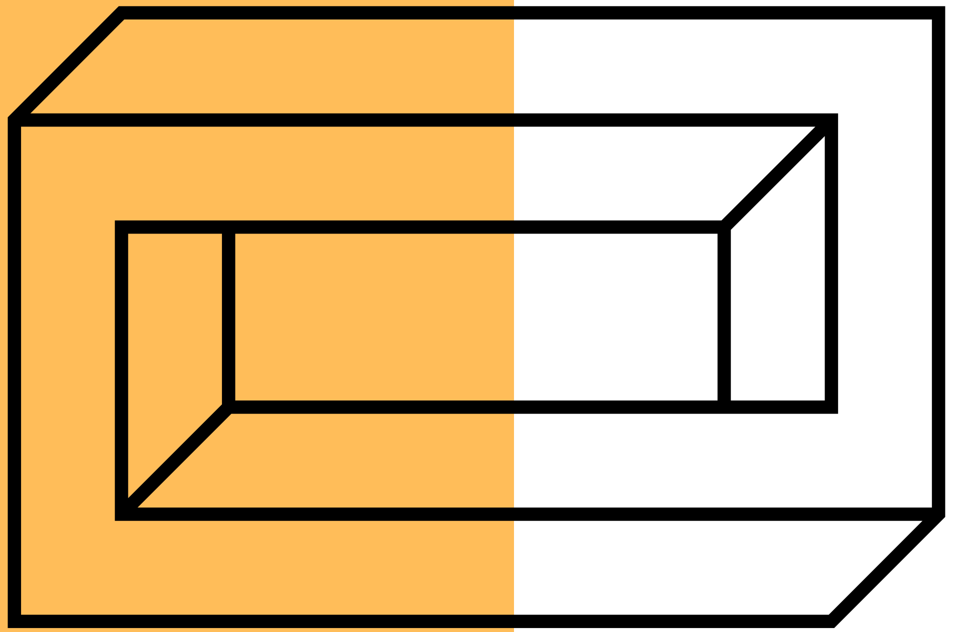
Sri Lanka Institute of Information Technology
August 2026

Introduction to Programming

Amila Alexander

What is Programming?



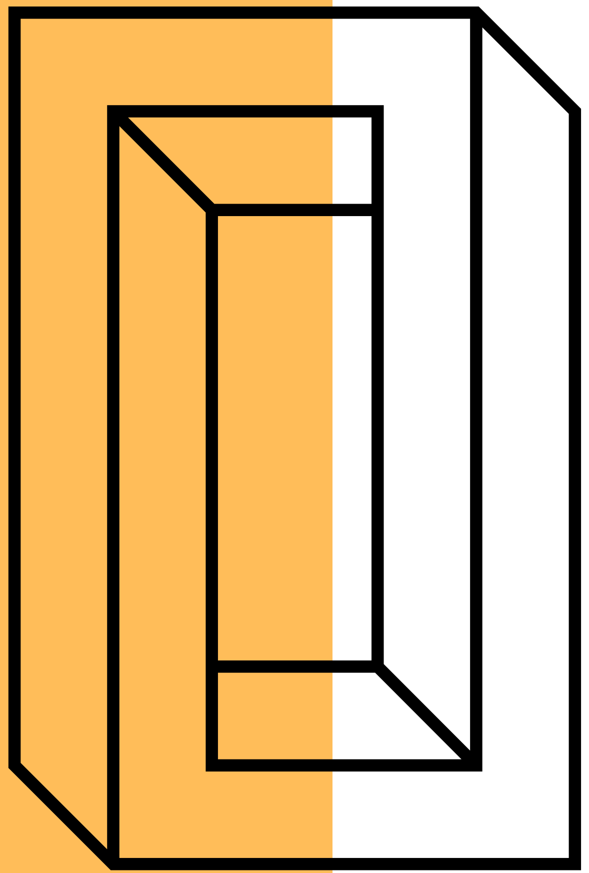
In simple terms, programming the process of giving a computer a set of instructions to follow

But, unlike how we give instructions to a human in English or Sinhala.

To speak to a computer, we must use a different kind of language.

This language is known as CODE

But, what is an instruction?



Go wash the clothes? 🧼

Clean the house? 🧹

Just like these are instructions for a person, a **line of code** is an instruction for a computer. It tells the computer to perform a specific task step by step, just like a to-do list!

Examples of Instructions we might give to a Computer

1 **Do this calculation**

→ e.g., *Add 5 to the user's score.*

2 **Store this data**

→ e.g., *Save the user's name in a database*

3 **Make a decision**

→ e.g., *If the user clicks a button, show a message.*

4 **Interact with the world**

→ e.g., *Send an email, fetch weather data, or control a robot arm.*

Where did Programming Start?

The world's first programmer is considered as Ada Lovelace

In the start programming was done using wires, plugs and switches. The concept of "software" wasn't invented yet. The Code was -hardware

Nowadays, computers speak in the language of electricity
1s and 0s (Binary)

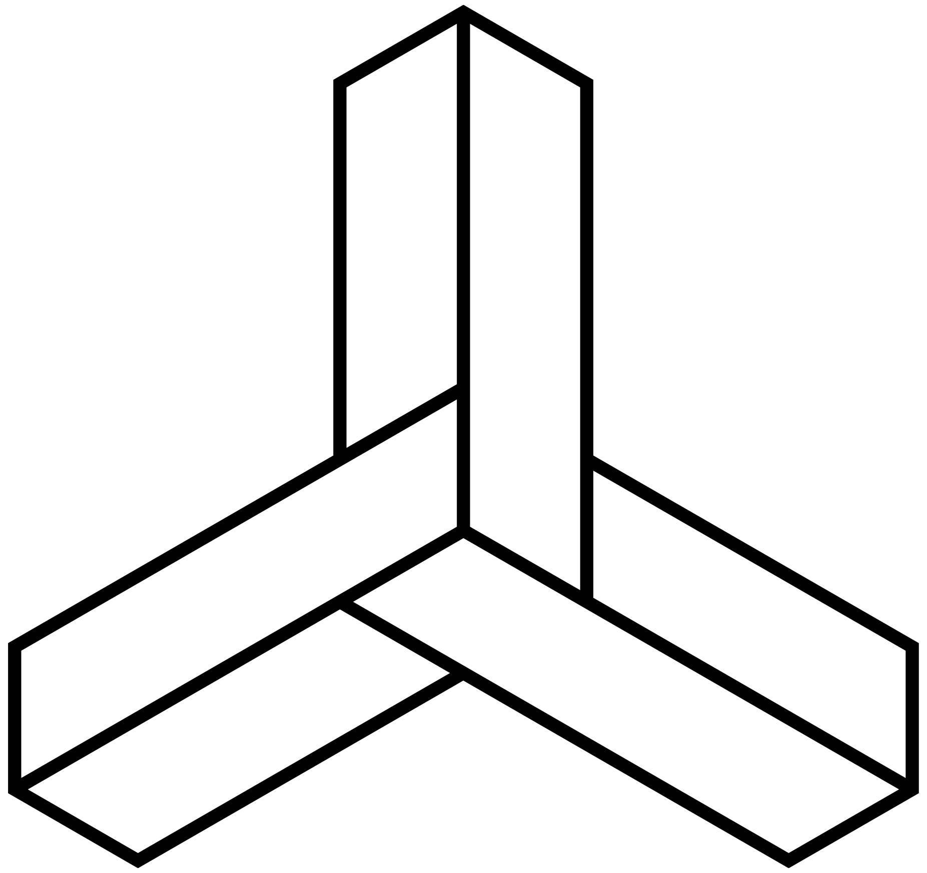
Machine Language

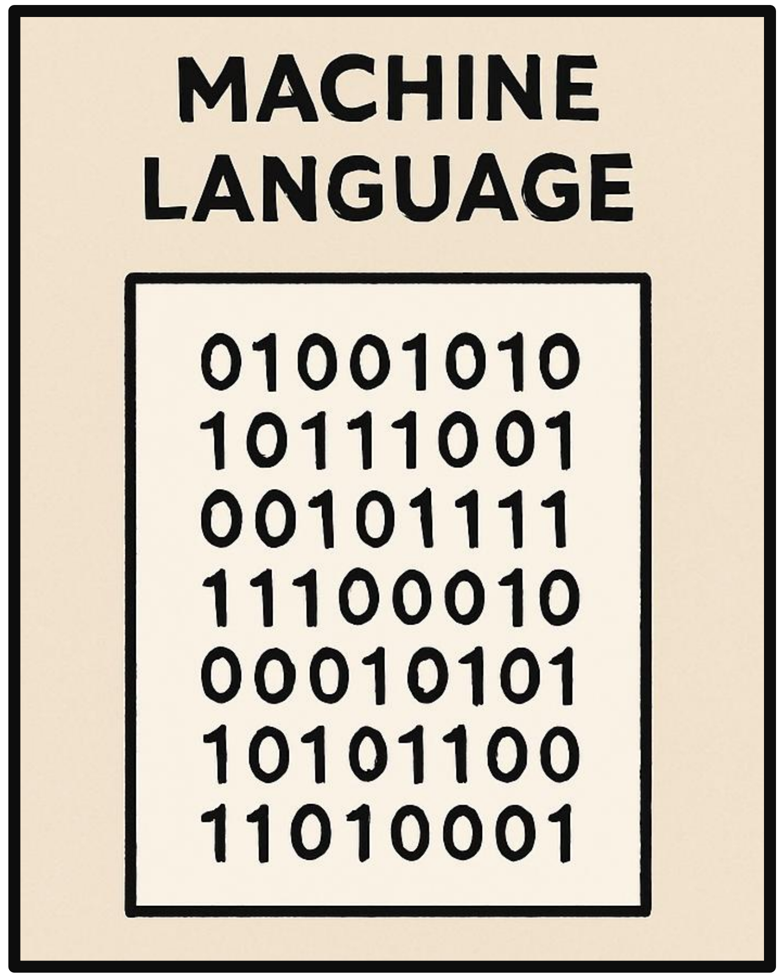
Every computer works from 101010s

Whether it's a PlayStation 5, calculator or even an A/C

Low Level Language

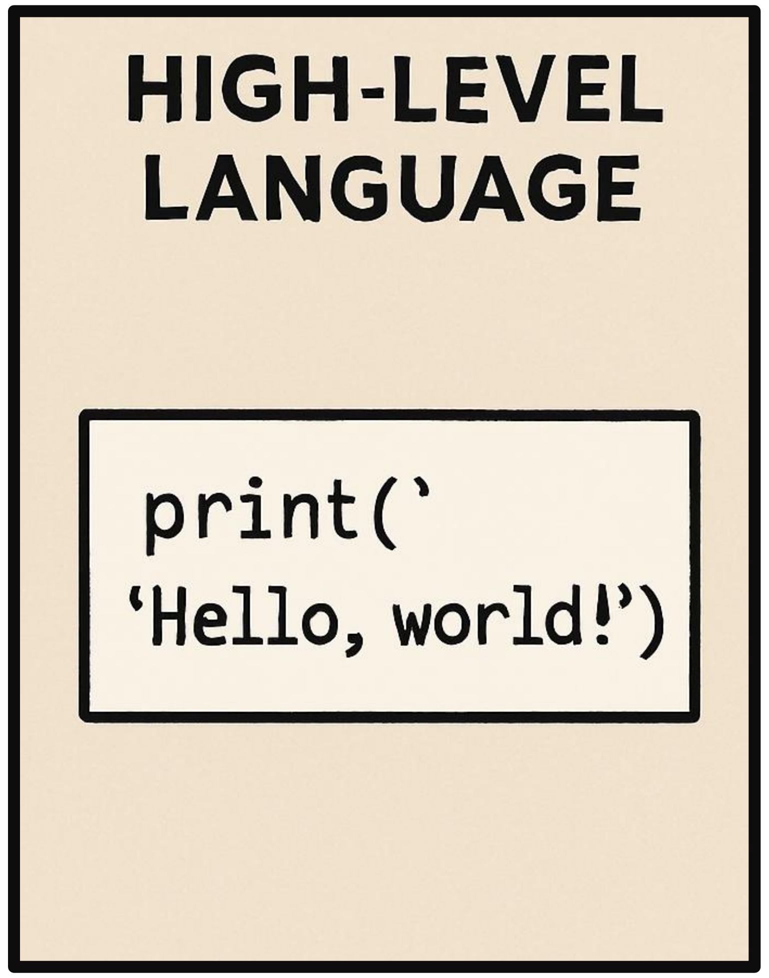
English words exist but still very difficult
Eg: Assembly Language





High Level

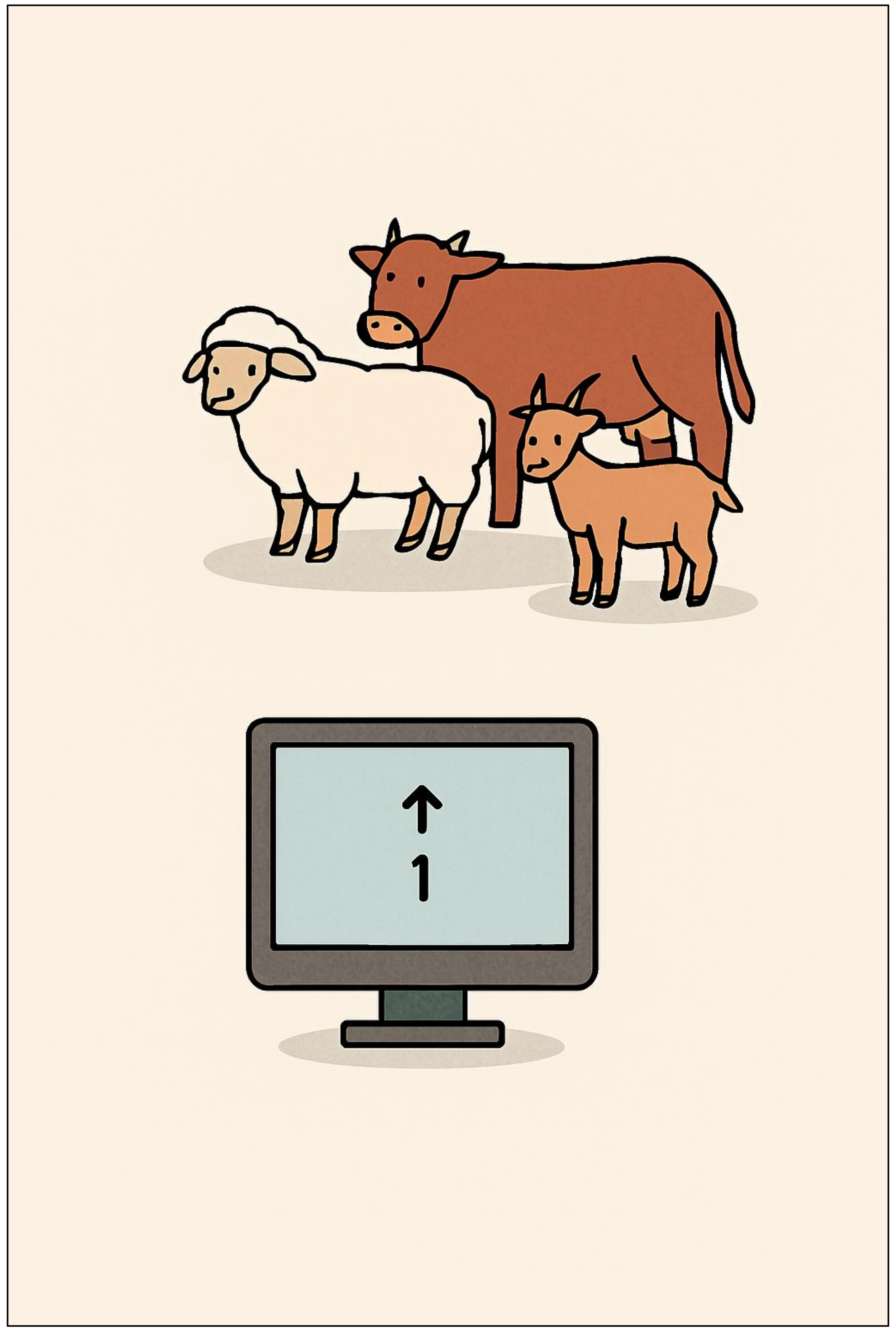
Easy to write, understand and learn
Eg: Python, Java etc



- Machine language is the native language of computers—binary instructions they understand directly.
- To bridge the gap, compilers translate the English-like code we write into machine language.
- For example, we might say: “Count the number of cows, sheep, and goats in the herd.”

- We count them by eye.

The computer uses a loop and a counter variable, incrementing it for each animal detected.



Core Programming Concepts

In summary,

These are some of the foundational building blocks of any code whether it's written for a simple calculator or a NASA rocket.

Show example of a code

MAKING DECISIONS

if conditions

LOOPS

Doing things multiple times

ARITHMETIC

Math & Logic

DATA & VARIABLES

Keeping track of things

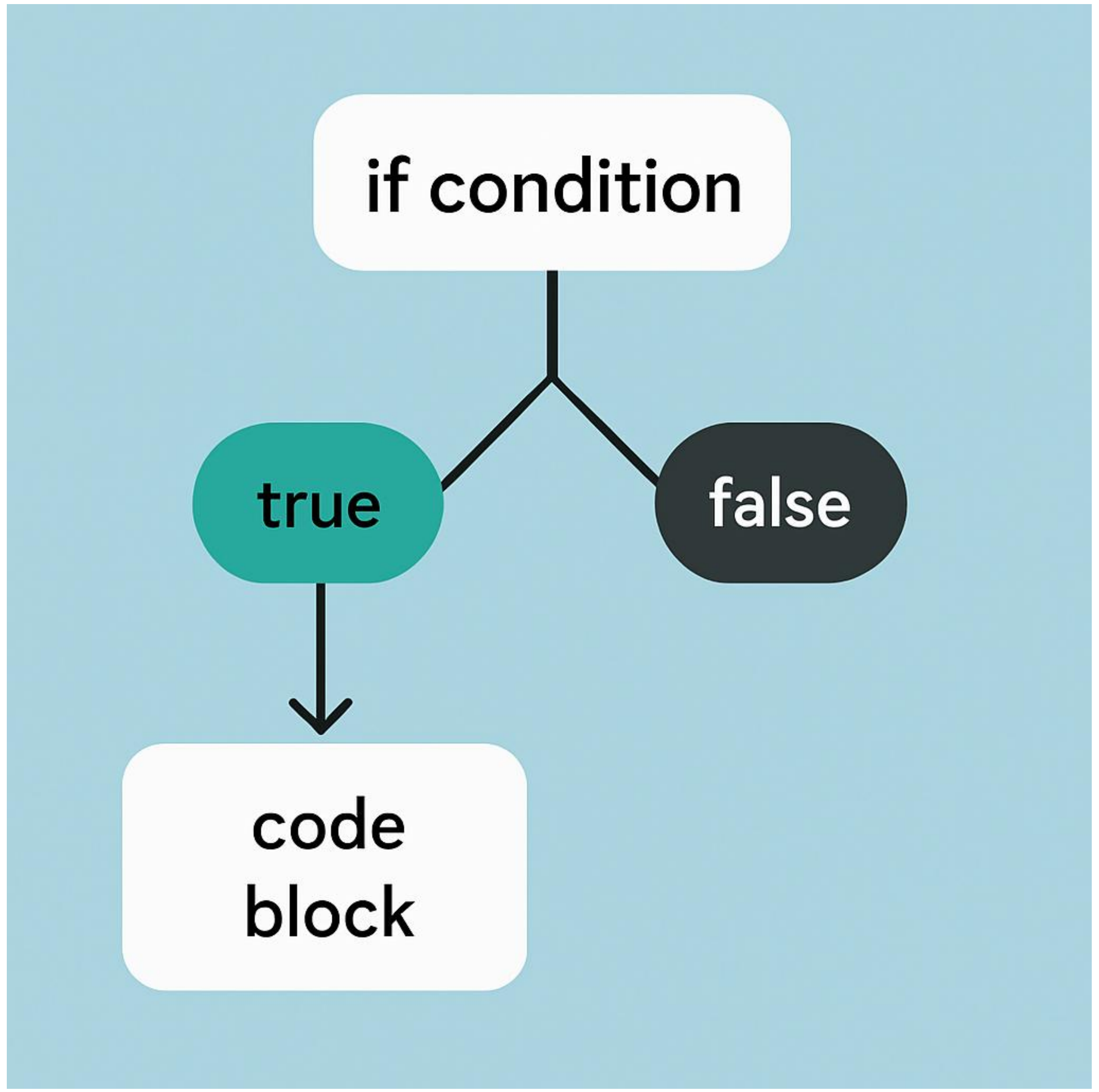
The humble “`if condition`”

An if condition has the ability of making a decision

If a statement is TRUE or **Truthy** the accompanied block runs

If the statement is FALSE or **Falsey**, the block does not run

Decision Maker



The Repeater “loops”

Loops are way of running a task that is repeated multiple times

Example if we want to create a program to find the average mark of each student for English

Instead of writing the “find average” of code for each student

It can be run in a loop

- How many times do you want it to repeat? 10 times? 20?
- Infinitely?

- Stop condition
- Increment Operation

Arithmetic & Logic

Basic Arithmetic

- **Addition (+)**: Combining values
- **Subtraction (−)**: Finding differences
- **Multiplication (×)**: Scaling values
- **Division (÷)**: Splitting values
- **Modulo (%)**: Finding remainders

Relational Operators

- Greater than (>)
- Less than (<)
- Equal (==)

Logic

- **AND (&&)**: True if both conditions are true
- **OR (||)**: True if at least one condition is true
- **NOT (!)**: Reverses the truth value

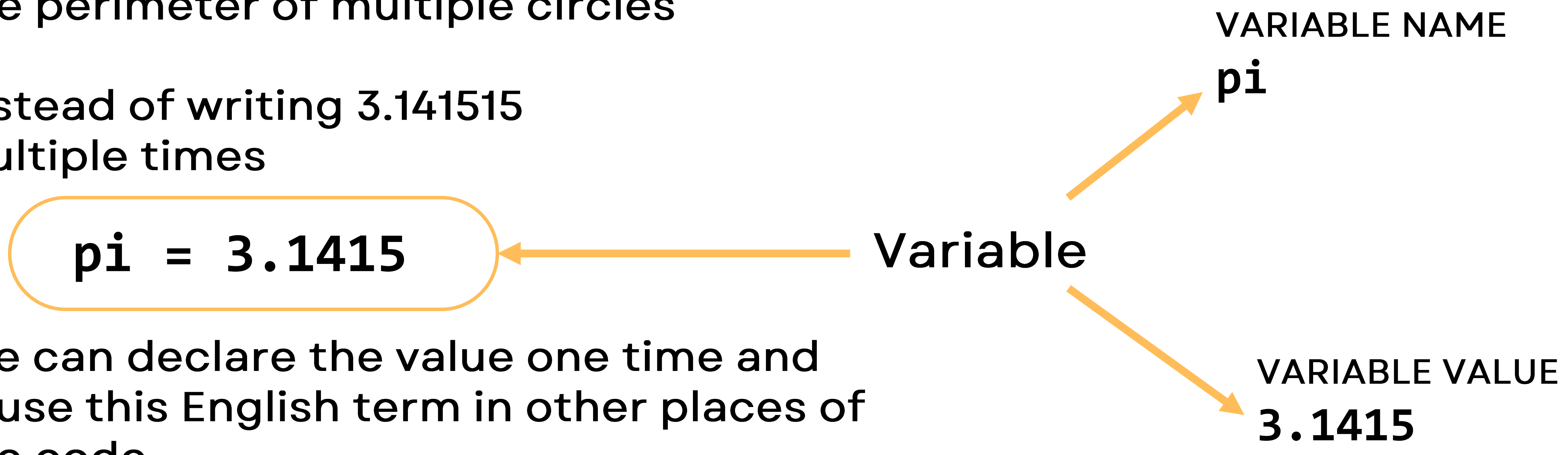
Variables and Data

Variables are simply names we assign to represent information

Example :

Imagine you’re creating a program to find the perimeter of multiple circles

Instead of writing 3.141515
Multiple times



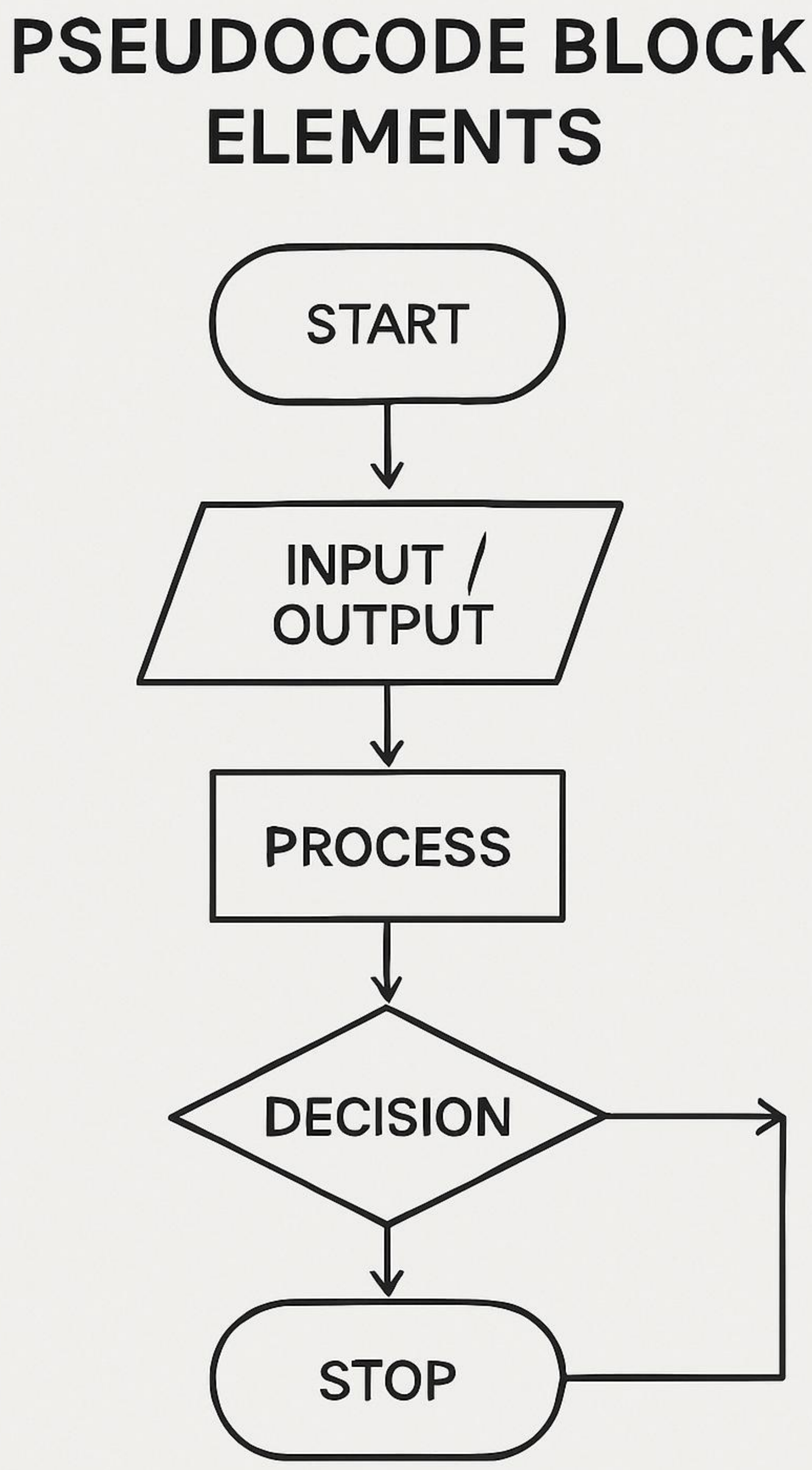
```
pi = 3.1415
```

We can declare the value one time and reuse this English term in other places of the code

Activity

Try to design the flow of an ATM machine

ATM



Make use of the knowledge we learned

- Checking operations (if conditions)
- Loops
- Arithmetic and Logic

You can use the pseudo code block elements shown on the left to draw the flow (if you like)

Try to identify the core concepts we used earlier like loops, conditions etc

EXAMPLE : Flow of the ATM machine

- User enters their card (check for valid card)
- Ask user for pin
- Perform the checking operations (verify the pin and make sure it's correct)
- **If** the pin is **CORRECT,** then allow the user to see their acc details
- **If** the pin is **INCORRECT** ask the user to renter, the pin
- If the pin is wrong more than 3 times, restrict the card usage and take the card in

- In the correct PIN is given, the user is asked what service they would like
- Do you wish to withdraw money? Do you wish to deposit money? Do you wisht o view you Acc. balance? Do you wish to obtain a printed statement?

Example : withdrawing Money

- Ask the user to enter the amount they require
- Perform the transaction
- Check if the user has sufficient account balance in their account.
- Subtract the amount from their account.

Summary of Concepts

ATM

💡 Concept	📝 Description	🧪 Examples Used in ATM Workflow
📦 **Variables**	Stores user information that might be reused later in the code	`pin, balance, amount, choice, attempts, sessionActive`
🔤 **Data Types**	Defines the type of data to ensure data is handled properly	`int, float, bool, string`
🔀 **Conditionals**	Checks user information like the PIN	`if, elif, else blocks for PIN check and transaction logic`
🔁 **Loops**	Repeats a block of code, like asking the user for the correct pin over and over again	`while sessionActive: keeps the session running`
⚖️ **Comparison**	Check if the account balance is sufficient etc	`inputPin == pin, amount <= balance, attempts >= 3`
➕ **Arithmetic Ops**	Performs mathematical calculations like subtracting the amount from account	`balance -= amount, balance += amount, attempts += 1`
⚪ **Boolean Logic**	Tracks if the ATM session is valid, if not return to idle screen	`sessionActive = True/False`

What about hardware?

How is your software going to interact with the real world

1

So far, each step discussed was about SOFTWARE

2

However, to deposit money a motor needs to spin.

* It needs to spin the exact number of times

* The money notes need to be categorized using vision or sensors

* The motorized door to allow accessing cash needs to opened

* The mechanism to dispense the card needs to be controlled.

→

Your system is a combination of both SOFTWARE and HARDWARE

THANK YOU!