Programming Languages and Program Development

Prepared by Dr. Walaa Khaled

Problem Statement:

Problem Statement help diagnose the situation so that your focus is on the problem, helpful tools at this stage include Algorithms and flowcharts for identifying the expected steps of a process. Therefore to solve any problem,

- Collect and analyze information and data
- Talk with people familiar with the problem
- If at all possible, view the problem first hand
- Confirm all findings

What is Programming

- Program – a very specific set of instructions (or command lines) that making a computer do what you want it to do

- Programming – the process of creating a program
  - the development of a solution to an identified program, and setting up of a related series of instructions which, when directed through computer hardware, will produce the desired results

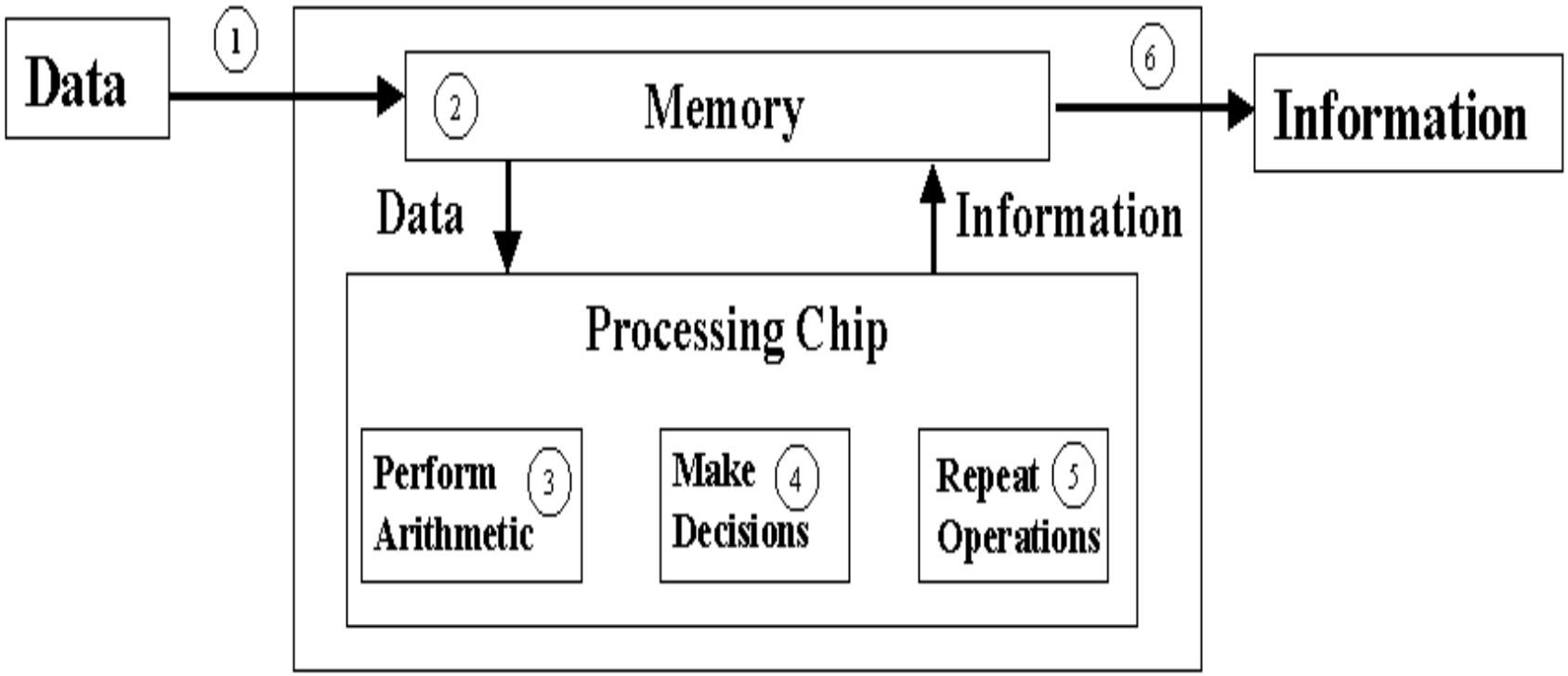
Steps in program development

1. Define the problem
2. Outline the solution
3. Develop the outline into an algorithm
4. Test the algorithm for correctness
5. Code the algorithm into a specific programming language
6. Run the program on the computer
7. Document and maintain the program

Define the Problem

- Divide the problem into three components (called IPO):
  - Inputs – what do you have?
  - Outputs – what do you want to have?
  - Processing
    - how do you go from inputs to outputs?
- A defining diagram is recommended

Define the Problem



Outline the Solution

- The major processing steps involved
- The major subtasks (if any)
- The major control structures (e.g. repetition loops)
- The major variables and record structures
- The mainline logic

Develop the Outline into an Algorithm

- Algorithm is a set of precise steps that describe exactly the tasks to be performed, and the order in which they are to be carried out
- Pseudocode (a form of structured English) is used to represent the solution algorithm

Test the Algorithm for Correctness

- The main purpose of desk checking the algorithm is to identify major logic errors early, so that they may be easily corrected
- Test data needs to be walked through each step in the algorithm, to check that the instructions described in the algorithm will actually do what they are supposed to

Code the Algorithm into a Specific Programming Language

- Only after all design considerations have been met should you actually start to code the program into your chosen programming language (e.g. Visual Basic, Java, C++)

Run the Program on the Computer

- This step uses a program compiler and programmer-designed test data to machine test the code for syntax errors
- Program complier translate high-level languages (e.g. VB) to low-level machine language before execution

Document and Maintain the Program

- Not the last step in the program development process
- An ongoing task from the initial definition of the problem to the final test result
- Involves both external documentation (such as hierarchy charts) and internal documentation that may have been coded in the program

Example 1: Circle

- Calculate the circumference and area
- IPO
  - I: radius
  - P: calculate circumference and area as

    circumference = 2 * P I* radius

    area=PI * radius ^ 2
  - O: radius, circumference, area

Algorithms and pseudocode

- A typical programming task can be divided into two phases:
- ***Problem solving phase***
  - produce an ordered sequence of steps that describe solution of problem
  - this sequence of steps is called an ***algorithm***
- ***Implementation phase***
  - implement the program in some programming language

Algorithms and pseudocode(cont.)

- First produce a general algorithm (one can use ***pseudocode***)
- Refine the algorithm successively to get step by step detailed ***algorithm*** that is very close to a computer language.
- ***Pseudocode*** is an artificial and informal language that helps programmers develop algorithms. Pseudocode is very similar to everyday English.

Algorithms and pseudocode(cont.)

Pseudocode common format

Input: READ, OBTAIN, GET

Output: PRINT, DISPLAY, SHOW

Compute: COMPUTE, CALCULATE, DETERMINE

Initialize: SET, INIT

Add one: INCREMENT, BUMP

Algorithms and pseudocode(cont.)

Pseudocode common format

IF-THEN-ELSE

```
IF condition THEN
        sequence 1
ELSE
        sequence 2
ENDIF
```

WHILE

```
WHILE condition
        sequence
ENDWHILE
```

FOR

```
FOR iteration bounds
        sequence
ENDFOR
```

NOT a language syntax

Algorithms and pseudocode *(cont.)*

- Example: Student pass

Algorithms and pseudocode *(cont.)*

Example : Student pass?

1. **GET Grade**
2. **If Grade is greater than or equal to 60 THEN**
3. **Print "passed"**
4. **ELSE**
5. **Print "failed**

The Flowchart

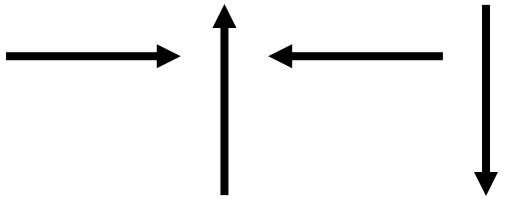
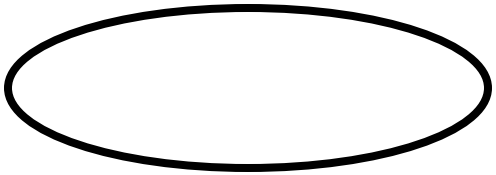
- (Dictionary) A schematic representation of a sequence of operations, as in a manufacturing process or computer program.
- (Technical) A graphical representation of the sequence of operations in an information system or program.
- Information system flowcharts show how data flows from source documents through the computer to final distribution to users.
- Program flowcharts show the sequence of instructions in a single program or subroutine.
- Different symbols are used to draw each type of flowchart.

The Flowchart

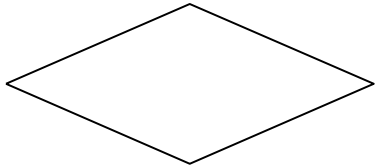
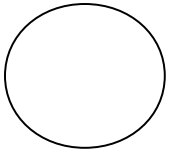
A Flowchart

- shows logic of an algorithm
- emphasizes individual steps and their interconnections
- e.g. control flow from one action to the next

Flowchart Symbols

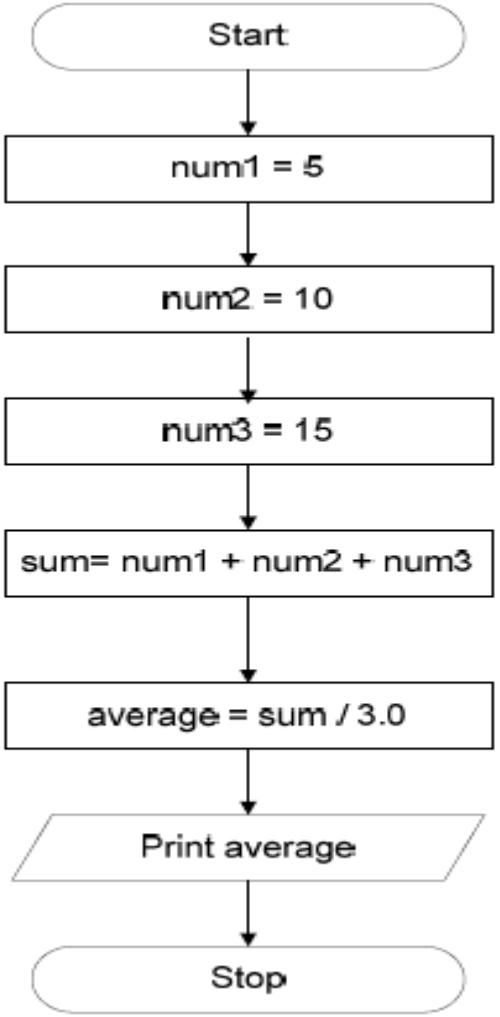
Symbol	Function
	Show the direction of data flow or logical solution.
	Indicate the beginning and ending of a set of actions or instructions (logical flow) of a module or program.
	Indicate a process, such as calculations, opening and closing files.

Flowchart Symbols

	Indicate input to the program and output from the program.
	Use for making decision. Either True or False based on certain condition.
	Use for doing a repetition or looping of certain steps.
	Connection of flowchart on the same page.
	Connection of flowchart from page to page.

The Flowchart *(cont.)*

Example:
Average of 3 numbers



Example 2

- Write an algorithm and draw a flowchart to convert the length in feet to centimeter.

Pseudocode:

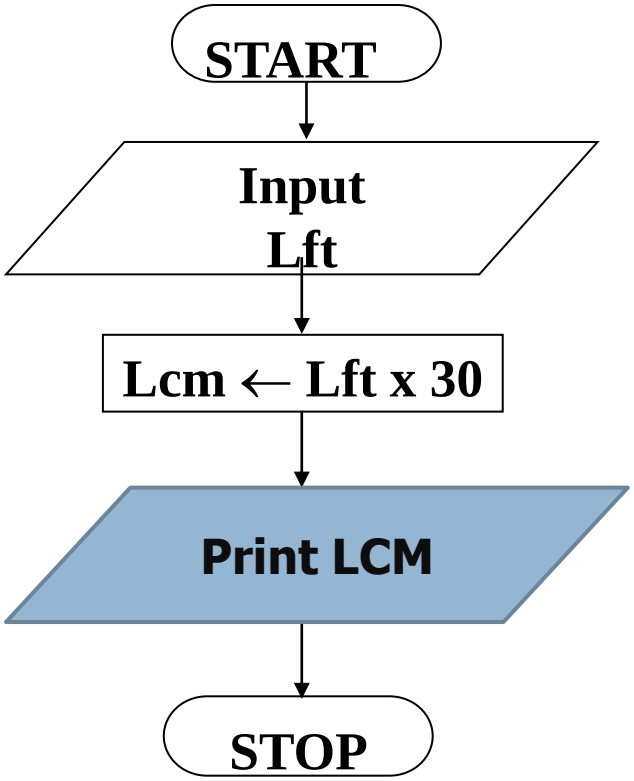
- Input the length in feet (Lft)
- Calculate the length in cm (Lcm) by multiplying LFT with 30
- Print length in cm (LCM)

Example 2

Algorithm

- Step 1: Input Lft
- Step 2: Lcm ← Lft x 30
- Step 3: Print Lcm

Flowchart



Example 3

Write an algorithm and draw a flowchart that will read the two sides of a rectangle and calculate its area.

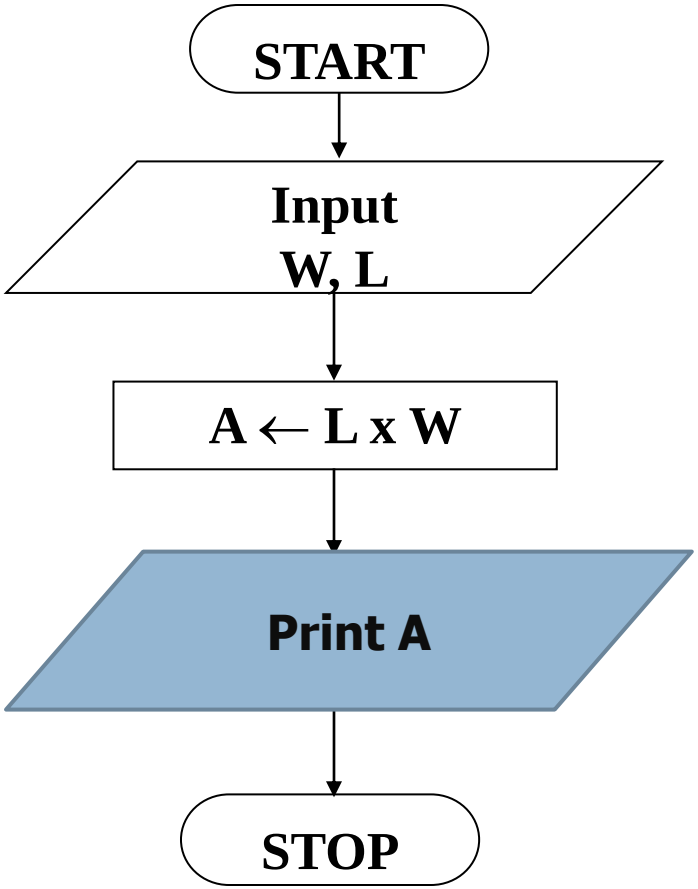
Pseudocode

- *Input the width (W) and Length (L) of a rectangle*
- *Calculate the area (A) by multiplying L with W*
- *Print A*

Example 3

Algorithm

- Step 1: Input W,L
- Step 2: A ← L x W
- Step 3: Print A



Example 4

- Write an algorithm and draw a flowchart that will calculate the roots of a quadratic equation

$$ax^2 + bx + c = 0$$

- Hint: **d** = sqrt ($b^2 - 4ac$), and the roots are: ***x*1** = (−*b* + *d*)/2*a* and ***x*2** = (−*b* − *d*)/2*a*

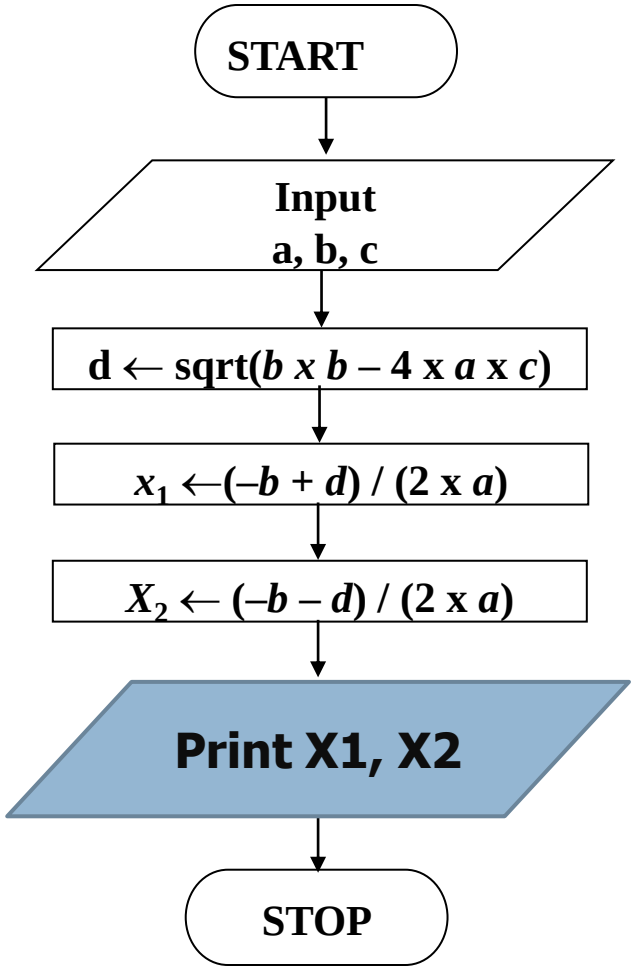
Example 4

Pseudocode:

- *Input the coefficients (a, b, c) of the quadratic equation*
- *Calculate* **d**
- *Calculate* ***x*1**
- *Calculate* **x2**
- *Print* x1 *and* x2

Example 4

- **Algorithm**:
  - Step 1: Input a, b, c
  - Step 2: $d \leftarrow \text{sqrt}\,(b \times b - 4 \times a \times c)$
  - Step 3: $x1 \leftarrow (-b + d) / (2 \text{ x } a)$
  - Step 4: $x2 \leftarrow (-b - d) / (2 \text{ x } a)$
  - Step 5: Print $x1$, $x2$

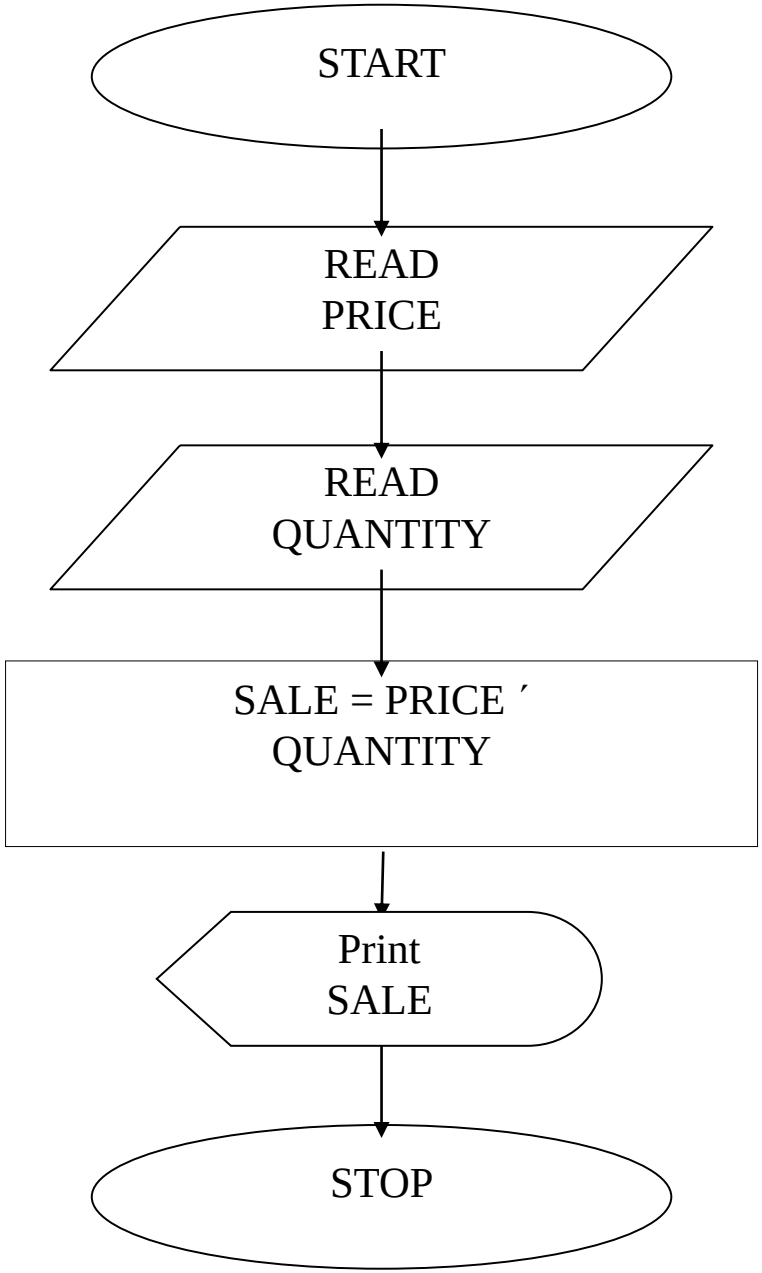


Example 5

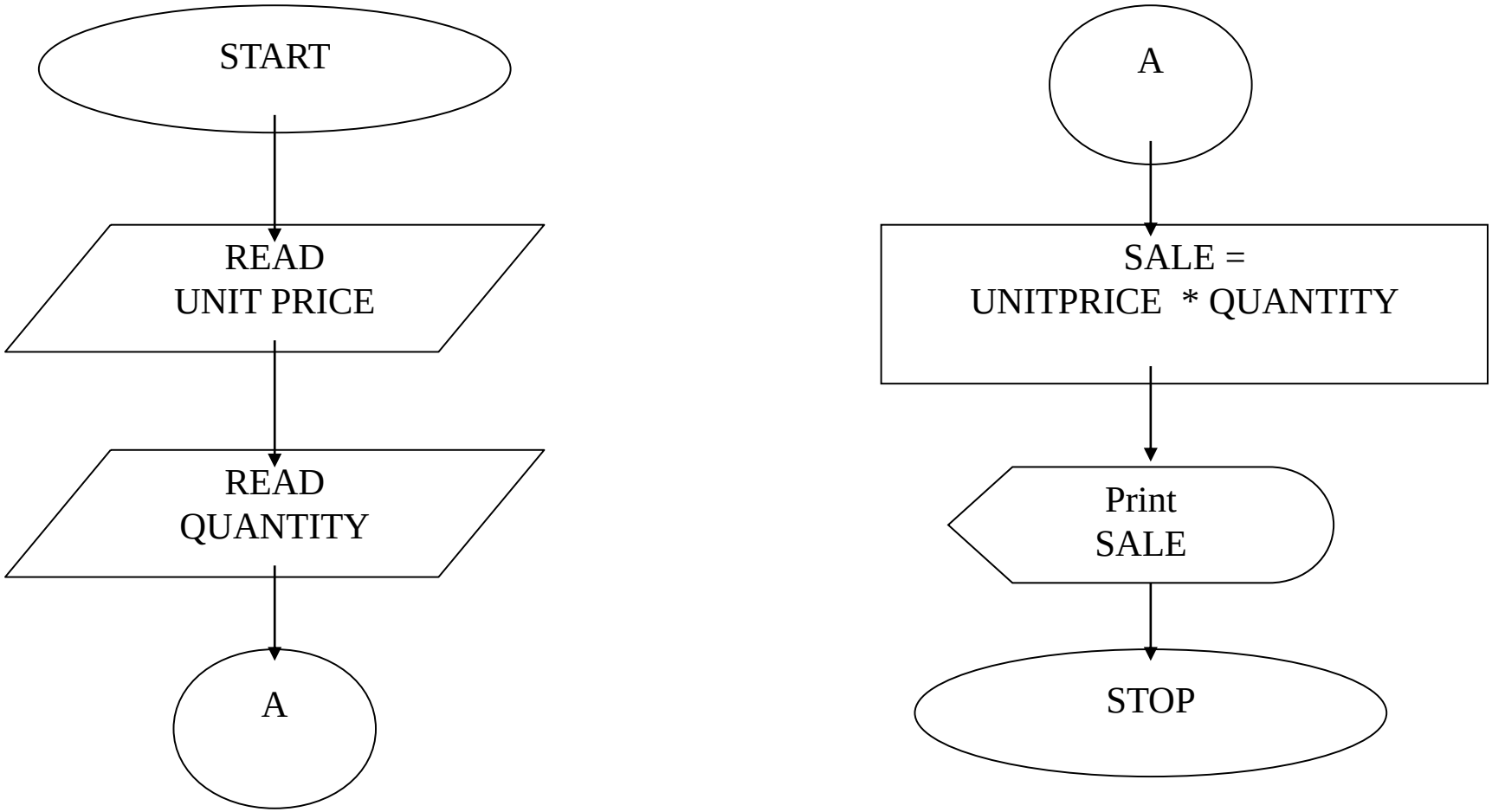
Draw a flowchart for a problem that to read two numbers. The first number represents the unit price of a product and the second number represents the quantity of the product sold. Calculate and print the total sale.

Example 5

- **Algorithm**:
  - Step 1: Read Price, Quantity
  - Step 2: Sale ← Price * quantity
  - Step 3: Print SALE



Example 5


START
READ
UNIT PRICE
READ
QUANTITY
A
A
SALE =
UNITPRICE * QUANTITY
Print
SALE
STOP

DECISION STRUCTURES

- The expression A>B is a logical expression
- *it describes a **condition** we want to test*
- ***if A>B is true (if A is greater than B)** we take the action on left*
- print the value of A
- ***if A>B is false (if A is not greater than B)** we take the action on right*
- print the value of B

IF–THEN–ELSE STRUCTURE

- The structure is as follows

If condition then

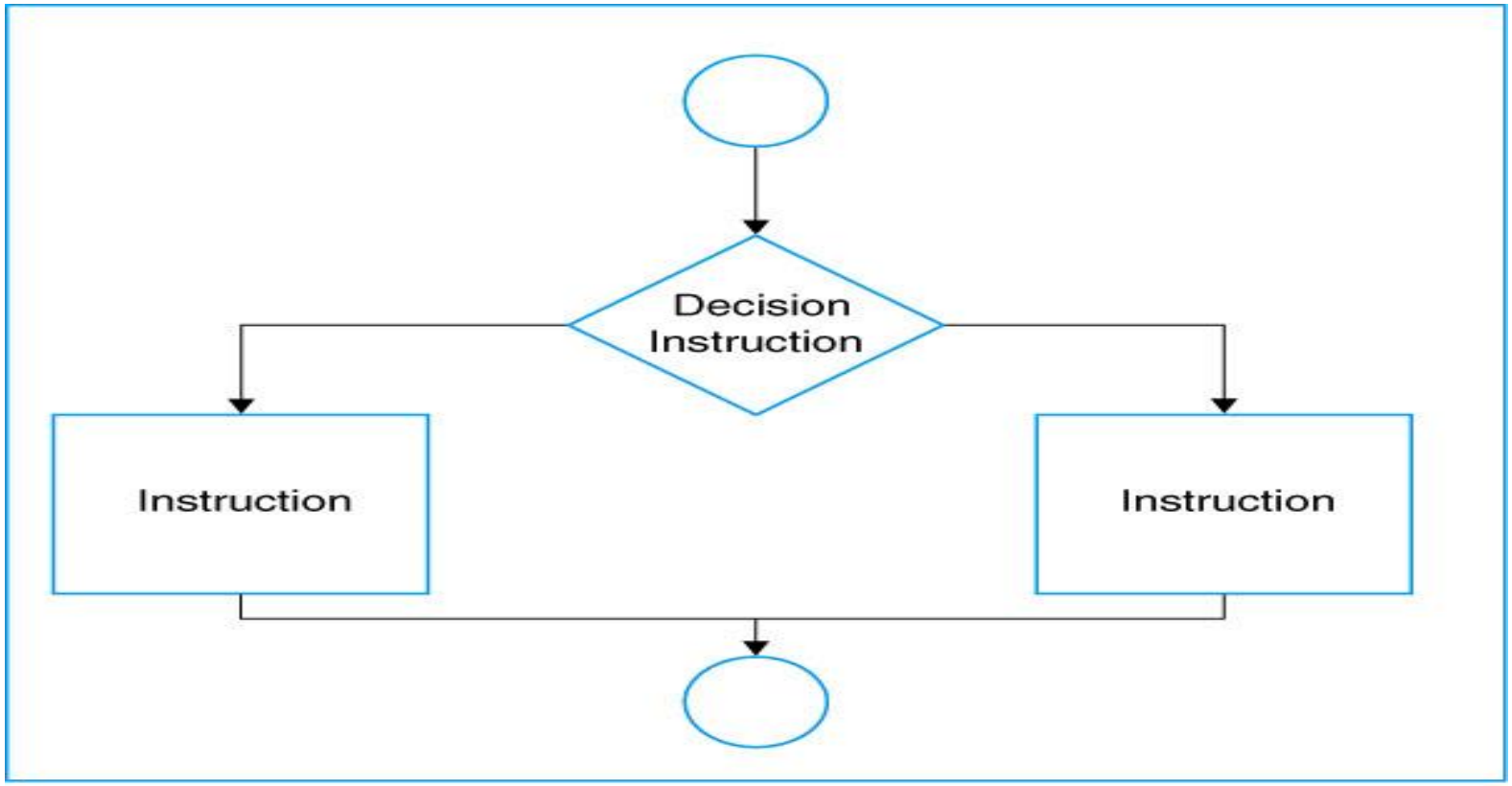
true alternative

else

false alternative

endif

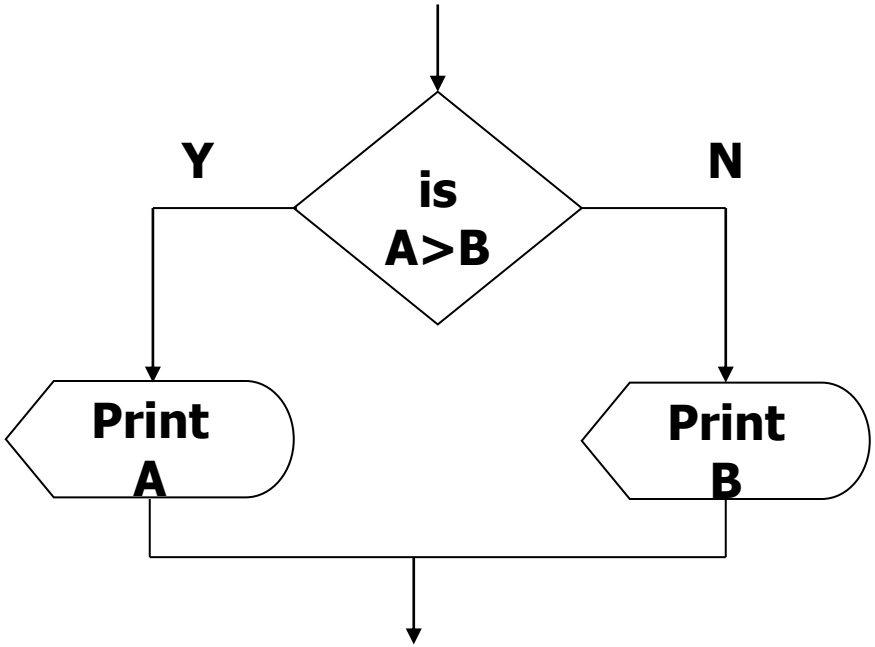
Decision Logic Structure



IF–THEN–ELSE STRUCTURE

- The algorithm for the flowchart is as follows:

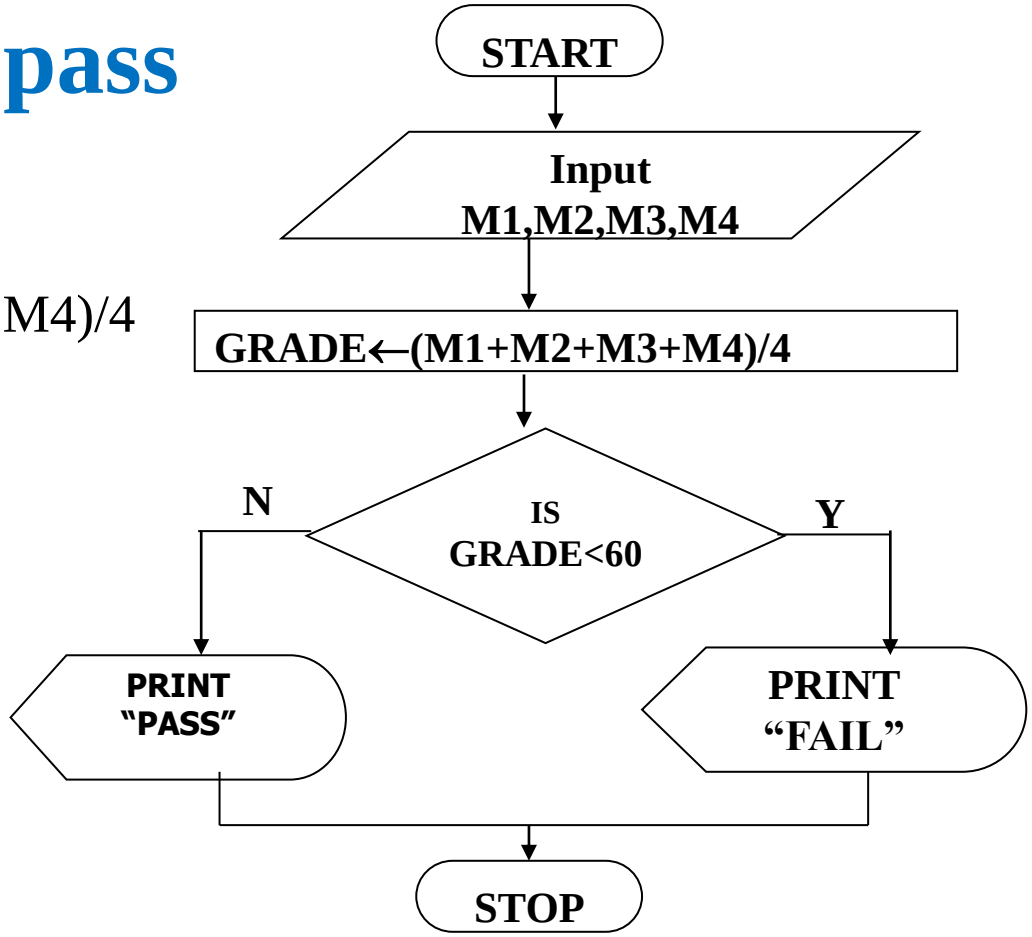
If A>B then
print A
else
print B
endif



Example 6

Example : Student pass

Step 1: Input M1,M2,M3,M4
Step 2: GRADE ← (M1+M2+M3+M4)/4
Step 3: if (GRADE <60) then
Print “FAIL”
else
Print “PASS”
endif



Relational Operators

Operator	Description
>	**Greater than**
<	**Less than**
=	**Equal to**
≥	**Greater than or equal to**
≤	**Less than or equal to**
≠	**Not equal to**

Example 7

- Write an algorithm that reads two values, determines the largest value and prints the largest value with an identifying message.

Example 7

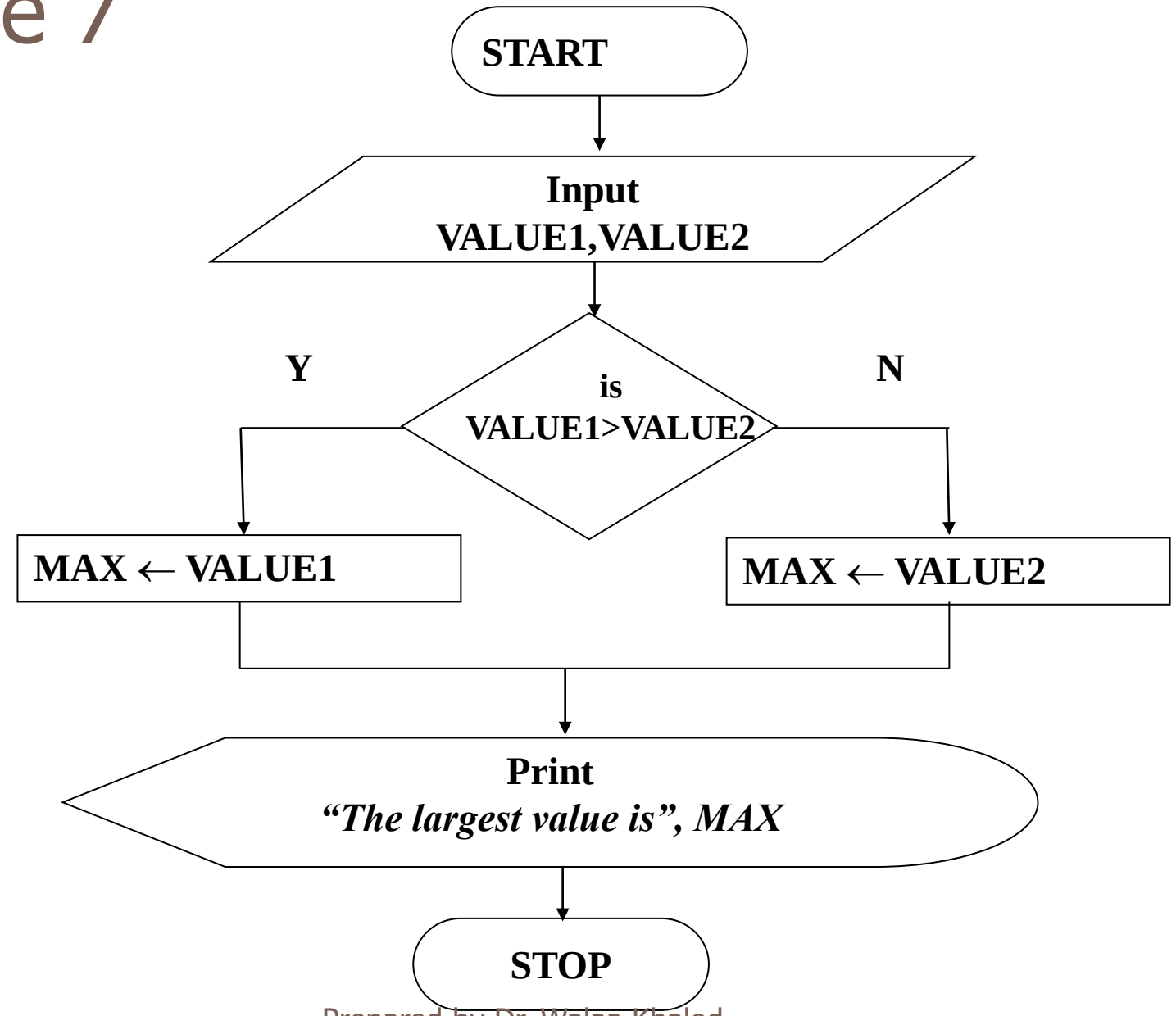
Algorithm

Step 1: *Input* VALUE1, VALUE2

Step 2: *if (*VALUE1 > VALUE2) *then*
MAX ← VALUE1
else
MAX ← VALUE2
endif

Step 3: *Print "The largest value is", MAX*

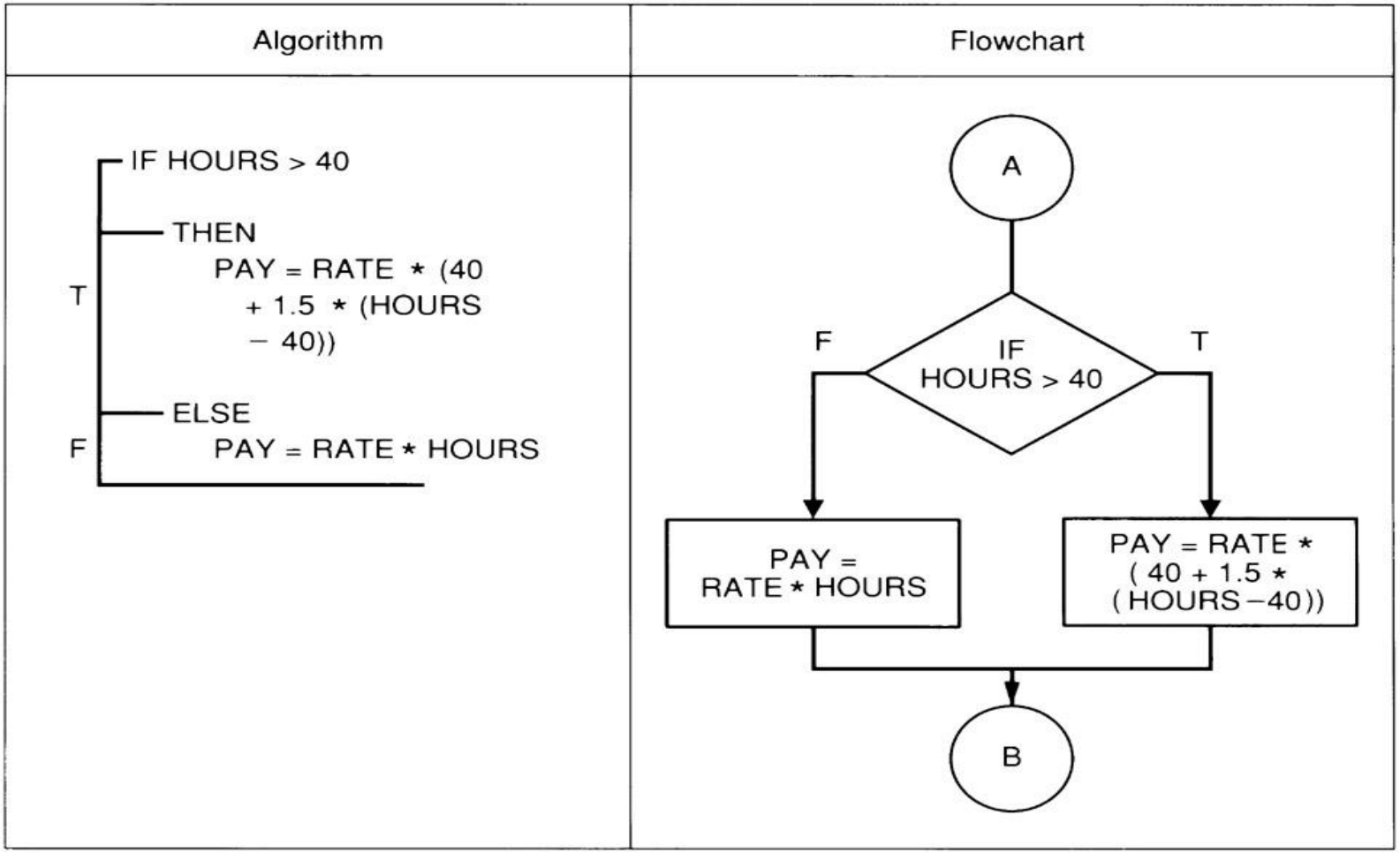
Example 7


START
Input
VALUE1,VALUE2
Y
is
VALUE1>VALUE2
N
MAX ← VALUE1
MAX ← VALUE2
Print
"The largest value is", MAX
STOP

Example

- Assume your are calculating pay at an hourly rate, and overtime pay(over 40 hours) at 1.5 times the hourly rate.
  - IF the hours are **greater** than 40, THEN the pay is calculated for **overtime**,
  - or ELSE the pay is calculated in the **usual way.**

Example Decision Structure



Note: For all flowcharts with decision blocks, T = TRUE and F = FALSE