

# Flow Chart/Pseudo Code

-Zubair uddin Shaikh

# PRE-PROGRAMMING PHASE

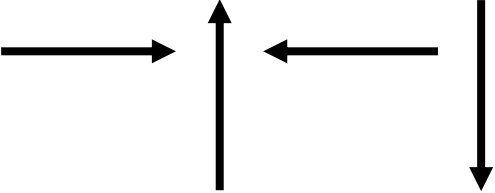
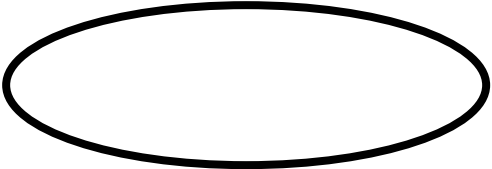

## ▶ Drawing the Program Flowcharts

- ▶ Flowchart is the **graphic** representations of the individual steps or actions to implement a particular module.
- ▶ The flowchart can be likened to the blueprint of a building. An architect draws a blueprint before beginning construction on a building, so the programmer draws a flowchart before writing a program.
- ▶ Flowchart is independent of any programming language.

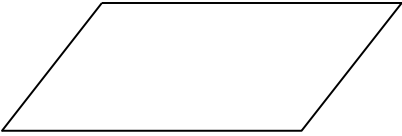
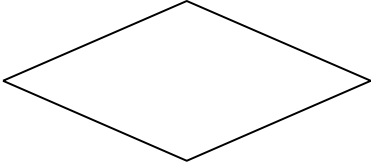

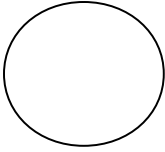
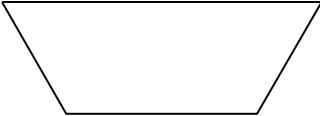
# PRE-PROGRAMMING PHASE

- ▶ Flowchart is the logical design of a program.
- ▶ It is the basis from which the actual program code is developed.
- ▶ Flowchart serves as documentation for computer program.
- ▶ The flowchart must be drawn according to definite rules and utilizes standard symbols adopted internationally.
- ▶ The International Organization for Standardization (IOS) was the symbols shown below (You can draw the symbols using ready-made flowcharting template):

# PRE-PROGRAMMING PHASE

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.

# PRE-PROGRAMMING PHASE

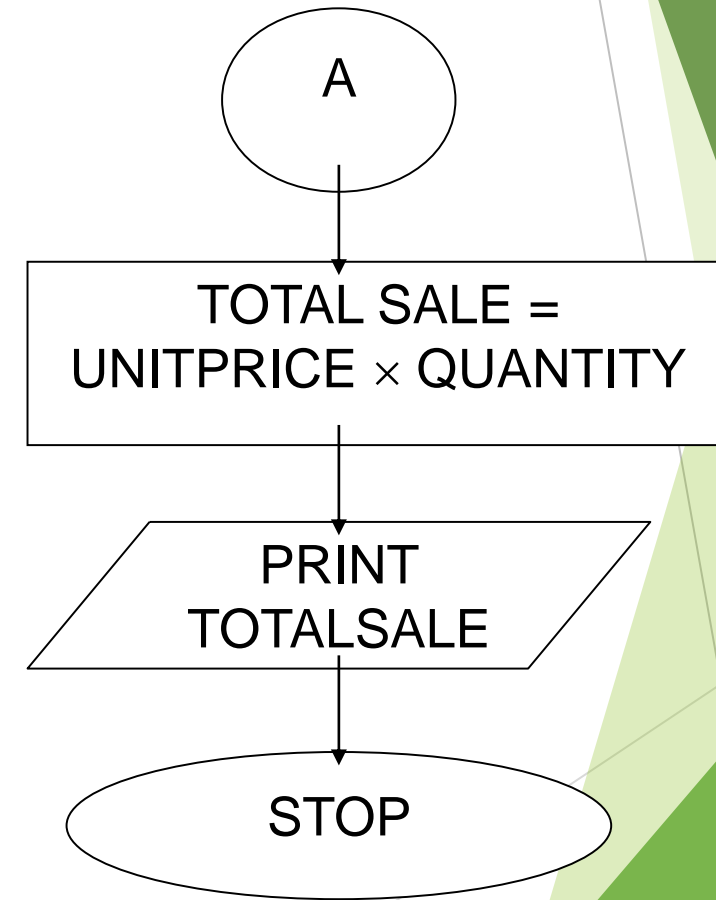
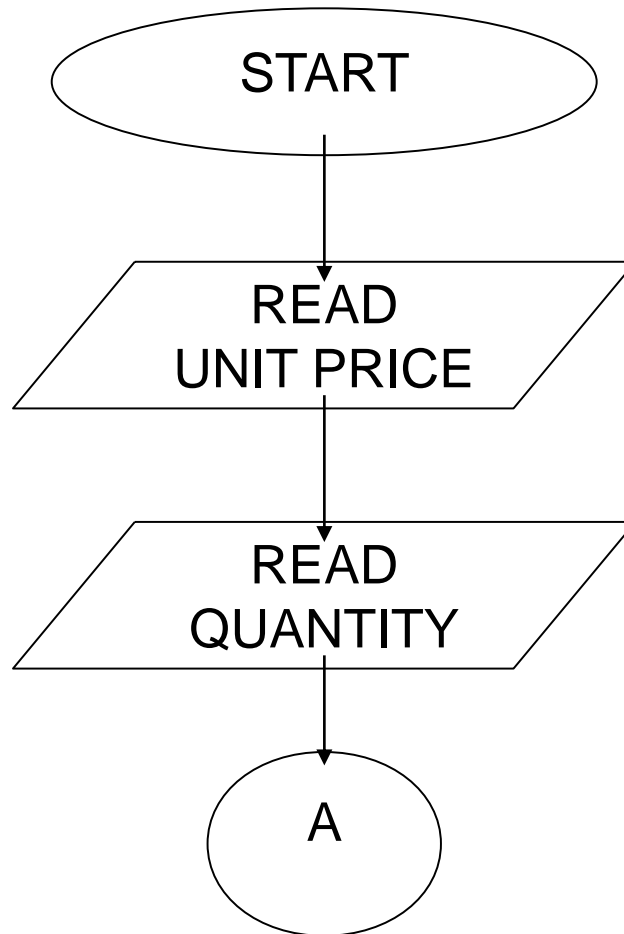
	Indicate input to the program and output from the program.
	Use for making decision. Either True or False based on certain condition.
	Use for doing a repetition or looping of certain steps.
	Connection of flowchart on the same page.
	Connection of flowchart from page to page.

# PRE-PROGRAMMING PHASE

## ► Example 2.3 : Sale Problem

- 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.
- Solution: Stepwise Analysis of the Sale Problem
  - Start of processing
  - Read the unit price
  - Read the quantity
  - Calculate total sale
  - Print total sale
  - Stop the processing

# PRE-PROGRAMMING PHASE



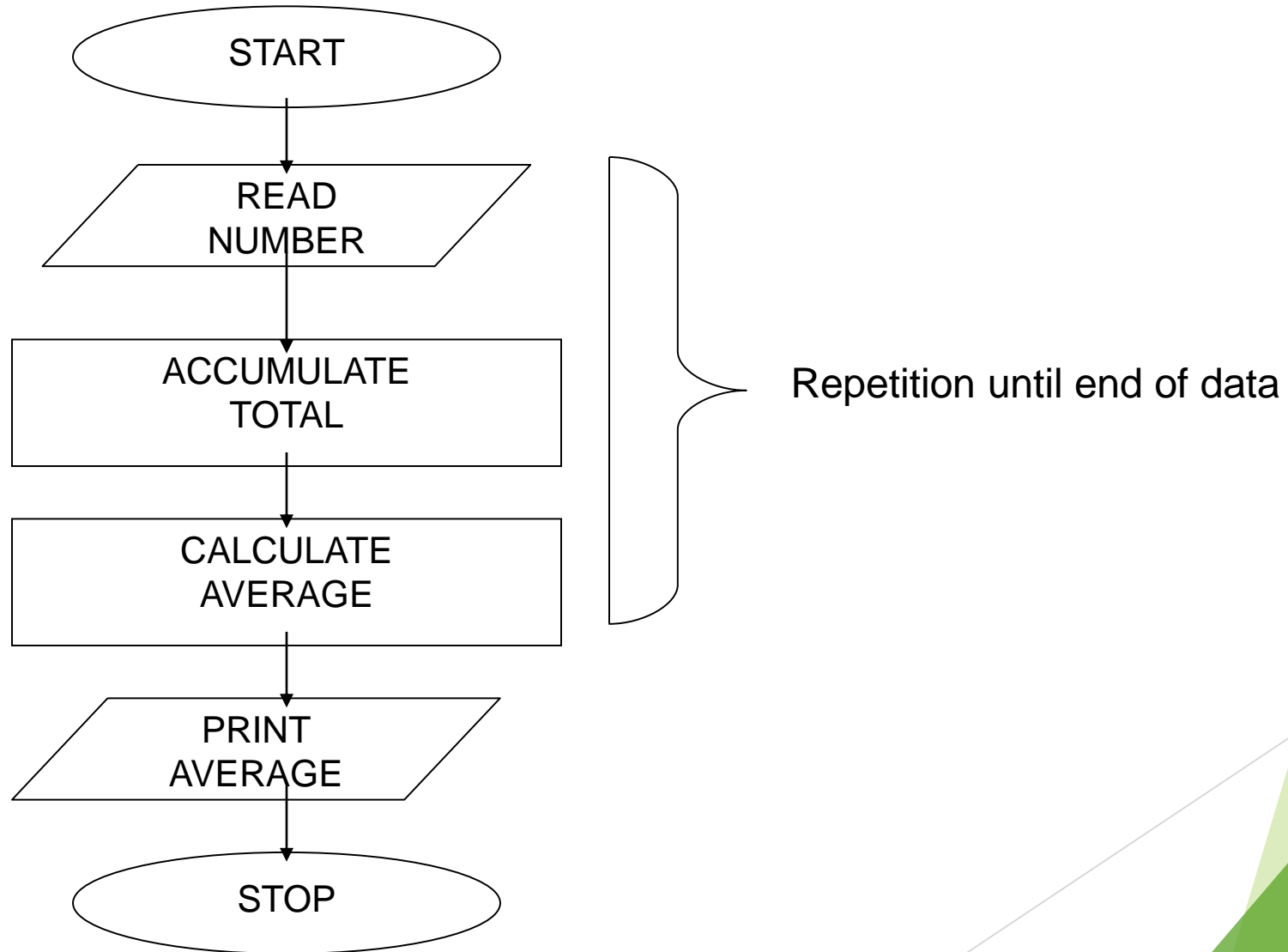
# PRE-PROGRAMMING PHASE

## ► Finding Average Problem

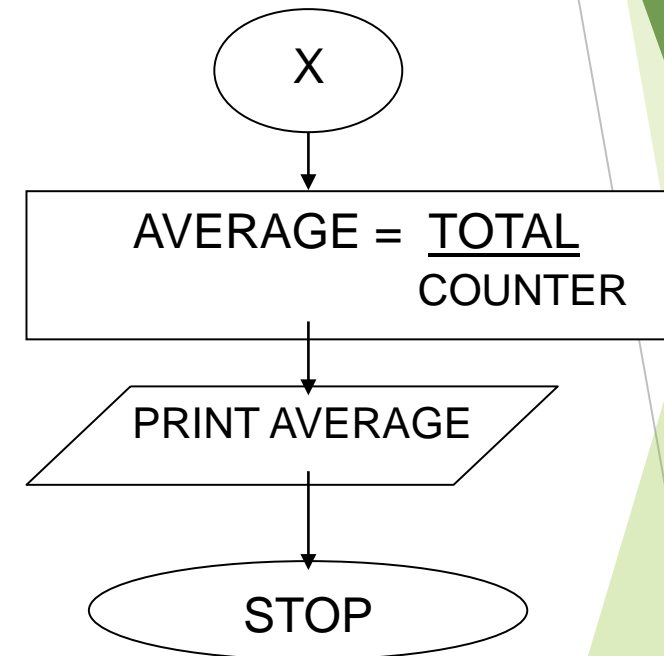
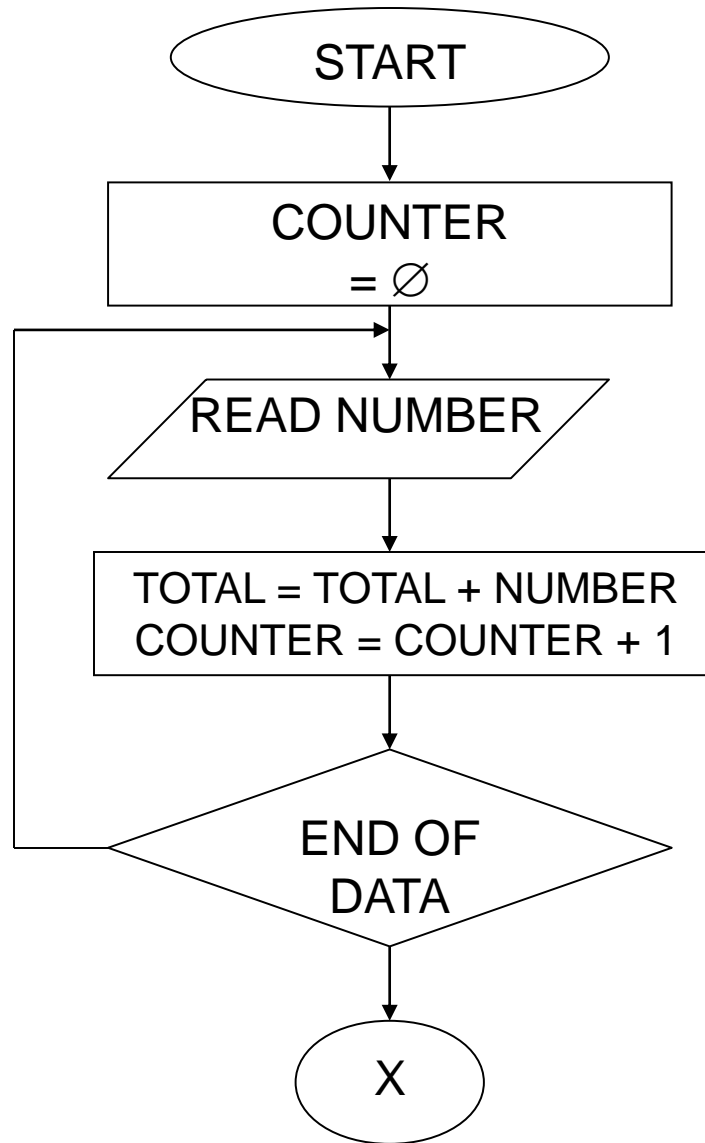
- Read a sequence of number, find the average of the number and print the average.
- Solution: Stepwise Analysis of Average Problem
  - Start the processing
  - Read a number
  - Add the number
  - Repeat reading until last data
  - Calculate the average
  - Print the average
  - Stop the processing

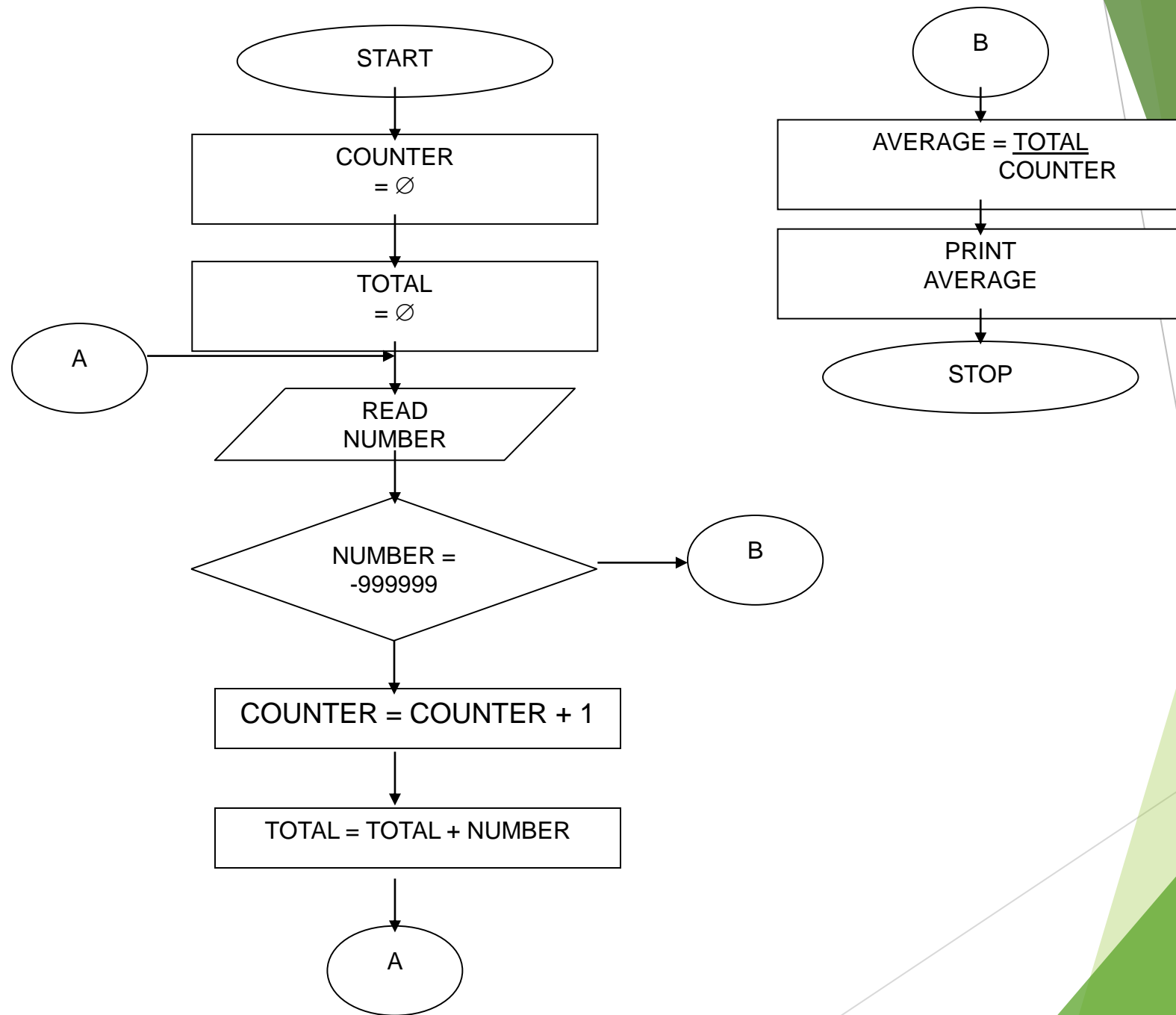


# PRE-PROGRAMMING PHASE



# PRE-PROGRAMMING PHASE

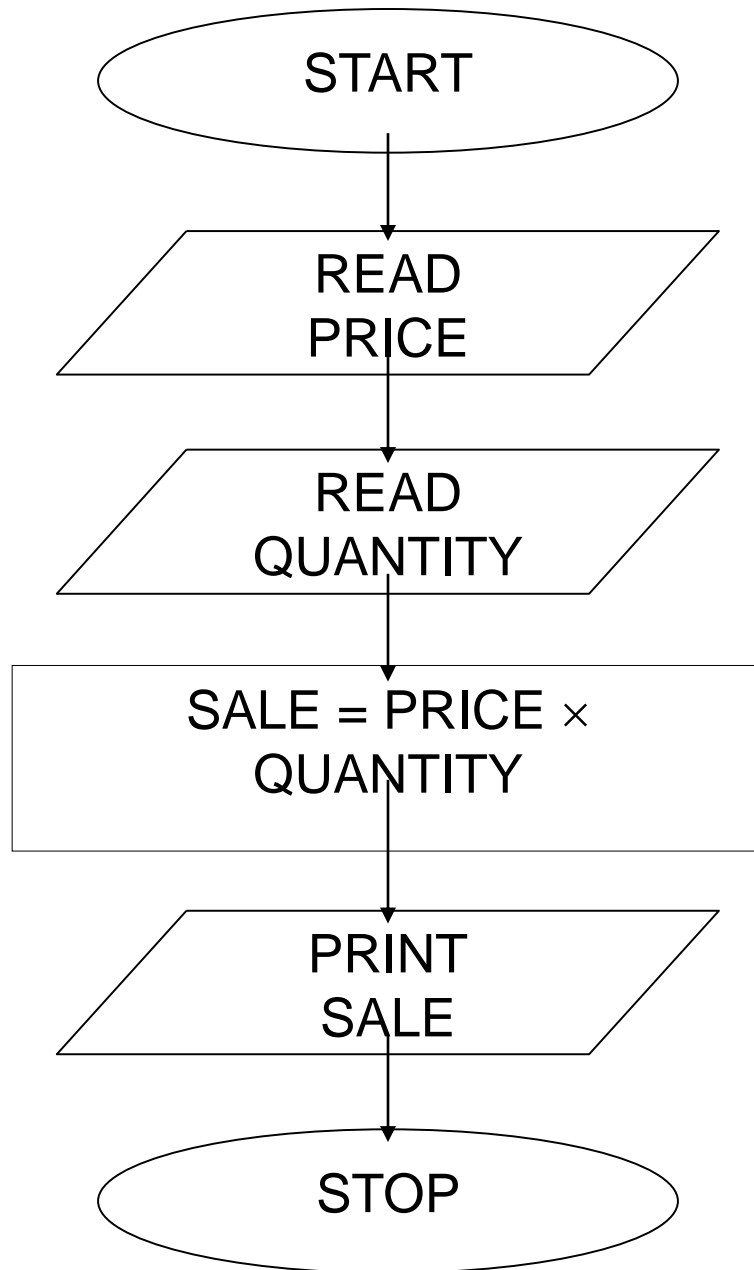




# PRE-PROGRAMMING PHASE

## ► Writing the Algorithm (Pseudocode)

- Pseudocode means an imitation computer code.
- It is used in place of symbols or a flowchart to describe the logic of a program. Thus, it is a set of instructions (descriptive form) to describe the logic of a program.
- Pseudocode is close to the actual programming language.
- Using the Pseudocode, the programmer can start to write the actual code.



Algorithm:

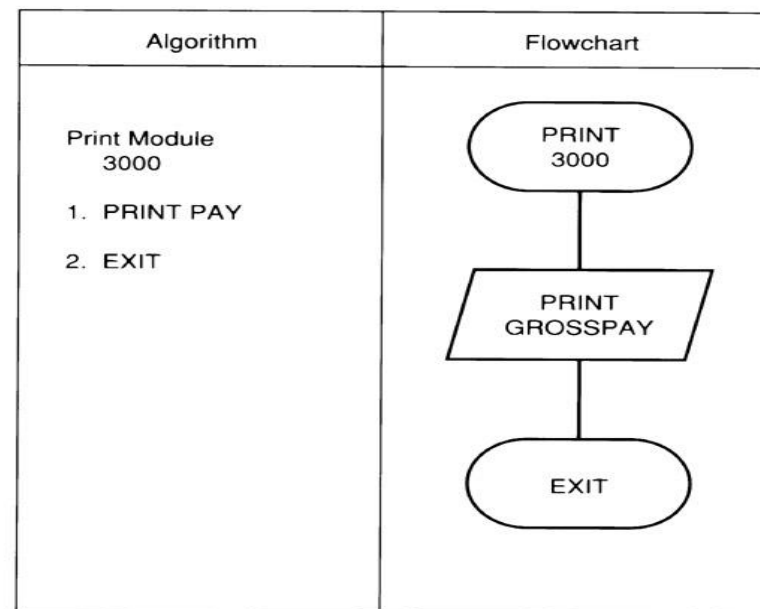
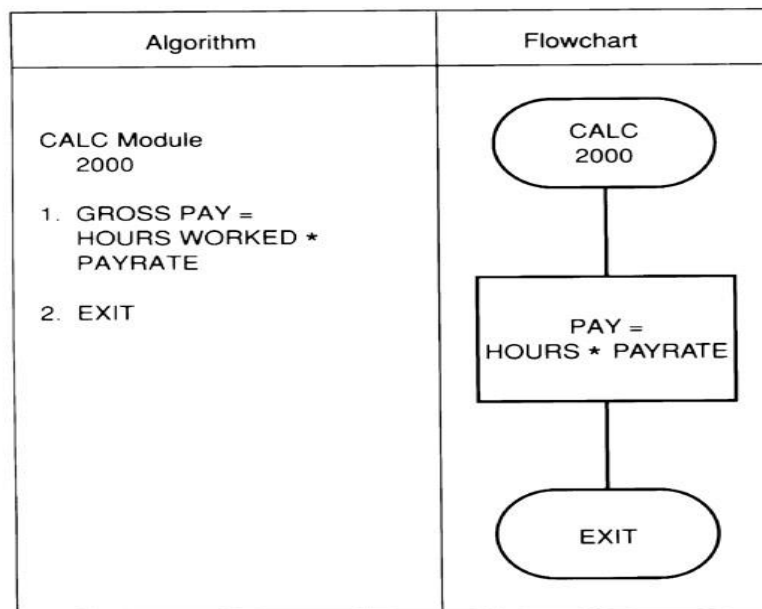
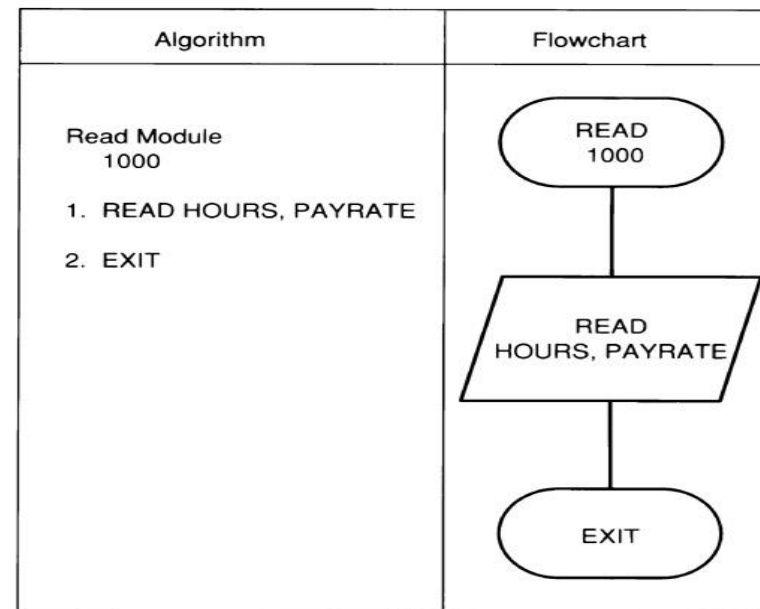
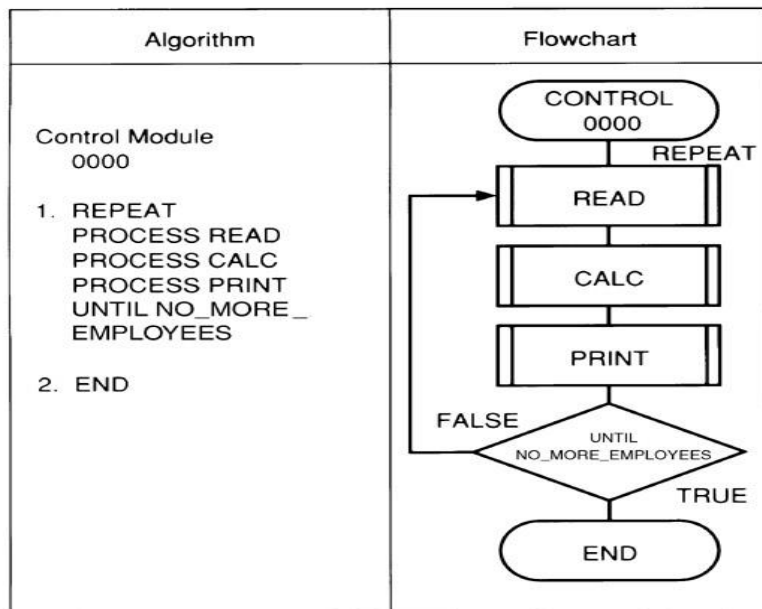
*Start*

*Read price, quantity*

*Sale = price x quantity*

*Print Sale*

*End*

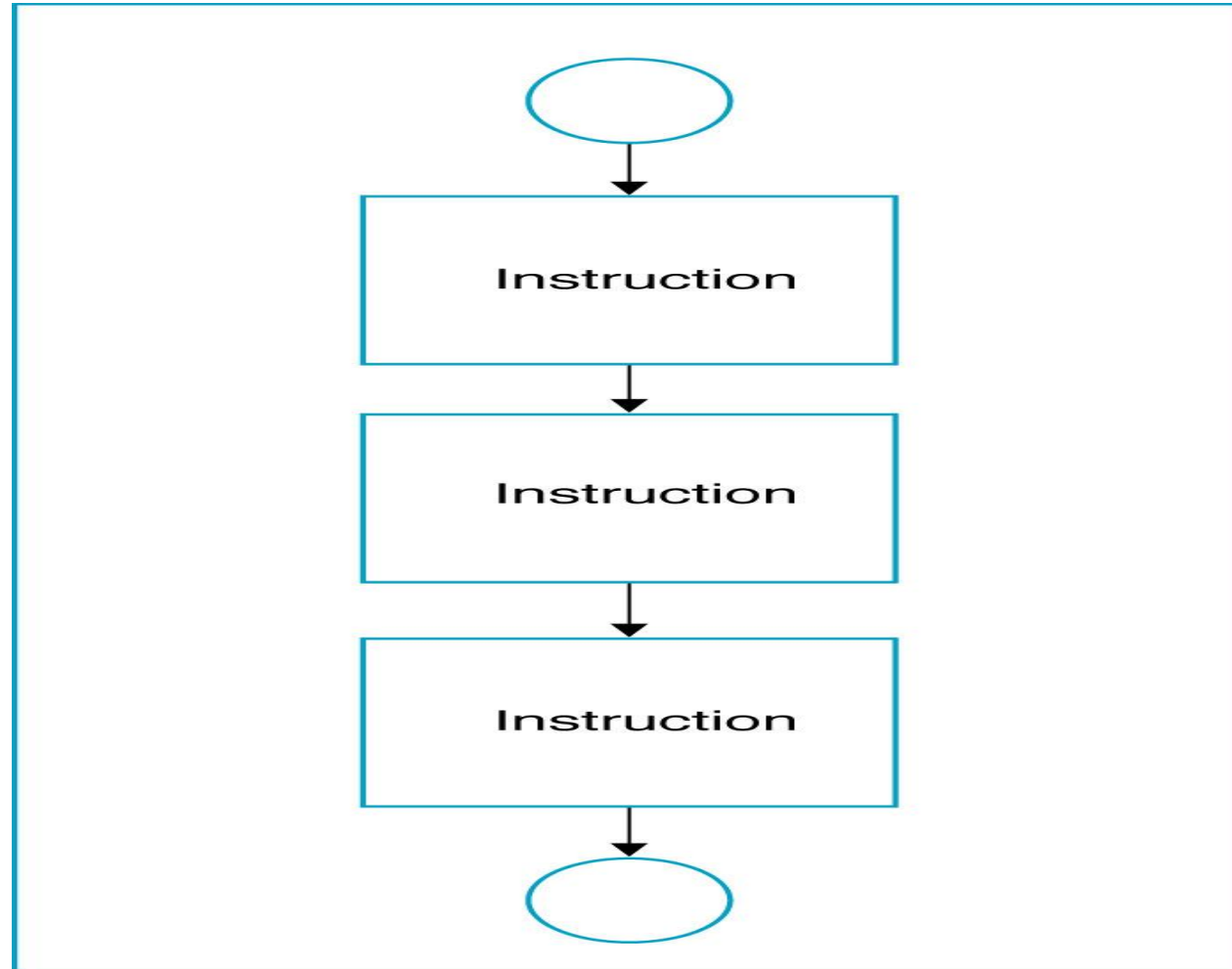


Example: Flowchart & Algorithm

# Structuring a Program

- ▶ Develop efficient computer solution to problems:
  1. Use Modules
  2. Use four logic structures
    - a. Sequential structure
      - Executes instructions one after another in a sequence.
    - b. Decision structure
      - Branches to execute one of two possible sets of instructions.
    - c. Loop structure
      - Executes set of instruction many times.
    - d. Case structure
      - Executes one set of instructions out of several sets.
  3. Eliminate rewriting of identical process by using modules.
  4. Use techniques to improve readability including four logic structure, proper naming of variables, internal documentation and proper indentation.

# Sequential Logic Structure

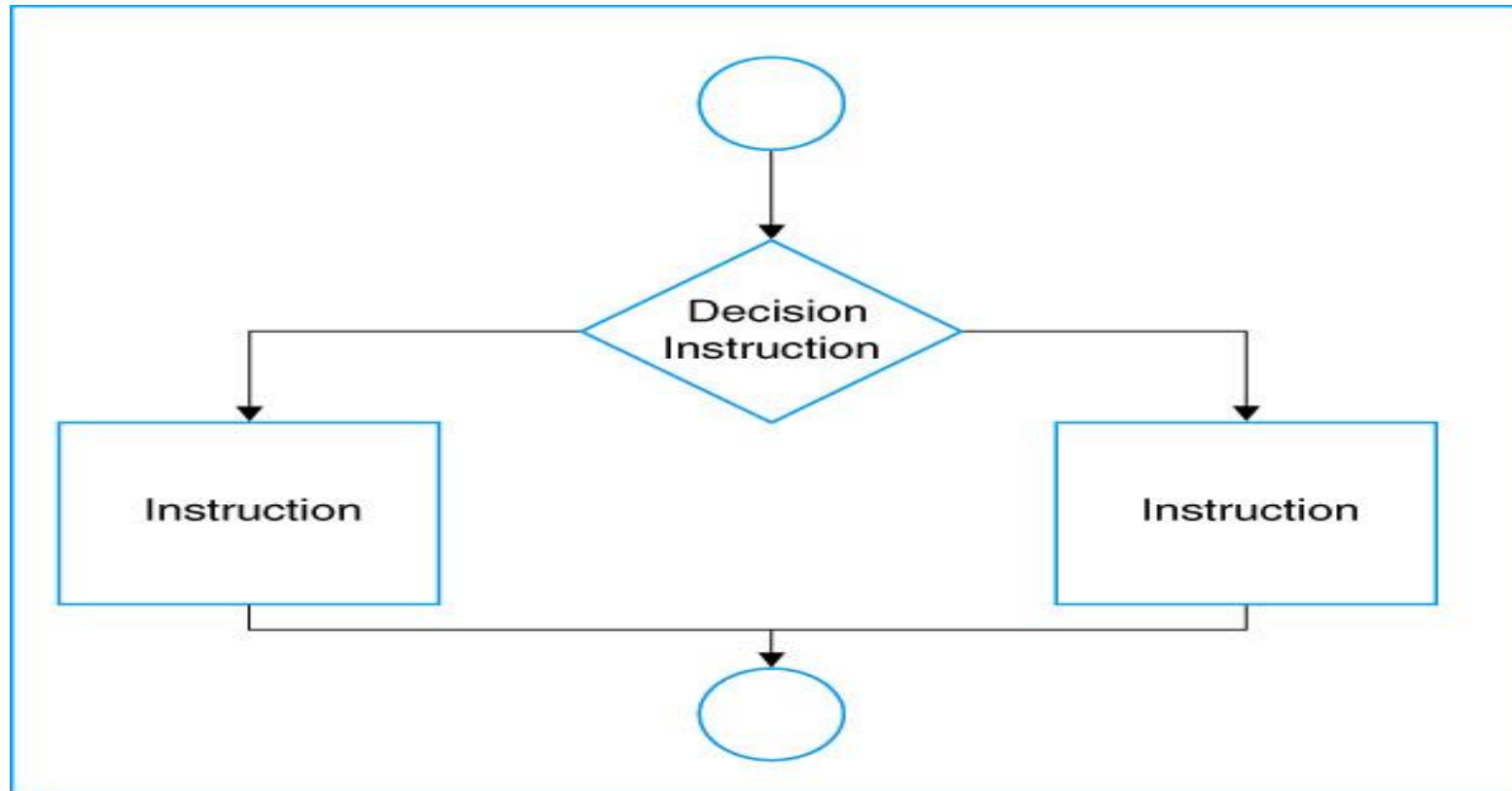




# The Decision Logic Structure

- ▶ Implements using the IF/THEN/ELSE instruction.
- ▶ Tells the computer that IF a condition is true, THEN execute a set of instructions, or ELSE execute another set of instructions
- ▶ ELSE part is optional, as there is not always a set of instructions if the conditions are false.
- ▶ Algorithm:  
    IF <condition(s)> THEN  
        <TRUE instruction(s)>  
    ELSE  
        <FALSE instruction(s)>

# Decision Logic Structure



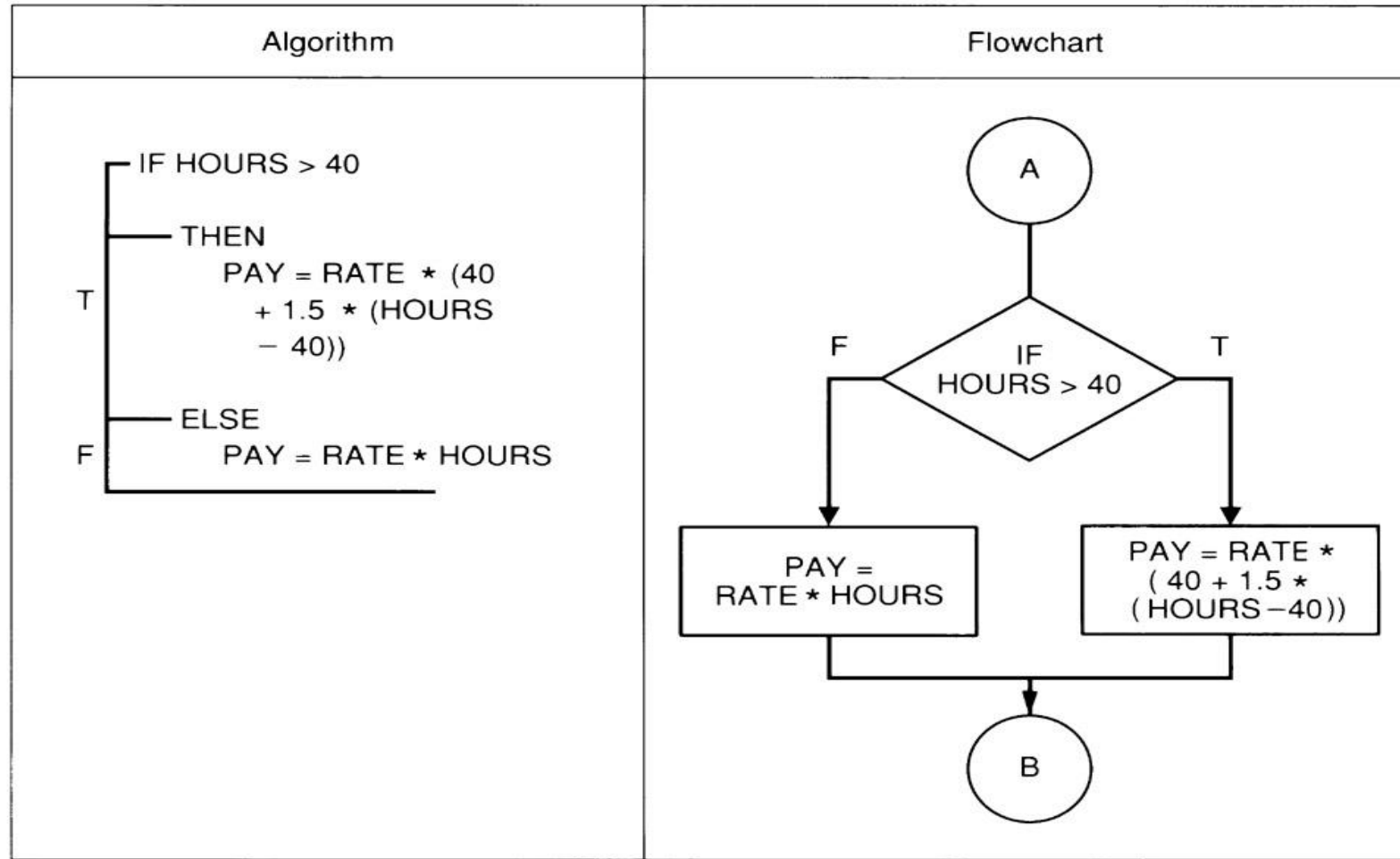
# Examples of conditional expressions

- ▶  $A < B$  (A and B are the same data type - either numeric, character, or string)
- ▶  $X + 5 \geq Z$  (X and Z are numeric data)
- ▶  $E < 5$  or  $F > 10$  (E and F are numeric data)
- ▶ DATAOK (DATAOK - logical datum)

# Example

- ▶ Assume you 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

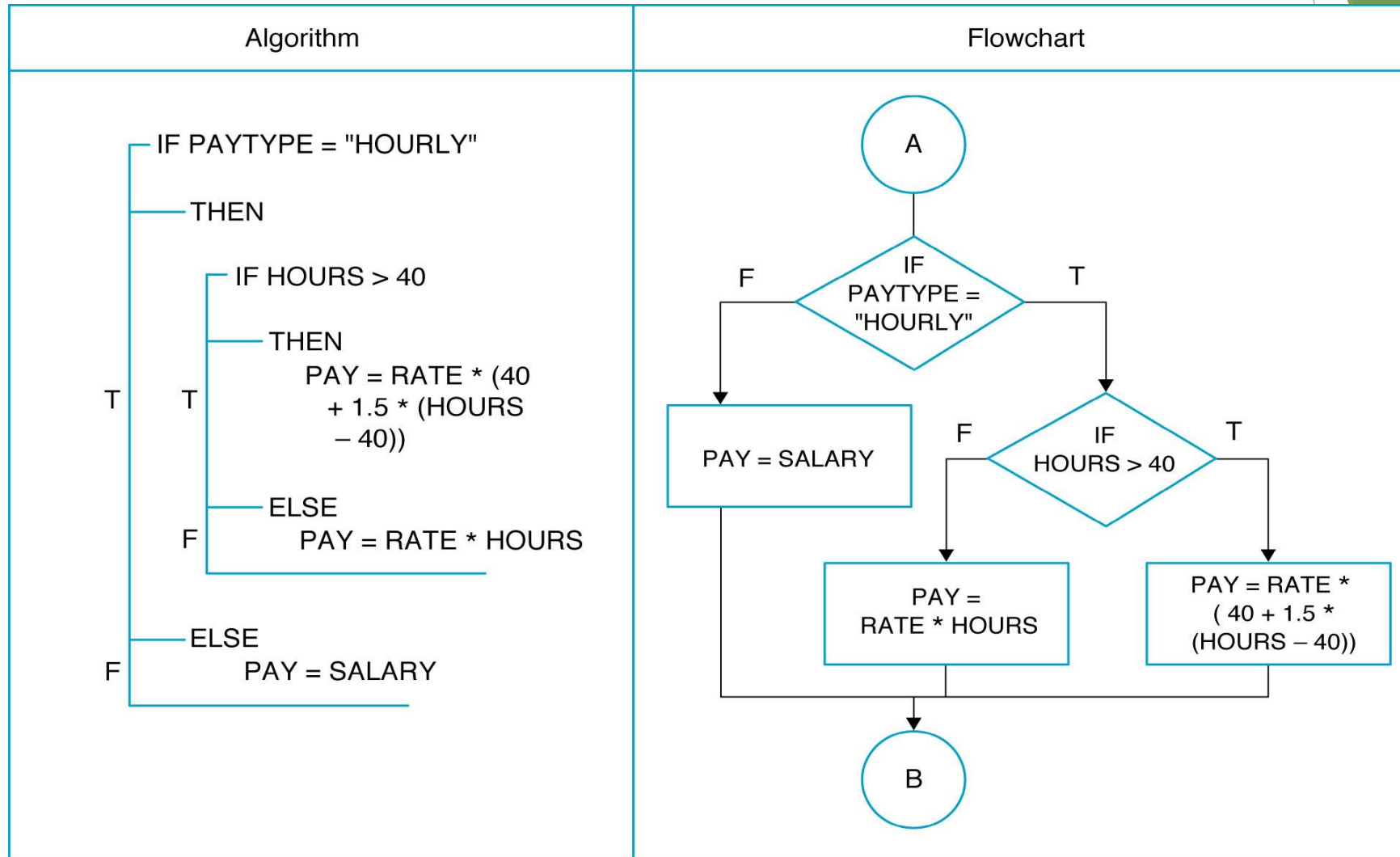


Note: For all flowcharts with decision blocks, T = TRUE and F = FALSE

# NESTED IF/THEN/ELSE INSTRUCTIONS

- ▶ Multiple decisions.
- ▶ Instructions are sets of instruction in which each level of a decision is embedded in a level before it.

# NESTED IF/THEN/ELSE INSTRUCTIONS



# EXERCISE

A hotel has a pricing policy as follows:

- 2 people : RM85
- 3 people : RM90
- 4 people : RM95
- Additional people : RM6 per person

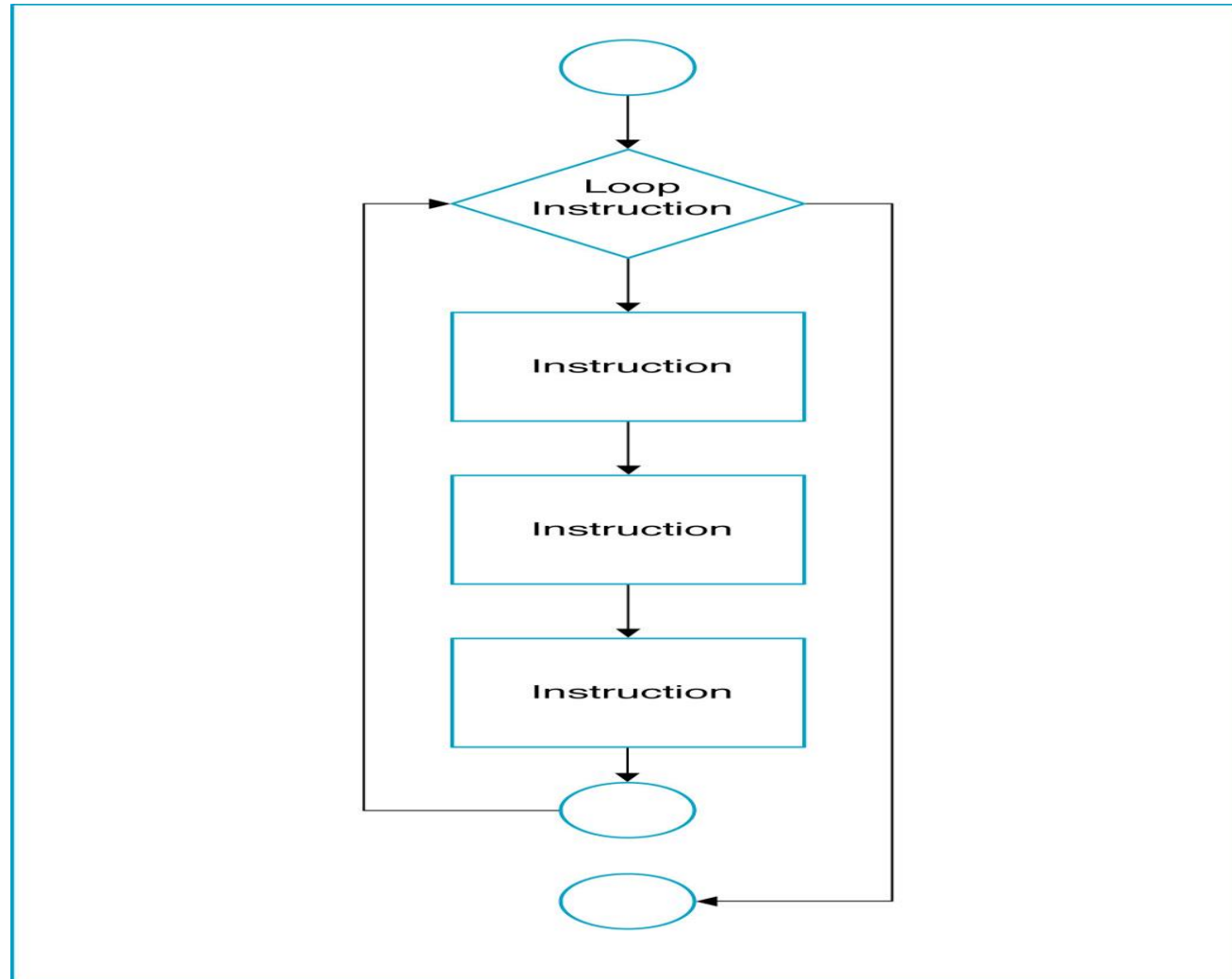
If the customer is staying on company business, there is a 20% discount. If the customer is over 60 years of age, there is a 15% discount. A customer does not receive both discount. Given the above data, print the cost of the room



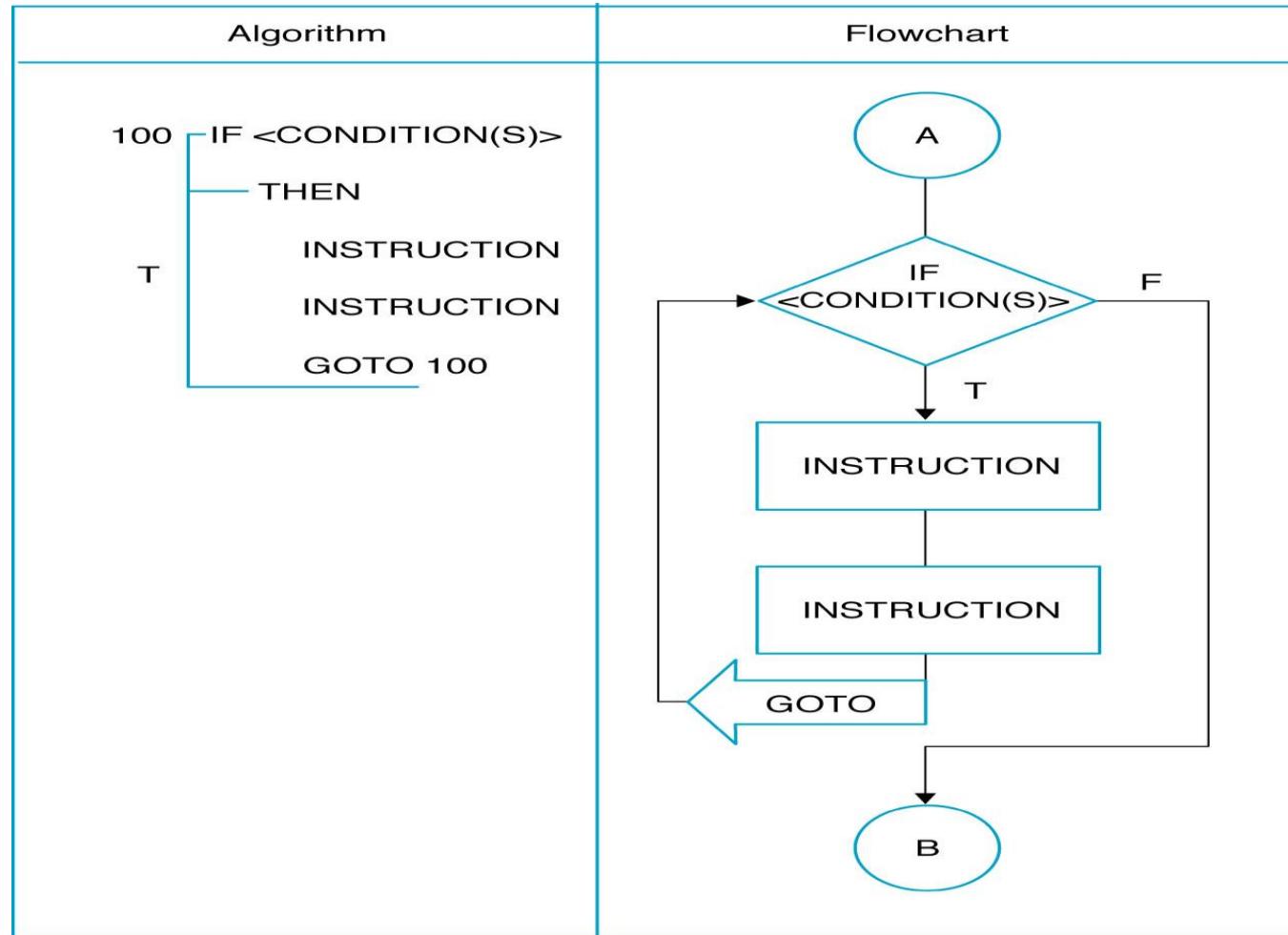
# The Loop Logic Structure

- ▶ Repeat structure
- ▶ To solve the problem that doing the same task over and over for different sets of data
- ▶ Types of loop:
  - ▶ WHILE loop
  - ▶ Do..WHILE loop
  - ▶ Automatic-Counter Loop

# Loop Logic Structure



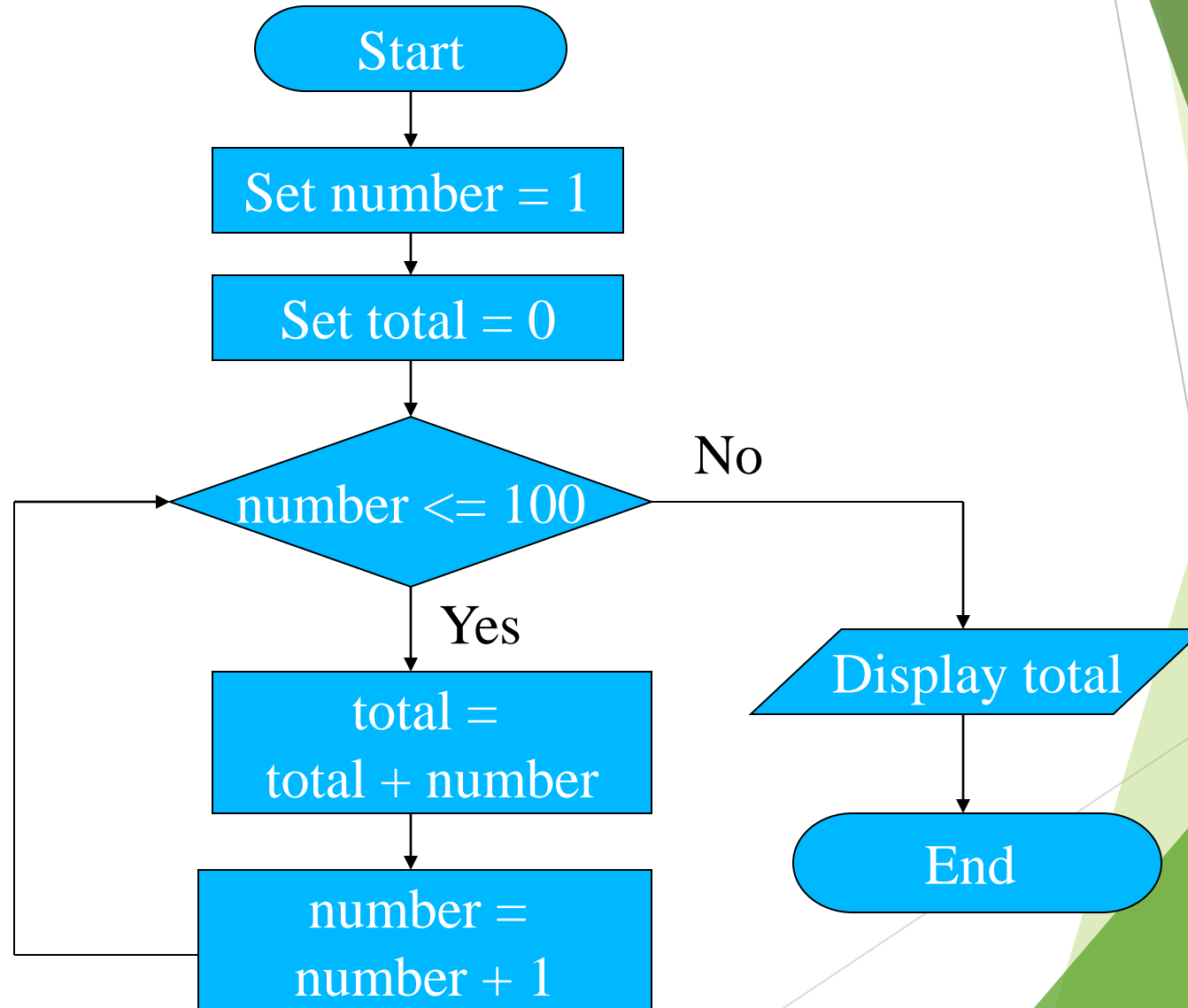
# WHILE loop



# WHILE loop

- ▶ Do the loop body if the condition is true.
- ▶ Example: Get the sum of 1, 2, 3, ..., 100.
  - ▶ Algorithm:
    - ▶ Set the number = 1
    - ▶ Set the total = 0
    - ▶ While (number <= 100)
      - ▶ total = total + number
      - ▶ number = number + 1
    - ▶ End While
    - ▶ Display total

# WHILE loop



# DO...WHILE Loop

- ▶ The body of the loop will process first before check the condition.
- ▶ Example: Get the sum of 1, 2, 3, ...100.

## Algorithm:

Set the number = 1

Set the total = 0

Do

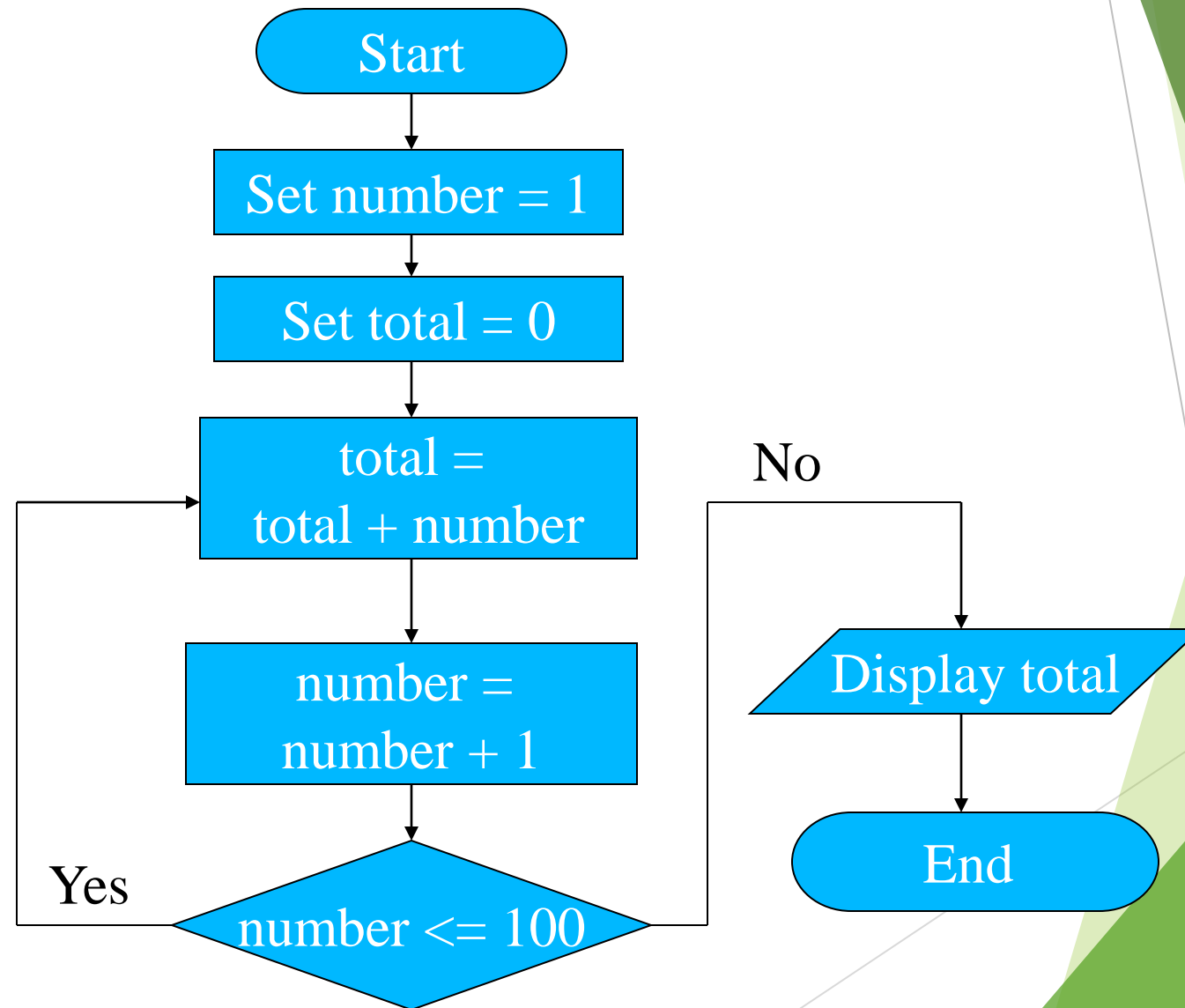
total = total + number

number = number + 1

While (number <= 100)

Display total

# DO...WHILE Loop

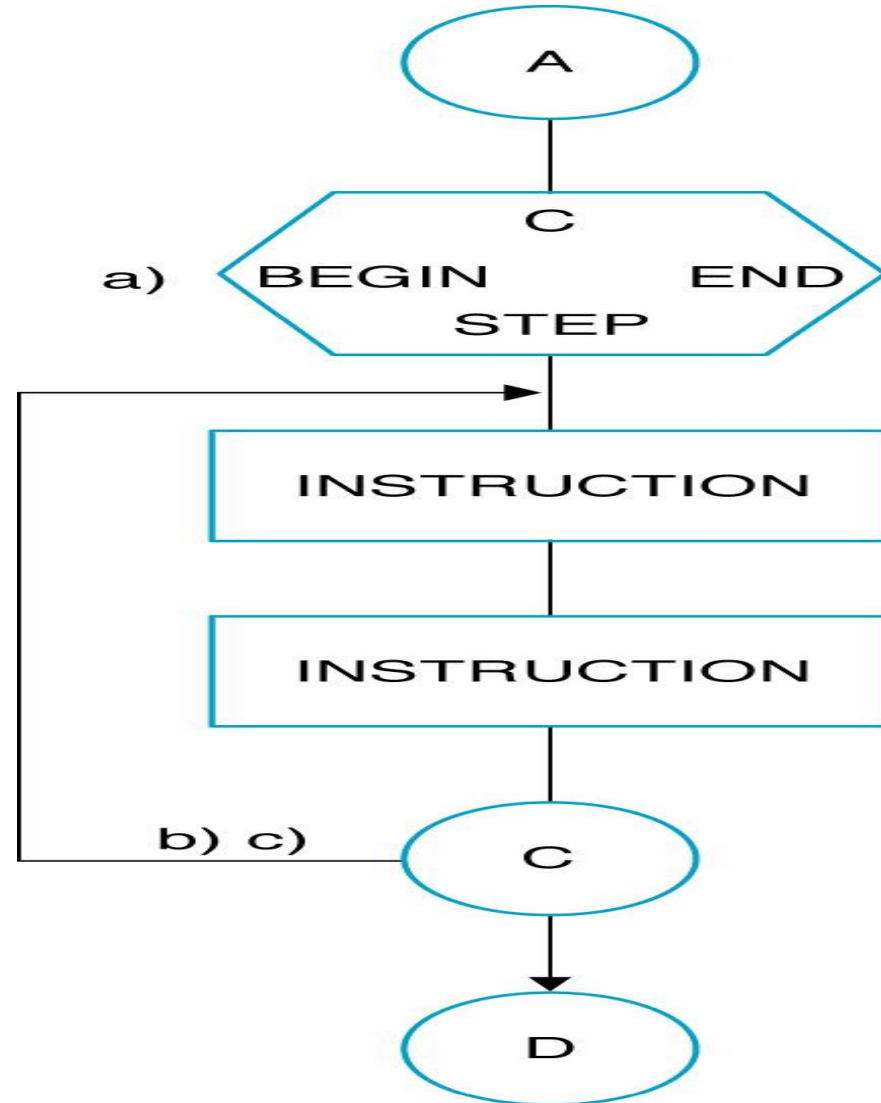


# Automatic Counter Loop

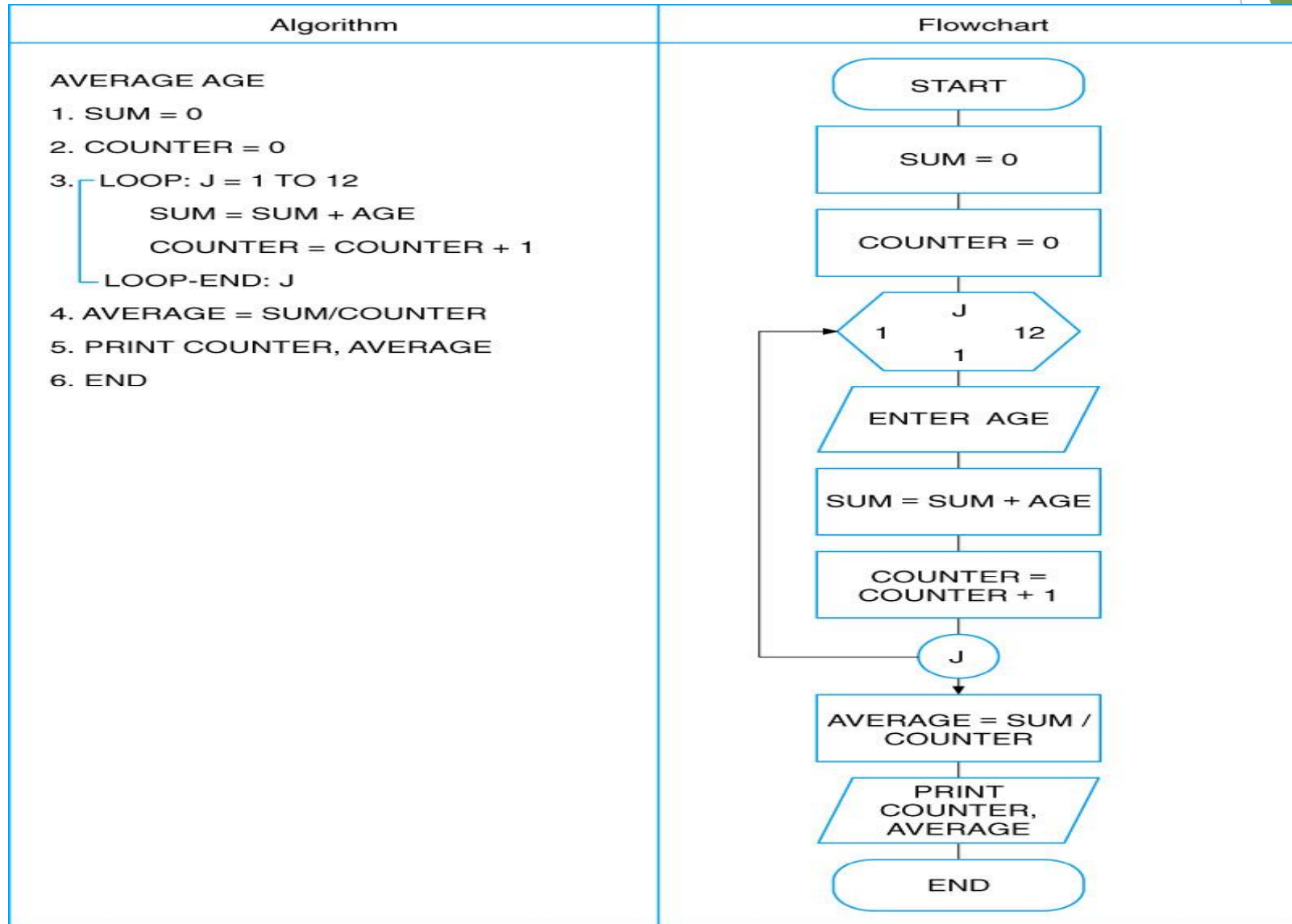
- ▶ Use variable as a counter that starts counting at a specified number and increments the variable each time the loop is processed.
- ▶ The beginning value, the ending value and the increment value may be constant. They should not be changed during the processing of the instruction in the loop.



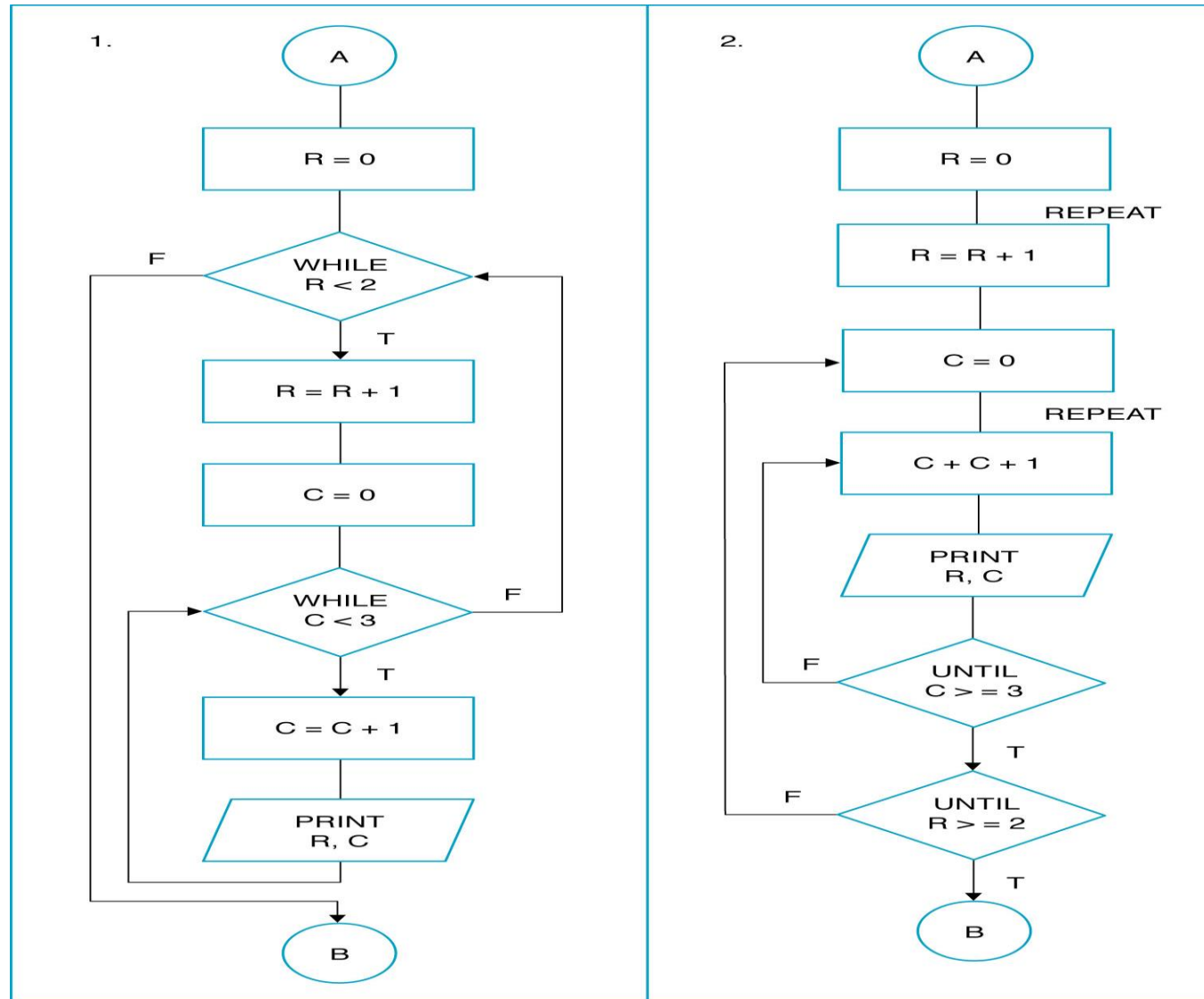
# Automatic-Counter Loop



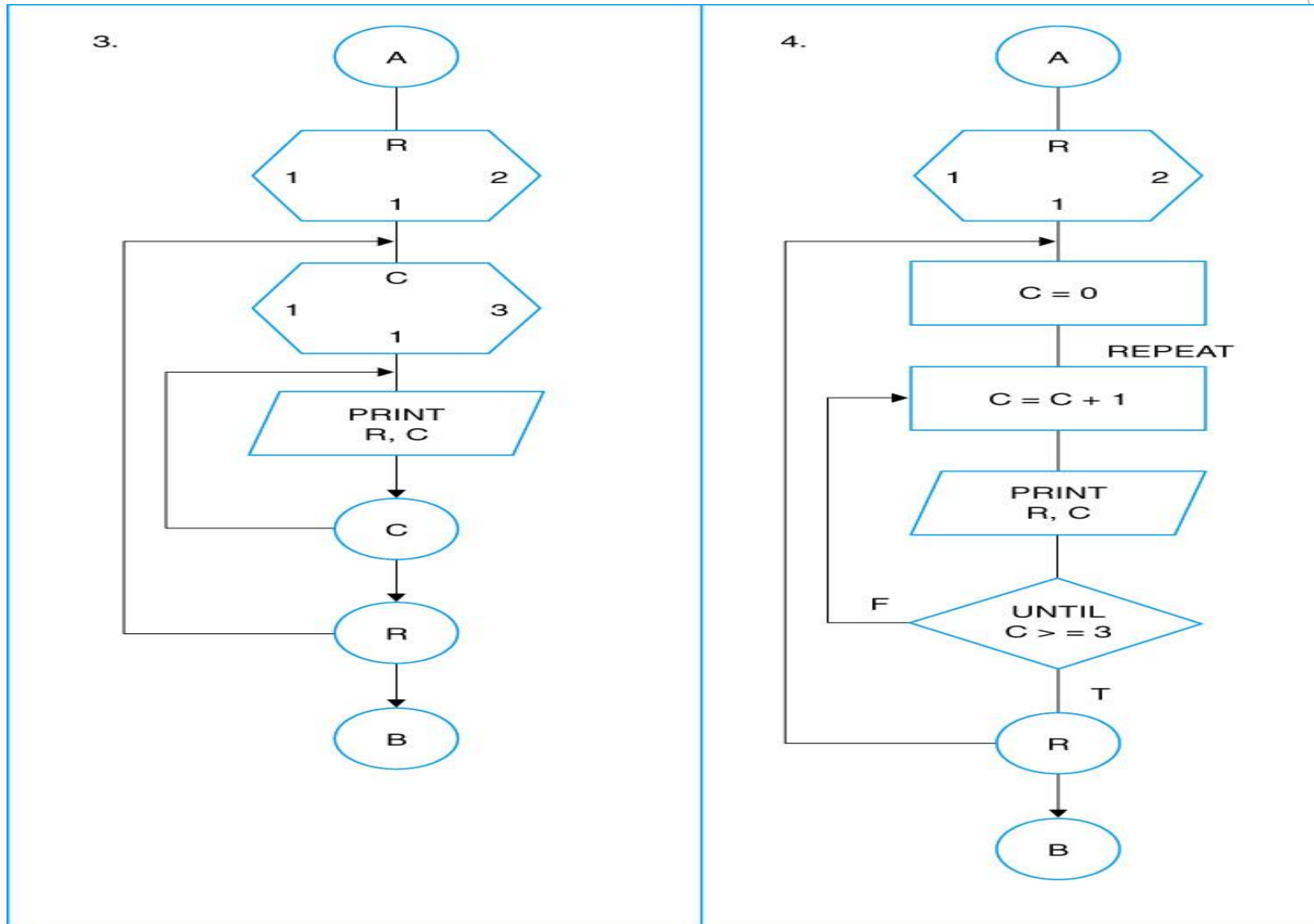
# Automatic-Counter Loop



# NESTED LOOP



# NESTED LOOP



# EXAMPLE OF NESTED LOOP

An algorithm to find total marks of test1, test2 and final and also average of the marks for each student in class SAK3100 Group 2.

```
LOOP: J = 1 TO 35
    SET JUMLAH = 0
    LOOP: K= 1 TO 3
        SET TEST = 0
        PRINT "MASUKKAN MARKAH TEST", K
        READ TEST
        JUMLAH = JUMLAH + TEST
    LOOP-END: K
    PURATA = JUMLAH / 3
    PRINT "MARKAH PELAJAR KE ", J, " = ", JUMLAH, "PURATA MARKAH = ", PURATA
LOOP-END: J
PRINT "ITU JUMLAH DAN PURATA MARKAH SEMUA PELAJAR DALAM KELAS SAK3100 K2"
END
```

# The Case Logic Structure

- ▶ Made up of several or many sets of instructions, only one of which will be selected by the user and executed by the computer

- ▶ Algorithm:

CASE OF VARIABLE

= constant1:

actions for VARIABLE = constant1

= constants2:

actions for VARIABLE = constant2

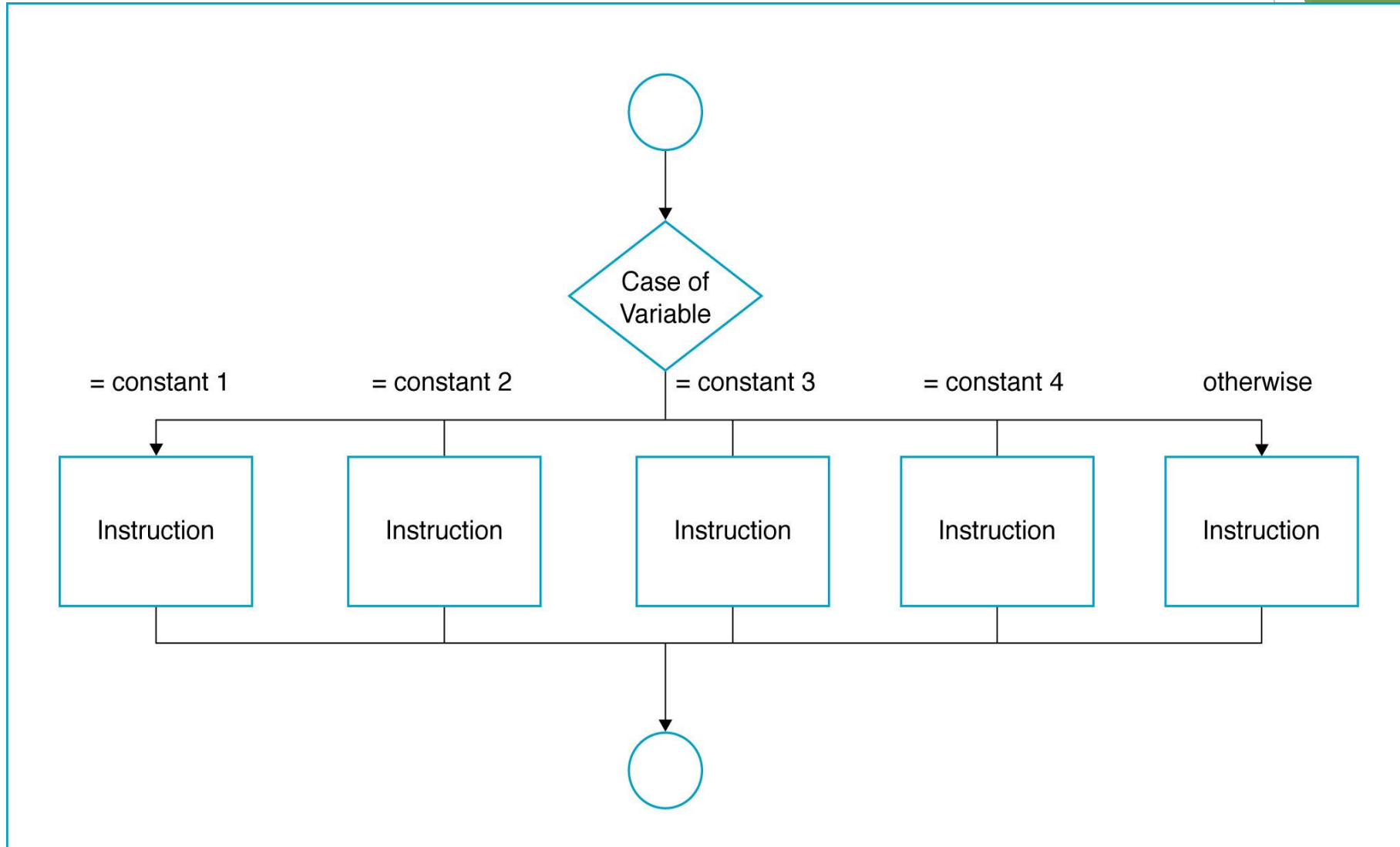
...

OTHERWISE:

Actions for VARIABLE = anything else

END-OF-CASE

# Case Logic Structure



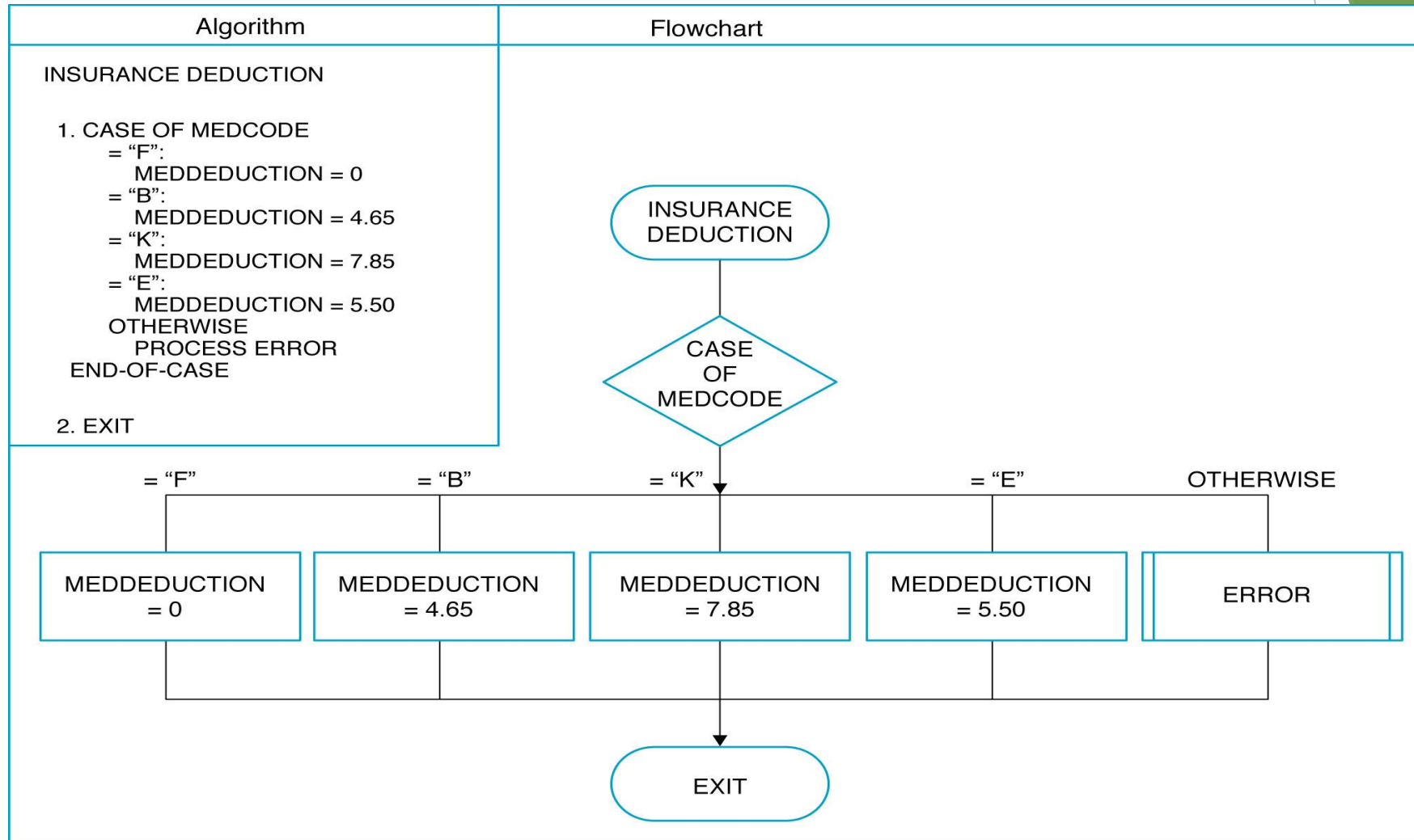
# Case Logic Structure

- Example: A company has four different medical plans. The programmer has given each plan a code corresponding to the beginning initial of the company: Plan 1 = F, Plan 2 = B, Plan 3 = K, Plan 4 = E.

The company pays for all of Plan 1. The individual has to pay for part of the others. The payroll deduction for Plan 2 = 4.65, for Plan 3 = 7.85, and for Plan 4 = 5.50. Any other codes are considered in error. Write the algorithm and draw the flowchart for a module to determine the payroll deduction.



# Example of Case Logic Structure



# Case Logic Structure

A module calculate an employee's pay as shown below. It has pay-type code and the pay rate to calculate the employee's pay.

H = hourly       $\text{pay} = \text{rate} * \text{hours}$

P = Piece work       $\text{pay} = \text{rate} * \text{number of pieces}$

C = commission       $\text{pay} = \text{commission} * \text{sales}$

S = salary       $\text{pay} = \text{salary}$