CS23101 - COMPUTATIONAL THINKING

UNIT 1 - INTRODUCTION TO COMPUTATIONAL THINKING

- Understanding the concepts:
  - Decomposition,
  - pattern recognition / data representation,
  - generalization / abstraction and
  - Algorithms.
- Complexity, Modeling, Automation, Analysis, visualization.

WHAT IS COMPUTATIONAL THINKING ?

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, 2014)**

WHY IS IT NEEDED ?

- 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.



KEY SKILLS FOR COMPUTATIONAL THINKING

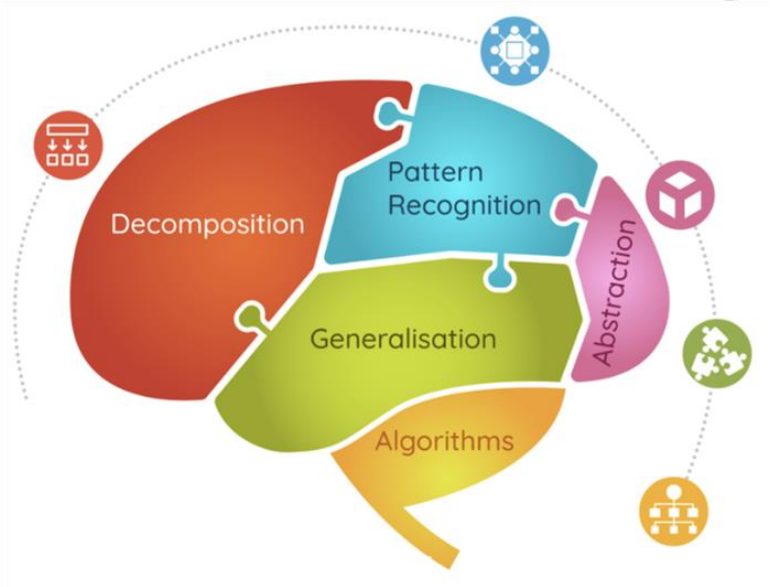
Decomposition - Breaking a big problem down into smaller parts

Pattern Recognition - Looking for similarities between and within problems

Generalisation - Adapting solutions to other problems to solve new ones

Abstraction - Taking the detail out of a problem and ignoring irrelevant information

Algorithm - Simple rules to follow that solve the problem



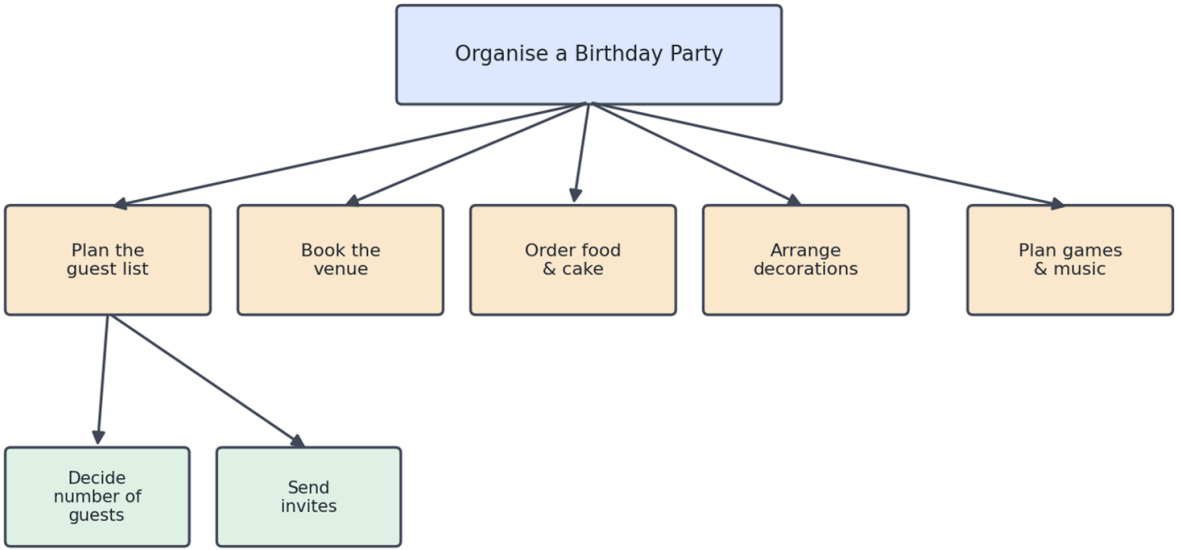
DECOMPOSITION

Breaking one large, complicated problem into smaller pieces that are each easy to solve on their own.

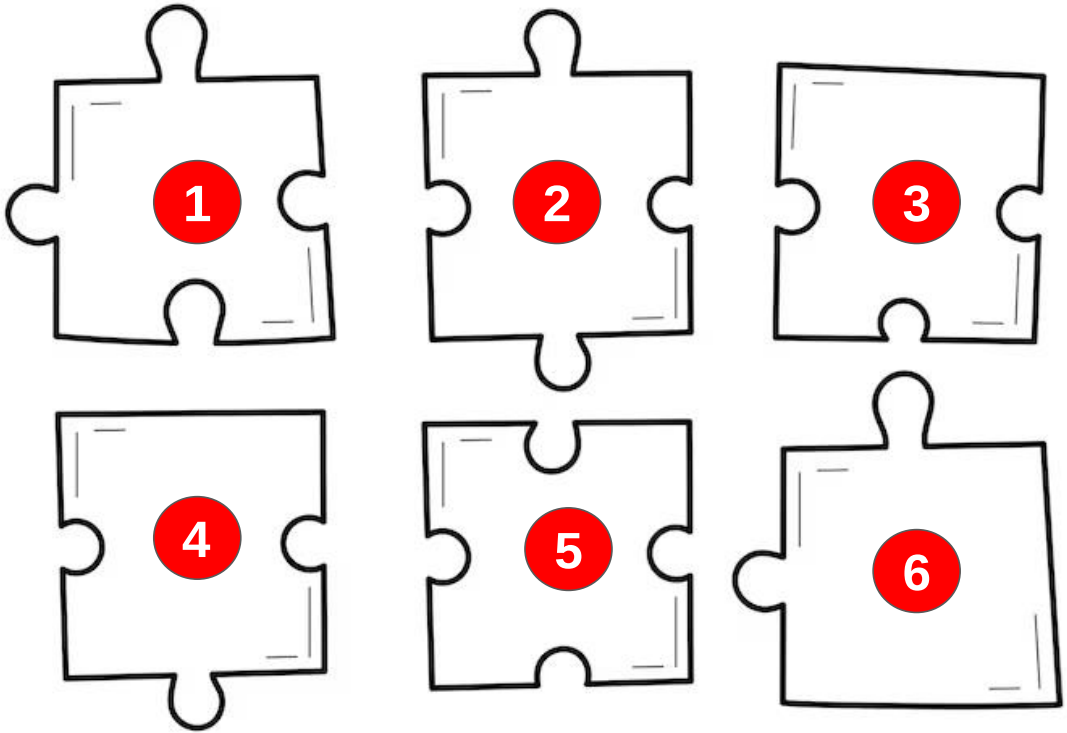
- The process of breaking a complex problem or system down into smaller, more manageable parts.
- Each smaller piece is easier to understand, test, and solve independently before combining the solutions to fix the overall problem

Why Decomposition matters ?

- Makes overwhelming problems easy to start
- Lets different people work on different parts at the same time
- Helps find and fix errors faster

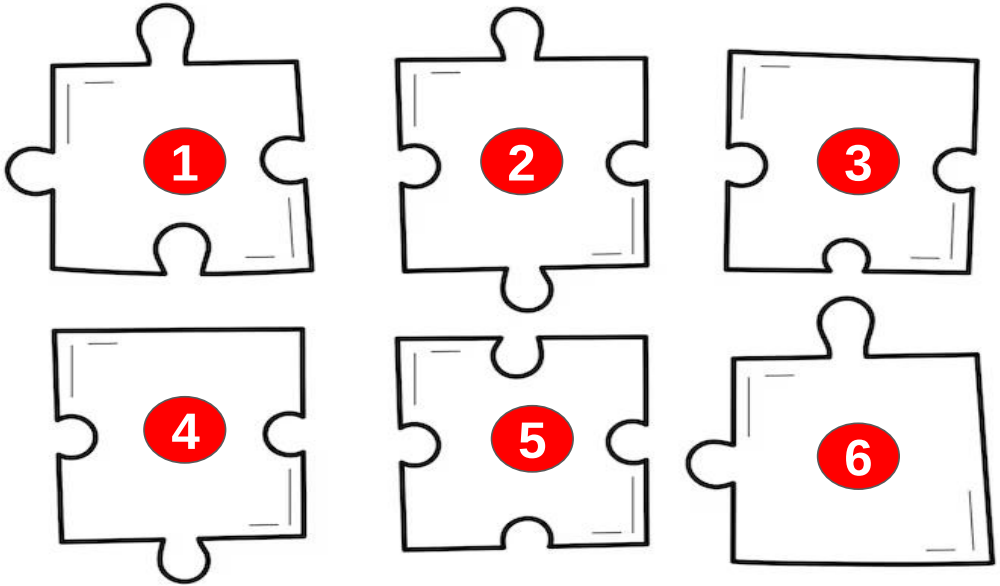


A single goal broken down into smaller sub-tasks, some of which break down further



Which piece would you pick first to solve and why ?

Answer in next page



Piece 6, because it has two straight edges, making it a corner piece.
A corner only fits in one place in the whole puzzle, so it's the fastest one to solve first.
This is decomposition.

Some Furbles are trying to escape from Count Pengula in the dead of night. To be free, they only have to cross one bridge, but the bridge is weak and can only hold two of them at a time. They also need to use a lantern to help them find the way safely across. The problem is they only have one Jack-o-lantern between them and the candle burns out in **seventeen minutes.** How can all of the group cross the bridge before the candle burns out?



Original Problem : Get everyone across the bridge safely before the lantern burns out in 17 minutes.

That is a large problem with several constraints:

- Only 2 people can cross at once.
- The lantern must always be carried.
- Someone may need to bring the lantern back.
- Everyone has to reach the other side.
- The total time must be ≤ 17 minutes.

Instead of solving all of this at once, **decompose** :

- Break the group into crossing pairs
- Solve the lantern-return problem
- Solve the next crossing
- Repeat the smaller problem

Furble	Time
	1
	2
	5
	10

Because of injuries, some Furbles move faster than others. The table shows the time in minutes it takes each to cross the bridge.

Constraints:

- Only 2 people can cross at once.
- The lantern must always be carried.
- Someone may need to bring the lantern back.
- Everyone has to reach the other side.
- The total time must be ≤ 17 minutes.

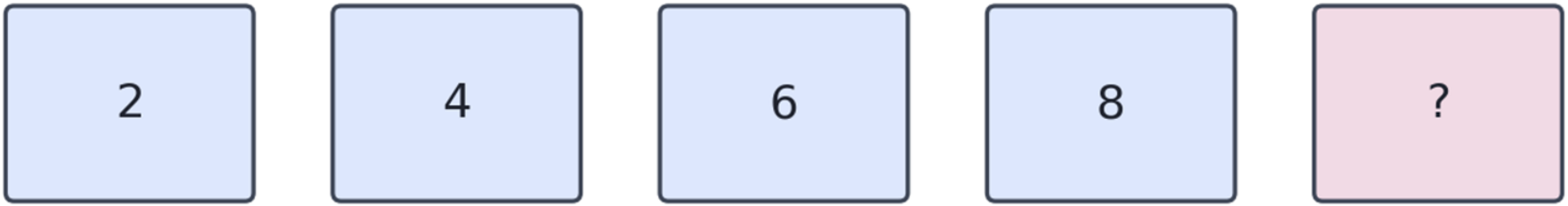
Answer in next page

First crossing		max(1,2)	**2**
First return		max(1)	**3**
Second crossing		max(5,10)	**13**
Second return		max(2)	**15**
Final crossing		max(1,2)	**17**

PATTERN RECOGNITION

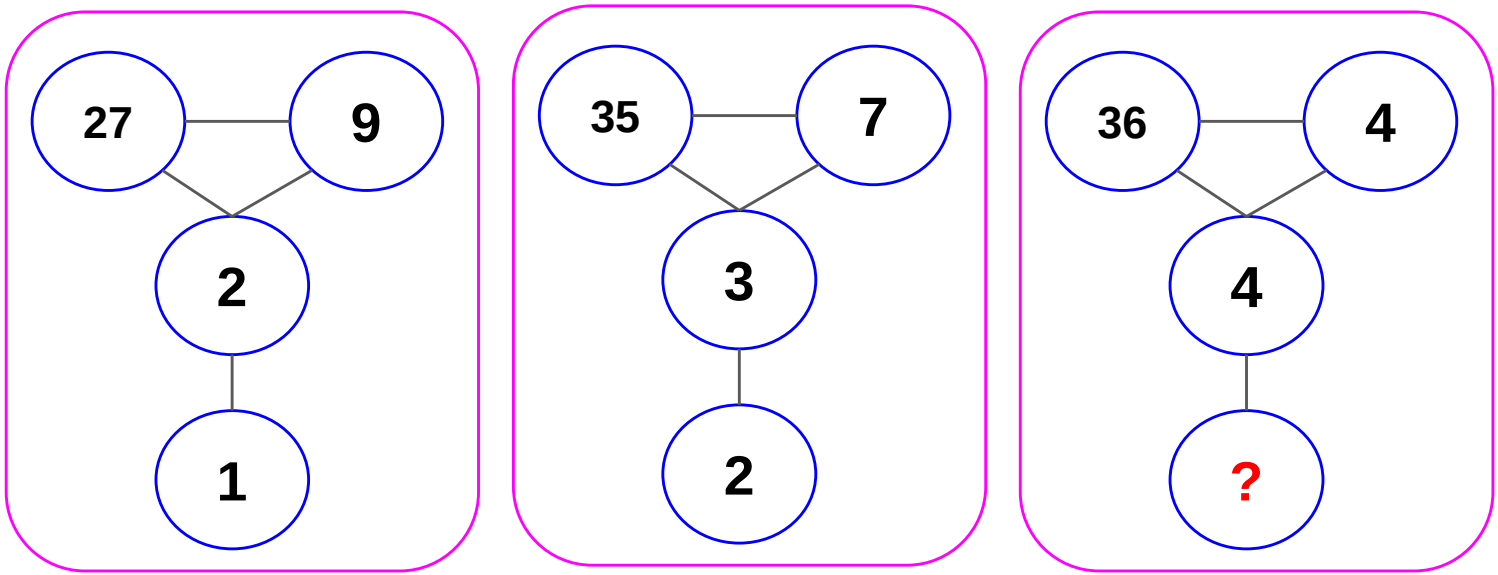
Looking for similarities between and within problems

- The process of looking for similarities, trends, and repeated structures within or across problems.
- Once a pattern is spotted, you can predict what comes next, or reuse a solution you already know.
- Used in weather forecasting, stock market trends, fraud detection in banking

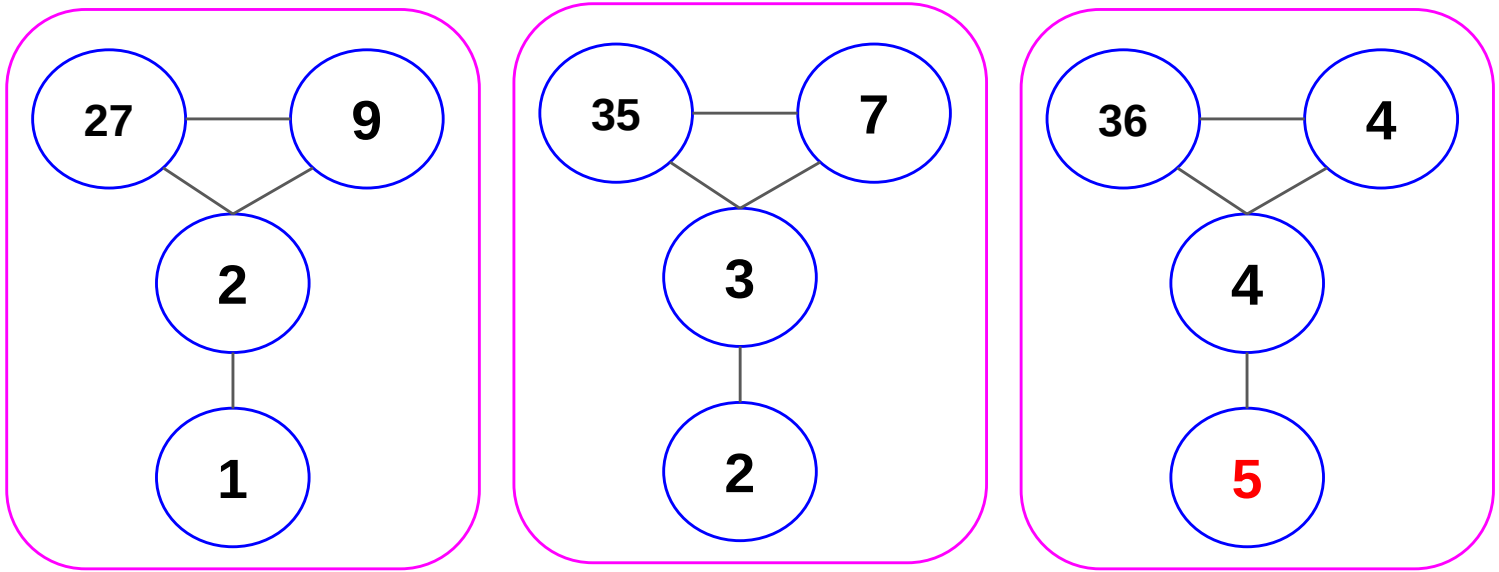


Problem : Spotting a repeating rule in a sequence.
Pattern noticed : Each number is 2 more than the number before it
Answer : The next number is **10**

Find the missing number :



Answer in next page



27 = 9 x (2+1)
35 = 7 x (3+2)
36 = 4 x (4 +5)

Find the missing number :

6	**18**	**42**
5	**15**	**?**
7	**36**	**71**

Options

a) 30
b) 25
c) 53
d) 45

Answer in next page

6	18	42
5	15	30
7	36	71

$6^2 + 18 - 12 = 42$

$7^2 + 36 - 14 = 71$

$5^2 + 15 - 10 = 30$

What pattern connects all the numbers in this grid ?

2	3	10	12	13
20	21	22	23	24
25	26	27	28	29
30	31	32	33	34
35	36	37	38	39
?	?	?	?	?

Answer in next page

2	3	10	12	13
20	21	22	23	24
25	26	27	28	29
30	31	32	33	34
35	36	37	38	39
200	201	202	203	204

The pattern is numbers whose English name starts with T

If CAB is coded as 6 and BED is coded as 40, then how will HAD be coded as ?

Answer in next page

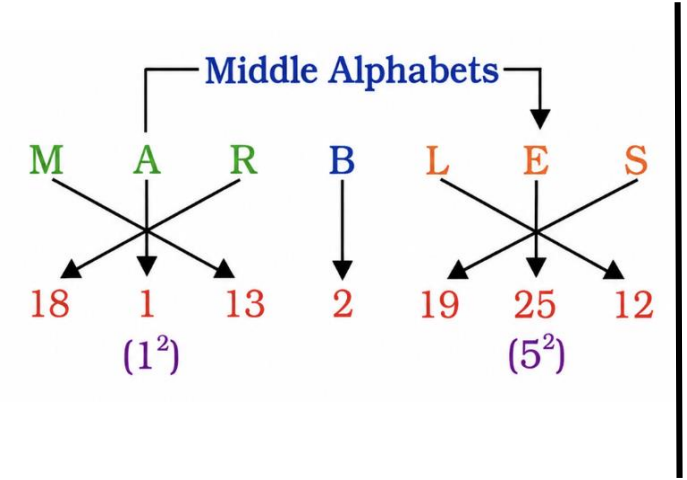
If CAB is coded as 6 and BED is coded as 40, then how will HAD be coded as ?

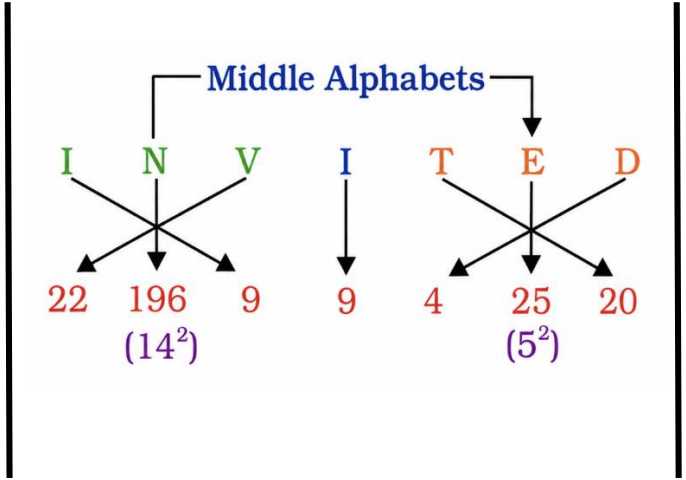
CAB = (3x1x2 = 6)
BED = (2x5x4 = 40)
HAD = (8x1x4 = 32)

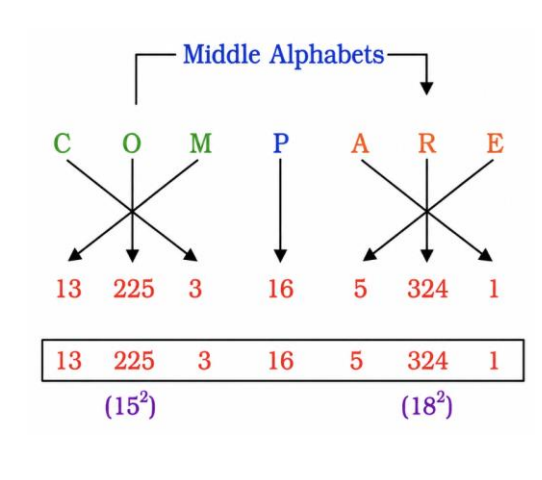
If MARBLES is coded as 181132192512 and
INVITED is coded as 221969942520,
then how will COMPARE be coded as ?

Answer in next page

If MARBLES is coded as 181132192512 and
INVITED is coded as 221969942520,
then how will COMPARE be coded as ?







Ans : 1322531653241

DATA REPRESENTATION

Translating real-world objects, concepts, and problems into a structured, digital format that a computer can process.

- Data representation is about how information - numbers, letters, pictures, sound, video - is converted into a form a computer can store and work with.
- At the lowest level, a computer only understands two states, usually written as 0 and 1

Binary

- Binary is the basic way computers represent and process data.
- It uses only two digits: **0 and 1**.
- All information in a computer is represented using combinations of these two digits.

ASCII

- ASCII (American Standard Code for Information Interchange) is a system used to represent characters such as letters, numbers, and symbols using numbers.
- Standard ASCII uses 7 bits and represents 128 characters.

Unicode

- Unicode is a universal character-encoding standard that represents characters from almost all languages, along with symbols and emojis.
- Unlike ASCII, Unicode can represent a very large number of different characters.

BINARY

The decimal number 5 represented in binary is:

101_2

4's place	2's place	1's place
1	0	1

$(1 \times 4) + (0 \times 2) + (1 \times 1) = 5$

So, 5 = 101_2

ASCII

The number 5 in ASCII is:

53 (Decimal)

Binary (7-bit) representation:

0	1	1	0	1	0	1

Representation	Value
Character	5
ASCII (Decimal)	53
ASCII (Binary 7-bit)	0110101

UNICODE

The number 5 in Unicode is:

U+0035 (Hexadecimal)

Binary (16-bit) representation:

0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Representation	Value
Character	5
Unicode (Hex)	U+0035
Unicode (Binary 16-bit)	0000 0000 0011 0101

Hands on Activity

Instructions

Arrange the cards from **64 to 1**. Turn cards **face up = 1** and **face down = 0**. Add the dots on the visible cards to make numbers from **1 to 127**. Observe the pattern as the cards flip while counting.

Test, Image, Video & Audio Formats

A file format is an agreed set of rules for how data is arranged inside a file, so any program following the same rules can open and understand it correctly.

TEXT	IMAGE	AUDIO	VIDEO
.txt .docx .csv	.jpg .png .gif	.mp3 .wav .aac	.mp4 .mov .avi
Stored as a sequence of character codes (ASCII / Unicode), one after another.	Stored as a grid of pixels; each pixel is a set of numbers for its colour (e.g. Red, Green, Blue).	Sound wave is measured (sampled) thousands of times per second and each sample is stored as a number.	Many still images (frames) shown one after another, usually with an audio track stored alongside.

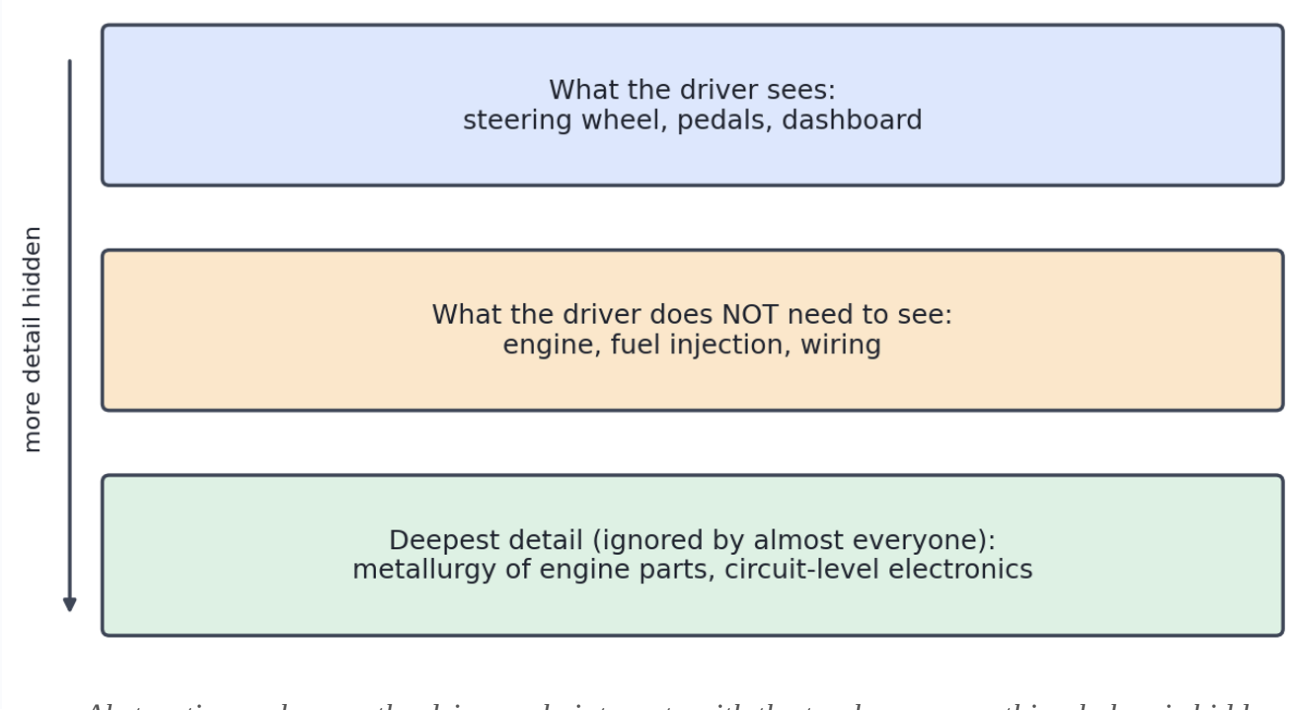
ABSTRACTION

Hiding the detail that doesn't matter for the task at hand, and keeping only what is essential to understand or use something.

- Abstraction is the process of filtering out – ignoring - the characteristics of patterns that we don't need in order to concentrate on those that we do.
- It is also the filtering out of specific details.
- 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 detail, 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’.

Where it is needed ?

- Any dashboard or control panel
- Maps standing in for real places
- APIs/remotes that hide internal complexity



Abstraction as layers: the driver only interacts with the top layer, everything below is hidden.

TASK

Can you add the statements below into the correct table columns.
Some of them are General Statements, others are Specific Details.

A cake needs ingredients.

A Victoria sponge cake has a layer of strawberry jam in the middle.

Cake needs a specified time to bake.

It takes 1 hour 45 minutes to cook a Christmas cake.

Know the specific ingredient quantities.

Banana cake uses 600g of sliced bananas.

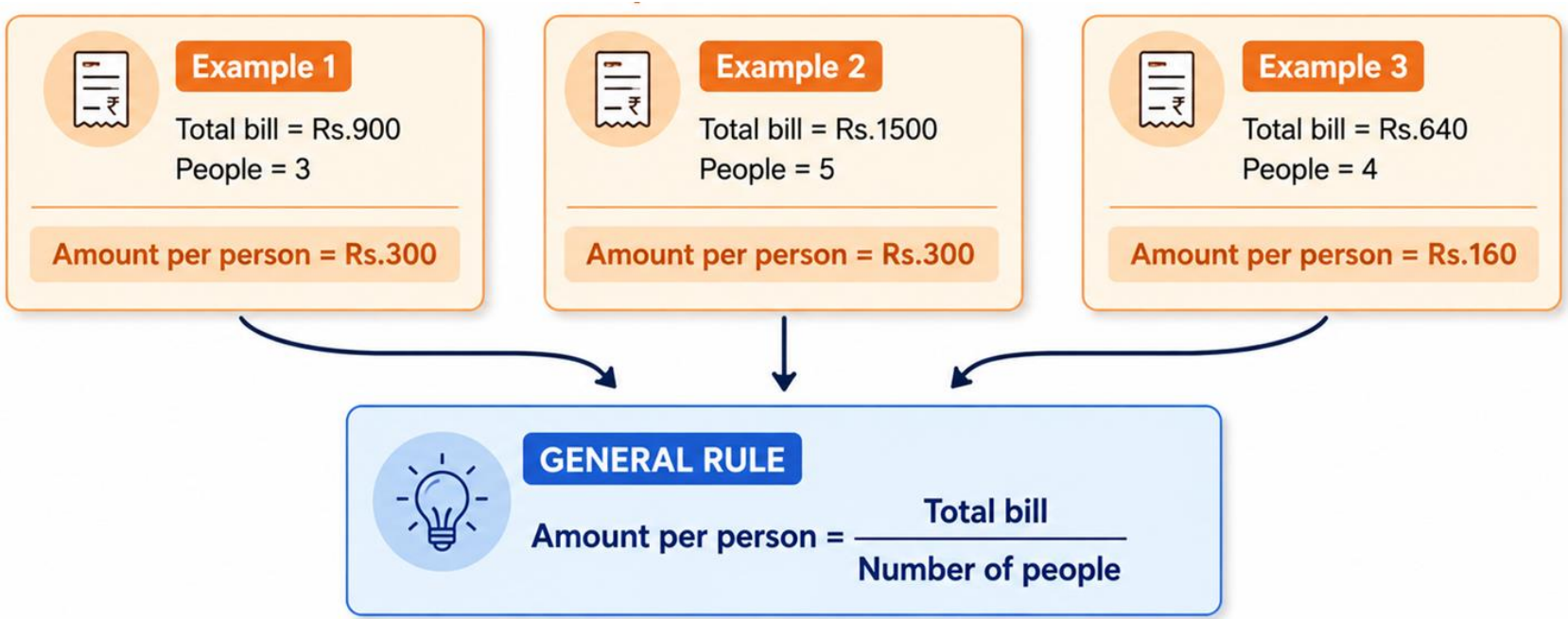
General Statements	Specific Details

GENERALISATION

Extending one specific solved solution into a rule that also works for a whole family of similar problems, not just the one case you started with.

Where it is needed ?

- A formula that works for any input, not one number
- A study method that works for any subject, not just one exam



one rule that now works for any bill, any group size.

ALGORITHM

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

- In an algorithm, each instruction is identified and the order in which they should be carried out is planned.
- Algorithms are often used as a starting point for creating a computer program, and they are sometimes written as a **flowchart** or in **pseudocode**.

- You use code to tell a computer what to do.
- Before you write code you need an **algorithm**.
- An **algorithm** is a list of rules to follow in order to solve a problem.
- **Algorithms** need to have their steps in the right order.

- When you write an algorithm you need to include precise, step-by-step instructions.
- **Task:** Can you put the following instructions in order for making a smoothie:

Imagine if we missed out one of the steps or reversed the order. We could end up switching on the blender with nothing in it. Or there could just be milk in the blender and no fruit.
That wouldn't make a very tasty smoothie, would it? In fact, it wouldn't make a smoothie at all!



People use lots of algorithms every day even if they don't realise it.
Choose from three problems and guess what the algorithm might be!

We need to make a cake!



The algorithm here is a cake recipe. You can find the algorithm to solve this problem in a cookbook!

We can't find the park!



The algorithm you need is a set of directions to get to the park! There might be different ways to the park so you can have different algorithms.

We need to build a toy!



The algorithm to follow here is the list of instructions that tell you how to make the toy.

Representing an algorithm : Pseudocode

- **Pseudocode is not a programming language**, it is a simple way of describing a set of instructions that does not have to use specific **syntax**.
- Writing in pseudocode is similar to writing in a programming language.
- Each step of the algorithm is written on a line of its own in sequence. Usually, **instructions** are written in **uppercase**, variables in **lowercase** and **messages** in **sentence case**.
- In pseudocode, **INPUT** asks a question. **OUTPUT** prints a message on screen.

- A simple program could be created to ask someone their name and age, and to make a comment based on these. This program represented in pseudocode would look like this:

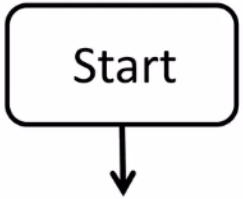
```
OUTPUT 'What is your name?'
INPUT user inputs their name
STORE the user's input in the name variable
OUTPUT 'Hello' + name
OUTPUT 'How old are you?'
INPUT user inputs their age
STORE the user's input in the age variable
IF age >= 70 THEN
    OUTPUT 'You are aged to perfection!'
ELSE
    OUTPUT 'You are a spring chicken!'
```

Representing an algorithm : Flow chart

- A flowchart is a diagram that represents a set of instructions.
- Flowcharts normally use standard symbols to represent the different instructions.
- There are few real rules about the level of detail needed in a flowchart.
- Sometimes flowcharts are broken down into many steps to provide a lot of detail about exactly what is happening.
- Sometimes they are simplified so that a number of steps occur in just one step.

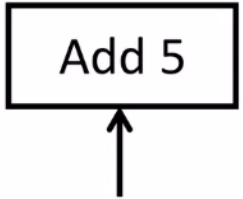
Representing an algorithm : Flow chart

Start / End



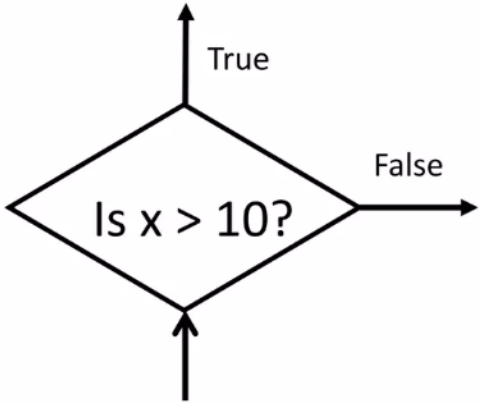
The boxes at the **start/end** are written using 'rounded rectangles'. A **start** symbol represents the start of a process and produces exactly one output. An **end** symbol represents the end of a process and has exactly one input.

Process



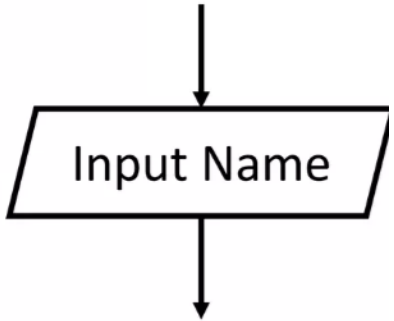
Process are written in a rectangular box. It represents an operation (or task) to be carried out.

Decision



When the algorithm has to make a **decision**, it is written in a rhombus. It has one input and two outputs.

Input / Output



Parallelograms show inputs and outputs

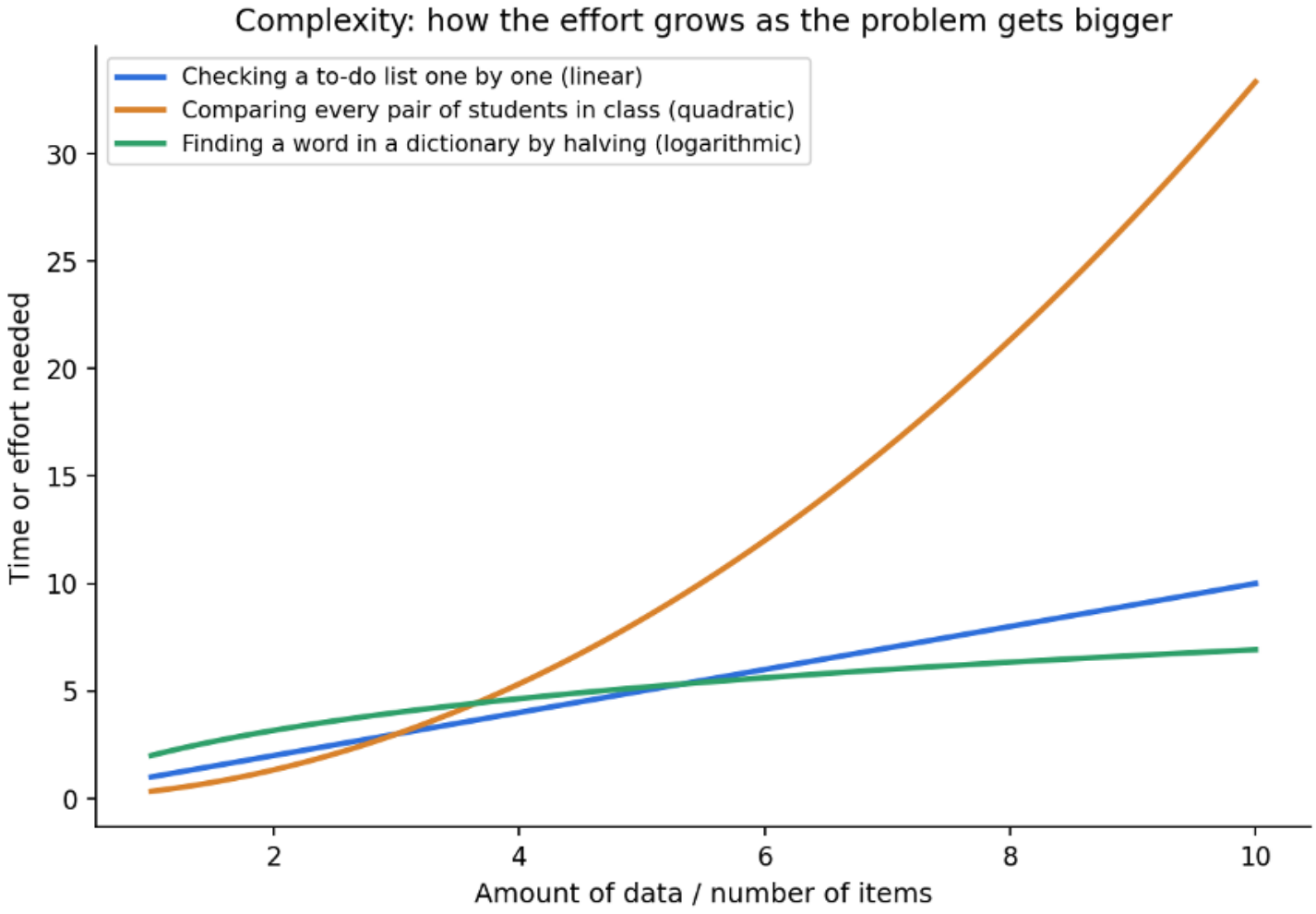
COMPLEXITY

Complexity is about understanding how difficult a problem is, and how much time or effort it takes to solve, especially as the problem gets bigger.

- A problem can be simple to state but hard to solve.
- As the size of the input grows, the work needed to solve it can grow slowly, or very fast.
- Understanding complexity helps you judge whether a solution will still work when the problem gets bigger.

Where is it needed ?

- Deciding if an app will still be fast with 1 million users
- Comparing two ways of organising a contact list
- Planning how long a task will realistically take to finish


Complexity: how the effort grows as the problem gets bigger
Checking a to-do list one by one (linear)
Comparing every pair of students in class (quadratic)
Finding a word in a dictionary by halving (logarithmic)
Time or effort needed
0
5
10
15
20
25
30
2
4
6
8
10
Amount of data / number of items

MODELING

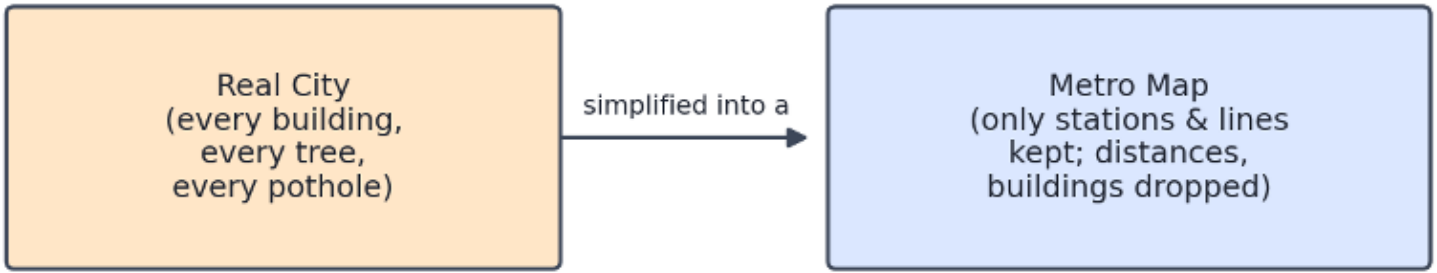
A model is a simplified version of something real, built by keeping only the parts needed to understand it or predict how it behaves.

- A model leaves out detail that does not affect the answer you need.
- The same real situation can have different models, depending on what question you are trying to answer.
- A model is judged by whether it is useful, not by whether it is perfectly accurate.

Where is it needed ?

- Weather forecasts, built from a model of the atmosphere
- A toy globe, a model of the Earth's shape and geography
- A budget spreadsheet, a model of your future spending

Modeling: building a simplified stand-in for something real,
keeping only what is useful for a purpose



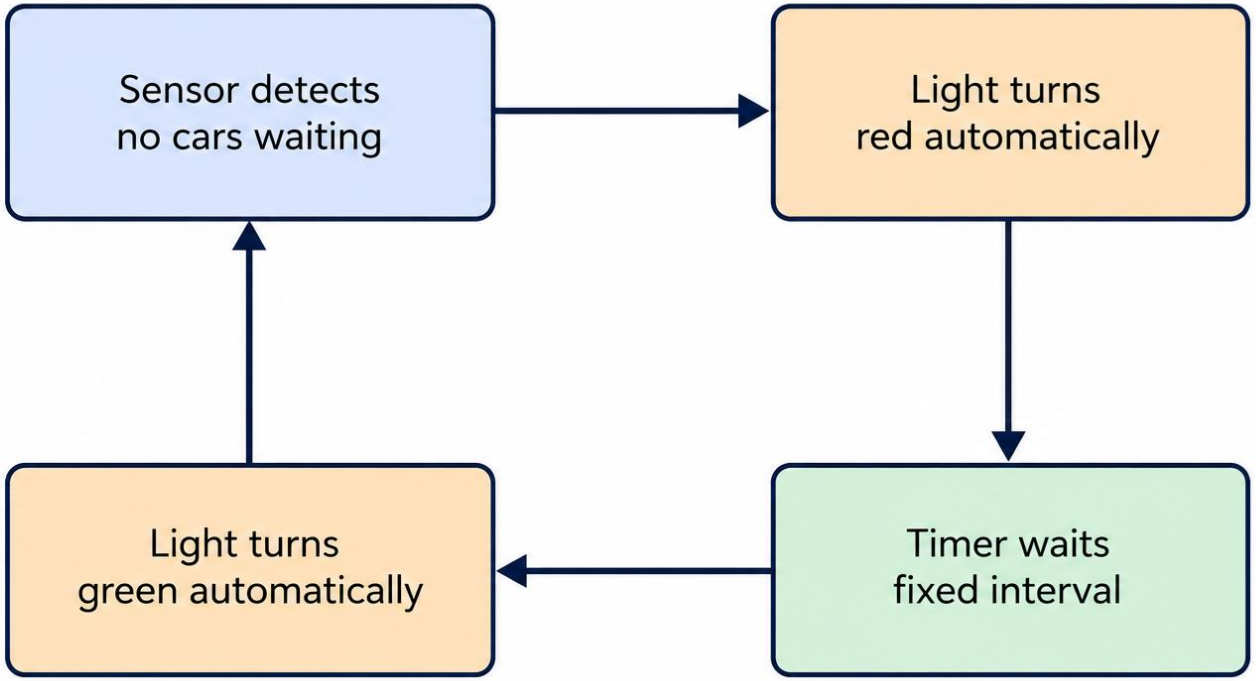
AUTOMATION

Automation means getting a machine or a program to carry out a set of steps by itself, without a person repeating them by hand each time.

- Automation works best on tasks that are repetitive and follow a fixed set of rules.
- Once an algorithm is written correctly, a computer can repeat it exactly, every single time.
- It removes the boring, repeated part of a task, freeing people for work that needs judgement.

Where is it needed ?

- A washing machine running its wash cycle unattended
- Auto-reply emails sent outside office hours
- A spreadsheet formula that updates a total by itself



Automation: a process that repeats and runs itself,
without a person deciding each step

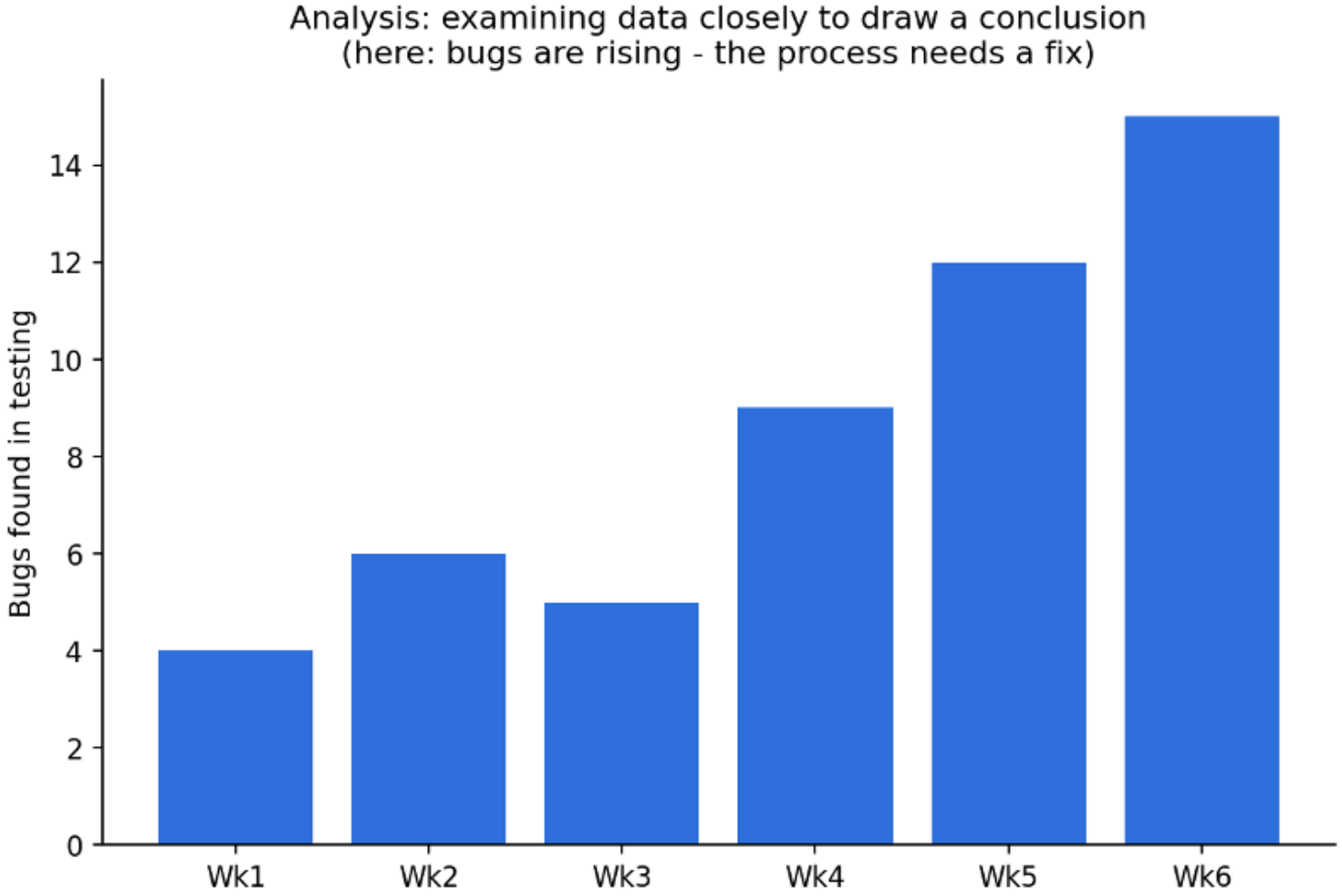
ANALYSIS

Analysis means examining data or a solution closely, breaking it down and comparing its parts, in order to understand it or judge how well it works.

- Analysis often comes after collecting data or trying out a solution.
- It involves looking for patterns, spotting problems, and comparing options against each other.
- Good analysis turns raw information into a decision you can actually act on.

Where is it needed ?

- Checking exam scores to find which topic students struggled with
- Comparing two phones on price, camera and battery before buying
- Reviewing a week's sales to spot the best and worst selling day



Analysing weekly test-bug data reveals a rising trend that needs attention

VISUALISATION

Visualization means presenting information as pictures, charts or diagrams, so patterns and relationships can be seen at a glance instead of read line by line.

- The human eye spots patterns in shapes and colours much faster than in rows of numbers.
- The right chart depends on the question, trends over time, parts of a whole, or comparisons.
- A misleading chart can distort the data, so choosing an honest, clear format matters.

Where is it needed ?

- A line graph showing a patient's temperature over a week
- A pie chart showing how a monthly budget is split
- A map coloured by state to show election results

Visualization: turning raw numbers into a picture that is easy to read at a glance

