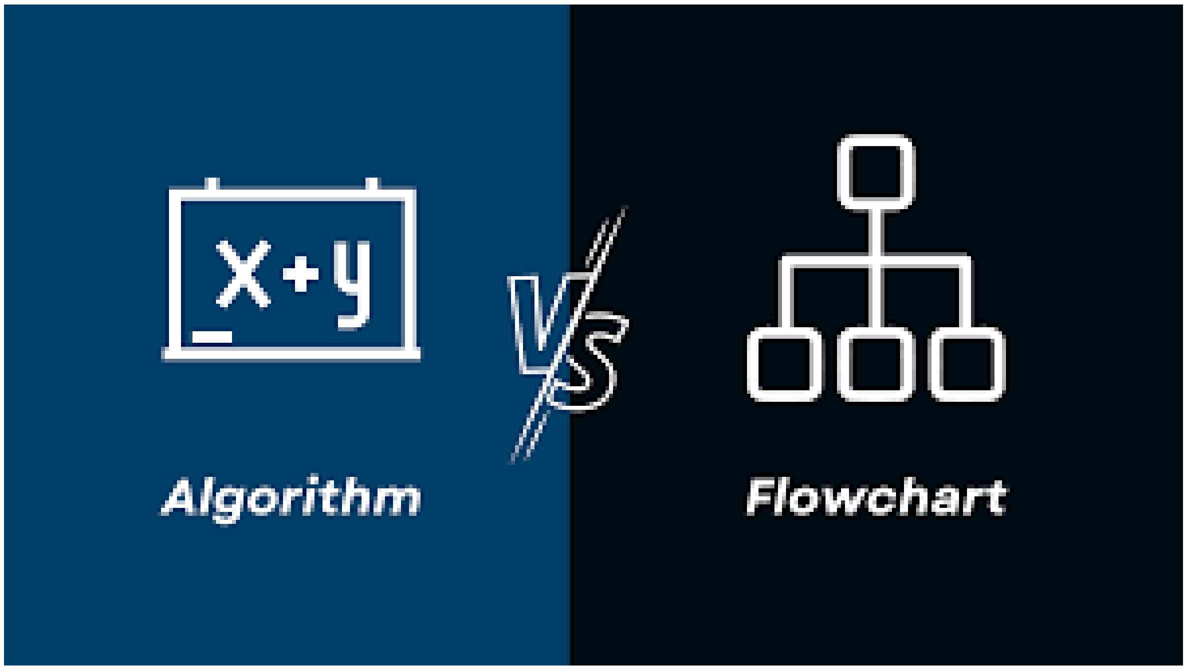
ALGORITHM & FLOWCHART



Mohannad Al-Kubaisi

INTRODUCTION

A computer is a useful tool for solving a great variety of problems. To make a computer do anything (i.e. solve a problem), you have to write a computer program. In a computer program you tell a computer, step by step, exactly what you want it to do. The computer then executes the program, following each step mechanically, to accomplish the end goal.

Algorithm and flowchart are the powerful tools for learning programming. An algorithm is a step-by-step analysis of the process, while a flowchart explains the steps of a program in a graphical way. Algorithm and flowcharts helps to clarify all the steps for solving the problem.

ALGORITHMS

The word “algorithm” relates to the name of the mathematician Al-khowarizmi, which means a procedure or a technique. Software Engineer commonly uses an algorithm for planning and solving the problems. An algorithm is a sequence of steps to solve a particular problem or algorithm is an ordered set of unambiguous steps that produces a result and terminates in a finite time.

Algorithm has the following characteristics

- **Input**: An algorithm may or may not require input.
- **Output**: Each algorithm is expected to produce at least one result.
- **Definiteness**: Each instruction must be clear and unambiguous.
- **Finiteness**: If the instructions of an algorithm are executed, the algorithm should terminate after finite number of steps.

The algorithm and flowchart include following three types of control structures.

1. **Sequence**: In the sequence structure, statements are placed one after the other and the execution takes place starting from up to down.
2. **Branching** (Selection): In branch control, there is a condition and according to a condition, a decision of either TRUE or FALSE is achieved. Generally, the ‘IF-THEN’ is used to represent branch control.
3. **Loop** (Repetition): The Loop or Repetition allows a statement(s) to be executed repeatedly based on certain loop condition e.g. WHILE, FOR loops.

HOW TO WRITE ALGORITHMS

Step 1: Define your algorithms input: Many algorithms take in data to be processed, e.g. to calculate the area of rectangle input may be the rectangle height and rectangle width.

Step 2: Define the variables: Algorithm's variables allow you to use it for more than one place. We can define two variables for rectangle height and rectangle width as HEIGHT and WIDTH (or H & W). We should use meaningful variable name.

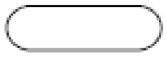
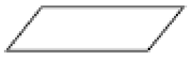
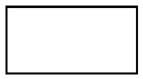
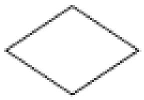
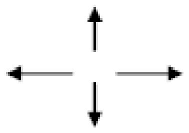
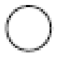
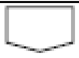
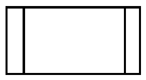
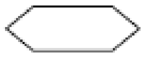
Step 3: **Outline the algorithm's operations**: Use input variable for computation purpose, e.g. to find area of rectangle multiply the HEIGHT and WIDTH variable and store the value in new variable (say) AREA. An algorithm's operations can take the form of multiple steps and even branch, depending on the value of the input variables.

Step 4: Output the results of your algorithm's operations: In case of area of rectangle output will be the value stored in variable AREA. if the input variables described a rectangle with a HEIGHT of 2 and a WIDTH of 3, the algorithm would output the value of 6.

FLOWCHART

The first design of flowchart goes back to 1945 which was designed by John Von Neumann. Unlike an algorithm, Flowchart uses different symbols to design a solution to a problem. It is another commonly used programming tool. By looking at a Flowchart can understand the operations and sequence of operations performed in a system. Flowchart is often considered as a blueprint of a design used for solving a specific problem.

Flowchart is diagrammatic /Graphical representation of sequence of steps to solve a problem. To draw a flowchart following standard symbols are use.

Symbol Name	Symbol	function
Oval		Used to represent start and end of flowchart
Parallelogram		Used for input and output operation
Rectangle		Processing: Used for arithmetic operations and data-manipulations
Diamond		Decision making. Used to represent the operation in which there are two/three alternatives, true and false etc
Arrows		Flow line Used to indicate the flow of logic by connecting symbols
Circle		Page Connector
		Off Page Connector
		Predefined Process /Function Used to represent a group of statements performing one processing task.
		Preprocessor
	|------------ -------- | |------------	Comments

The language used to write algorithm is simple and similar to day-to-day life language. The variable names are used to store the values. The value store in variable can change in the solution steps. In addition some special symbols are used as below.

Assignment Symbol (← or =) is used to assign value to the variable.

e.g. to assign value 5 to the variable HEIGHT, statement is

HEIGHT ← 5 or HEIGHT = 5

The symbol '=' is used in most of the programming language as an assignment symbol, the same has been used in all the algorithms and flowcharts in the manual.

The statement C = A + B means that add the value stored in variable A and variable B then assign/store the value in variable C.

The statement R = R + 1 means that add I to the value stored in variable R and then assign/store the new value in variable R, in other words increase the value of variable R by 1

Mathematical Operators

Operator	Meaning	Example
+	Addition	A + B
-	Subtraction	A – B
*	Multiplication	A * B
/	Division	A / B
^	Power	A^3 for A^3
%	Reminder	A % B

Relational Operators

Operator	Meaning	Example
<	Less than	A < B
<=	Less than or equal to	A <= B
= or ==	Equal to	A = B
# or !=	Not equal to	A # B or A !=B
>	Greater than	A > B
>=	Greater tha or equal to	A >= B

Logical Operators

Operator	Example	Meaning
AND	A < B AND B < C	Result is True if both A<B and B<C are true else false
OR	A< B OR B < C	Result is True if either A<B or B<C are true else false
NOT	NOT (A >B)	Result is True if A>B is false else true

Selection control Statements

Selection Control	Example	Meaning

IF (Condition) Then
…
ENDIF
	IF (X > 10) THEN
	  Y=Y+5
	ENDIF
		If condition X>10 is True execute the statement between THEN and ENDIF

IF (Condition) Then
…
ELSE
…..
 	
ENDIF
	IF (X > 10) THEN
	  Y=Y+5
	ELSE
	  Y=Y+8
	  Z=Z+3
	ENDIF
		If condition X>10 is True execute the statement between THEN and ELSE otherwise execute the statements between ELSE and ENDIF

Loop control Statements

Selection Control	Example	Meaning

WHILE (Condition)
DO
..
..
ENDDO
	WHILE (X < 10)
	 DO
	   print x
	   x=x+1
	ENDDO
		Execute the loop as long as the condition is TRUE

DO
….
…
UNTILL (Condition)
	DO
	   print x
	   x=x+1
	UNTILL (X >10)
		Execute the loop as long as the condition is false

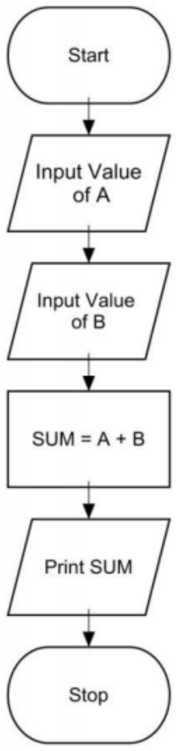
GO TO statement also called unconditional transfer of control statement is used to transfer control of execution to another step/statement. . e.g. the statement GOTO n will transfer control to step/statement n.

Note: We can use keyword INPUT or READ or GET to accept input(s) /value(s) and keywords PRINT or WRITE or DISPLAY to output the result(s).

Algorithm & Flowchart to find the sum of two numbers

Algorithm

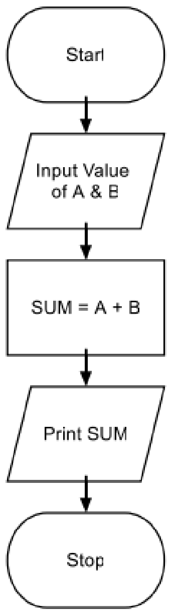
```
Step-1   Start
Step-2   Input first  numbers say A
Step-3   Input second number say B
Step-4   SUM = A + B
Step-5   Display SUM
Step-6   Stop
```



OR

Algorithm

```
Step-1   Start
Step-2   Input two numbers say A & B
Step-3   SUM = A + B
Step-4   Display SUM
Step-5   Stop
```

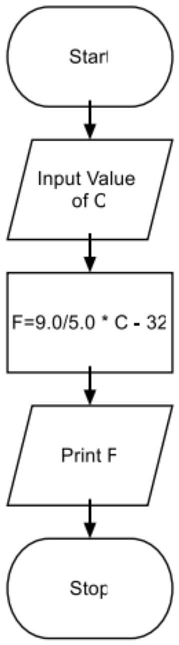


Algorithm & Flowchart to convert temperature from Celsius to Fahrenheit

C : temperature in Celsius
F : temperature Fahrenheit

Algorithm

```
Step-1   Start
Step-2   Input temperature in Celsius say C
Step-3   F = (9.0/5.0 x C) + 32
Step-4   Display Temperature in Fahrenheit F
Step-5   Stop
```

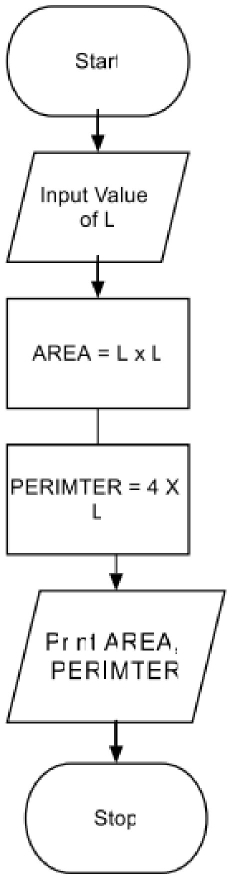


Algorithm & Flowchart to find Area and Perimeter of Square

L : Side Length of Square
AREA : Area of Square
PERIMETER : Perimeter of Square

Algorithm

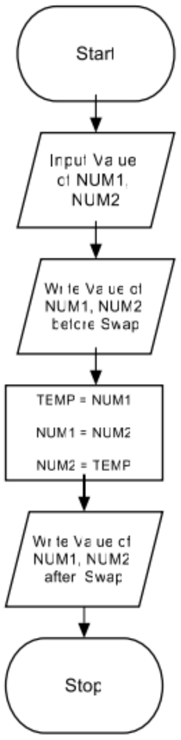
```
Step-1   Start
Step-2   Input Side Length of Square say L
Step-3   Area = L x L
Step-4   PERIMETER = 4 x L
Step-5   Display AREA, PERIMETER
Step-6   Stop
```



Algorithm & Flowchart to Swap Two Numbers using Temporary Variable

Algorithm

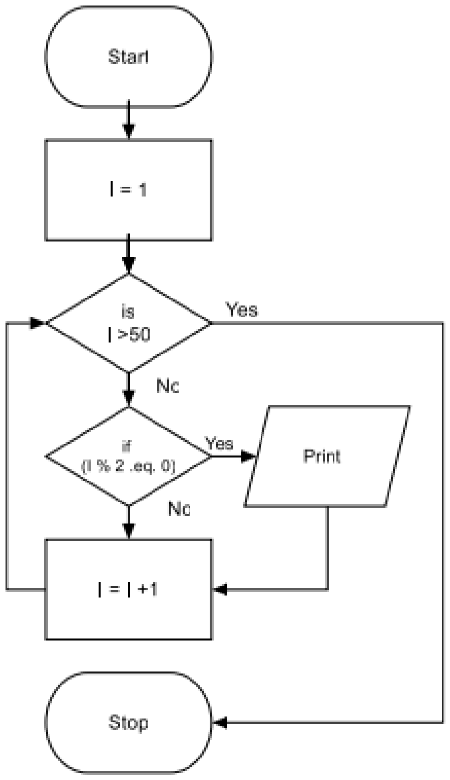
```
Step-1   Start
Step-2   Input Two Numbers Say NUM1,NUM2
Step-3   Display Before Swap Values NUM1, NUM2
Step-4   TEMP = NUM1
Step-5   NUM1 = NUM2
Step-6   NUM2 = NUM1
Step-7   Display After Swap Values NUM1,NUM
Step-8   Stop
```


Start
Input Value of NUM1, NUM2
Write Value of NUM1, NUM2 before Swap
TEMP = NUM1
NUM1 = NUM2
NUM2 = TEMP
Write Value of NUM1, NUM2 after Swap
Stop

Algorithm & Flowchart to find Even number between 1 to 50

Algorithm

```
Step-1   Start
Step-2   I = 1
Step-3   IF (I >50)  THEN
             GO TO Step-7
         ENDIF
Step-4   IF (  (I % 2) =0)  THEN
             Display I
         ENDIF
Step-5   I = I + 1
Step-6   GO TO Step--3
Step-7   Stop
```


Start
I = 1
is I >50
Yes
No
if (I % 2 .eq. 0)
Yes
Print
No
I = I +1
Stop