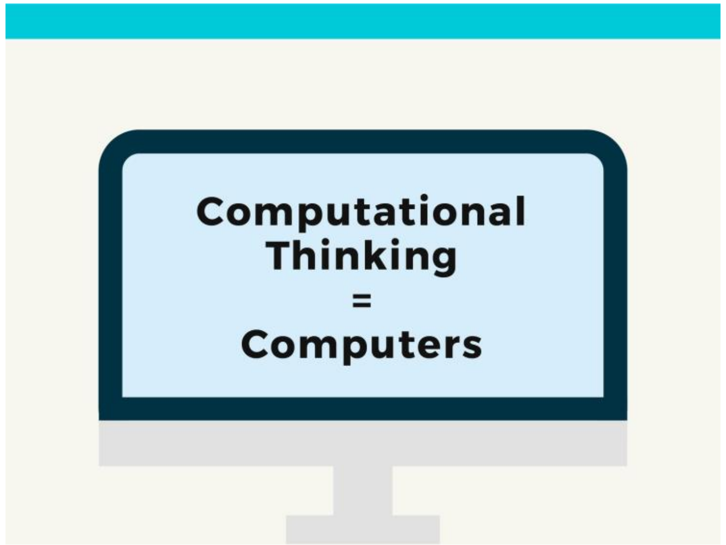
With computational thinking skills, students will have:

- Confidence and persistence when working on complex problems
- Higher skills in handling the ambiguity in given problems
- Ability to deal with open-ended problems
- Better communication and collaboration skills

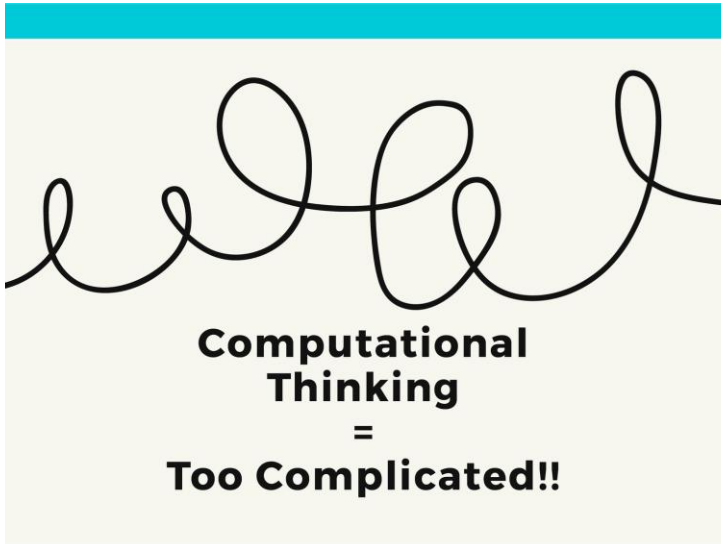
Common Misconceptions



Computational Thinking is the logic and skillset behind Coding and Programming



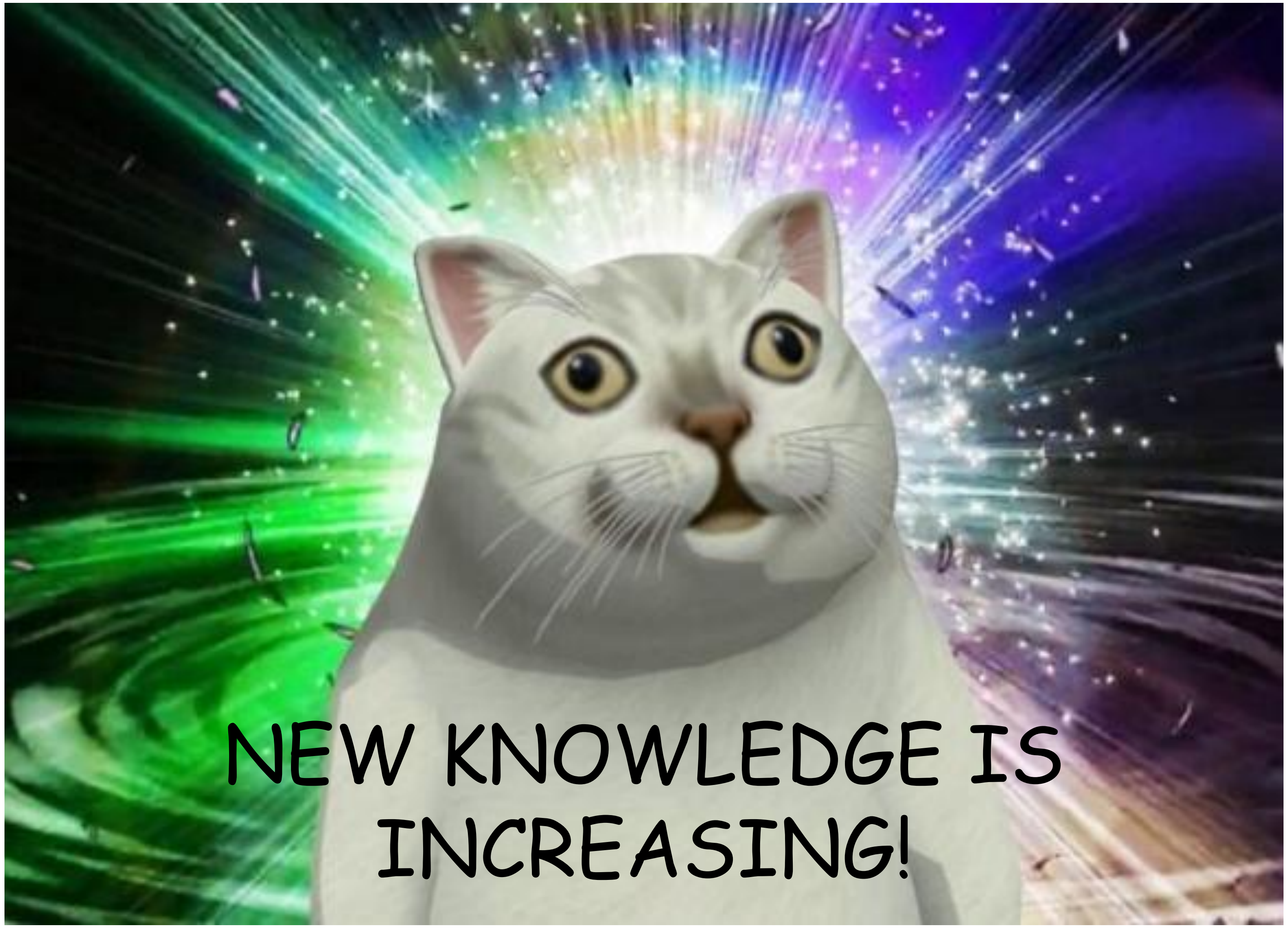
Computational thinking is a soft, transferable skill that isn't strictly tied to technology and computers.



According to various education experts, ages 4 to 7 is an appropriate time to pick up Computational Thinking

Summary

- CT is a problem solving approach. Thus, it is applicable in a diverse array of fields.
- In relation to programming, CT is an approach to problem-solving where the ultimate aim is to provide a solution whose form means it is ready to be programmed into a computer.
- CT consists of 4 pillars, which are decomposition, pattern recognition, abstraction, and algorithm. We do not necessarily use them altogether to solve every problem.


NEW KNOWLEDGE IS
INCREASING!

Task

🧩 **Problem: Organizing a Science Fair**

Scenario:

Your school is organizing a IT fair with 20 student projects. Each project needs a table, a poster board, and 2 hours of presentation time. You are in charge of scheduling and organizing the event efficiently.

Using computational thinking, describe how you would:

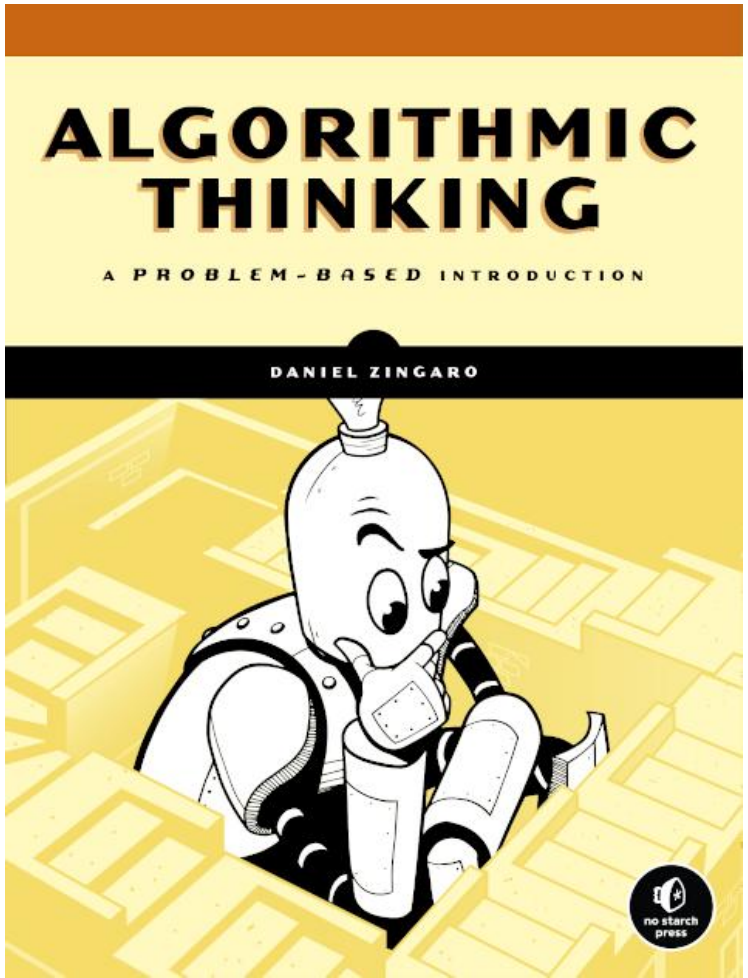
References

1. Karl, Beecher. "Computational Thinking: A Beginner's Guide to Problem-Solving and Programming." Swindon, UK: BCS, The Chartered Institute for IT (2017).
2. McClelland, Katharine, and L. A. Grata. "Review of the Importance of Computational Thinking in K-12." Proceedings of the eLmL (2018): 2-34.
3. https://www.tinythinkers.org/benefits

Further References

1. Inggriani Liem in Bukatalks. 2018. “Computational Thinking”. https://www.youtube.com/watch?v=_6D0ks7wvtI
2. http://www.bebras.or.id/

Next week



- We will learn about **Algorithmic Thinking & Flowchart**. Please read references about it.
- It is **strongly encouraged** to do **self study and self-paced practicum.**

THANK YOU!