Introduction to Computer Programming

- Week 2

- *Eng. Sylvain Manirakiza -*

Sund - Sept 14th, 2025

Devotion

Be devoted to one another in love. Honor one another above yourselves."
— Romans 12:10 (NIV)

Reflection

As we go deeper into programming, we'll face problems that require collaboration, not competition. This verse reminds us:

- 🤝 Let's build a culture of support — where no question is "too simple" to ask.
- 🌱 Let's respect each other's learning pace — we're not racing, we're growing.
- 🎯 Let's honor each other's efforts, even when the code fails — because every error is part of the journey.

Quiz

Quick recap of Day 1

1. What is Algorithm

An algorithm is a series of instructions, once executed correctly, leads to a given result /Solve the problem

2. What is Programming

A process of creating a set of instructions or commands that a computer can understand and execute through programming Language

Categories of Computer Programming

1. Applications Development
2. System Programming

Programming Language

Programming languages provide a way for humans to communicate with computers, enabling them to carry out tasks efficiently and accurately. *e.g. Python, Java, C++. C#*

Categories of Programming Languages

✔ High-Level, Low level, Scripting Languages, Compiled Languages, Interpreted Languages, Domain-Specific Languages, etc .

Quick recap of Day 1

Programming with
Pseudocodes

A simplified, informal way of describing an algorithm or program's logic using plain language and basic programming-like structures, without following strict syntax rules of any programming language. It's used to plan and explain code in a way that's easy to understand.

Example

Write **pseudocode** to calculate the average of three numbers

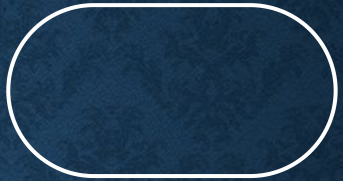
```
Start
1.  Input num1,num2,num3
2.  Sum = num1+num2+num3
3.  Avg = Sum/3
4.  Output Avg
End
```

FLOWCHART

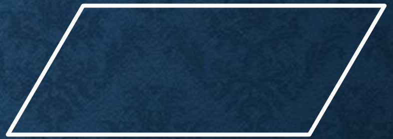
- A flowchart is the graphical or pictorial representation of an algorithm with the help of different symbols, shapes, and arrows to demonstrate a process or a program.
- The main purpose of using a flowchart is to analyze different methods.

Common symbols applied in a flowchart:

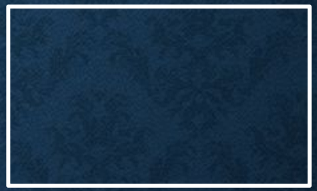
Terminal Box –Start / End



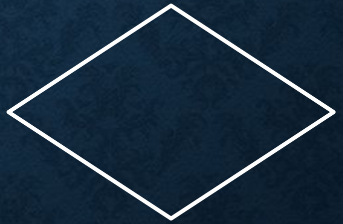
Input / Output



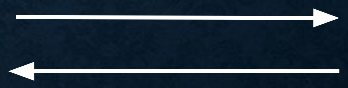
Process / Instruction



Decision



Connector / Arrow



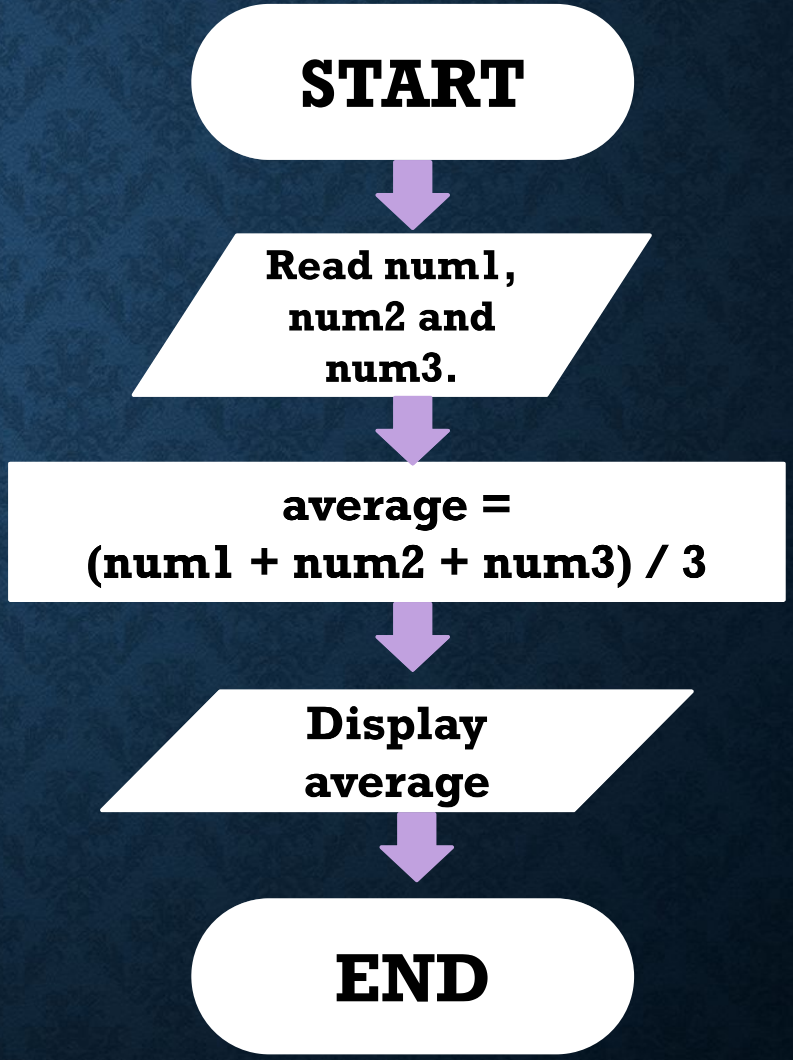
1. Compute the average of three numbers.

ALGORITHM

- Step 1: Start
- Step 2: Declare variables num1, num2, num3 and average.
- Step 3: Read values of num1, num2 and num3.
- Step 4: Find the average using the formula:

  average = (num1 + num2 +num3) / 3
- Step 5: Display average.
- Step 6: End.

FLOWCHART



Chap 2. Variables and Operations

Today’s Objectives

Objective 1

- Define variables and their purpose in programming.

Objective 2

- Explain variable declaration and data types.

Objective 3

- Demonstrate assignment operations and expressions.

Objective 4

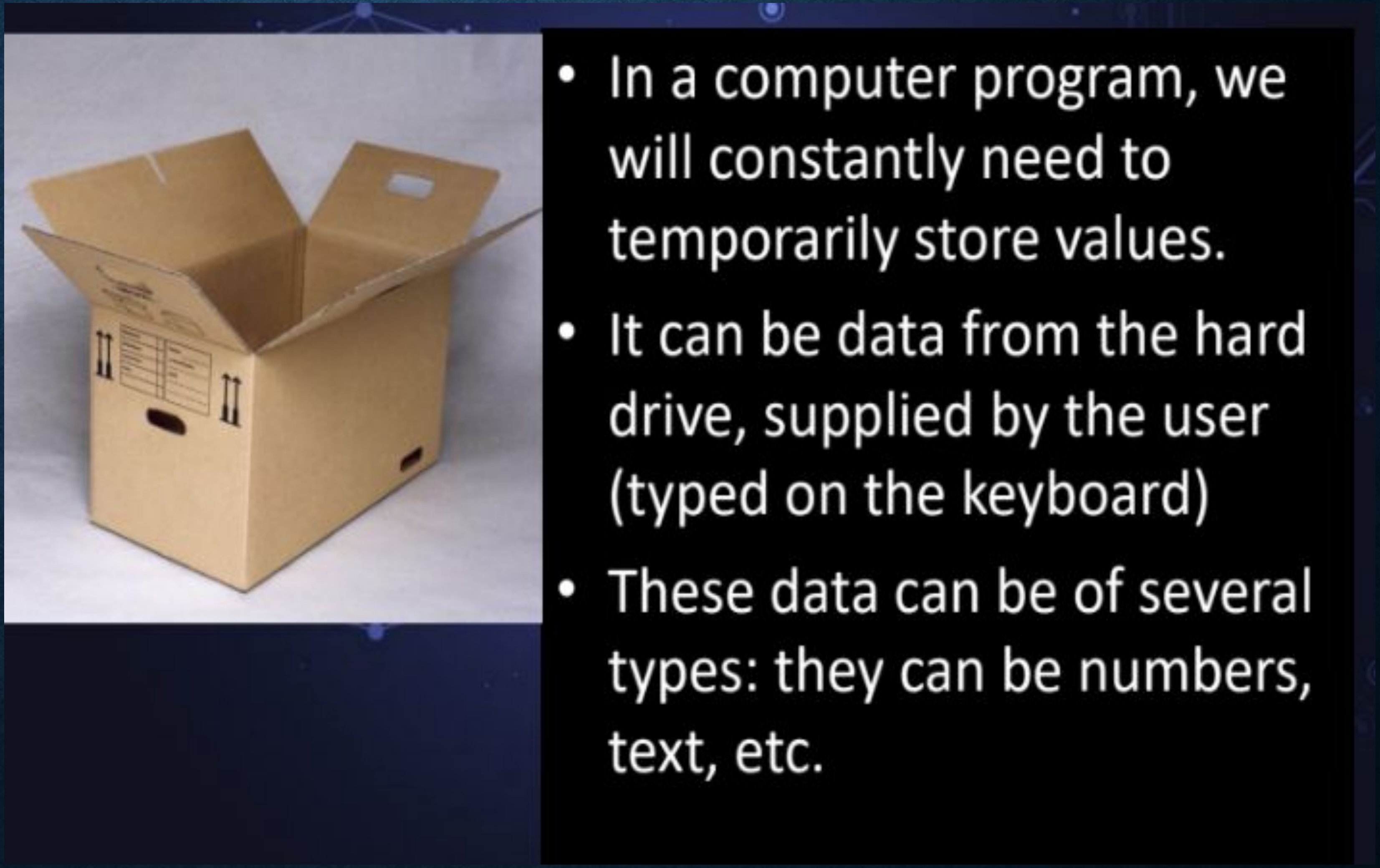
- Solve examples involving variables, operations, and control structures.

What are the variables used for?

A variable is like a labeled box where you store information.

Example

Store user age in a program with a variable named Age.



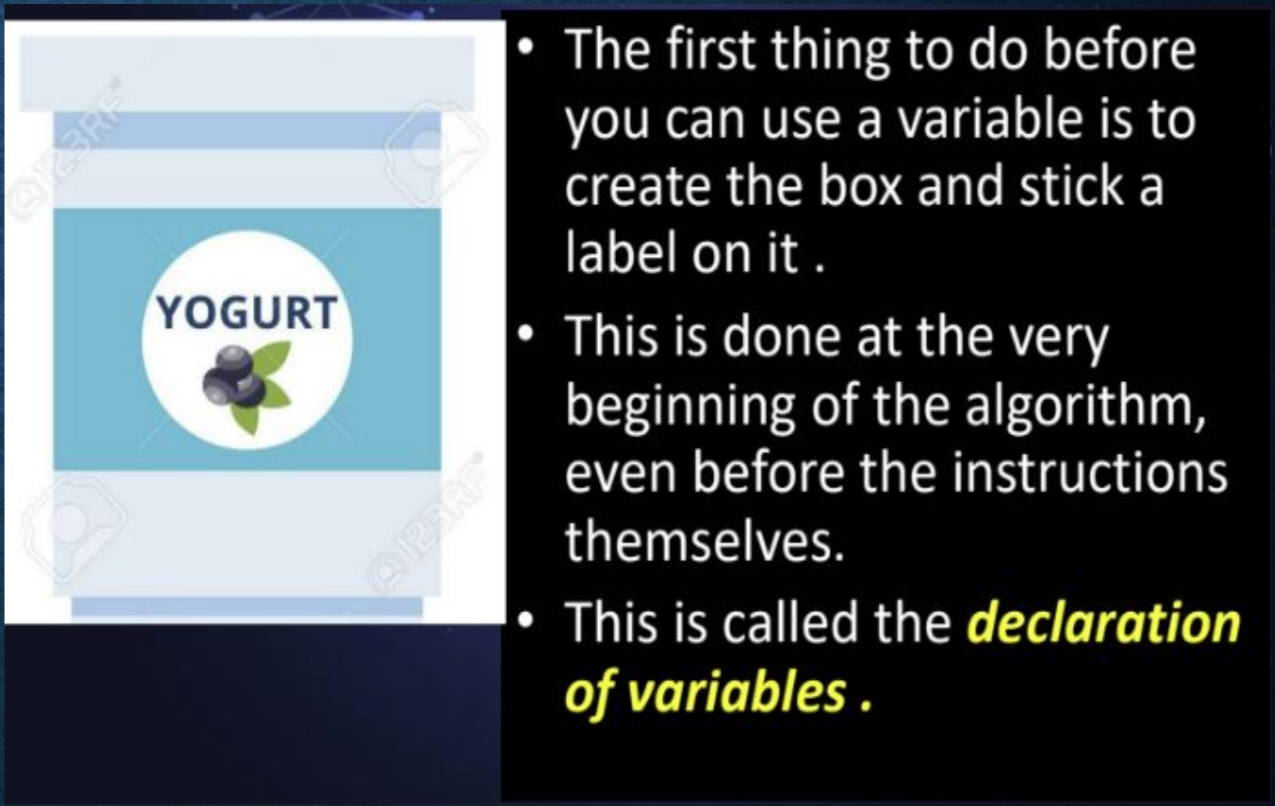
- In a computer program, we will constantly need to temporarily store values.
- It can be data from the hard drive, supplied by the user (typed on the keyboard)
- These data can be of several types: they can be numbers, text, etc.

Importance of Variables

- Why We Need Variables: For storing inputs, intermediate calculations, and outputs.
- Examples:
  - Input: User's name.
  - Calculation: Sum of two numbers.

Declaration of variables

- Syntax: Variable Name in Type
- Rules:
  - Must begin with a letter.
  - No spaces or special characters allowed.
  - Example: Variable A in Numeric



- The first thing to do before you can use a variable is to create the box and stick a label on it .
- This is done at the very beginning of the algorithm, even before the instructions themselves.
- This is called the ***declaration of variables .***

Data Types Overview

1. Numeric: Integers, Floats.
2. Alphanumeric: Strings (e.g., names or messages).
3. Boolean: True/False.
   - Example: Declare a string variable for a user's name: Variable Name in Alphanumeric.

1. Numeric Data Types

- Byte:
  - A data type that can store integers from 0 to 255.
  - Smallest memory footprint (8 bits)
  - *Example: "Variable Age in Byte"*
  - *Use case:*
    - *Efficient for storing small numbers like age or counts*
    - *Small counts (e.g., ages in a kindergarten class).*

1. Numeric Data Types …

- Single: A numeric type capable of storing larger integers, ranging from -32,768 to 32,767.
  - Larger memory (16 bits).
  - Can handle both positive and negative numbers
  - Example: `Variable Score in Single`.
  - Use case: Useful for storing exam scores or inventory counts.

1. Numeric Data Types …

Integers:

- Represents whole numbers (no decimals) within a larger range, often system-dependent.
- Even larger memory (usually 32 or 64 bits, depending on the system).
- Used when the range of numbers might exceed what a Single can handle.
    - Example: Variable Population in Integer

      Population ← 125000.

1. Numeric Data Types …

Float/Real/Double:

- Used for storing decimal values with higher precision.
- Example: `Variable Temperature in Float`.
- Use case: Measuring temperatures, monetary values, or scientific data.

1. Numeric

Why Default to Integer?

- Ease of Use: Integer supports a wide range without worrying about memory constraints in modern systems.
- Standard Practice: Many programming languages (e.g., Python, Java) use int as the default for whole numbers.
- Future-Proofing: If the data grows or exceeds expected limits, Integer prevents overflow errors that could occur with Byte or Single.
-

Comparison of Numeric and Alphanumeric Types

Type	Range	Example	Use Case
Byte	0 to 225	*Ages in years*	Small integers
Single	-32,768 to 32,767	*Number of books in a library*	Moderate integers
Integer	System-dependent (large)	*Population count*	Large whole numbers
Floats/Real	Decimal numbers	*Temperature: 36.5*	Measurements or currency
Strings	Sequence of characters	v *Name: "John Doe"*	Text data
Boolean	True/False	*IsEligible: True*	Conditional logic

Alphanumeric Data Types

- Usage: To store text values, like names, Messages or labels.
- Key Operations: Concatenation (e.g., combining `"Hello"` + `"World"` → `"HelloWorld"`).
- Strings handle both text and sequences of numbers (e.g., storing phone numbers as strings).
- Example:
  - Variable Name in String

Variable Name ← "Alice"

Boolean Data Types

- A simple data type that can hold one of two values: \`True\` or \`False\`
- *Example:*
  - *Variable IsEligible in Boolean*
  - *IsEligible ← True*
- Use case: Storing conditions like whether a user is logged in or if an item is available.

Assignment Operations

- Explanation: Assign values to variables using the assignment operator (←).
- Example:
  - FirstNumber ← 12
  - SecondNumber ← FirstNumber + 5

Modifying Variables

- Example:
  - Increment: Counter ← Counter + 1
  - Update: Total ← Total + Price

Differences Between Algorithms

- Example:
  - FirstCharacter ← "Hello"
  - SecondCharacter ← FirstCharacter
- It Shows how data assignment flows.

Expressions and Arithmetic Operators

- Operators: Addition (+), Subtraction (-), Multiplication (*), Division (/).
- Example:
  - Total ← Price * Quantity

Concatenation with Strings

- Operator: Use & to join strings.
- Example:
  - FullName ← "John" & "Doe" → FullName = "JohnDoe"

Boolean Operators

- Types: AND, OR, NOT.
- Example:
  - IsEligible AND IsMember → True if both are True.

Truth Table Examples

- **AND** Truth Table:
  - True AND True = True.
  - True AND False = False.

Why Truth Tables Are Useful?

- They help visualize and validate the behavior of Boolean expressions.
- They are essential for debugging logical operations in code.
- They are foundational in digital circuits and computational logic.

Truth Tables Application

Determining whether a person is eligible for a discount based on two conditions:

- They must be a loyal customer (LoyalCustomer = True).
- They must have spent at least $100 (SpentOver100 = True).
- If both conditions are met, they get a discount. Otherwise, they do not.

Pseudocode

```
Variables LoyalCustomer, SpentOver100, EligibleForDiscount as Boolean
Start
  If LoyalCustomer AND SpentOver100 Then
    EligibleForDiscount ← True
  Else
    EligibleForDiscount ← False
  End If
  Write "Discount Eligibility: ", EligibleForDiscount
End
```

Variables

Exercises

Input and Output/Read and Write

- Input:Read user-provided data.
- Output: Display calculated or stored results.
- Example:
  - Ask for name, output “Hello, [Name]!”
-

Control Structures

- Definition: Constructs that dictate the flow of program execution.
- Purpose: Control the logic and decisions of a program.
- Example:
  - If Score > 50, then "Pass", else "Fail".

Ternary Operator

- Syntax: Condition ? “Value” : “Value2”
- Example:
  - Result ← (Number % 2 == 0) ? "Even" : "Odd"

Ternary Operator

- A ternary operator is a shorthand way of writing an if-else statement in a single line. It is called a "ternary" operator because it involves three operands:
  - Condition: A Boolean expression (True or False).
  - Value if True: The value or operation to execute if the condition is True.
  - Value if False: The value or operation to execute if the condition is False.

Example in Pseudo-Code

Checking if a number is odd or even.

```
DECLARE Number as Integer
DECLARE Result as String
Start
Write “Enter any number: ”
Read Number
  Result ← (Number % 2 == 0) ? "Even" : "Odd"
  Write "The number is: ", Result
End
```

Example in Pseudo-Code

Checking if a number is odd or even using an "If-Else"

```
Start
   Declare Number as Integer
   Declare Result as String

   Write "Enter a number:"
   Read Number

   If Number % 2 == 0 Then
      Result ← "Even"
   Else
      Result ← "Odd"
   End If

   Write "The number is: ", Result
End
```

2nd Example in Pseudo-Code

Checking if Someone is eligible to vote or not.

```
DECLARE Age as Integer
DECLARE Eligibility as String
Start
  Write “Enter your Age: ”
  Read Age
  Eligibility ← (Age>0) ? "Eligible to vote" : "Not eligible to vote"
  Write "You are: ", Eligibility
End
```

Recap and Summary

- Key Concepts:
  - Variables, data types, assignment operations, and control structures.

Group Assignment

Click here to access it