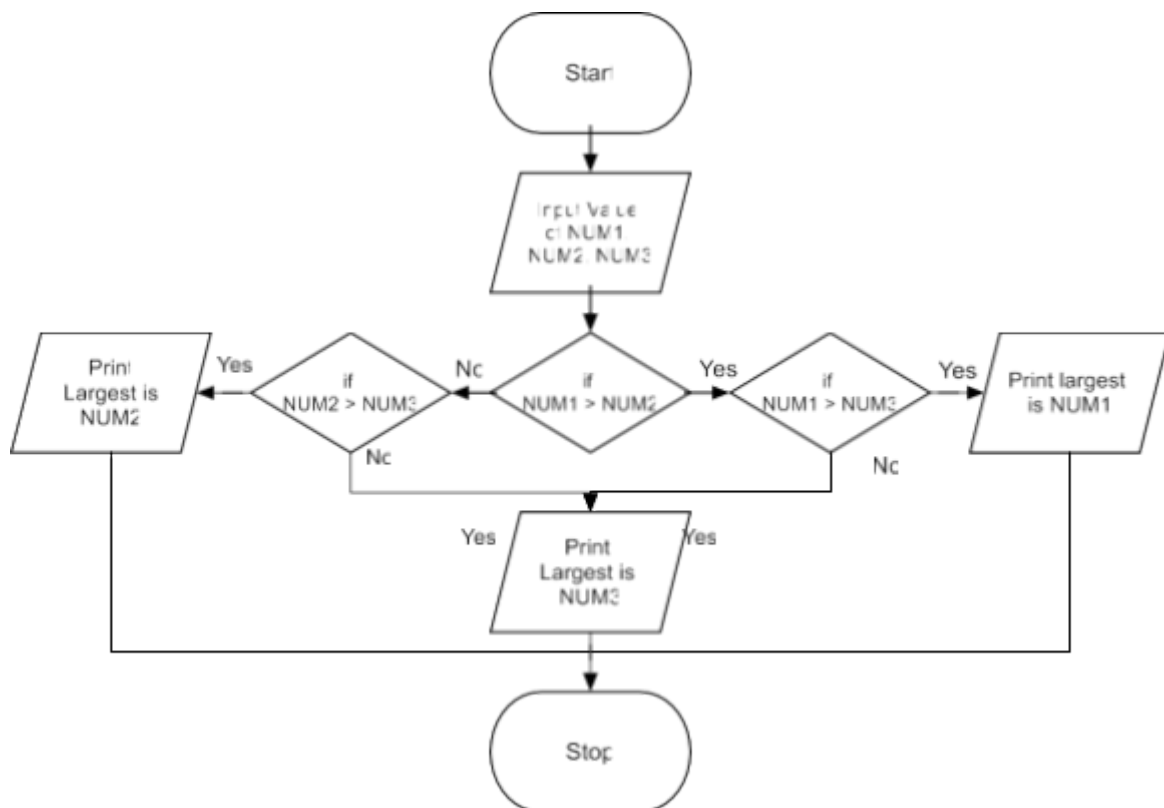


# ALGORITHM & FLOWCHART MANUAL

## for STUDENTS



Alpha Beta Gamma  
Instructor  
Computer & Instrumentation Center

..

## ALGORITHM:

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. 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.





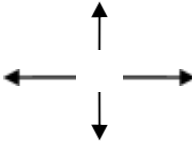





## FLOWCHART:

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 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** (  $\leftarrow$  or  $=$  ) is used to assign value to the variable.

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

HEIGHT  $\leftarrow$  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 1 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 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) .. .. ENDWhile	WHILE ( $X < 10$ ) print x x=x+1 ENDWhile	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

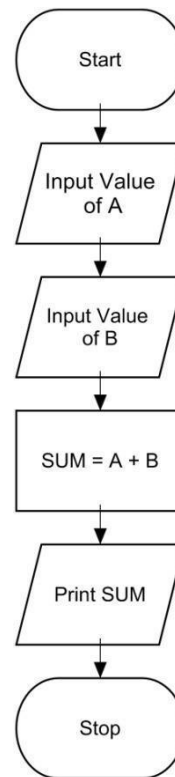
Step-2 Input first numbers say A

Step-3 Input second number say B

Step-4  $SUM = A + B$

Step-5 Display SUM

Step-6 Stop



OR

### Algorithm

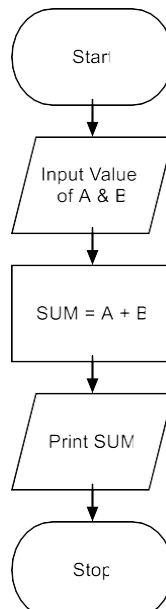
Step-1 Start

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

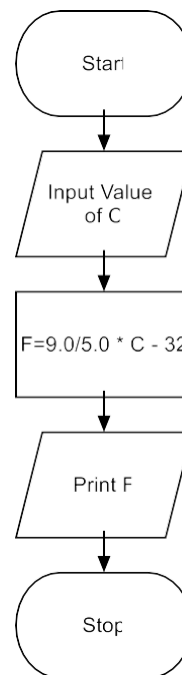
Step-1 Start

Step-2 Input temperature in Celsius say C

Step-3  $F = (9.0/5.0 \times C) + 32$

Step-4 Display Temperature in Fahrenheit F

Step-5 Stop



## Algorithm & Flowchart to convert temperature from Fahrenheit to Celsius

C : temperature in Celsius  
F : temperature Fahrenheit

### Algorithm

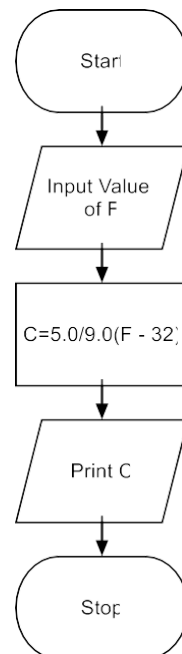
Step-1 Start

Step-2 Input temperature in Fahrenheit say F

Step-3  $C = 5.0/9.0 (F - 32)$

Step-4 Display Temperature in Celsius C

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

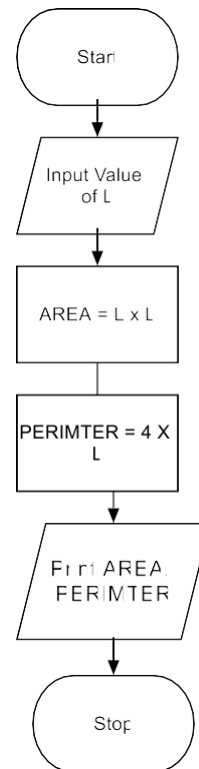
Step-2 Input Side Length of Square say L

Step-3  $\text{Area} = L \times L$

Step-4  $\text{PERIMETER} = 4 \times L$

Step-5 Display AREA, PERIMETER

Step-6 Stop



## Algorithm & Flowchart to find Area and Perimeter of Rectangle

L : Length of Rectangle  
B : Breadth of Rectangle  
AREA : Area of Rectangle  
PERIMETER : Perimeter of Rectangle

### Algorithm

Step-1 Start

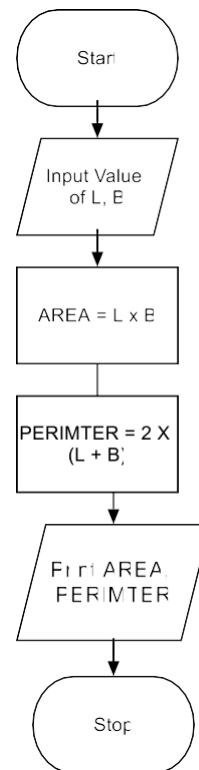
Step-2 Input Side Length & Breadth say L,

B Step-3  $\text{Area} = L \times B$

Step-4  $\text{PERIMETER} = 2 \times (L + B)$

Step-5 Display AREA, PERIMETER

Step-6 Stop



## Algorithm & Flowchart to find Area and Perimeter of Circle

R : Radius of Circle  
AREA : Area of Circle  
PERIMETER : Perimeter of Circle

### Algorithm

Step-1 Start

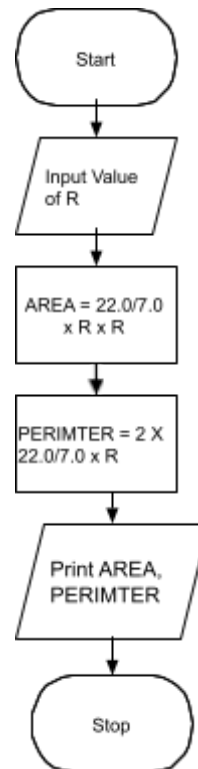
Step-2 Input Radius of Circle say

R Step-3 Area =  $22.0/7.0 \times R \times R$

Step-4 PERIMETER =  $2 \times 22.0/7.0 \times R$

Step-5 Display AREA,

PERIMETER Step-6 Stop



## Algorithm & Flowchart to find Area & Perimeter of Triangle (when three sides are given)

A : First Side of Triangle  
B : Second Side of Triangle  
C : Third Side of Triangle  
AREA : Area of Triangle  
PERIMETER : Perimeter of Triangle

### Algorithm

Step-1 Start

Step-2 Input Sides of Triangle A,B,C

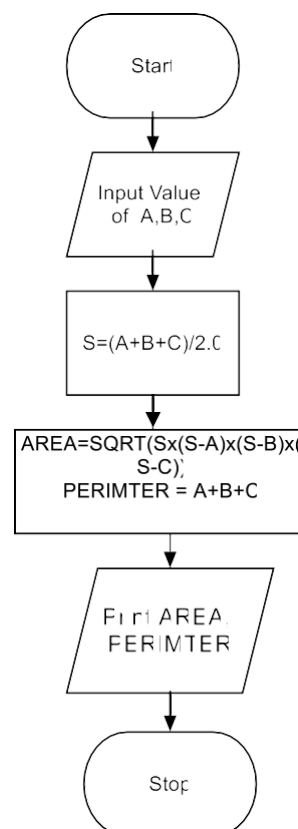
Step-3  $S = (A + B + C) / 2.0$

Step-4  $AREA = \text{SQRT}(S \times (S-A) \times (S-B) \times (S-C))$

Step-5 PERIMETER =  $S1 + S2 + S3$

Step-6 Display AREA,

PERIMETER Step-7 Stop



..

## Algorithm & Flowchart to find Simple Interest

P : Principle Amount  
N : Time in Years  
R : % Annual Rate of Interest  
SI : Simple Interest

### Algorithm

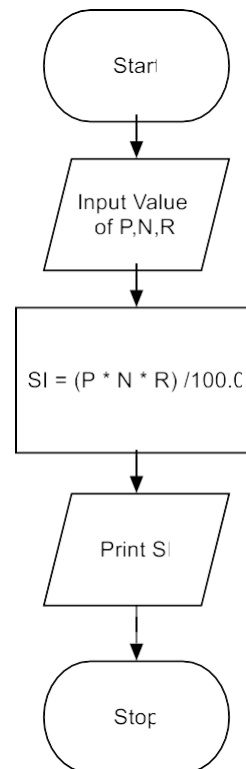
Step-1 Start

Step-2 Input value of P, N, R

Step-3  $SI = (P \times N \times R) / 100.0$

Step-4 Display SI F

Step-6 Stop



## Algorithm & Flowchart to find Compound Interest

P : Principle Amount  
N : Time in Years  
R : % Annual Rate of Interest  
CI : Compound Interest

### Algorithm

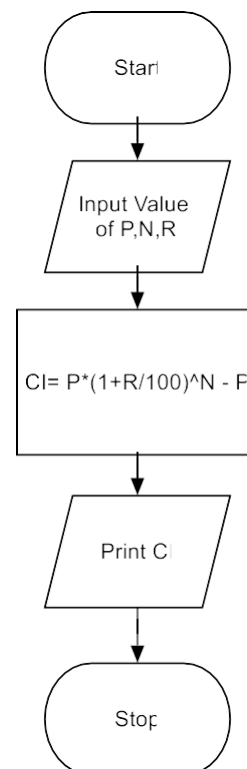
Step-1 Start

Step-2 Input value of P, N, R C

Step-3  $CI = P(1+R/100)^N - P$

Step-4 Display CI

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

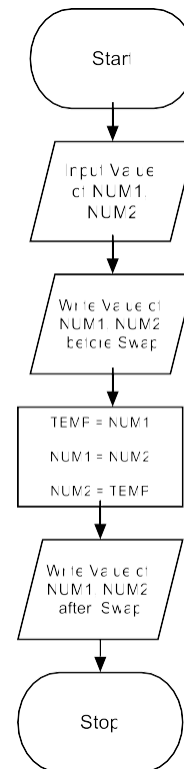
Step-4  $TEMP = NUM1$

Step-5  $NUM1 = NUM2$

Step-6  $NUM2 = NUM1$

Step-7 Display After Swap Values NUM1, NUM2

Step-8 Stop



## Algorithm & Flowchart to Swap Two Numbers without using temporary variable

### Algorithm

Step-1 Start

Step-2 Input Two Numbers Say A, B

Step-3 Display Before Swap Values A, B

Step-4  $A = A + B$

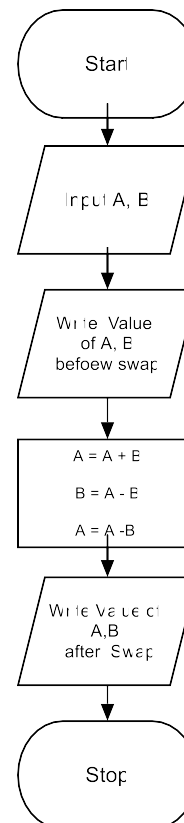
Step-5  $B = A - B$

Step-6  $A = A - B$

Step-7

Step-7 Display After Swap Values A, B

Step-8 Stop



## Algorithm & Flowchart to find the smallest of two numbers

### Algorithm

Step-1 Start

Step-2 Input two numbers say

NUM1, NUM2

Step-3 IF NUM1 < NUM2 THEN

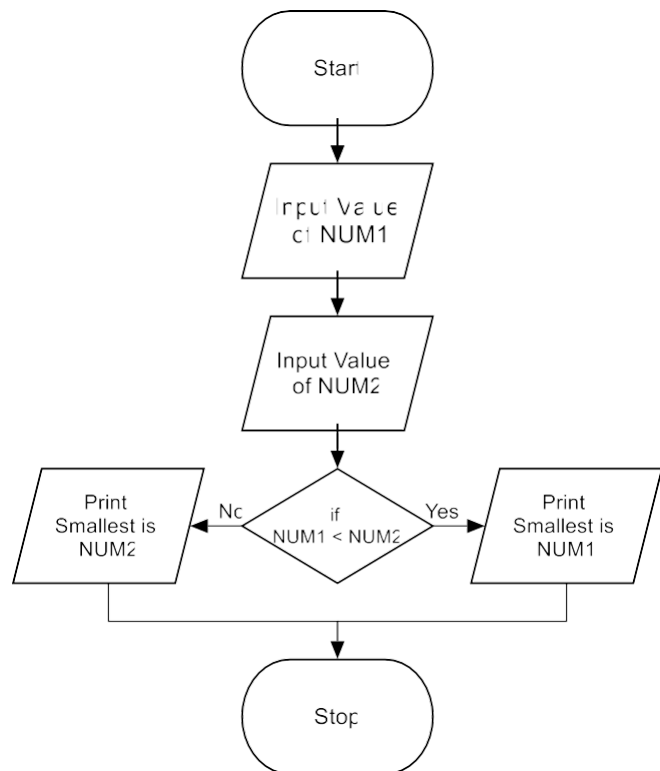
print smallest is NUM1

ELSE

print smallest is NUM2

ENDIF

Step-4 Stop



## Algorithm & Flowchart to find the largest of two numbers

### Algorithm

Step-1 Start

Step-2 Input two numbers say

NUM1, NUM2

Step-3 IF NUM1 > NUM2 THEN

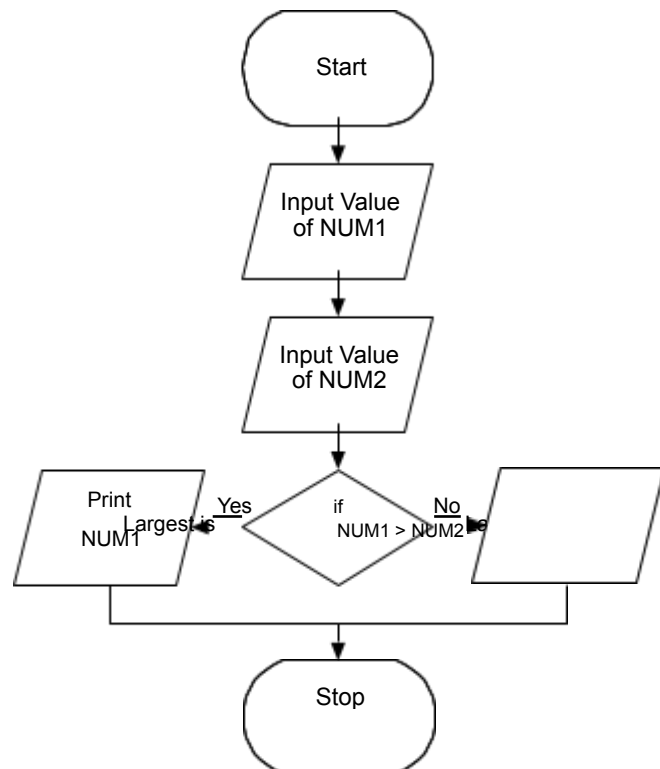
print largest is NUM1

ELSE

print largest is NUM2

ENDIF

Step-4 Stop



## Algorithm & Flowchart to find the largest of three numbers

### Algorithm

Step-1 Start

Step-2 Read three numbers say num1,num2, num3

Step-3 if num1>num2 then go to step-5

Step-4 IF num2>num3 THEN

print num2 is largest

ELSE

print num3 is largest

ENDIF

GO TO Step-6

Step-5 IF num1>num3 THEN

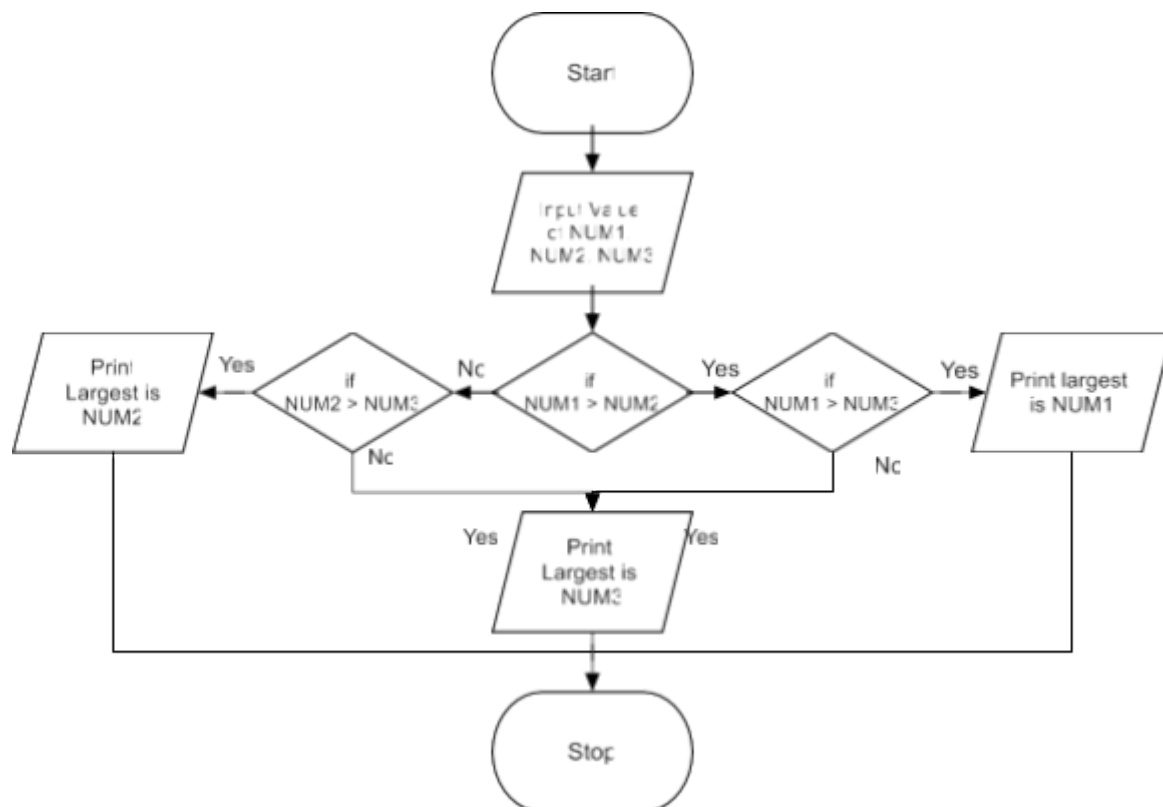
print num1 is largest

ELSE

print num3 is largest

ENDIF

Step-6 Stop



## Algorithm & Flowchart to find the largest of three numbers (an another way)

### Algorithm

Step-1 Start

Step-2 Read three numbers say A,B,C

Step-3 BIG = A

Step-4 IF B > BIG THEN

BIG = B

ENDIF

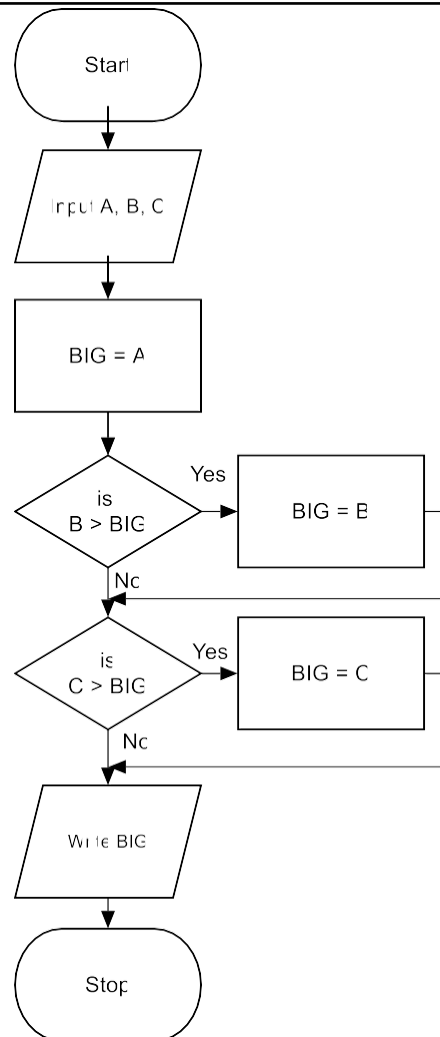
Step-5 IF C > BIG THEN

BIG = C

ENDIF

Step-6 Write

BIG Step-7 Stop



## Algorithm & Flowchart to find Even number between 1 to 50

### Algorithm

Step-1 Start

Step-2  $I = 1$

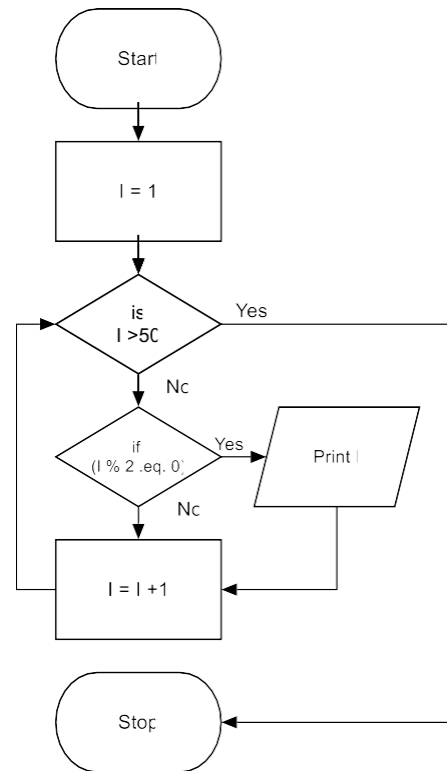
Step-3 IF ( $I > 50$ ) THEN  
GO TO Step-7  
ENDIF

Step-4 IF ( $(I \% 2) = 0$ ) THEN  
Display I  
ENDIF

Step-5  $I = I + 1$

Step-6 GO TO

Step-3 Step-7 Stop



## Algorithm & Flowchart to find Odd numbers between 1 to n where n is a positive Integer

### Algorithm

Step-1 Start

Step-2 Input Value of N

Step-3  $I = 1$

Step-4 IF ( $I > N$ ) THEN  
GO TO Step-8  
ENDIF

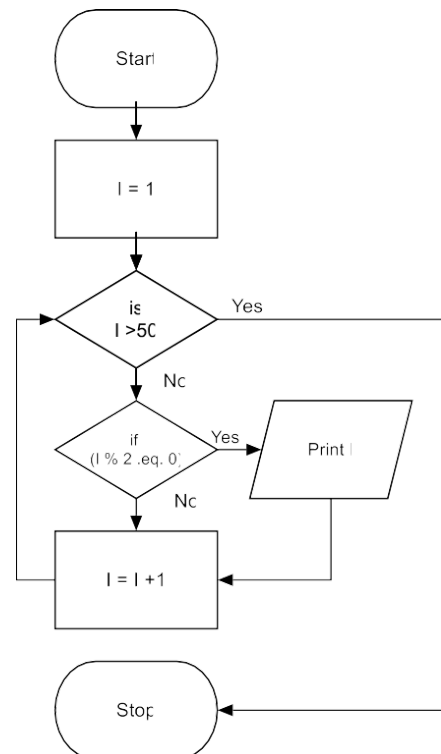
Step-5 IF ( $(I \% 2) = 1$ ) THEN  
Display I  
ENDIF

Step-6  $I = I + 1$

Step-7 GO TO

Step-4 Step-8

Stop





..

## Algorithm & Flowchart to find sum of series $1+2+3+.....+N$

### Algorithm

Step-1 Start

Step-2 Input Value of N

Step-3  $I = 1$ ,  $SUM = 0$

Step-4 IF ( $I > N$ ) THEN

GO TO Step-8

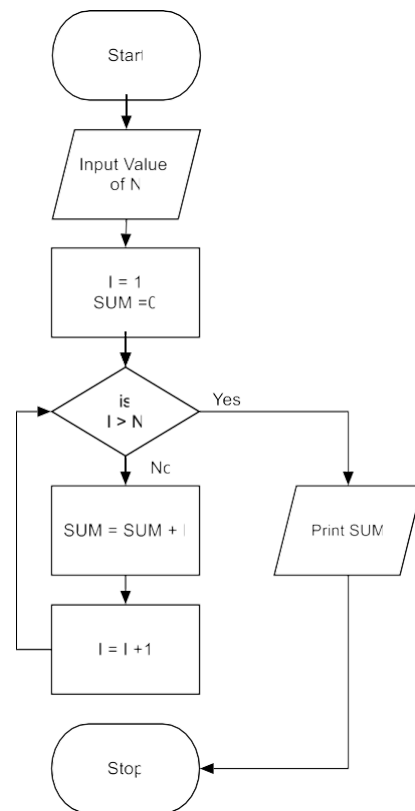
ENDIF

Step-5  $SUM = SUM + I$

Step-6  $I = I + 1$

Step-7 Go to step-4

Step-8 Display value of  
SUM Step-9 Stop



## Algorithm & Flowchart to find sum of series $1+3+5+.....+N$ , Where N is positive odd Integer

### Algorithm

### Algorithm

Step-1 Start

Step-2 Input Value of N

Step-3  $I = 1$ ,  $SUM = 0$

Step-4 IF ( $I > N$ ) THEN

GO TO step 8

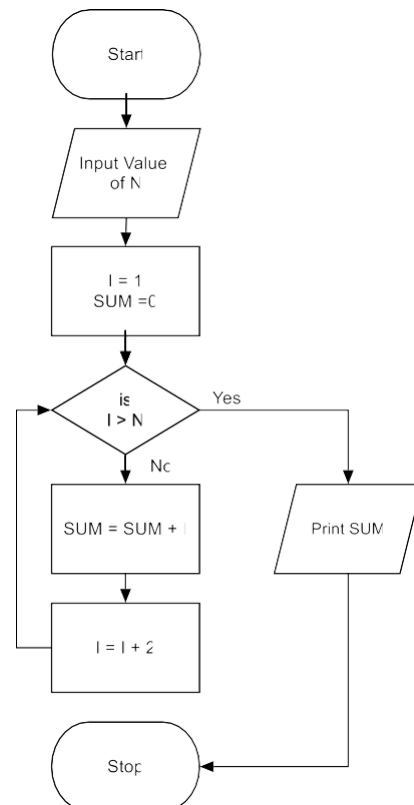
ENDIF

Step-5  $SUM = SUM + I$

Step-6  $I = I + 2$

Step-7 Go to step-4

Step-8 Display value of  
SUM Step-9 Stop



..

## Algorithm & Flowchart to find sum of series $1 - X + X^2 - X^3 \dots X^N$

### Algorithm

Step-1 Start

Step-2 Input Value of N, X

Step-3  $I = 1$ ,  $SUM = 1$ ,  $TERM = 1$

Step-4 IF ( $I > N$ ) THEN

GO TO Step-9

ENDIF

Step-5  $TERM = - TERM * X$

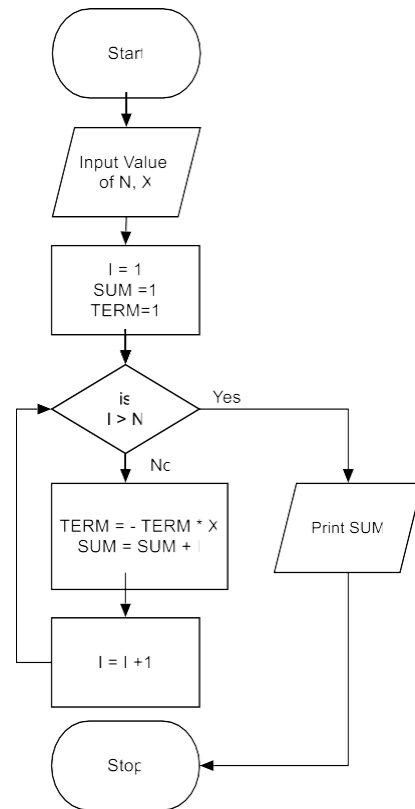
Step-6  $SUM = SUM + TERM$

Step-7  $I = I + 1$

Step-8 Go to step-4

Step-9 Display value of SUM

Step-10 Stop



## Algorithm & Flowchart to print multiplication Table of a number

### Algorithm

Step-1 Start

Step-2 Input Value of NUM

Step-3  $I = 1$

Step-4 IF ( $I > 10$ ) THEN

GO TO Step 9

ENDIF

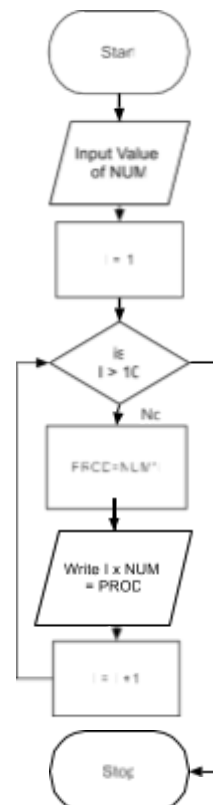
Step-5  $PROD = NUM * I$

Step-6 WRITE  $I \times$  NUM  $=$  PROD

Step-7  $I = I + 1$

Step-8 Go to step-4

Step-9 Stop



..

### Algorithm & Flowchart to generate first n Fibonacci terms 0,1,1,2,3,5...n ( $n > 2$ )

#### Algorithm

Step-1 Start

Step-2 Input Value of N

Step-3 A=0, B=1, COUNT=2

Step-4 WRITE A, B

Step-5 IF (COUNT > N) then go to step 12

Step-6 NEXT= A + B

Step-7 WRITE NEXT

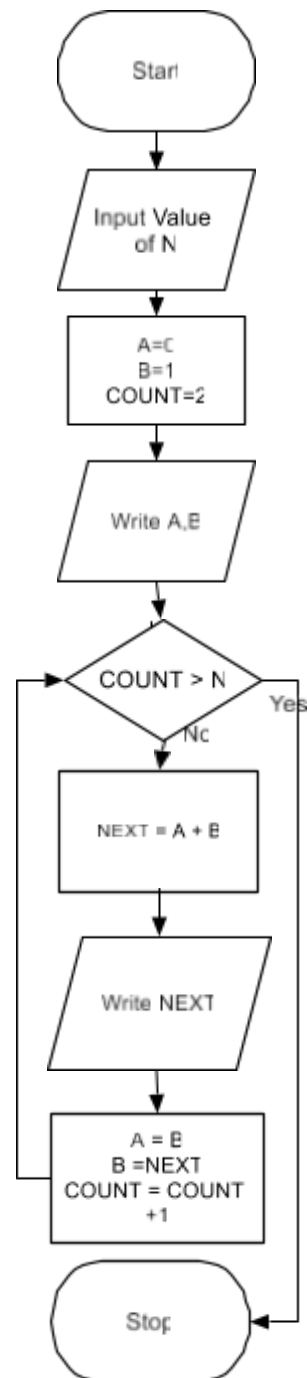
Step-8 A=B

Step-9 B=NEXT

Step-10 COUNT=COUNT + 1

Step-11 Go to step-4

Step-12 Stop



## Algorithm & Flowchart to find sum and average of given series of numbers

### Algorithm

Step-1 Start

Step-2 COUNT=0

Step-3 SUM=0

Step-4 Input NUM (next number in series)

Step-5 SUM= SUM +NUM

Step-6 COUNT=COUNT+1

Step-7 IF More Number in Series then

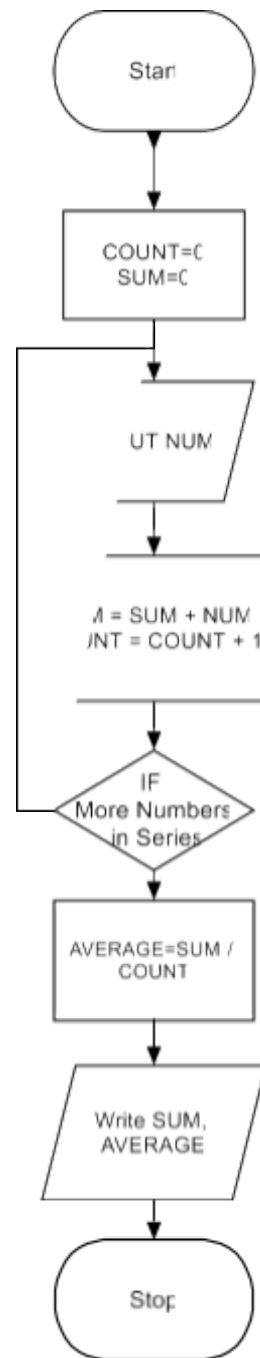
GOTO Step-4

ENDIF

Step-8 AVERAGE=SUM / COUNT

Step-9 WRITE SUM, AVERAGE

Step-10 Stop



## Algorithm & Flowchart to find Roots of Quadratic Equations $AX^2+BX+C=0$

### Algorithm

Step-1 Start

Step-2 Input A,B,C

Step-3  $DISC = B^2 - 4 A * C$

Step-4 IF (DISC < 0) THEN

Write Roots are Imaginary

Stop

ENDIF

Step-5 IF (DISC==0) THEN

Write Roots are Real and Equal

$X1 = -B/(2*A)$

Write Roots are X1,X1

Stop

ENDIF

Step-6 IF (DISC > 0)

Write Roots are Real and Unequal

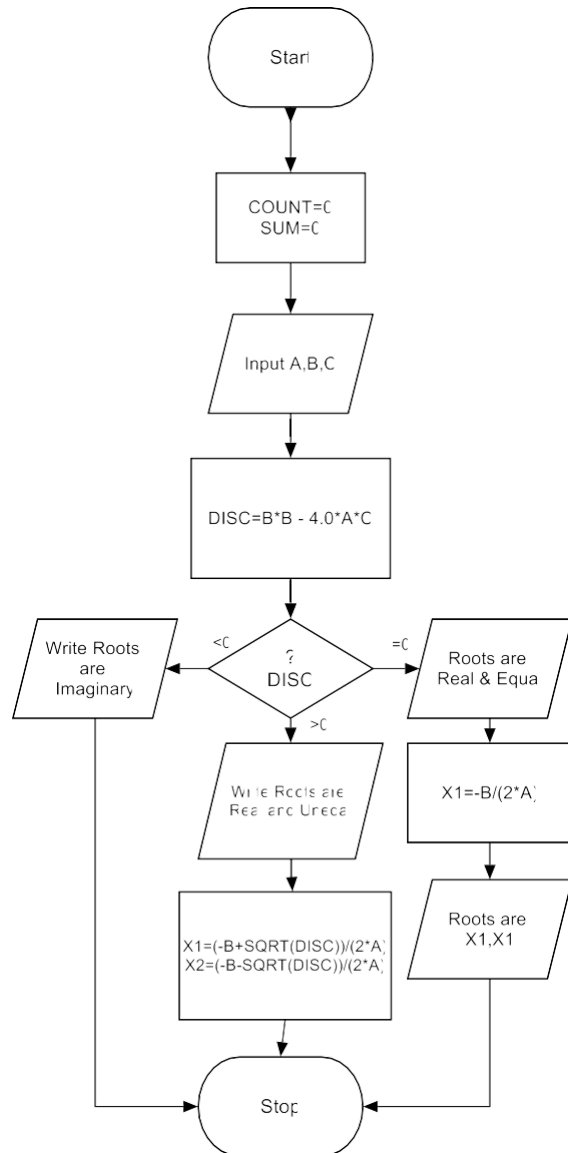
$X1 = (-B + \text{SQRT}(\text{DISC})) / (2*A)$

$X2 = (-B - \text{SQRT}(\text{DISC})) / (2*A)$

Write Roots are X1,X2

Stop

ENDIF



## Algorithm & Flowchart to find if a number is prime or not

### Algorithm

Step-1 Start

Step-2 Input NUM

Step-3  $R = \text{SQRT}(\text{NUM})$

Step-4  $I = 2$

Step-5 IF (  $I > R$  ) THEN

Write NUM is Prime Number

Stop

ENDIF

Step 6 IF (  $\text{NUM} \% I == 0$  ) THEN

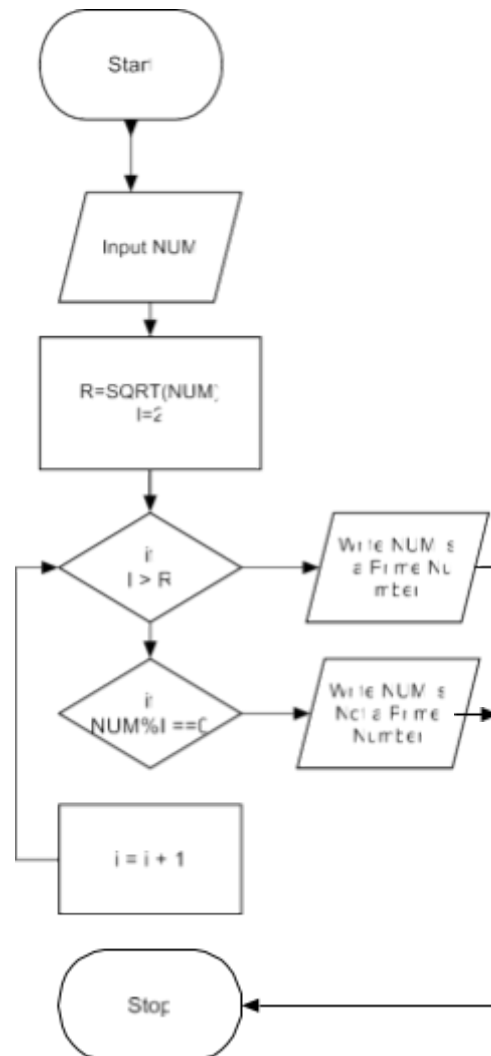
Write NUM is Not Prime

Stop

ENDIF

Step-7  $I = I + 1$

Step-8 Go to Step-5



## Algorithm & Flowchart to find GCD and LCM of two numbers

### Algorithm

Step-1 Start

Step-2 Read two number A, B

Step-3 IF (A > B) THEN

N = A

D = B

ELSE

N = B

D = A

ENDIF

Step-4  $r = N/D$

Step-5 WHILE (r !=

0) DO

N = D

D = r

$r = N \% D$

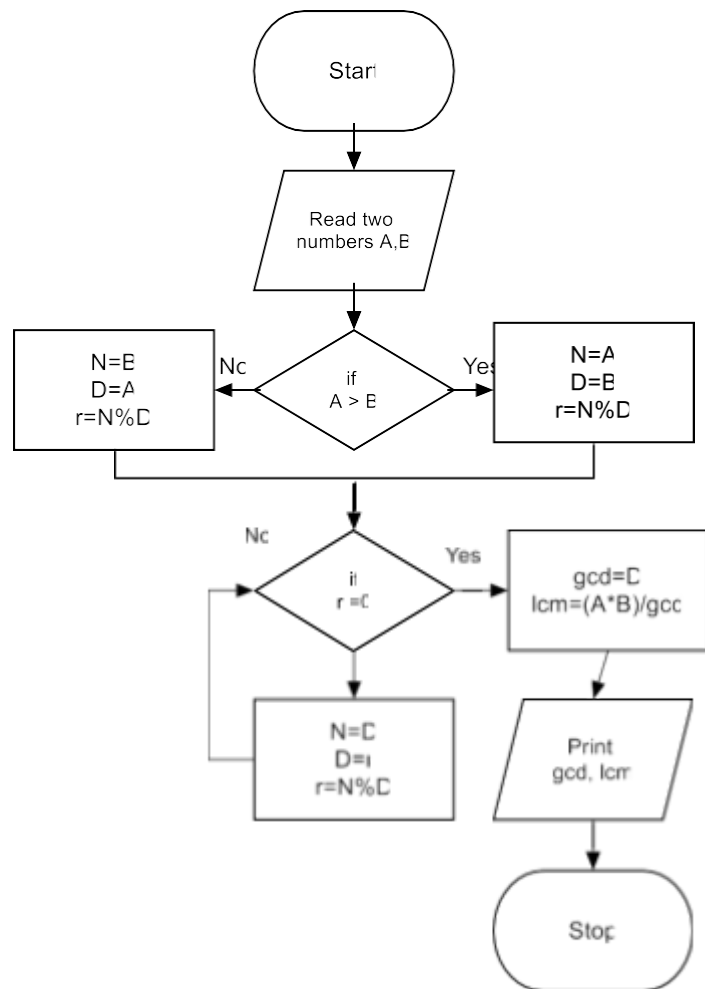
DONE

Step-6  $\text{gcd} = d$

Step-7  $\text{lcm} = (a*b)/\text{gcd}$

Step-8 Display gcd, lcm

Step-9 Stop



...

## Algorithm & Flowchart to find Factorial of number n ( $n!=1 \times 2 \times 3 \times \dots \times n$ )

### Algorithm

Step-1 Start

Step-2 Read number N

Step-3 FACT=1 CTRL=1

Step-4 WHILE (CTRL <= N)

DO

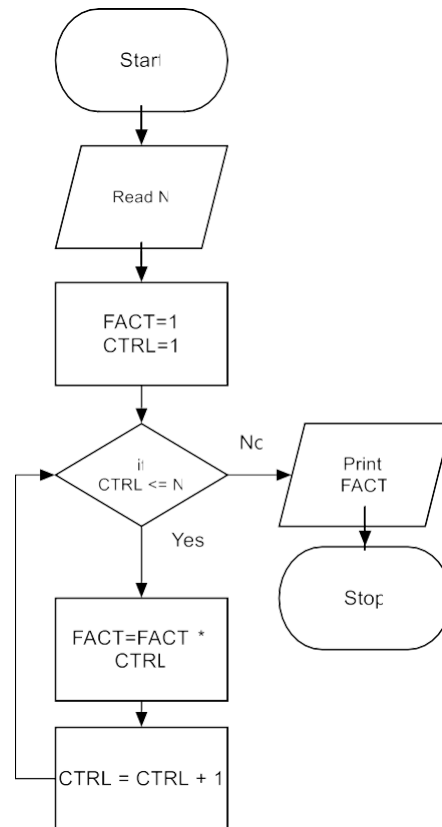
FACT=FACT\*I

CTRL=CTRL+1

DONE

Step-5 Display FACT

Step-6 Stop



## Algorithm & Flowchart to find all the divisor of a number

### Algorithm

Step-1 Start

Step-2 Read number N

Step-3 D=1

Step-4 WHILE (D <

N) DO

IF (N % D == 0) THEN

PRINT D

ENDIF

D=D+1

DONE

Step-5 Stop

