

FLOWCHARTS

• Problem Solving:

-A **problem** can be defined as a difference between a desired situation and current situation.

-A **problem solution** is a procedure, or method, for transforming the current situation to the desired one.

-In computer science, an **algorithm**, explicitly states the steps leading to the solution, which must be transmitted to the computer.

-In the problem-solving phase of computer programming, the **algorithms** can be designed through the use of **flowchart** or **pseudo-code**.

• Flowchart:

-A flowchart provides a **visual** representation of the patterns the algorithm follows in order to show the logic of program.

-It is a **diagram** made up of boxes, diamonds and other shapes, connected by arrows.

-Each **shape** represents a step in the process, while the **arrows** show the order in which they occur.



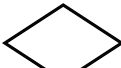

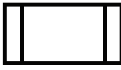

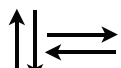
-Flowcharting **combines** symbols and flow lines in order to show figuratively the operation of an algorithm.

-Generally drawn in **early stages** of formulating computer solutions.

-Facilitate **communication** between programmers and business people.

-Helpful in **understanding** the logic of complicated and lengthy problems.

• Flowchart Symbols:

Symbol	Name	Function
	Process	Indicates any type of internal operation inside the Processor or Memory
	Input/Output	Used for any Input/Output (I/O) operation. Indicates that the computer is to obtain data or output results
	Decision	Used to ask a question that can be answered in a binary format (Yes/No, True/False)
	Connector	Allows the flowchart to be drawn without intersecting lines or without a reverse flow
	Predefined Process	Used to invoke a subroutine
	Terminal	Indicates the starting or ending of the program process, or interrupt program
	Flow Lines	Shows direction of flow

• Flowchart General Rules:

1- All flow charts start with a terminal.

2- All boxes of the flowchart are connected with Arrows (not lines).

3- Flowchart symbols have an entry point on the top of the symbol with no other entry points.

4- The exit point for all flowchart symbols is on the bottom except for the decision symbol.

5- The decision symbol has two exit points; these can be on the sides or the bottom and one side.

6- A flowchart flows from top to bottom. However, an upward flow can be shown as long as it does not exceed 3 symbols.

7- Connectors are used to connect breaks in the flowchart. Examples are:

- From one page to another page.
- From the bottom of the page to top of the same page.
- An upward flow of more than 3 symbols.

8- Subroutines (predefined processes) have their own and independent flowcharts.

9- All flowcharts end with a terminal.

• **Design Structures:**

- Sequence:** One statement is executed after another
- Selection/Decision:** Statements can be executed or skipped depending on whether a condition evaluates to True or False
- Repetition:** Statements are executed repeatedly until a condition evaluates to True or False

• **Keywords:**

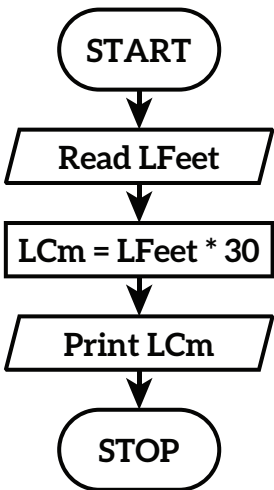
- Input:** Read / Get / Accept
- Initialize:** Set / Init
- Declare:** Declare
- Processing:** Compute / Calculate / Set / Increment
- Output:** Print / Display / Show

• **Relational Operators:**

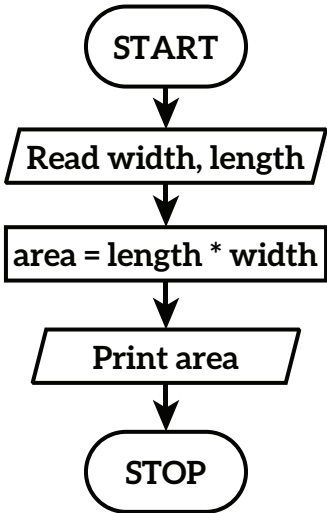
Operator	Description
>	Greater than
<	Less than
=	Equal to
>=	Greater than or equal to
<=	Less than or equal to
<>	Not equal to

1] **Sequence Structure:**

+ **Example (1) :** Draw a flowchart to convert the length in feet to centimeter.



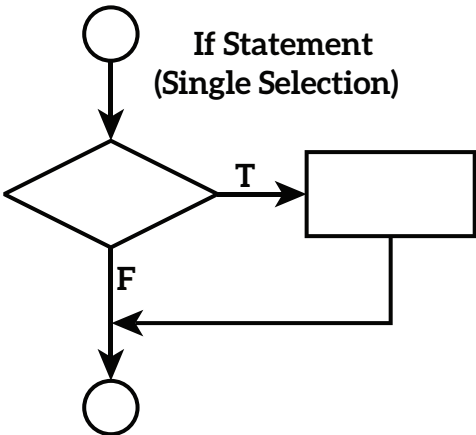
+ **Example (2) :** Draw a flowchart that will read the two sides of a rectangle and calculate its area



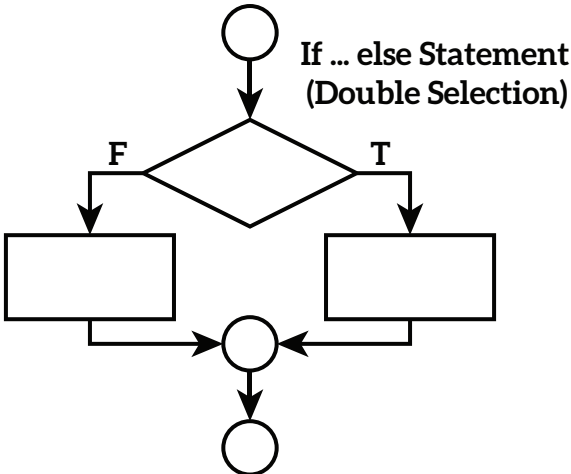
2] **Selection Structure:**

(A) **IF-THEN Structure:**

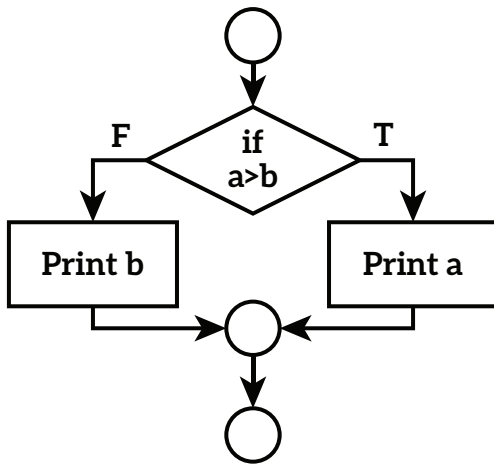
The IF-THEN statement allows us to choose whether to execute a set of program statements based on a test.



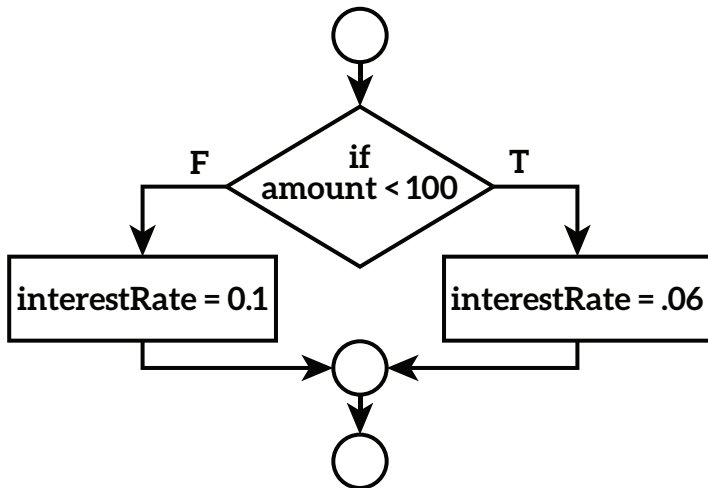
(B) **IF-THEN-ELSE Structure:**



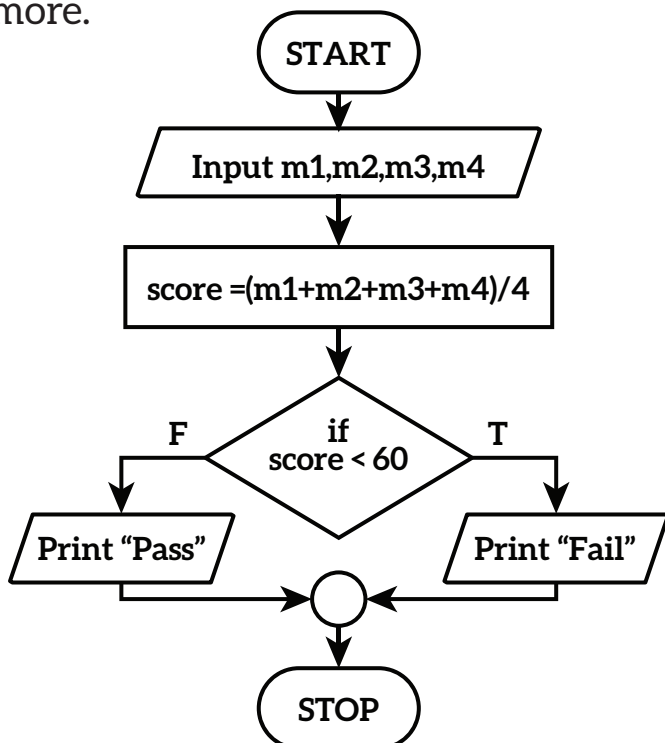
+ **Example (1)** : An algorithm that prints the larger of two numbers.



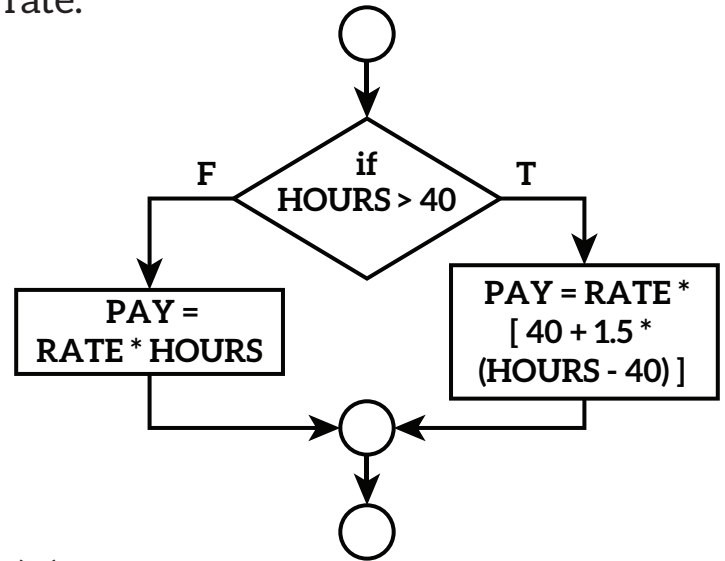
+ **Example (2)** : An algorithm that calculates the interest rate for the user according to his balance.



+ **Example (3)** : Student passes if the average score of his four subjects 60 or more.



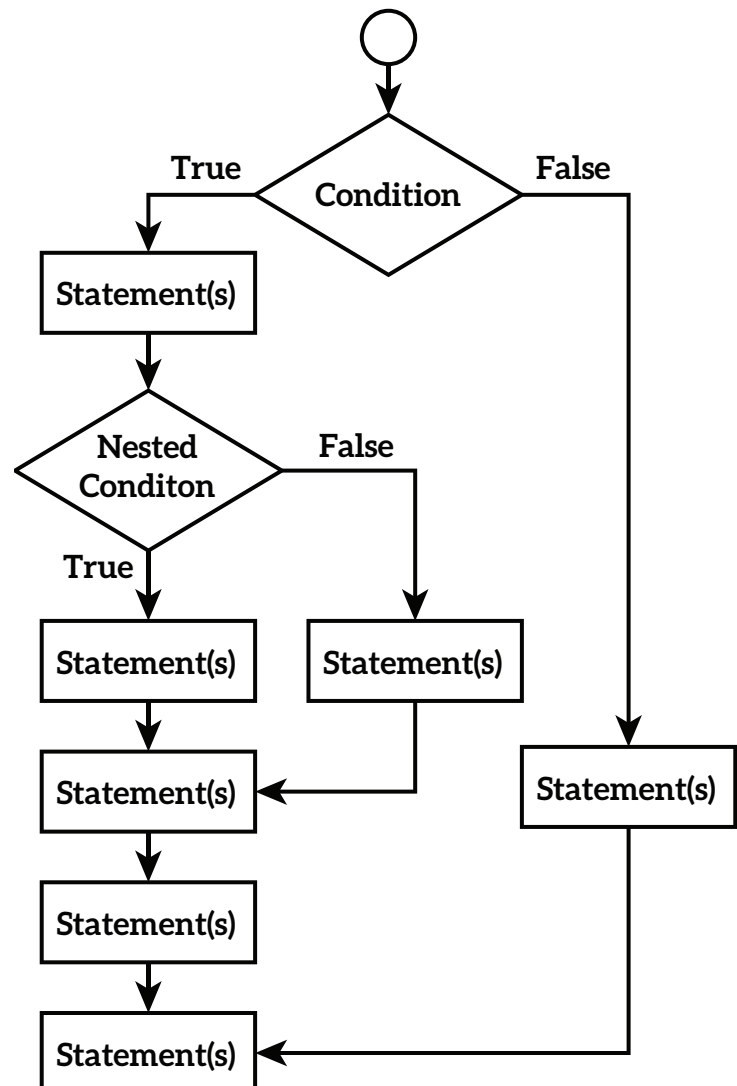
+ **Example (4)** : Calculate the employees pay at an hourly rate, and overtime pay (over 40 hours) at 1.5 times the hourly rate.



(C) Nested IF:

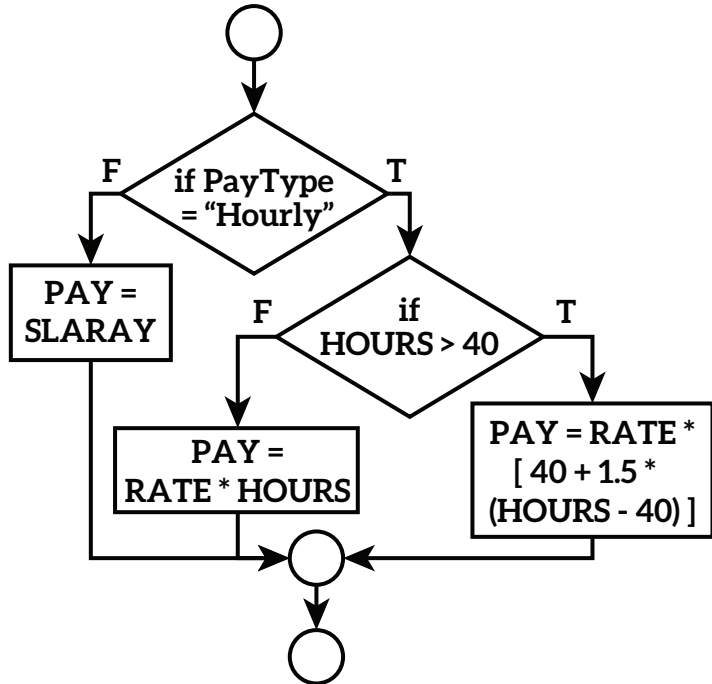
-Multiple decisions.

-Instructions are sets of instructions in which each level of a decision is embedded in a level before it.

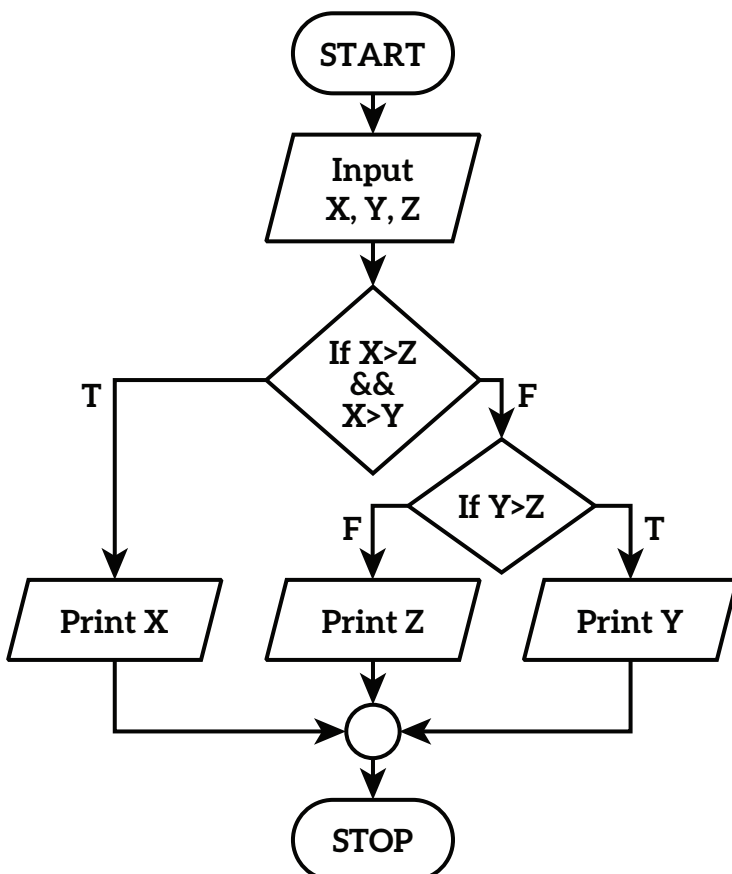


+ Example (1): - A company has two payment policies. Some employees have fixed monthly salary, while some are paid on an hourly basis.

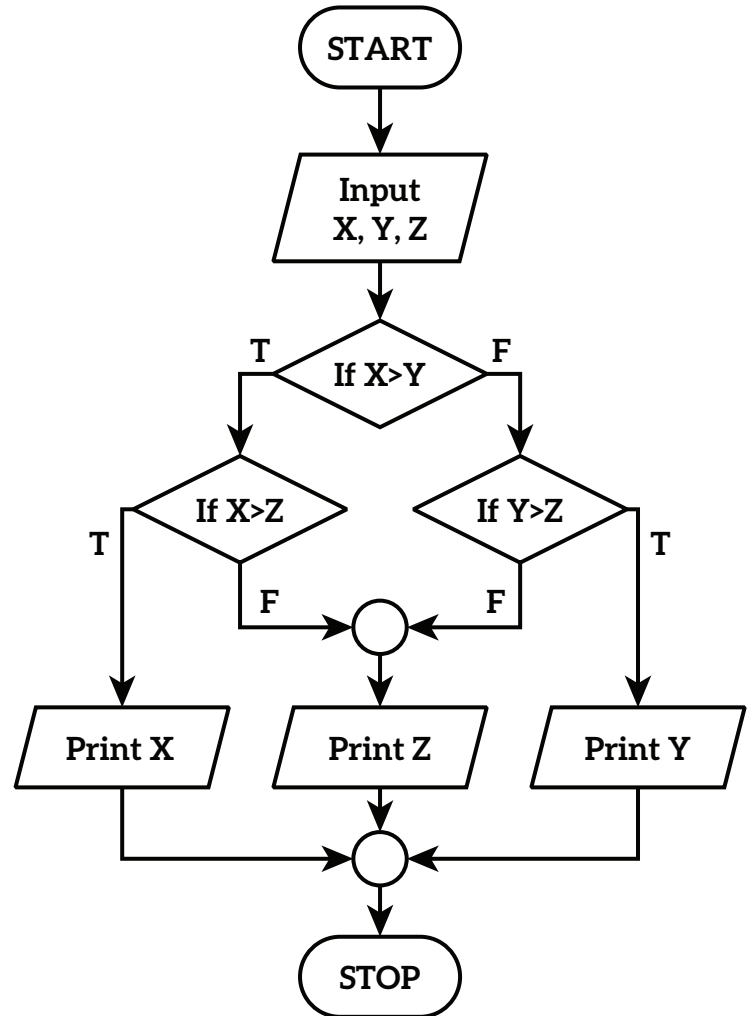
- A rate is fixed for one working hour.
- Employees paid on an hourly basis who exceed 40 working hours get a bonus for the extra hours (at 1.5 times the hourly rate).



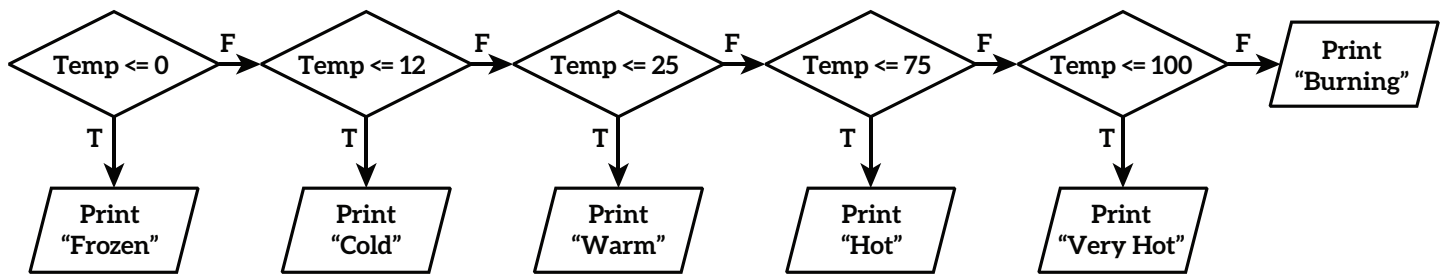
+ Example (2): Largest of 3 numbers



Another solution for Example (2)

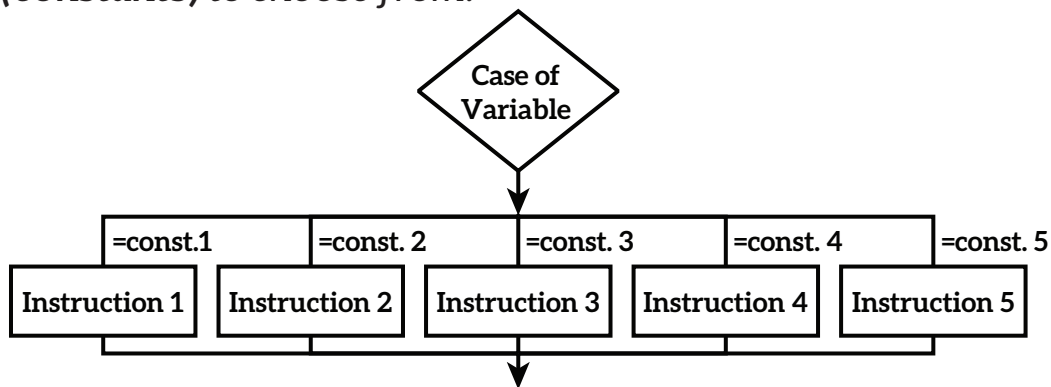


+ **Example (3):** A program that provides weather prediction according to expected temperature.



(D) Case Selection:

- Repeating the IF ... ELSE statements a number of times can be somewhat confusing.
- An alternative method provided in a number of languages is to use a selector determined by the alternative **fixed values**.
- Selection structures are called **case selection** structures when there are **two or more alternatives (constants)** to choose from.



+ **Example:** A company has four different medical plans. Each plan is given a code corresponding to the following: Plan 1 = F, Plan 2 = B, Plan 3 = K, Plan 4 = E. The company pays all for Plan 1. The individual has to partly pay for the other plans. The payroll deduction for Plan 2 = 4.65, for Plan 3 = 7.85, and for Plan 4 = 5.50. Any other codes are considered errors. Write the algorithm for a module to determine the payroll deduction.

Algorithm:

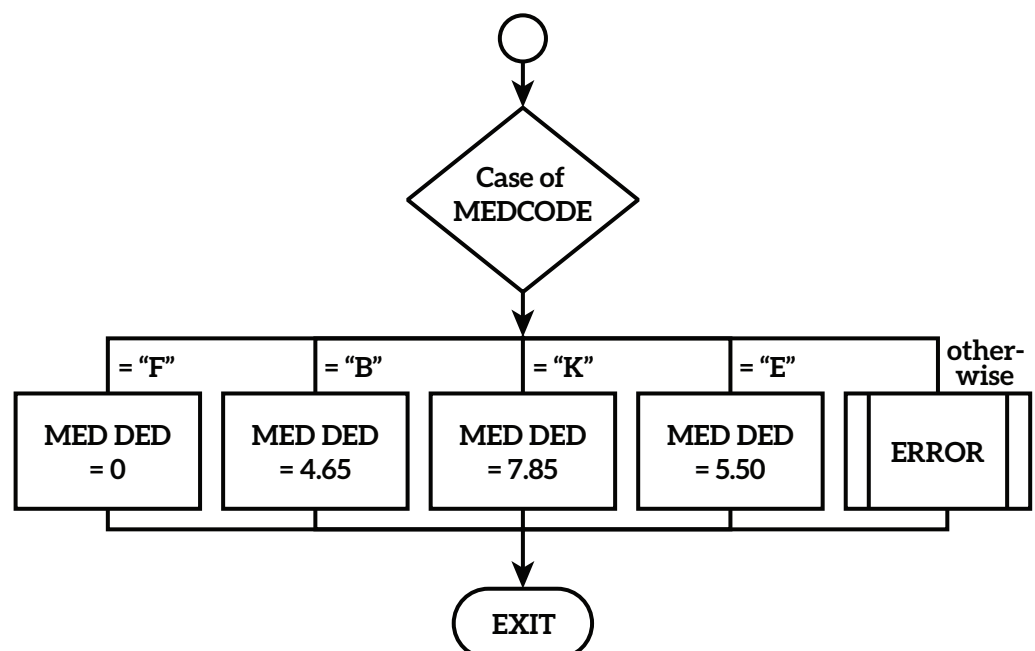
INSURANCE DEDUCTION
CASE OF MEDCODE

= "F":
MEDDEDUCTION = 0
= "B":
MEDDEDUCTION = 4.65
= "K":
MEDDEDUCTION = 7.85
= "E":
MEDDEDUCTION = 5.50

OTHERWISE

PROCESS ERROR

END-OF-CASE



3] Repetition Structures:

- **Advantages of Repetition**

Structures:

- One of the **basic logical structures** of computer programming.
- Defining **loops** allows computers to **perform particular tasks repeatedly**.
- You can write one set of instructions that operates on **multiple separate sets of data**.

- The Repetition structure can be implemented using:

- (A) **WHILE Loop**
- (B) **DO WHILE Loop**
- (C) **FOR Loop**

- Any program instruction that repeats a statement or sequence of statements a number of times is called an **iteration** or a **loop**.

- The commands used to create iterations or loops are all based on **logical tests**.

- The statements that execute within a loop are known as **the loop body**.

- A **counter** is any numeric variable you use to count the number of times an event has occurred.

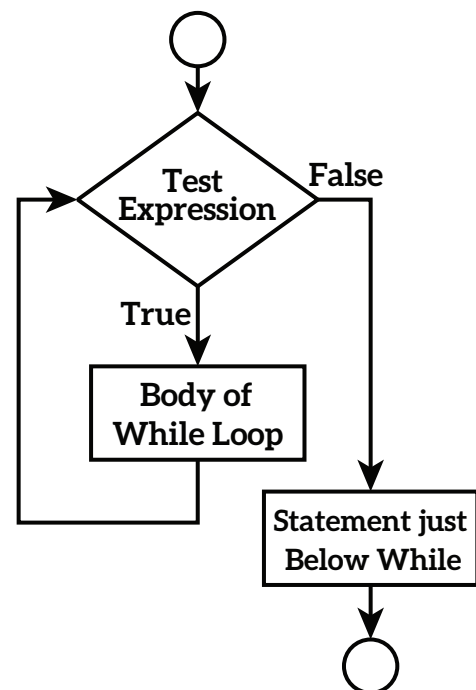
- Every time you add 1 to a variable is called **incrementing** the variable.

(A) **WHILE Loop:**

- This type of conditional loop **tests** for terminating condition at the **beginning of the loop**.

- No action** is performed at all if the first test causes the terminating condition to evaluate as false.

- Sentinel-controlled**. (controlled by user's input)



- **How WHILE Loop Works?**

- The while loop evaluates the condition expression.
- If the condition expression is true, statements inside the body of while loop are executed.
- Then, the condition expression is evaluated again. This process goes on until the condition expression is false.
- When the condition expression is false, while loop is terminated.

- Make sure that the condition is actually going to turn false at some point in time.

- If the condition is always true, then you will end up in an **infinite loop**.

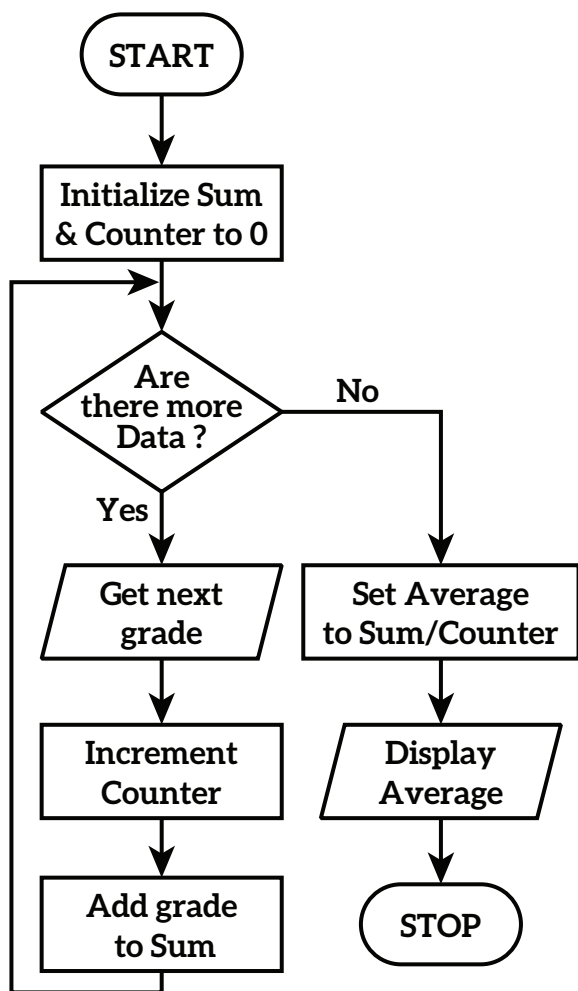
- + **Problem:** Calculate and report the average grade for a class

- **Discussion:** The average grade equals the sum of all grades divided by the number of students

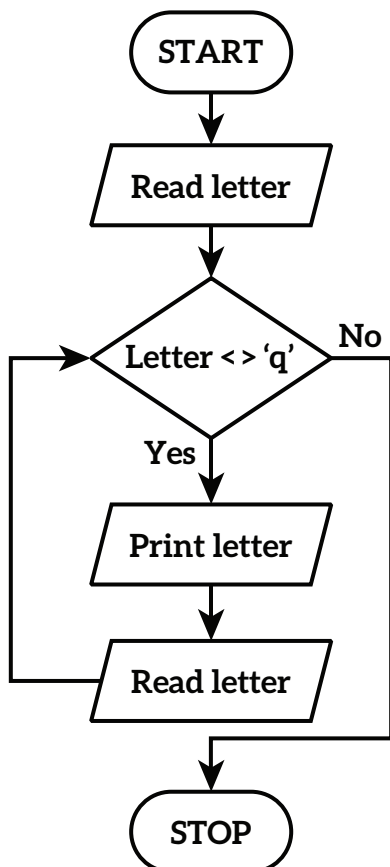
- **Input:** Student grades

- **Processing:** Find the sum of the grades; count the number of students; calculate average

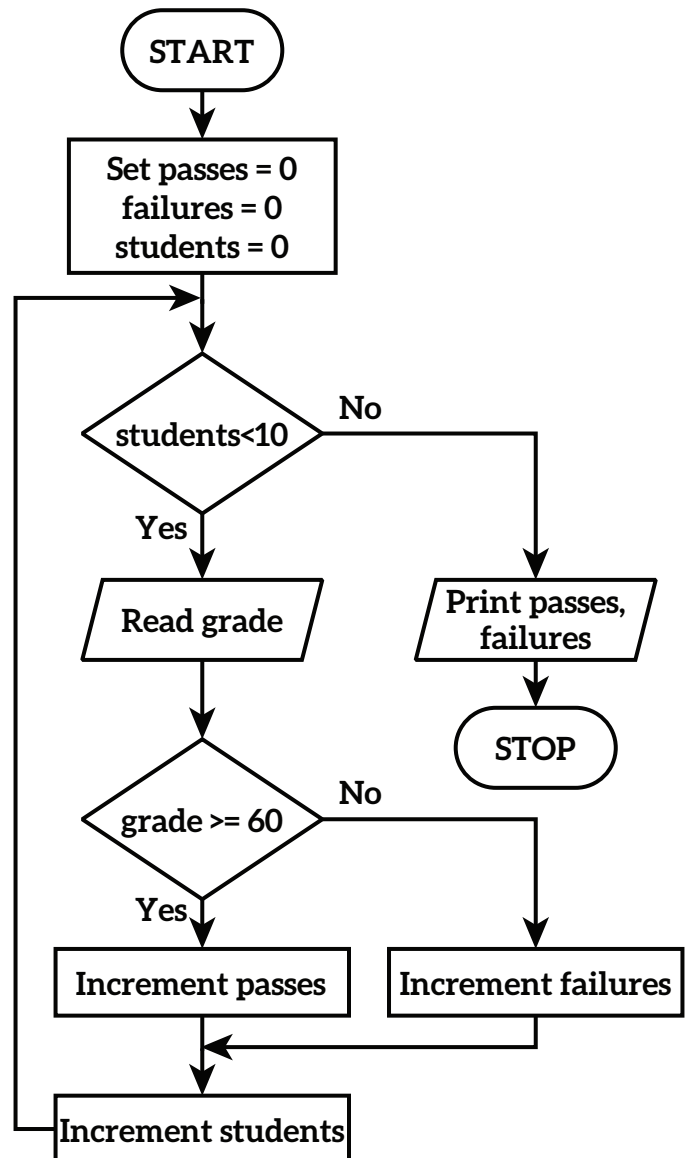
- **Output:** Average grade



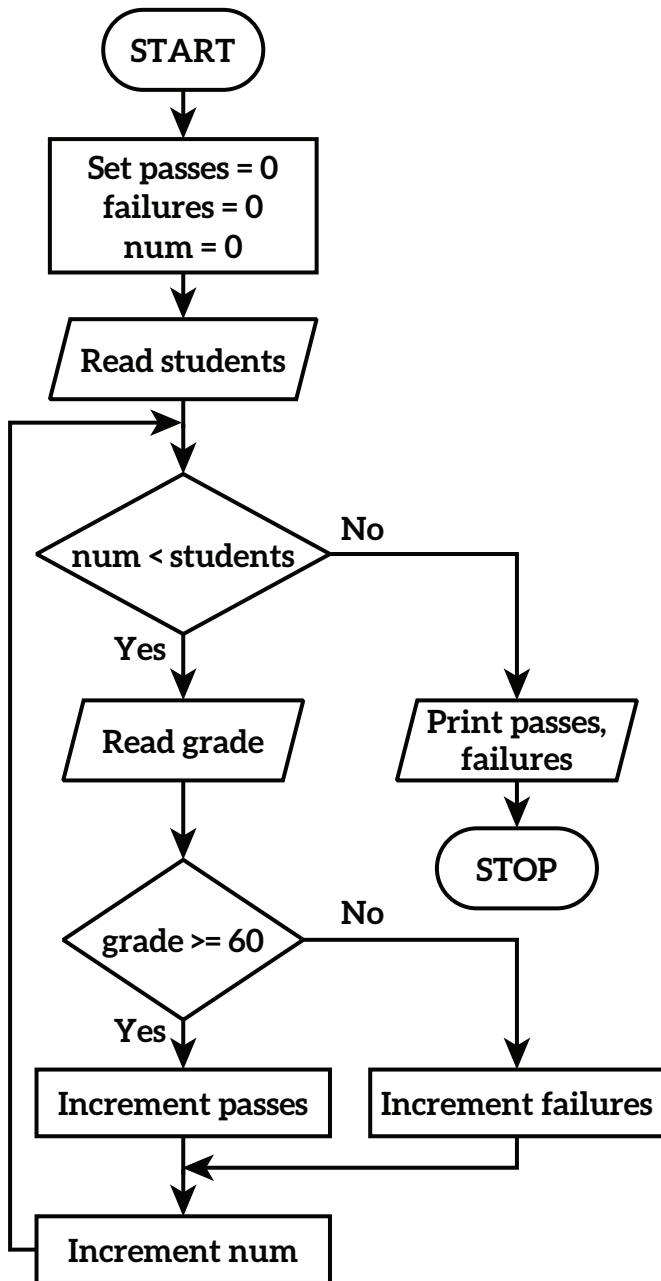
+ **Example (1)** : An algorithm to print out each character typed at a keyboard until the character 'q' is entered.



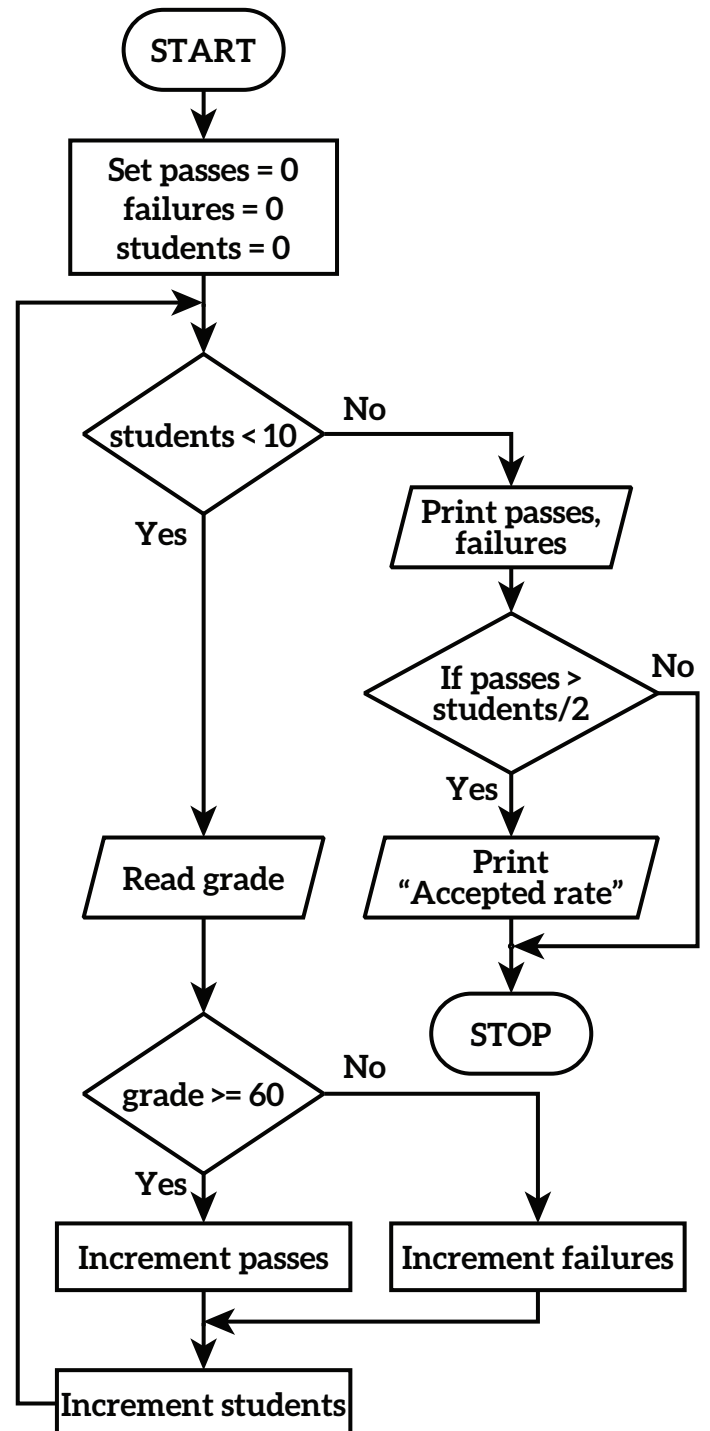
+ **Example (2)** : An algorithm that reads the exam results for **10 students** and displays the numbers of passed and failed students. A student passes if his mark is greater than or equal to 60.



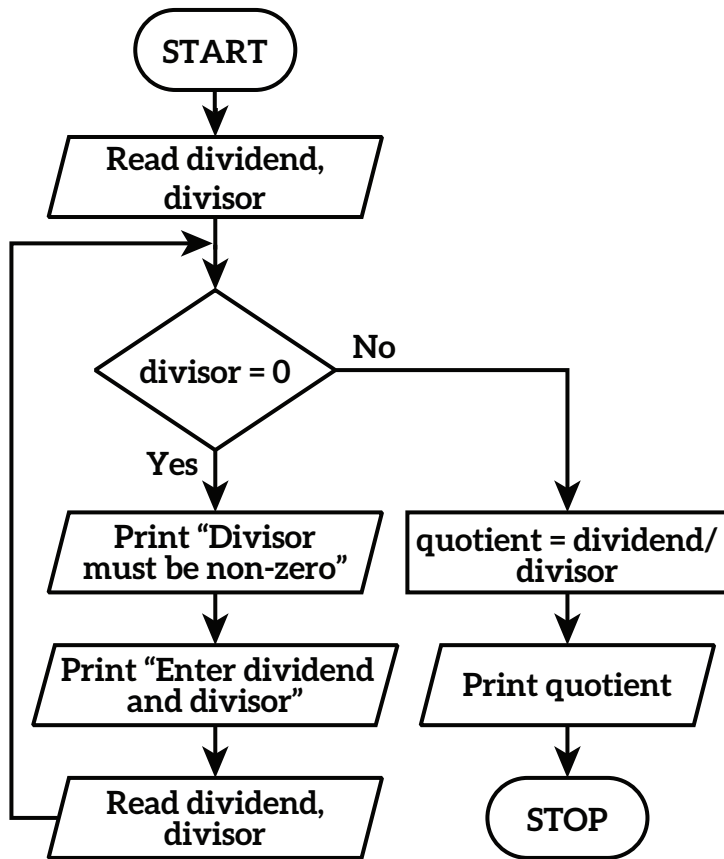
+ **Example (2) cont.** : An algorithm that reads the exam results for a **number of students** and displays the numbers of passed and failed students. A student passes if his mark is greater than or equal to 60.



+ **Example (3)** : An algorithm that reads the exam results for 10 students and displays the numbers of passed and failed students. A student passes if his mark is greater than or equal to 60. If more than half of the students pass, the success rate is accepted by the school.



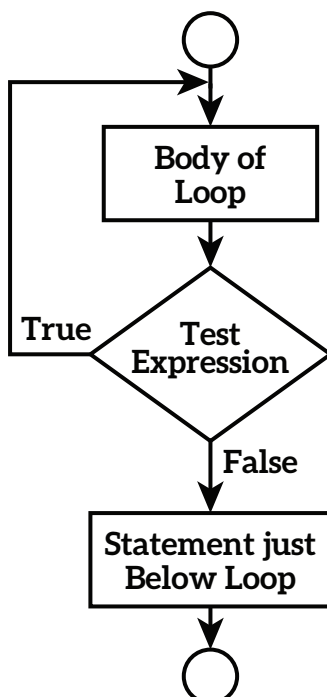
+ **Example (4)** : Get two numbers from the user and display their quotient. Make sure that the divisor number is not zero.



(B) DO WHILE Loop:

-This type of conditional loop **tests** for terminating condition at **the end of the loop**.

-**Executes at least once** before checking the condition.



• How DO WHILE Loop Works?

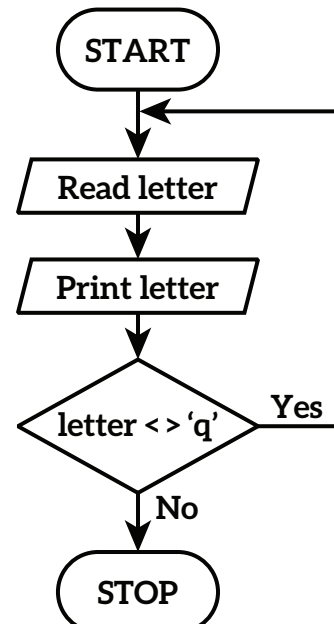
- The codes inside the body of loop is executed at least once. Then, only the condition expression is checked.
- If the condition expression is true, the body of loop is executed. This process continues until the condition expression becomes false.
- When the test expression is false, DO WHILE loop is terminated.

• DO WHILE loop has the convenience in the following situations:

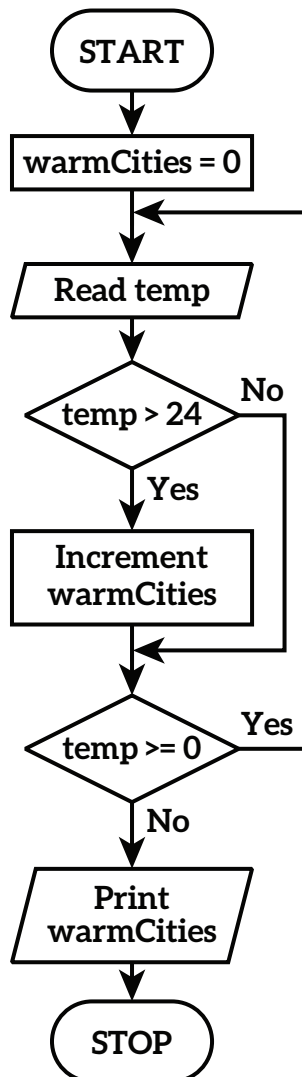
- The condition is not known or even defined until inside the loop
- Loop should execute at least once
- When you are asking a question, whose answer will determine if the loop is repeated

• Just like with a WHILE loop, it is possible to end up in an **infinite loop**, so it is very important to make sure the end condition will always be met eventually.

+ **Example (1)** : An algorithm to print out each character typed at a keyboard until the character 'q' is entered.

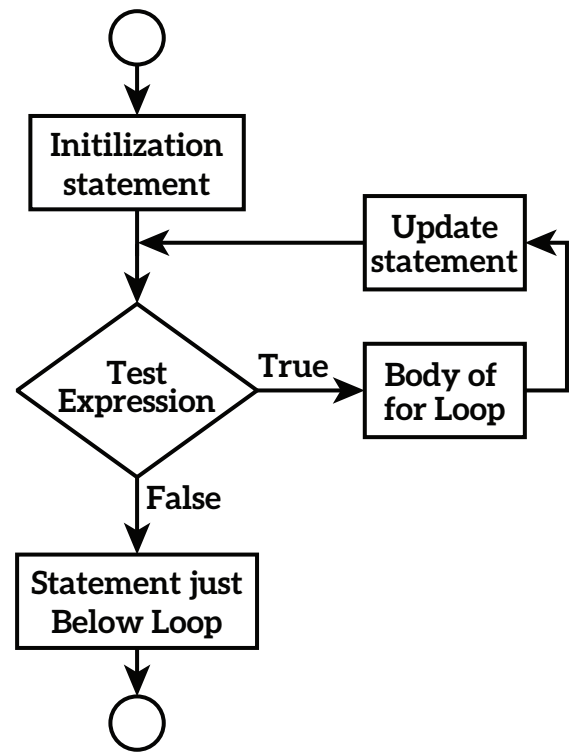


+ **Example (2)** : An algorithm that accepts readings of city temperatures and displays the number of warm cities (>24). The program stops when user enters a negative number.



(C) FOR Loop:

- Used when the **number of iterations** is **known** in advance.
- Uses an **initialization** of the variable as a starting point.
- The **condition** depends on the value of the variable.
- The variable is **incremented** on each iteration until it reaches the required value.

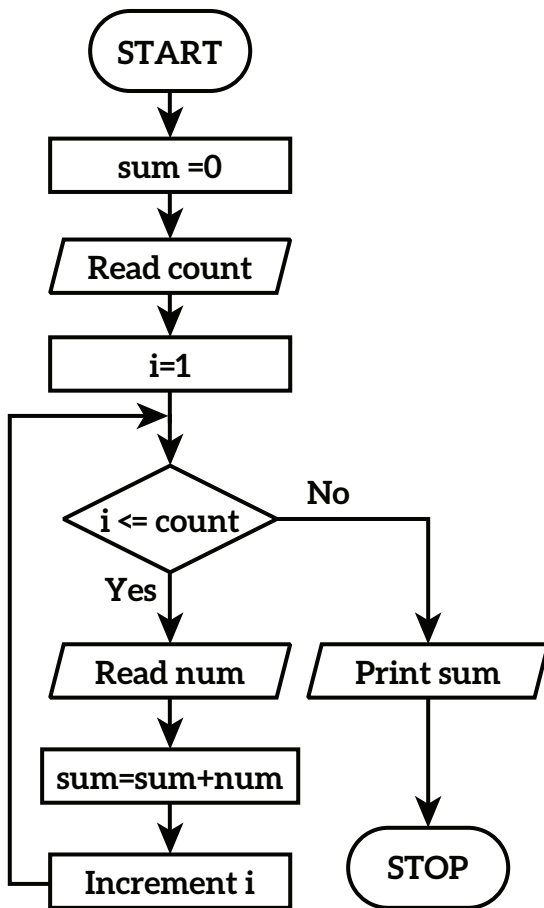


- The **starting state section** allows you to either declare a variable and give it a value or give a value to an already existing variable.
- The **condition section** tells the program that as long as the conditional expression is true the loop should continue to repeat itself.
- The **increment/update section** is the easiest way for a for loop to handle changing of the variable. It is possible to do things like $x+1$, $x = x - 10$, or even $x = \text{random}(5)$

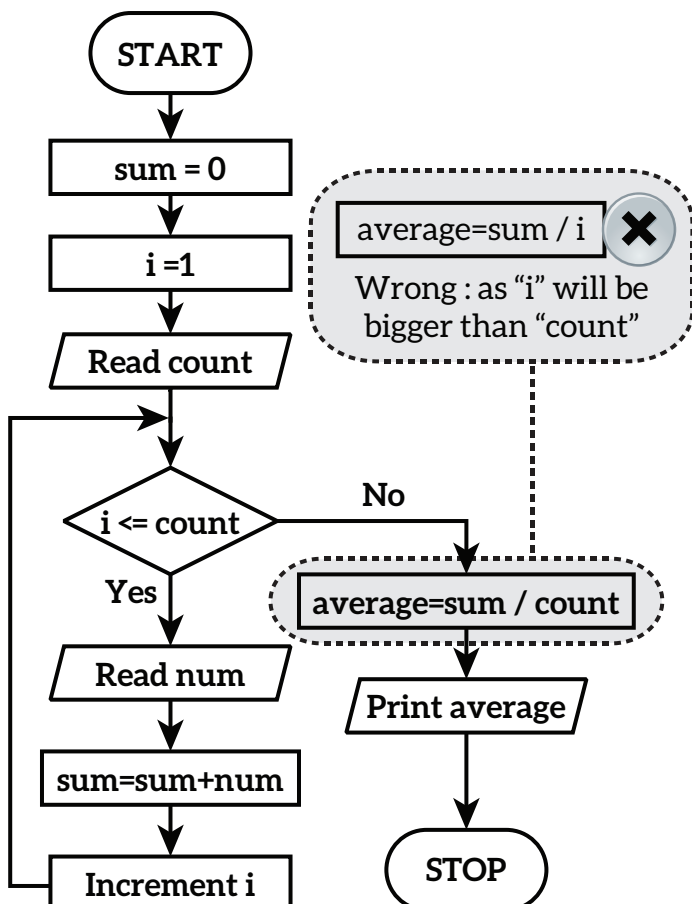
• How FOR Loop Works?

1. The starting statement is executed only once at the beginning.
2. Then, the condition expression is evaluated.
3. If the condition expression is false, FOR loop is terminated. But if it is true, statements inside body of FOR loop are executed and the increment expression is executed.
4. Again, the condition expression is evaluated and this process repeats until the condition expression is false.

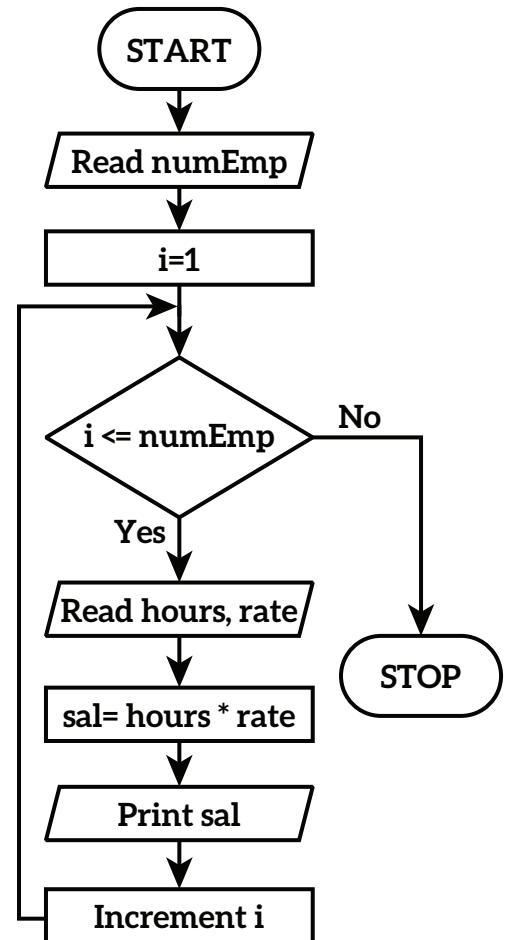
+ **Example (1)** : An algorithm to calculate the **sum** of a set of numbers.



+ **Example (2)** : An algorithm to calculate the **average** of a set of numbers.



+ **Example (3)** : An algorithm to calculate the salary for the hourly-paid employees.



• Which Loop to Use?

There is usually more than one way to solve almost every programming problem

-In case the program performs an indefinite number of iterations, as long as a certain condition remains true: **Use WHILE**

WHILE

-In case the program performs an indefinite number of iterations and does not need a condition at the start: **Use DO..WHILE**

DO..WHILE

-In case the program performs a specific number of iterations: **Use FOR**

#Exercise:

● Complete:

1. Each represents a step in the process, while the show the order in which they occur.
2. The algorithms can be designed through the use of or
3. The has two exit points; these can be on the sides or the bottom and one side.
4. Selection structures are called selection structures when there are two or more constants to choose from.
5. If the condition is always true, then you will end up in an
6. The statements that execute within a loop are known as the
7. The are used in case an upward flow will traverse more than three symbols.
8. In the design structure, statements can be executed or skipped depending on whether a condition evaluates to True or False.
9. The in the FOR loop handles changing the value of the variable.
10. The loop is used when processes should be executed at least once.
11. In flowchart, the symbol is used to ask a question that can be answered with binary format.

● Which Loop to Use?

A company calculates the weekly wages of its workers. The user submits the number of normal and overtime hours of the workers. Payment for normal working hour is 14 while for overtime working hour is 18...

1. If the number of workers is 10
2. If the user enters many workers and wants to choose when to stop
3. If we know that the company has at least one worker and the user wants to choose when to stop

ans. : 1-For 2-While 3-Do While

● Put True (T) or False (F):

1. All boxes of the flowchart are connected with lines.
2. The WHILE loop is used in case the program performs an indefinite number of iterations, as long as a certain condition remains true.
3. Nested IF is used in case of multiple embedded decisions.
4. An algorithm can be defined as a difference between a desired situation and current situation.
5. The FOR loop is used in case the program performs an indefinite number of iterations and does not need a condition at the start.
6. Every Nested IF can be represented using Case selection.

● Practice:

1. Flowchart for program that provides recommendations for hotels by interpreting the hotel stars ratings made by users. Each star rate represents a specific meaning.
2. Flowchart for program that will output the square of any number input until the number input is zero.