



Algorithm and Flowchart

ComPuter Engineering, KMUTT

Algorithm & Flowchart Manual

ALGORITHM

The word “*algorithm*” relates to the name of the Persian mathematician, Muḥammad ibn Mūsā al-Khwārizmī, or 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. In the case of TRUE, one of the two branches is explored; but in the case of FALSE condition, the other alternative is taken. 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.

Advantages of algorithm

- It is a step-wise representation of a solution to a given problem, which makes it easy to understand.
- An algorithm uses a definite procedure.
- It is not dependent on any programming language, so it is easy to understand for anyone even without programming knowledge.
- Every step in an algorithm has its own logical sequence so it is easy to debug.

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 e.g. instead of using H & W use HEIGHT and WIDTH as 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.

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 (\Downarrow or $=$) is used to assign value to the variable., e.g. to assign value 5 to the variable LENGTH, statement is

LENGTH \Downarrow 5 or LENGTH = 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 $Z = X + Y$ means that add the value stored in variable X and variable Y then assign/store the value in variable Z.

The statement $A = A + 1$ means that add 1 to the value stored in variable A and then assign/store the new value in variable A, in other words increase the value of variable A 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 than 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).

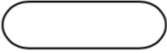
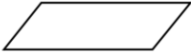

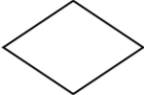
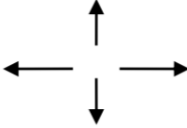



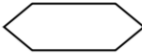

FLOWCHART

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 one 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.

Advantages of flowchart:

- Flowchart is an excellent way of communicating the logic of a program.
- Easy and efficient to analyze problem using flowchart.
- During program development cycle, the flowchart plays the role of a blueprint, which makes program development process easier.
- After successful development of a program, it needs continuous timely maintenance during the course of its operation. The flowchart makes program or system maintenance easier.
- It is easy to convert the flowchart into any programming language code.

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

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

Example 1: Write an algorithm and a flowchart to find the sum of two numbers.

Algorithm	Flowchart
<p>STEP 1: Start</p> <p>STEP 2: Input the first number say X</p> <p>STEP 3: Input the second number say Y</p> <p>STEP 4: $SUM = X + Y$</p> <p>STEP 5: Display SUM</p> <p>STEP 6: Stop</p>	<pre>graph TD; Start([Start]) --> InputA[/Input Value of A/]; InputA --> InputB[/Input Value of B/]; InputB --> Process[SUM = A + B]; Process --> PrintSUM[/Print SUM/]; PrintSUM --> Stop([Stop]);</pre>

<p>STEP 1: Start</p> <p>STEP 2: Input two numbers say X, Y</p> <p>STEP 3: $SUM = X + Y$</p> <p>STEP 4: Display SUM</p> <p>STEP 5: Stop</p>	<pre>graph TD; Start([Start]) --> InputAB[/Input Value of A & E/]; InputAB --> Process[SUM = A + E]; Process --> PrintSUM[/Print SUM/]; PrintSUM --> Stop([Stop]);</pre>
---	--

Example 2: Write an algorithm and a flowchart to convert temperature from Celsius to Fahrenheit.

Algorithm	Flowchart
<p>C: temperature in Celsius F: temperature Fahrenheit</p> <p>Step 1: Start Step 2: Input temperature in Celsius say C Step 3: $F = (9.0/5.0 \times C) + 32$ Step 4: Display Temperature in Fahrenheit F Step 5: Stop</p>	<pre> graph TD Start([Start]) --> Input[/Input Value of C/] Input --> Process[F=9.0/5.0 * C - 32] Process --> Output[/Print F/] Output --> Stop([Stop]) </pre>

Example 3: Write an algorithm and a flowchart to convert temperature from Fahrenheit to Celsius.

Algorithm	Flowchart
<p>C: temperature in Celsius F: temperature Fahrenheit</p> <p>Step 1: Start Step 2: Input temperature in Fahrenheit F say C Step 3: $C = 5.0/9.0(F-32)$ Step 4: Display Temperature in Celsius C Step 5: Stop</p>	<pre> graph TD Start([Start]) --> Input[/Input Value of F/] Input --> Process[C=5.0/9.0(F - 32)] Process --> Output[/Print C/] Output --> Stop([Stop]) </pre>

Example 4: Write an algorithm and a flowchart to find Area and Perimeter of Square.

Algorithm	Flowchart
<p>L : Side Length of Square AREA : Area of Square PERIMETER : Perimeter of Square Algorithm</p> <p>Step 1: Start Step 2: Input Side Length of Square say L Step 3: Area= $L \times L$ Step 4: PERIMETER = $4 \times L$ Step 5: Display AREA, PERIMETER Step 6: Stop</p>	<pre> graph TD Start([Start]) --> Input[/Input Value of L/] Input --> Area[AREA = L x L] Area --> Perimeter[PERIMETER = 4 x L] Perimeter --> Output[/Print AREA, PERIMETER/] Output --> Stop([Stop]) </pre>

Example 5: Write an algorithm and a flowchart to find Area and Perimeter of Rectangle.

Algorithm	Flowchart
<p>L : Length of Rectangle B : Breadth of Rectangle AREA : Area of Rectangle PERIMETER : Perimeter of Rectangle</p> <p>Step 1: Start Step 2: Input Side Length & Breadth say L, B Step 3: Area= $L \times B$ Step 4: PERIMETER = $2 \times (L + B)$ Step 5: Display AREA, PERIMETER Step 6: Stop</p>	<pre> graph TD Start([Start]) --> Input[/Input Value of L, B/] Input --> Area[AREA = L x B] Area --> Perimeter[PERIMETER = 2 x (L + B)] Perimeter --> Output[/Print AREA, PERIMETER/] Output --> Stop([Stop]) </pre>

Exercise 1: Write an algorithm and a flowchart to find Simple Interest.

Algorithm	Flowchart
<p>P : Principle Amount N : Time in Years R : % Annual Rate of Interest SI : Simple Interest</p>	

Exercise 2: Write an algorithm and a flowchart to find Compound Interest.

Algorithm	Flowchart
<p>P : Principle Amount N : Time in Years R : % Annual Rate of Interest CI : Compound Interest</p>	

Exercise 3: Write an algorithm and a flowchart to swap two numbers with temporary variable.

Algorithm	Flowchart

Exercise 4: Write an algorithm and a flowchart to swap two numbers without temporary variable.

Algorithm	Flowchart

Exercise 5: Write an algorithm and a flowchart to find the largest of two numbers.

Exercise 6: Write an algorithm and a flowchart to find the largest of three numbers.

Exercise 7: Write an algorithm and a flowchart to find even numbers between 1 to 50.

Exercise 8: Write an algorithm and a flowchart to find sum of series $1 + 2 + 3 + \dots + N$.

Exercise 9: Write an algorithm and a flowchart to find sum of series $1 - X + X^2 - X^3 + \dots + X^N$.

Exercise 10: Write an algorithm and a flowchart to print multiplication table of a number.

Exercise 11: Write an algorithm and a flowchart to find sum of series $1 - X + X^2 - X^3 + \dots + X^N$.