Python&Math Initiative Project

(PyMath Project)

WP2 - ALGORITHMS AND MATHEMATICS

Day 1, Session 2: Flowchart Basics & Algorithmic Operators

Prof. Dr. Turgay Tugay BİLGİN

Bursa Technical University

10.11.2025

Contents

CONTENTS

01

Introduction & Review

02

Session Objectives

03

Flowchart Symbols

04

Operators

05

Decision Symbol

06

Even/Odd Example

Contents

CONTENTS

01. Piecewise Func. Example

02. Summary & Next Steps

01

Introduction & Review

Review of Session 1: What We Learned

Algorithm

A clear, sequential, and finite set of steps to solve a problem.

5 Characteristics

Input, Process, Output, Finiteness, Definiteness.

Flowgorithm

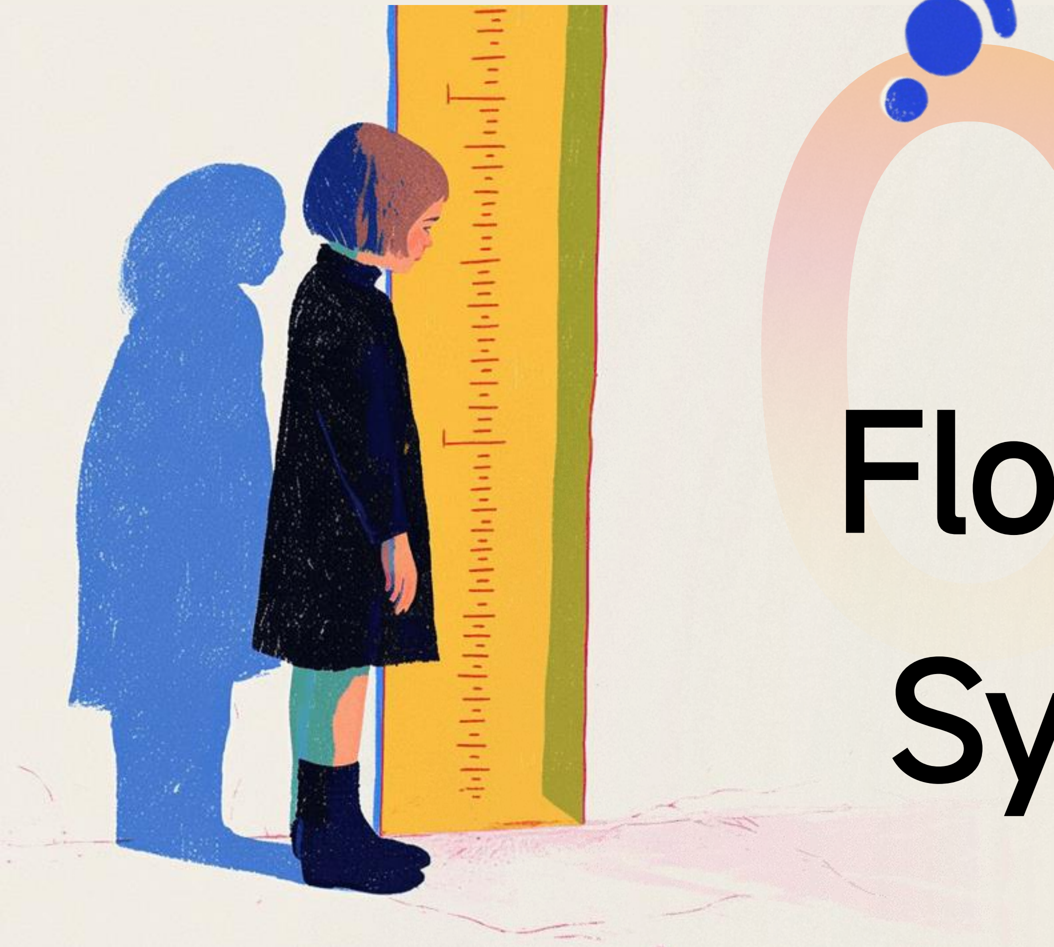
Basic Symbols: Declare, Input, Process, Output. Designed algorithms for addition and division.

02

Session Objectives

Session 2: Objectives

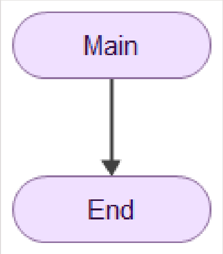
- Distinguish between Assignment, Comparison, and Logical operators.
- Understand how to use these operators in mathematical problems.
- Master the Decision (If) symbol in flowcharts.
- Design flowcharts using If/Else and nested If/Else.



03

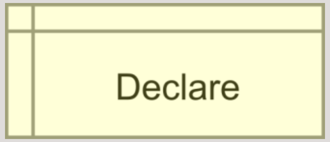
Flowchart Symbols

Flowchart Symbols: Review & New



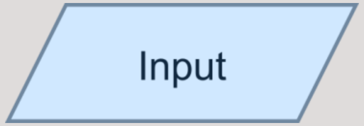
Main / End

Start/Stop



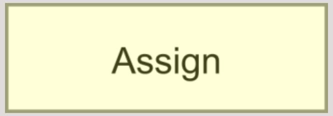
Declare

Creates variables



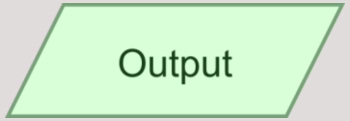
Input

Gets data



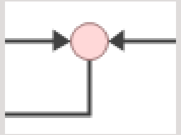
Assign

Performs assignment



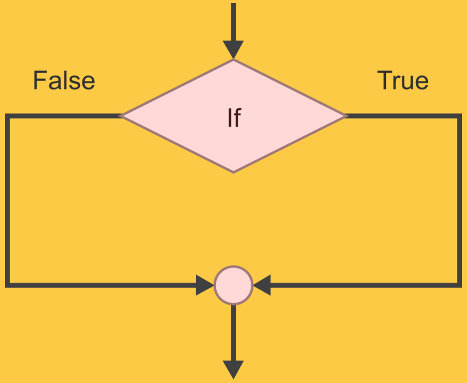
Output

Displays results



Arrows

Shows direction



If (NEW)

Splits path based on condition

04

Operators

The Building Blocks of Logic: Operators

Operators are special symbols that allow the algorithm to perform "actions" like calculations and comparisons.

Assignment

Comparison

Logical

Operator 1: The Assignment Operator (=)

It "assigns" or "puts" a value into a variable. It calculates the value on the right and stores it in the variable on the left.

IMPORTANT: This is NOT mathematical "equality". This is an "assignment" action.

Used inside the `Process` symbol in Flowgorithm.

Usage:

Variable = Value

Examples:

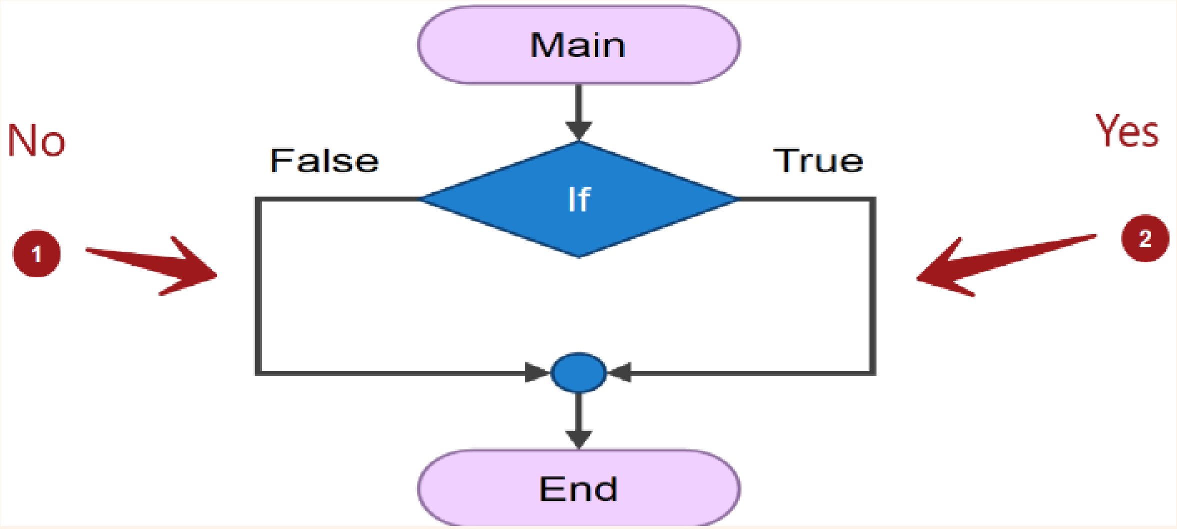
- `number1 = 5`
- `name = "Ahmet"`
- `total = number1 + number2`

Operator 2: Comparison

They compare two values and produce a single answer: TRUE or FALSE.

These operators allow the algorithm to "make decisions".

Used inside the `Decision` (If) and `Loop` symbols.



Common Comparison Operators

Flowgorithm / Python	Math Symbol	Meaning	Example (number = 5)	Result
`==`	=	Is equal to?	`number == 5`	**True**
`!=`	≠	Is not equal to?	`number != 10`	**True**
`>`	>	Is greater than?	`number > 3`	**True**
`<`	<	Is less than?	`number < 2`	**False**
`>=`	≥	Is greater than or equal to?	`number >= 5`	**True**
`<=`	≤	Is less than or equal to?	`number <= 4`	**False**

Assignment (=) vs. Comparison (==)

Assignment (=)

Used to **give** a value to a variable.

```
x = 5
```

"Let the value of x be 5"

Used in a `Process` box.

Comparison (==)

Used to **check if** two values are equal.

```
x == 5
```

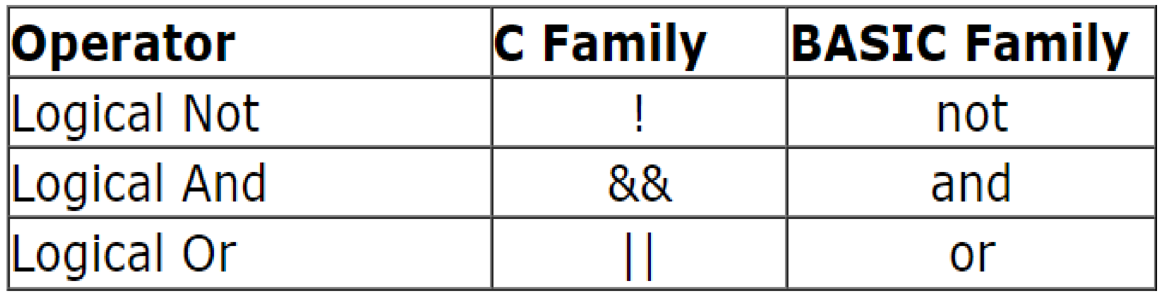
"Is the value of x equal to 5? -> True/False"

Used in a `Decision` box.

Operator 3: Logical

They combine multiple True/False conditions to produce a single TRUE or FALSE result.

Used for making complex decisions.



Operator	C Family	BASIC Family
Logical Not	!	not
Logical And	&&	and
Logical Or	||	or

- ✓ and
- ✓ or
- ✓ not

Logical Operator: and

Returns `True` only if ALL conditions are true.

If even one is false, the result is `False`.

(x > 0)
TRUE

∩

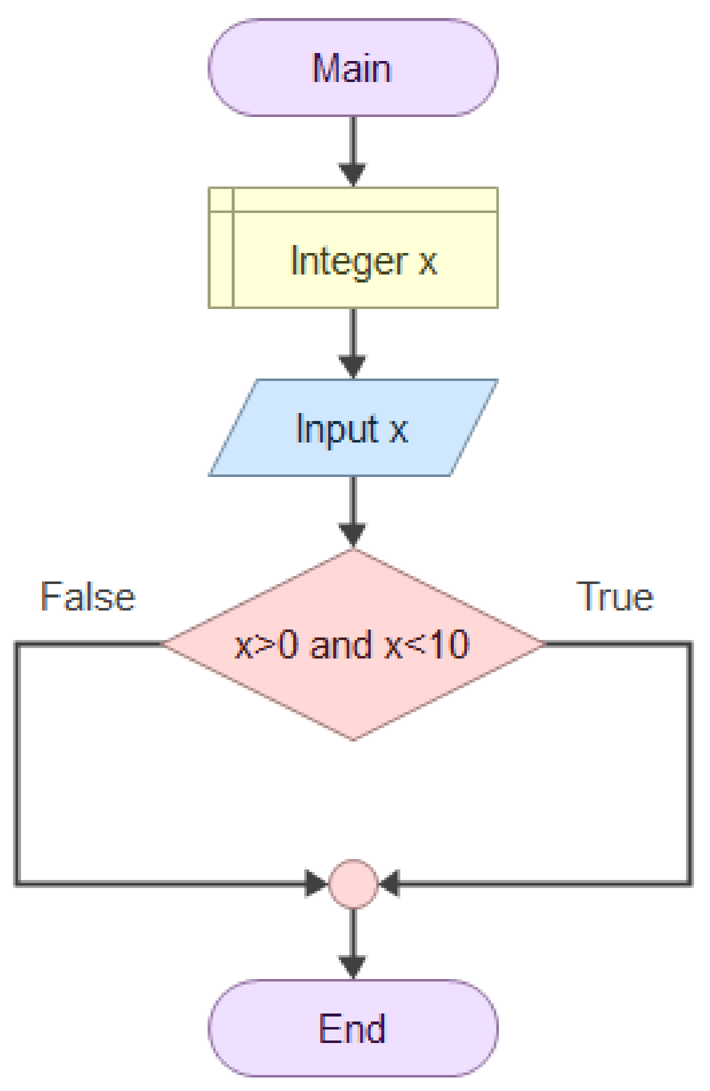
(x < 10)
TRUE

=

(x > 0) and (x < 10)
TRUE

Mathematical Equivalent: Intersection (∩)

Usage: "Check if a number is between 0 and 10."



Logical Operator: or

Returns `True` if at least ONE condition is true. It is only `False` if ALL conditions are false.

(grade < 0)
FALSE

∪

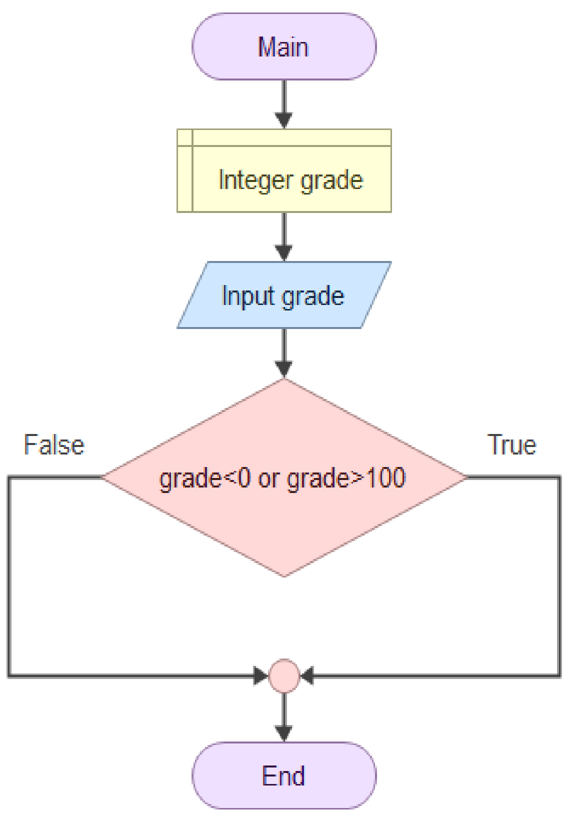
(grade > 100)
TRUE

=

(grade < 0) or (grade > 100)
TRUE

Mathematical Equivalent: Union (U)

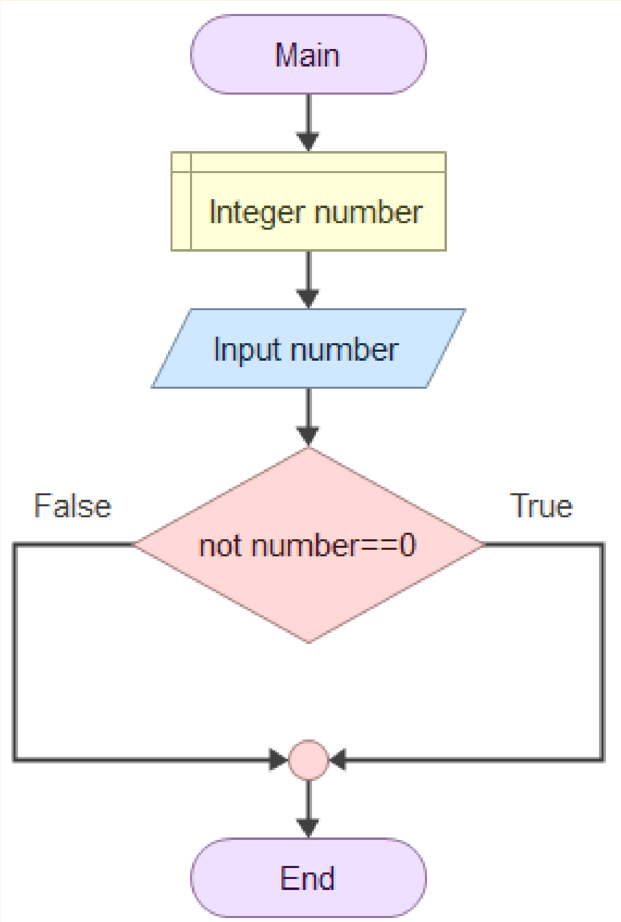
Usage: "If the exam grade is invalid, show an error."



Logical Operator: not

It reverses the result of a condition.

Usage: "If the number is NOT zero..."



True becomes False
False becomes True

number == 0 → not (number == 0)

TRUE → FALSE

number == 0 → not (number == 0)

FALSE → TRUE

Operator Precedence

Just like in mathematics (multiplication before addition), algorithmic operators have an order of operations.

1. Parentheses `( )` (Always first)

2. Mathematical Operators (`*`, `/`, `+`, `-`, `%`)

3. Comparison Operators (`==`, `>`, `<`, `!=`)

4. Logical Operators (`not`, `and`, `or`)

Example:

(5 + 3) > 7 and 10 != 11

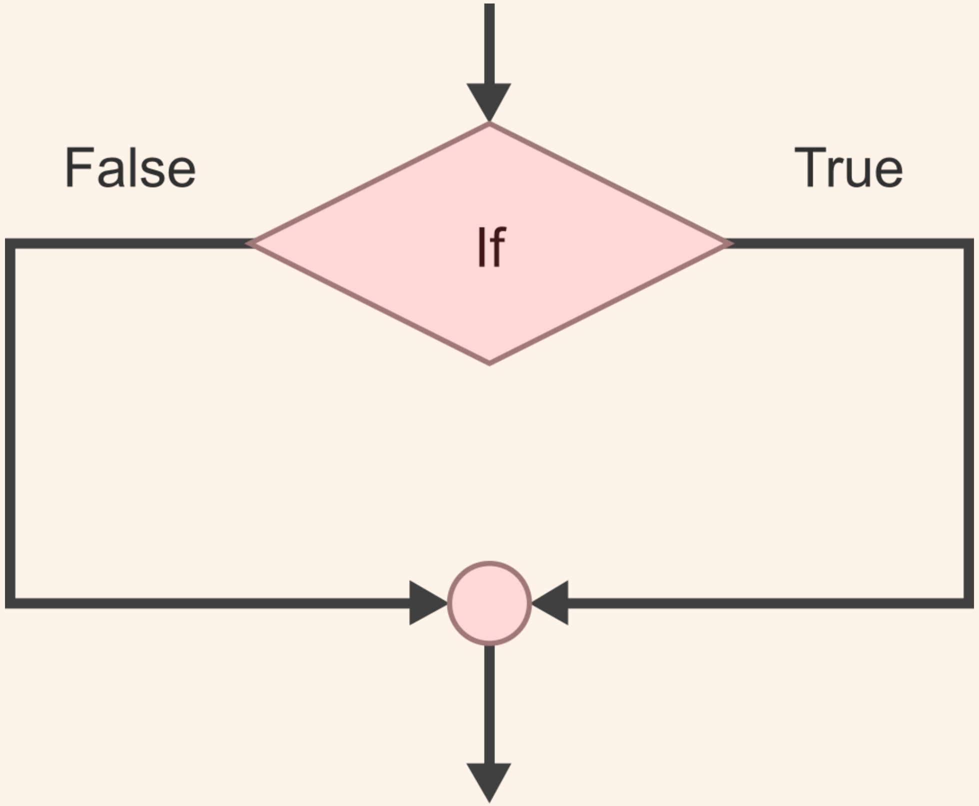
Step 1: `8 > 7 and 10 != 11`

Step 2: `True and True`

Step 3: Result: True

05

Decision Symbol

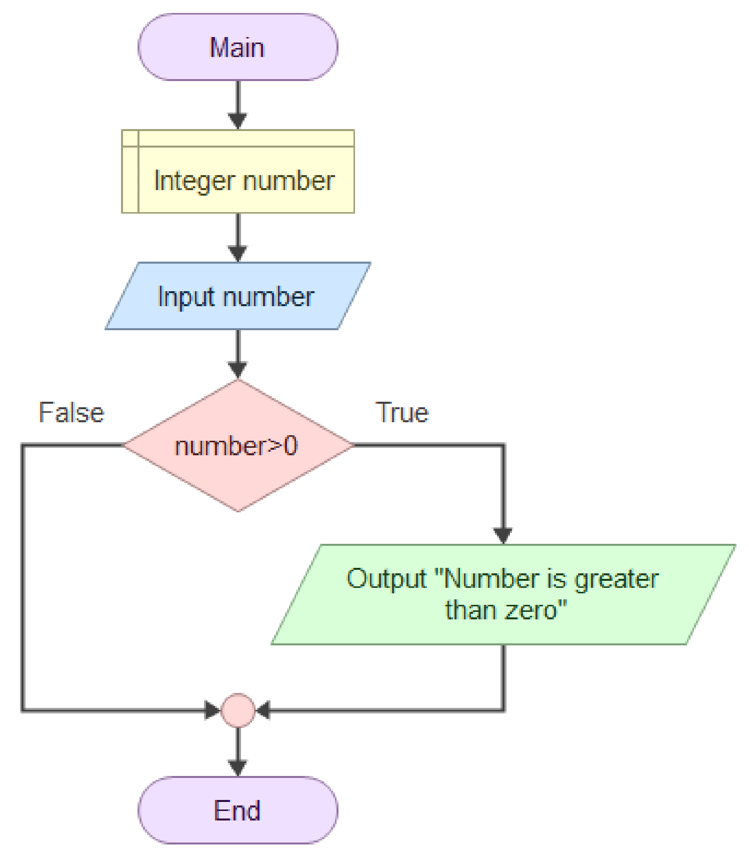


Symbol 6: Decision

- **Flowgorithm Name:** `If`
- **Meaning:** Decision / Condition
- **What it does:** The "fork in the road" for the algorithm, allowing different behaviors.
- **Usage:** Write a condition inside that results in `True` or `False`.

Example: `If` Only (Positive Check)

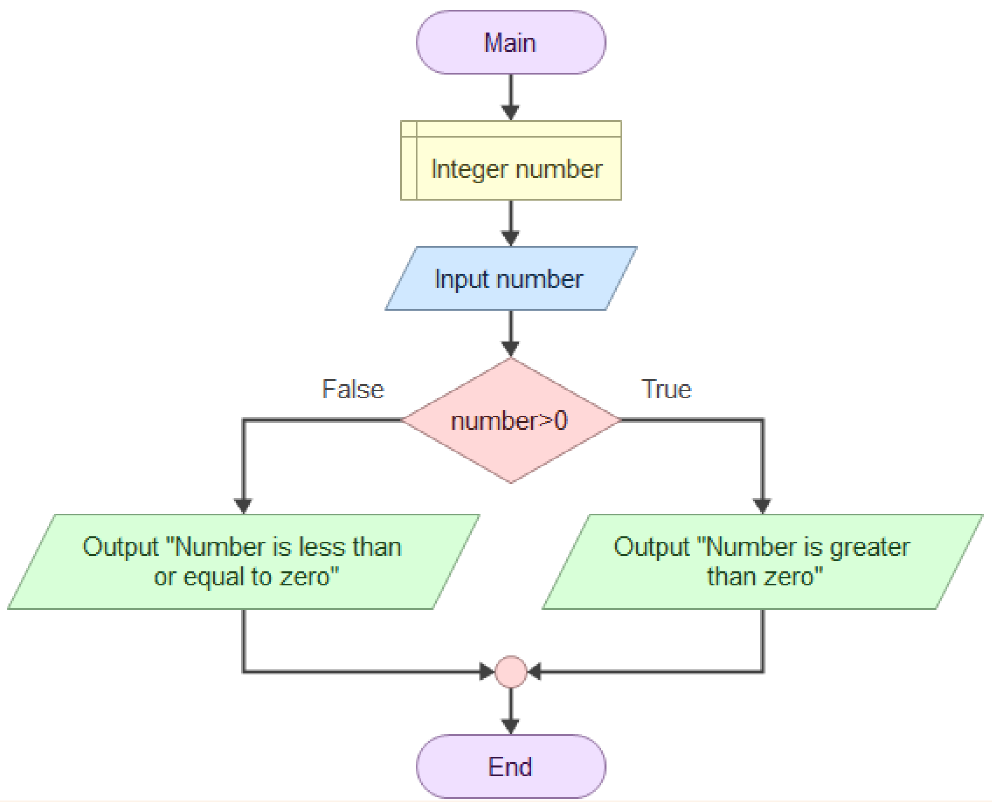
Problem: If the entered number is positive, print "Positive". Otherwise, do nothing.



Logic: We only care about the `True` path. The `False` path remains empty.

Example: `If / Else` (Positive / Not Positive)

Problem: If the entered number is positive, print "Positive". Otherwise, print "Not Positive".


Main
Integer number
Input number
False
number>0
True
Output "Number is less than or equal to zero"
Output "Number is greater than zero"
End

Logic: We care about both the `True` and `False` paths.

06

Even/Odd Application

Algorithm Example: Even or Odd?

Problem: Design an algorithm that takes an integer and prints "Even" or "Odd".

What we need:

1. Get a number from the user (`Input`).
2. Make a decision (`If`).
3. Print the result (`Output`).

Even/Odd: The Mathematical Logic

The Rule

If a number's remainder is 0 when divided by 2, it is EVEN. Otherwise, it is ODD.

The Operator

We need the Modulus operator. It gives us the remainder of a division.

The Modulus Operator: %

What it does:

It gives the remainder of the first number divided by the second.

Our Algorithm's Condition:

```
(number % 2) == 0
```

Examples:

`10 % 2` results in `0`. (10 is Even)

`11 % 2` results in `1`. (11 is Odd)

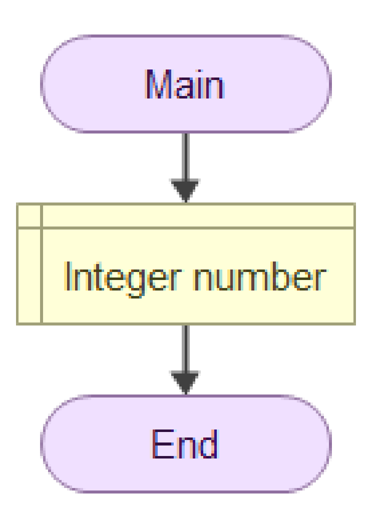
`4 % 2` results in `0`. (4 is Even)

`17 % 5` results in `2`.

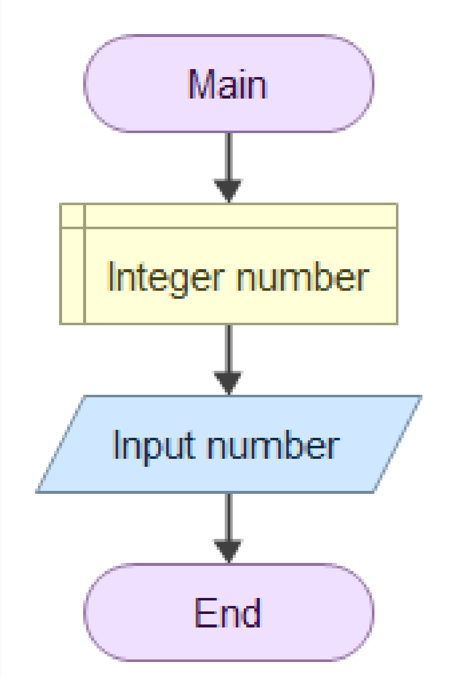
Even/Odd: Textual Algorithm (Pseudo code)

```
1  Start.

2  Declare an integer variable `number`.

3  Get the value for `number` from the user.

4  IF (`number % 2`) `== 0` is True:
   Output "The number is Even".

5  ELSE: Output "The number is Odd".

6  Stop.
```

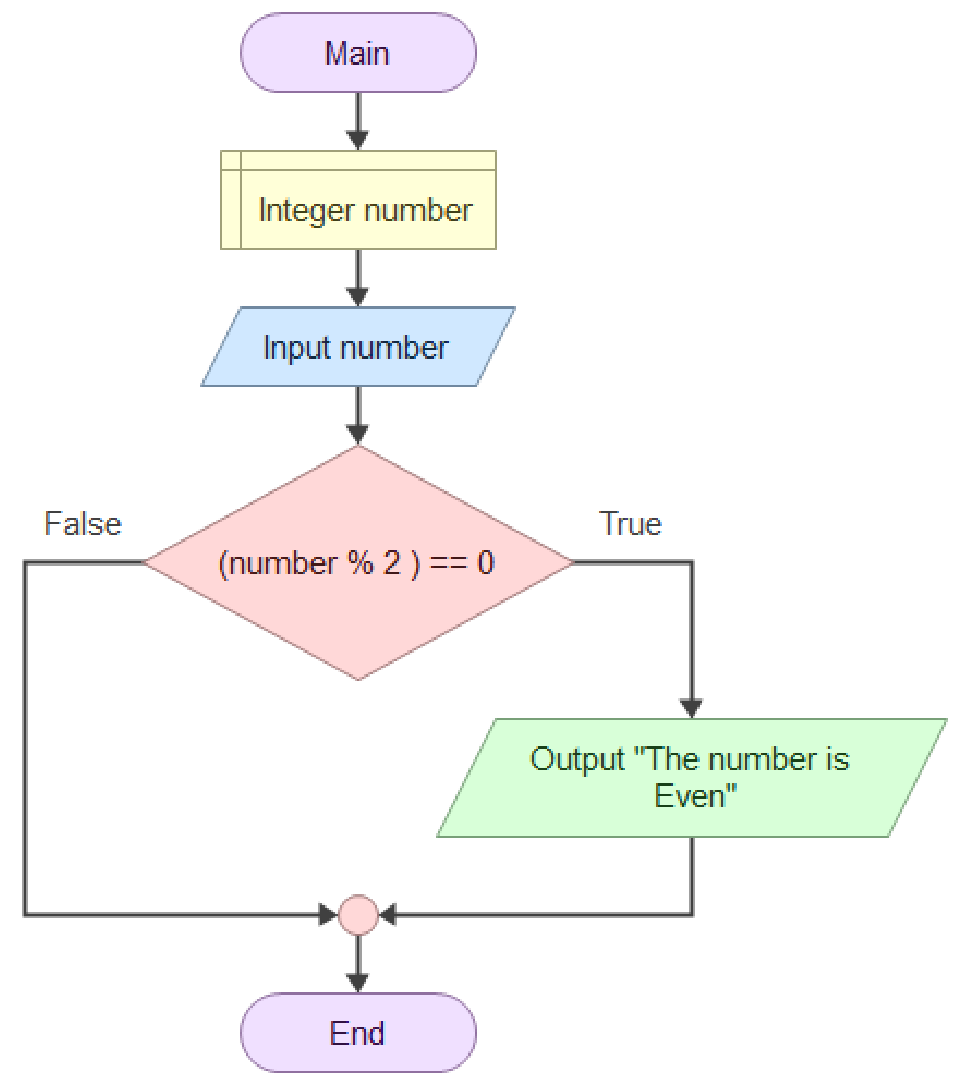
Flowgorithm: Even/Odd (Step 1)



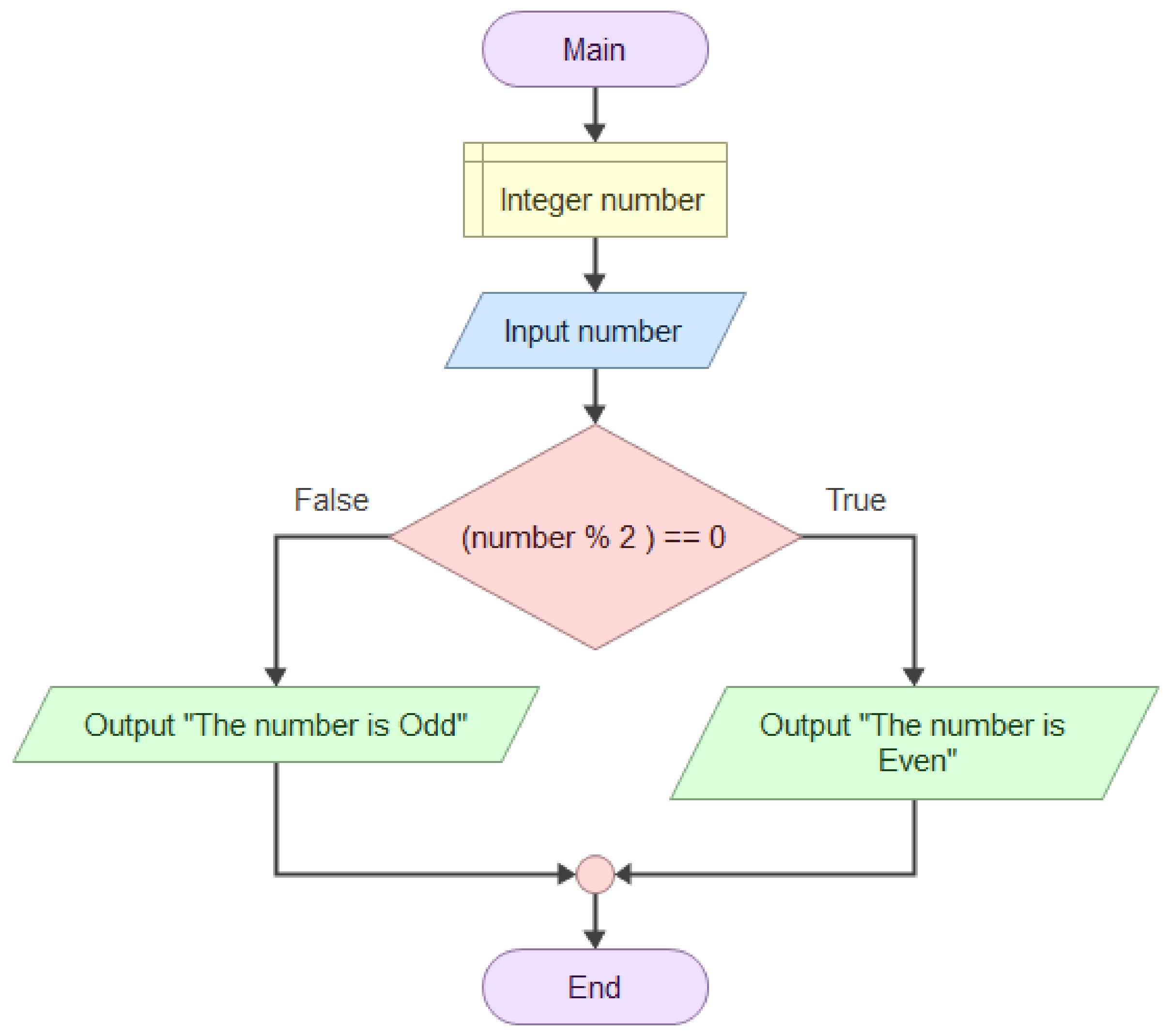
Flowgorithm: Even/Odd (Step 2)



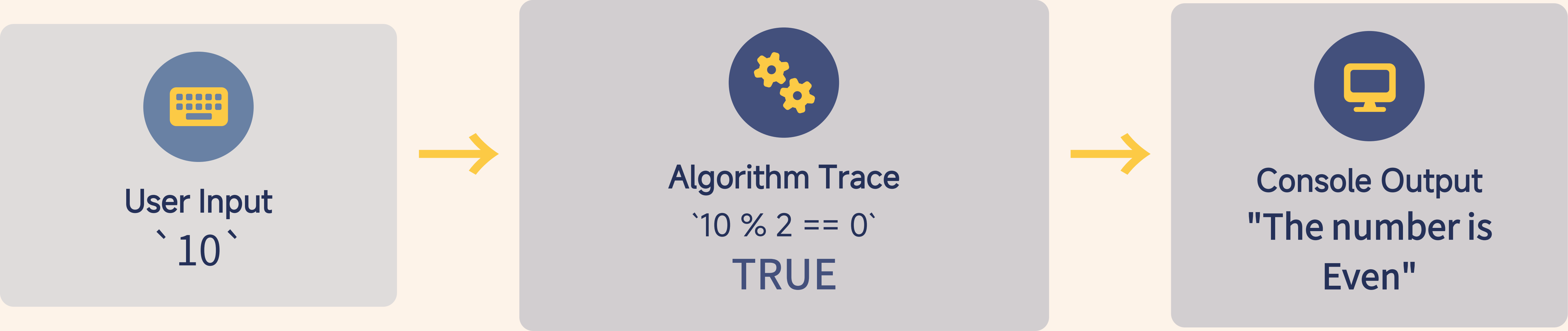
Flowgorithm: Even/Odd (Step 3)



Flowgorithm: Even/Odd (Step 4)



Flowgorithm: Testing (Test 1)



User Input

` 10 `

Algorithm Trace

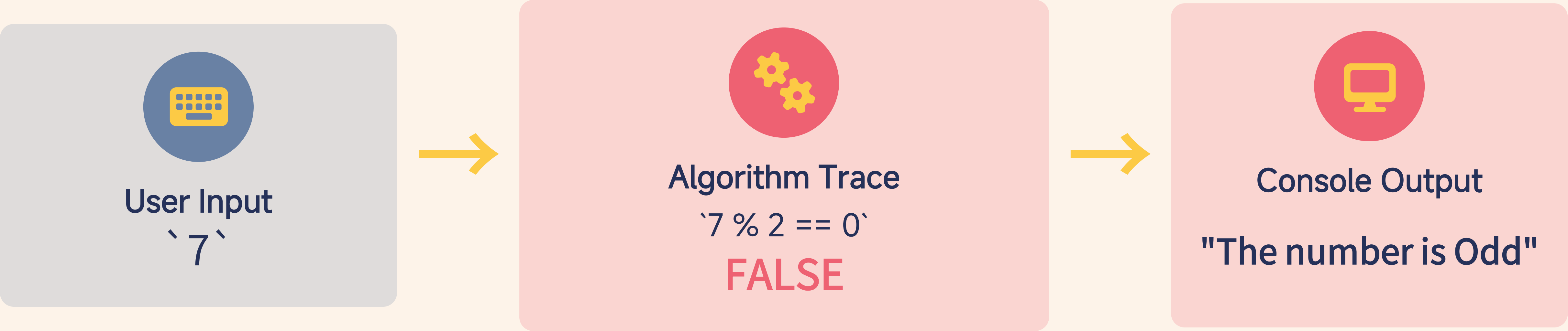
`10 % 2 == 0`

TRUE

Console Output

"The number is Even"

Flowgorithm: Testing (Test 2)



User Input

` 7 `

Algorithm Trace

`7 % 2 == 0`

FALSE

Console Output

"The number is Odd"

07

Piecewise Function Algorithm

Algorithm Example 2: Piecewise Function

Problem: Calculate the value of `y` based on a user's input for `x`.

Our Function:

$$f(x) = \begin{cases} x^2, & x < 0 \\ 10, & x = 0 \\ 2x + 5, & x > 0 \end{cases}$$

This requires more than one `If/Else`. We need to *nest* our decisions.

`y = x^2`
(if `x < 0`)

`y = 10`
(if `x = 0`)

`y = 2x + 5`
(if `x > 0`)

Piecewise Function: The Logic

1

Step 1: Ask "Is `x < 0`?"

- IF TRUE: Calculate `y = x * x`. Done.
- IF FALSE: Go to Step 2.

2

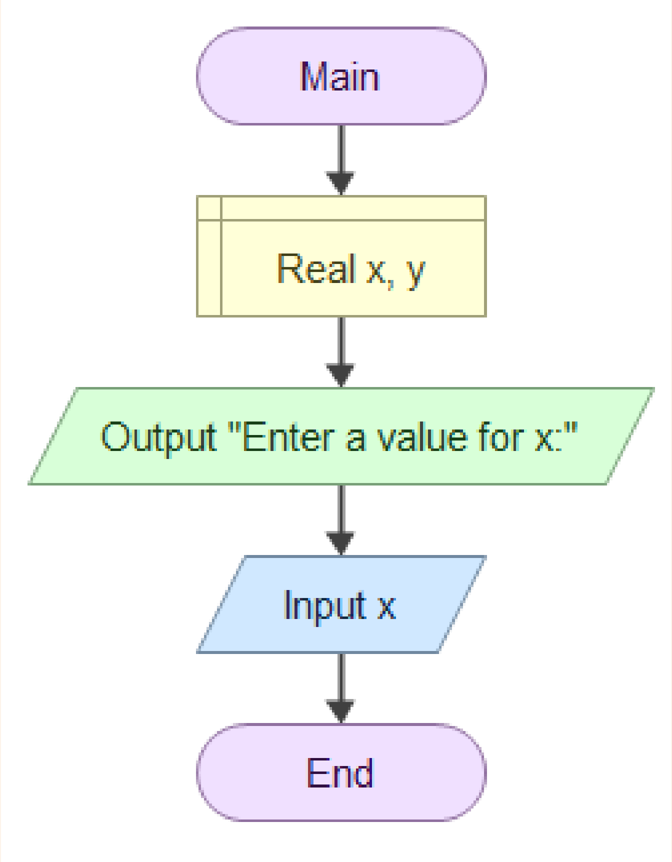
Step 2 (Nested): Ask "Is `x == 0`?"

- IF TRUE: Set `y = 10`. Done.
- IF FALSE: It must be > 0. Calculate `y = 2x + 5`. Done.

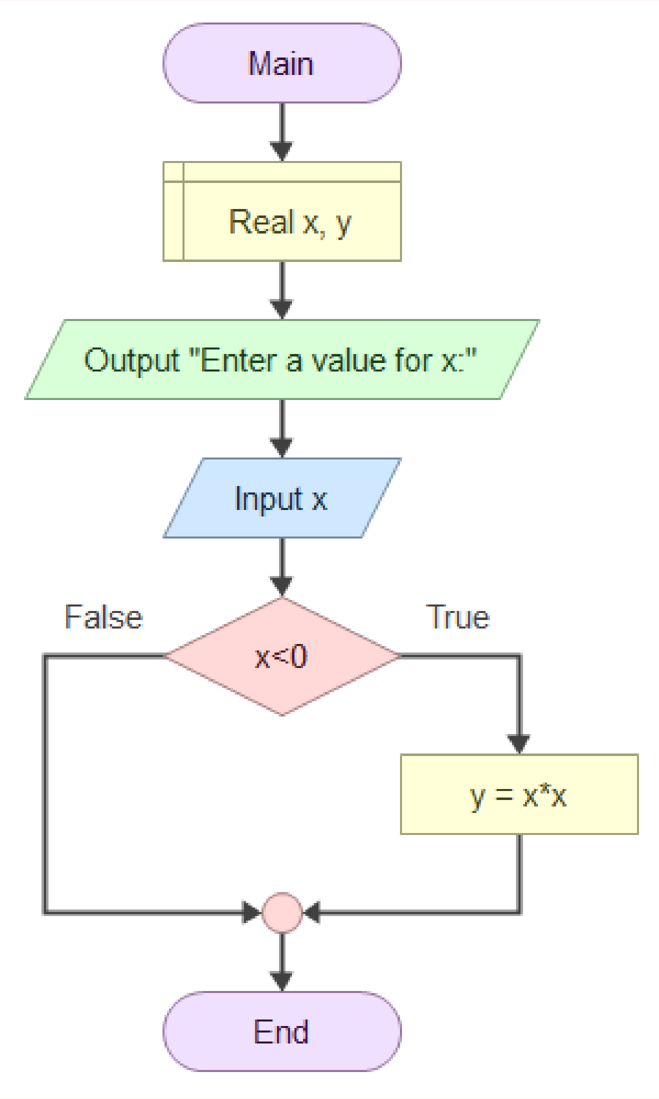
Piecewise Function: Textual Algorithm

```
1 Start.
2 Declare `Real` variables: `x`, `y`.
3 Output: "Enter a value for x:".
4 Input: `x`.
5 IF `x < 0` is True: Process: `y = x * x`.
    6 ELSE:
        7 IF `x == 0` is True: Process: `y = 10`.
            8 ELSE: Process: `y = (2 * x) + 5`.
9 Output: "The value of y is: " & y
10 Stop.
```

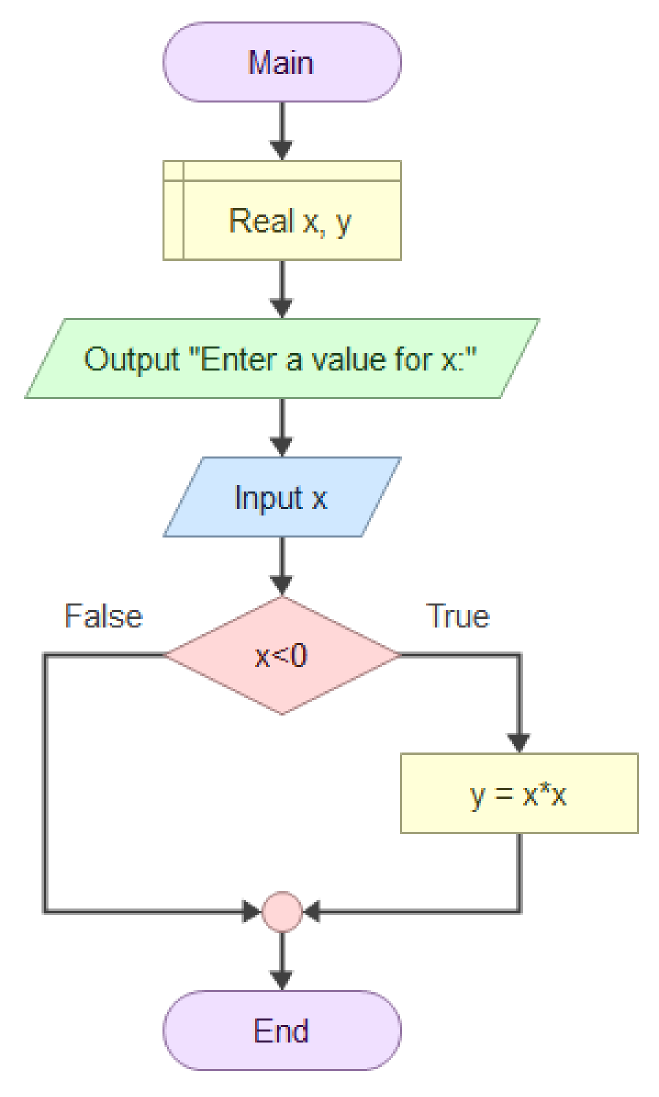
Flowgorithm: Piecewise (Step 1)



Flowgorithm: Piecewise (Step 2)



Flowgorithm: Piecewise (Step 3)



Completed Flowchart & Test

Test Run:

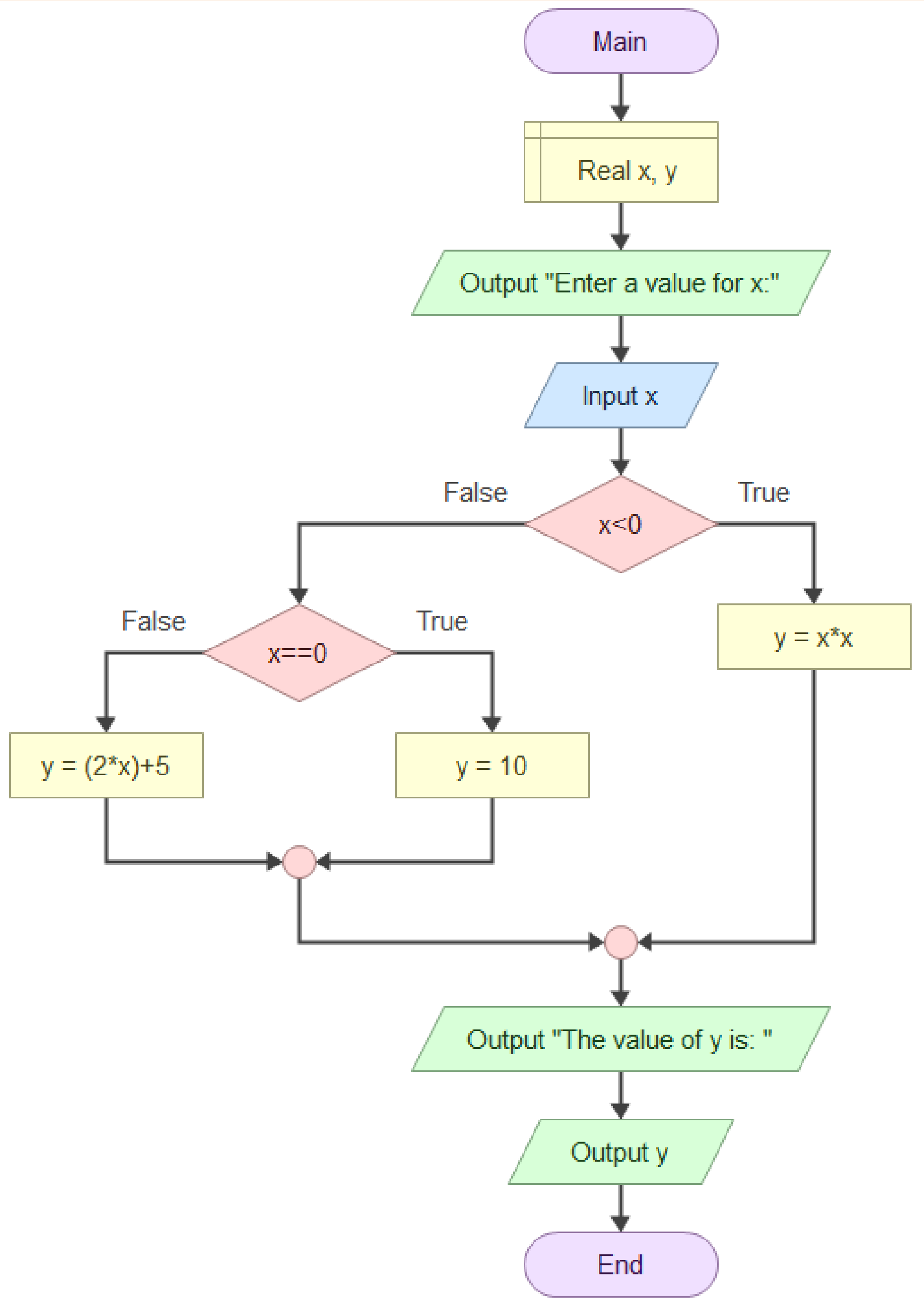
Input: `x = 4`

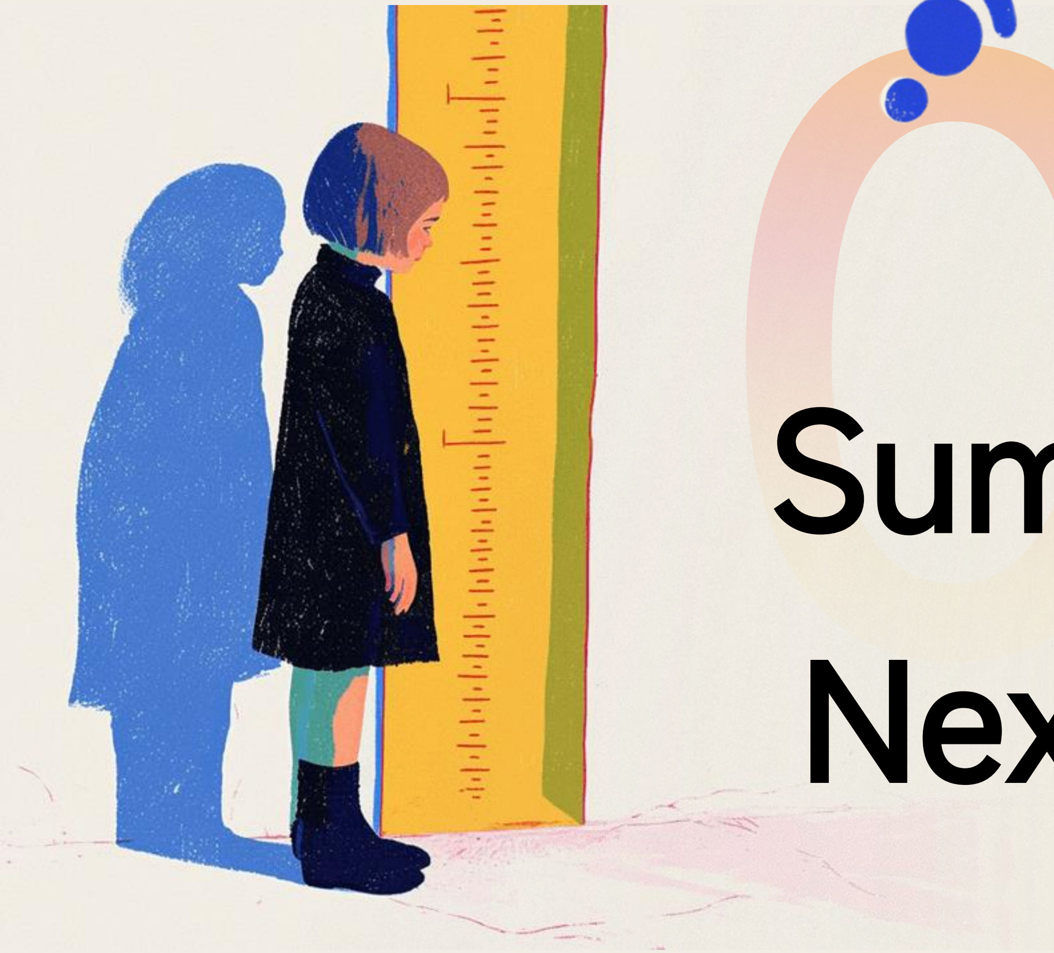
1. `x < 0`? (4 < 0) -> `False`.

2. `x == 0`? (4 == 0) -> `False`.

3. `Process: y = (2 * 4) + 5` -> `y` is now 13.

4. `Output: "The value of y is: 13"`





08

Summary & Next Steps

What We Learned

Operators:

- **Assignment (`=`):** Assigns a value.
- **Comparison (`==`, `>`, `<`...):** Returns `True`/`False`.
- **Logical (`and`, `or`, `not`):** Combines conditions.
- **Modulus (`%`):** Finds the remainder.

Decision Making:

- **`If` Symbol:** Splits path based on a condition.
- **`If/Else`:** Used for the Even/Odd check.
- **Nested `If/Else`:** Solved the Piecewise Function.

Do you have any questions?

End of Day 1 Section 2

Prof. Dr. Turgay Tugay BİLGİN
https://www.linkedin.com/in/ttbilgin