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.

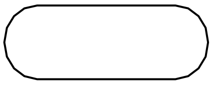
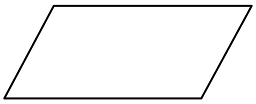
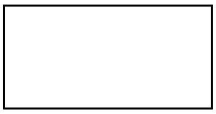
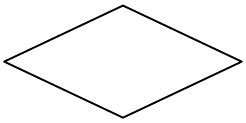
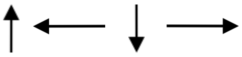
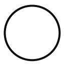
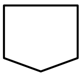
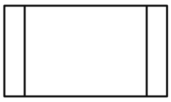
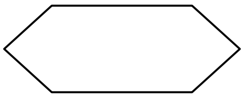
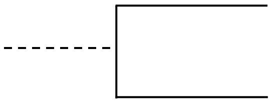
FLOWCHART

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 its operation. The flowchart makes program or system maintenance easier.
- It is easy to convert the flowchart into any programming language code.

Flowchart is graphical representation of sequence of steps to solve a problem. To draw a flowchart, the following standard symbols are use:

Symbol Name	Symbol	Function
Terminal		Used to represent start and end of flowchart.
Input/Output		Used for input and output operation
Process		Used for arithmetic operations and data-manipulations
Decision		Used to represent the operation in which there are two/three alternatives, true and false, yes or no, etc.
Flow line		Used to indicate the flow of logic by connecting symbols
On – page connector		Connects two or more parts of a flowchart, which are on the same page.
Off – page connector		Connects two parts of a flowchart which are spread over different pages
Predefined process/function		Used to represent a group of statements performing one processing task.
Preparation / Initialization		Preprocess. This is used to prepare memory for repetition of an action
Comments / Annotation		This is used to describe action or variables

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
^	Exponent	A^3 for A^3
%	Remainder / Modulo	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

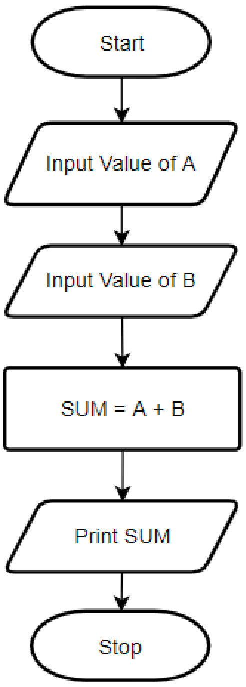
Selection Control	Example	Meaning
WHILE (Condition) DO .. .. ENDDO	WHILE (X < 10) DO print x x=x+1 ENDDO	Execute the loop as long as the condition is TRUE
DO …. … UNTILL (Condition)	DO print x x=x+1 UNTILL (X >10)	Execute the loop as long as the condition is false

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).

Problem: Find the sum of two numbers

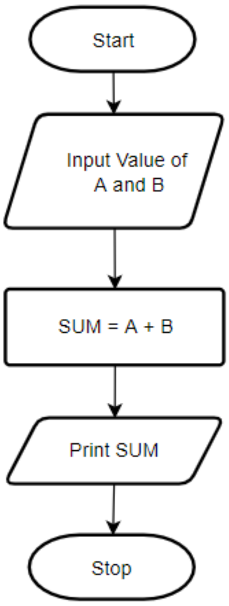
```
Step 1    Start
Step 2    Input first numbers say A
Step 3    Input first numbers say B
Step 4    SUM = A + B
Step 5    Display SUM
Step 6    Stop
```



OR

Problem: Find the sum of two numbers

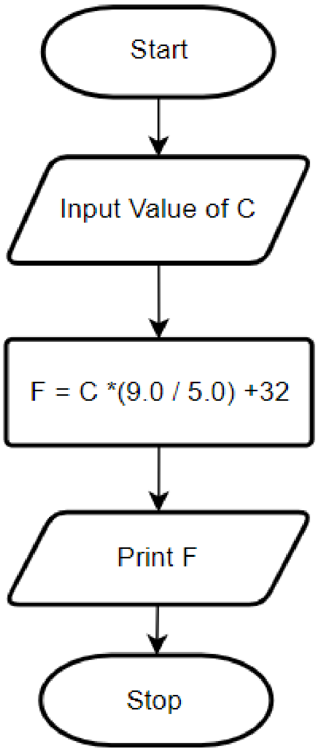
```
Step 1    Start
Step 2    Input first numbers say A and B
Step 3    SUM = A + B
Step 4    Display SUM
Step 5    Stop
```



Problem: Convert temperature from Celsius to Fahrenheit

C : temperature in Celsius
F : temperature in Fahrenheit

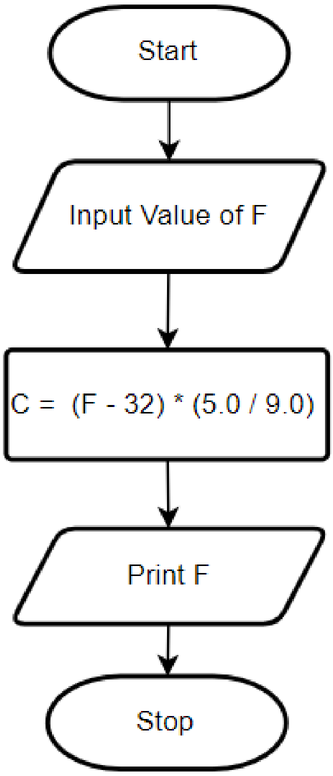
```
Step 1    Start
Step 2    Input temperature in Celsius (C)
Step 3    F = C *(9.0 / 5.0) +32
Step 4    Display temperature in Fahrenheit (F)
Step 5    Stop
```



Problem: Convert temperature from Fahrenheit to Celsius

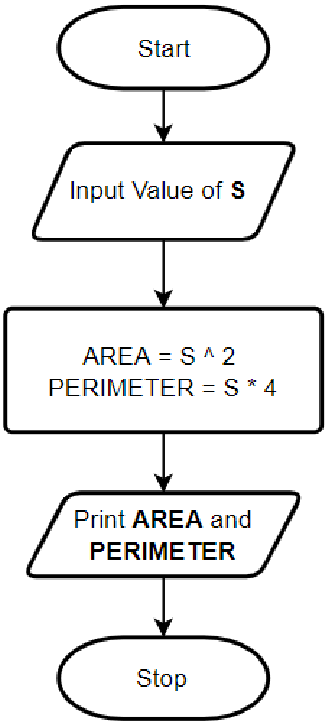
C : temperature in Celsius
F : temperature in Fahrenheit

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



Problem: Find the Area and Perimeter of a Square

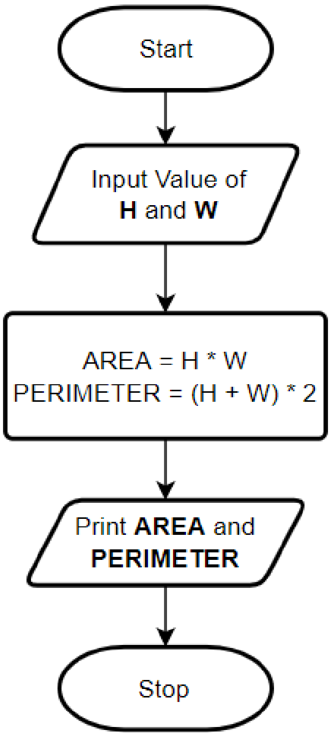
S : Side length
AREA : Area of a Square
PERIMETER : Perimeter of a Square



Step 1	Start
Step 2	Input side length (S)
Step 3	AREA = S x S
Step 4	PERIMETER = 4 * S
Step 5	Display AREA and PERIMETER
Step 6	Stop

Problem: Find the Area and Perimeter of a Rectangle

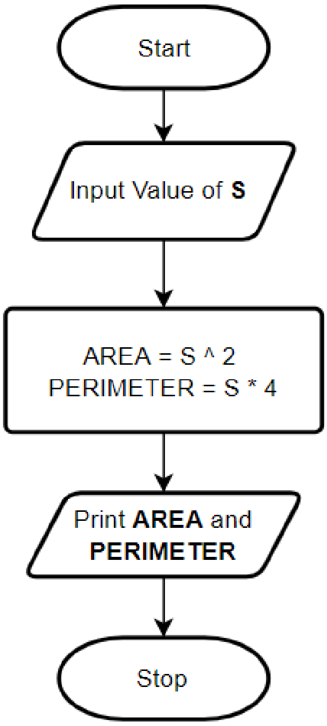
H : Height W: Width
AREA : Area of a Rectangle
PERIMETER : Perimeter of a Rectangle


Start
Input Value of **H** and **W**
AREA = H * W
PERIMETER = (H + W) * 2
Print **AREA** and **PERIMETER**
Stop

Step 1	Start
Step 2	Input height (H) and width (W)
Step 3	AREA = H x W
Step 4	PERIMETER = (H + W) * 2
Step 5	Display AREA and PERIMETER
Step 6	Stop

Problem: Find the Area and Perimeter of a Circle

R : Radius
AREA : Area of a circle
PERIMETER : Perimeter of a circle (circumference)

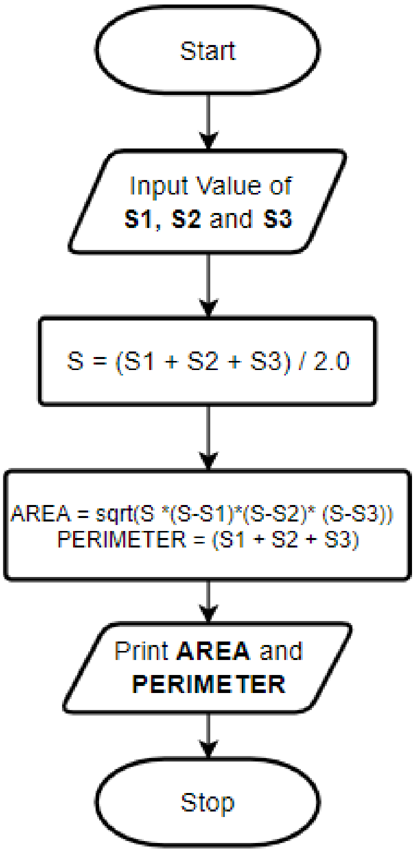


```
Step 1    Start
Step 2    Input radius  (R)
Step 3    AREA = (22.0 / 7.0) * R ^ 2
Step 4    PERIMETER = 2 * (22.0/ 7.0) * R
Step 5    Display AREA and PERIMETER
Step 6    Stop
```

NOTE: $\pi = 22.0 / 7.0$

Problem: Find the Area and Perimeter of a Triangle

S1 : First side of the triangle
S2 : First side of the triangle
S3 : First side of the triangle
AREA : Area of a triangle
PERIMETER : Perimeter of a triangle


Start
Input Value of **S1**, **S2** and **S3**
S = (S1 + S2 + S3) / 2.0
AREA = sqrt(S *(S-S1)*(S-S2)* (S-S3))
PERIMETER = (S1 + S2 + S3)
Print **AREA** and **PERIMETER**
Stop

```
Step 1    Start
Step 2    Input sides of the triangle (S1, S2, S3)
Step 3    S = ( S1 + S2 + S3 ) / 2.0
Step 4    AREA = sqrt ( S * (S – S1) *(S – S2)* (S – S3))
Step 5    PERIMETER = S1 + S2 + S3
Step 6    Display AREA and PERIMETER
Step 7    Stop
```

Note : S : Semi perimeter

Problem: Swap two numbers using temporary variable

Step 1 Start

Step 2 Input two numbers (NUM1 and NUM2)

Step 3 Display NUM1 and NUM2 before swap

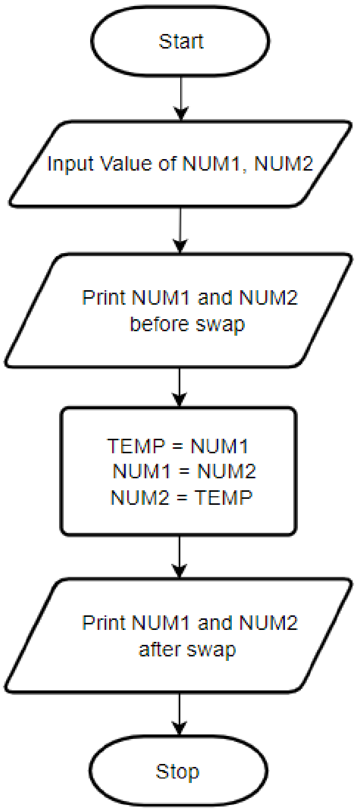
Step 4 TEMP = NUM1

Step 5 NUM1 = NUM2

Step 6 NUM2 = TEMP

Step 7 Display NUM1 and NUM2 after swap

Step 8 Stop


Start
Input Value of NUM1, NUM2
Print NUM1 and NUM2 before swap
TEMP = NUM1
NUM1 = NUM2
NUM2 = TEMP
Print NUM1 and NUM2 after swap
Stop

Problem: Swap two numbers without using temporary variable

Step 1 Start

Step 2 Input two numbers (NUM1 and NUM2)

Step 3 Display NUM1 and NUM2 before swap

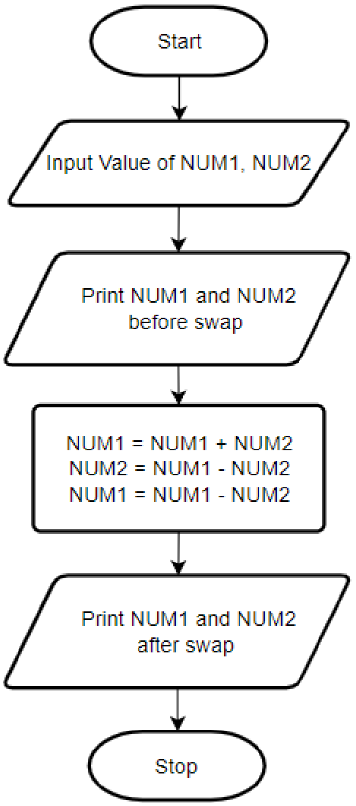
Step 4 NUM1 = NUM1 + NUM2

Step 5 NUM2 = NUM1 – NUM2

Step 6 NUM1 = NUM1 – NUM2

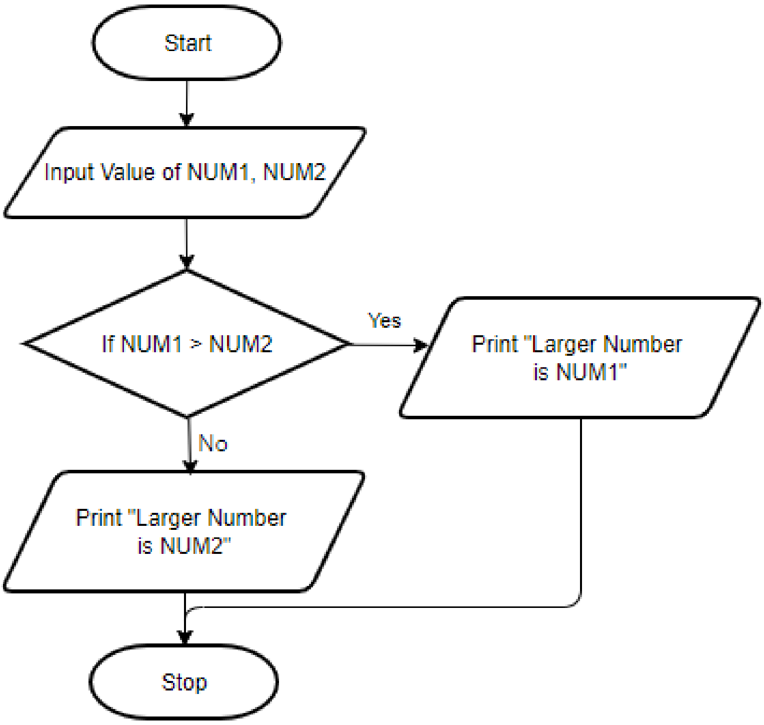
Step 7 Display NUM1 and NUM2 after swap

Step 8 Stop


Start
Input Value of NUM1, NUM2
Print NUM1 and NUM2 before swap
NUM1 = NUM1 + NUM2
NUM2 = NUM1 - NUM2
NUM1 = NUM1 - NUM2
Print NUM1 and NUM2 after swap
Stop

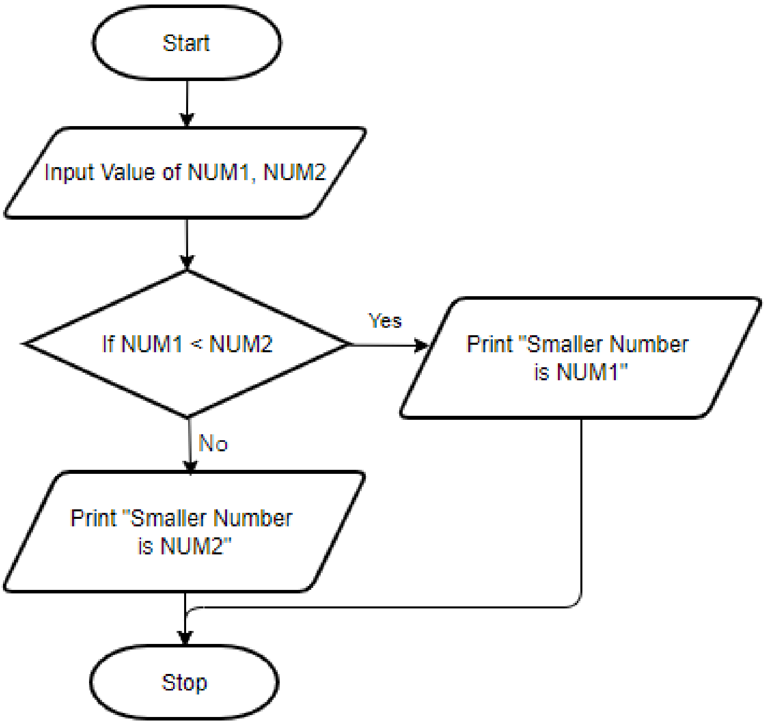
Problem: Find the larger of two numbers

```
Step 1      Start

Step 2      Input two numbers
            (NUM1 and NUM2)

Step 3      IF NUM1 > NUM2
            THEN
                Display NUM1
            ELSE
                Display NUM2
            ENDIF

Step 4      Stop
```


Start
Input Value of NUM1, NUM2
If NUM1 > NUM2
Yes
Print "Larger Number is NUM1"
No
Print "Larger Number is NUM2"
Stop

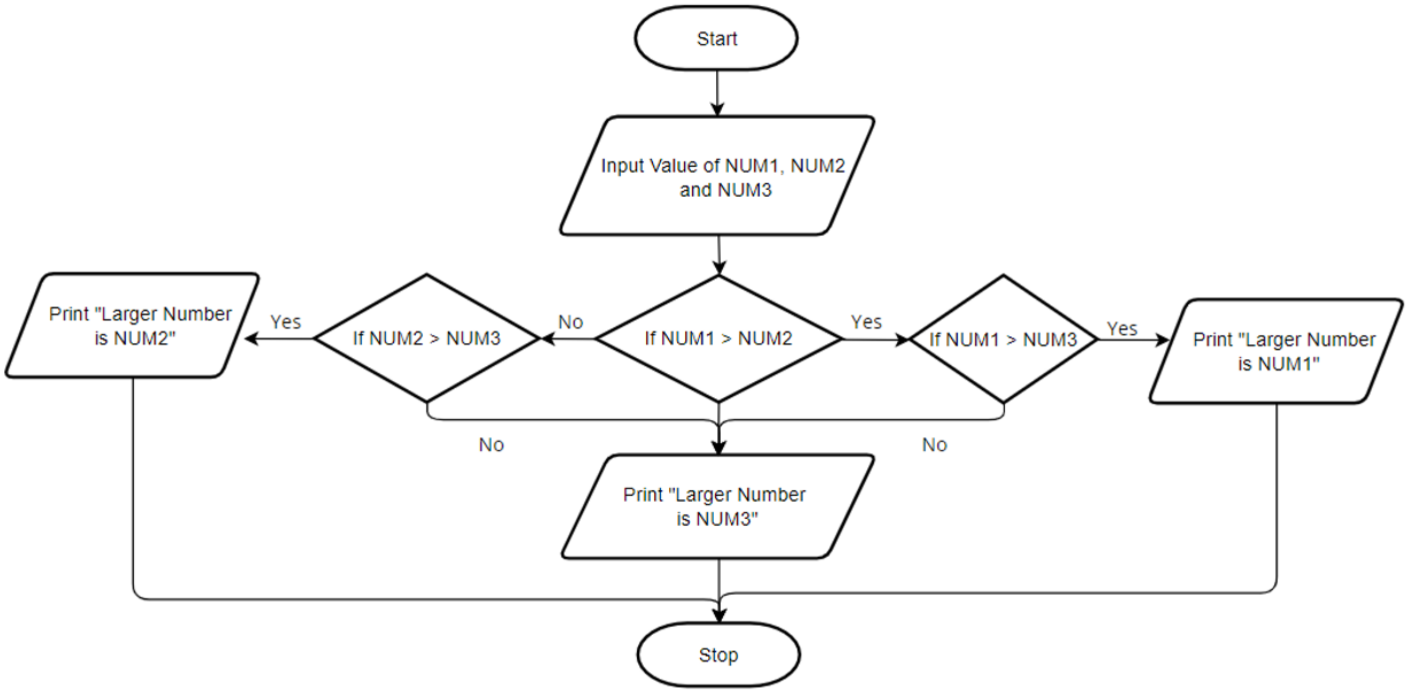
Problem: Find the smaller of two numbers

```
Step 1   Start

Step 2   Input two numbers
         (NUM1 and NUM2)

Step 3   IF NUM1 < NUM2 THEN
             Display NUM1
         ELSE
             Display NUM2
         ENDIF

Step 4   Stop
```


Start
Input Value of NUM1, NUM2
If NUM1 < NUM2
Yes
Print "Smaller Number is NUM1"
No
Print "Smaller Number is NUM2"
Stop

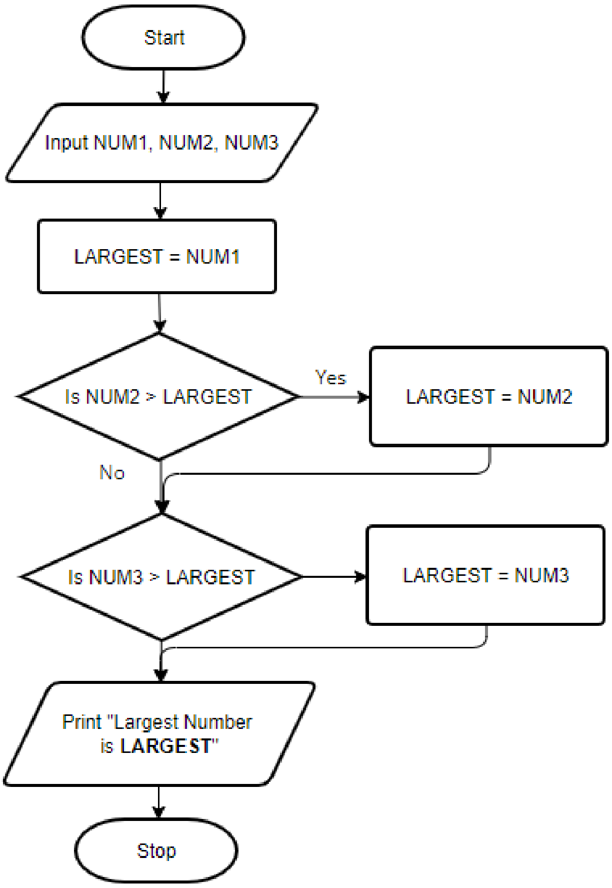
Problem: Find the largest of three numbers

```
Step 1   Start

Step 2   Input three numbers (NUM1, NUM2 and NUM3)

Step 3   IF NUM1 > NUM2 THEN
             GOTO Step 5
         ENDIF

Step 4   IF NUM2 > NUM3 THEN
             Display NUM2
         ELSE
             Display NUM3
         ENDIF
         GOTO Step 6

Step 5   IF NUM1 > NUM3 THEN
             Display NUM1
         ELSE
             Display NUM3
         ENDIF

Step 6   Stop
```


Start
Input Value of NUM1, NUM2 and NUM3
If NUM1 > NUM2
No
Yes
If NUM2 > NUM3
Yes
Print "Larger Number is NUM2"
If NUM1 > NUM3
Yes
Print "Larger Number is NUM1"
No
No
Print "Larger Number is NUM3"
Stop

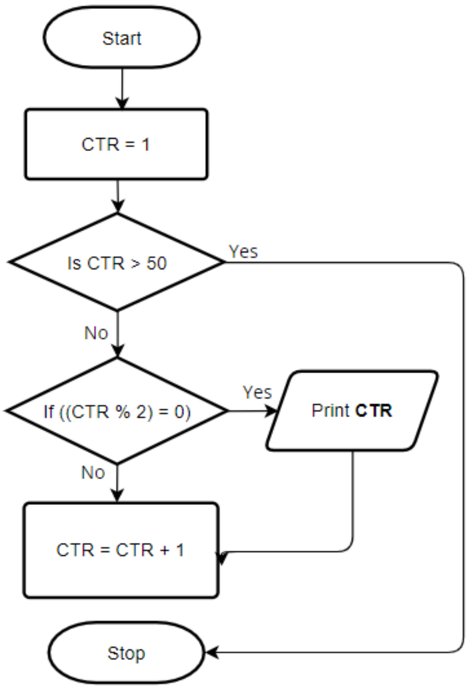
Problem: Find the largest of three numbers (another way)

```
Step 1   Start

Step 2   Input three numbers
         (NUM1, NUM2 and NUM3)

Step 3   LARGEST = NUM1

Step 4   IF NUM2 > LARGEST THEN
             LARGEST = NUM2
         ENDIF

Step 4   IF NUM3 > LARGEST THEN
             LARGEST = NUM3
         ENDIF

Step 5   Display LARGEST

Step 6   Stop
```



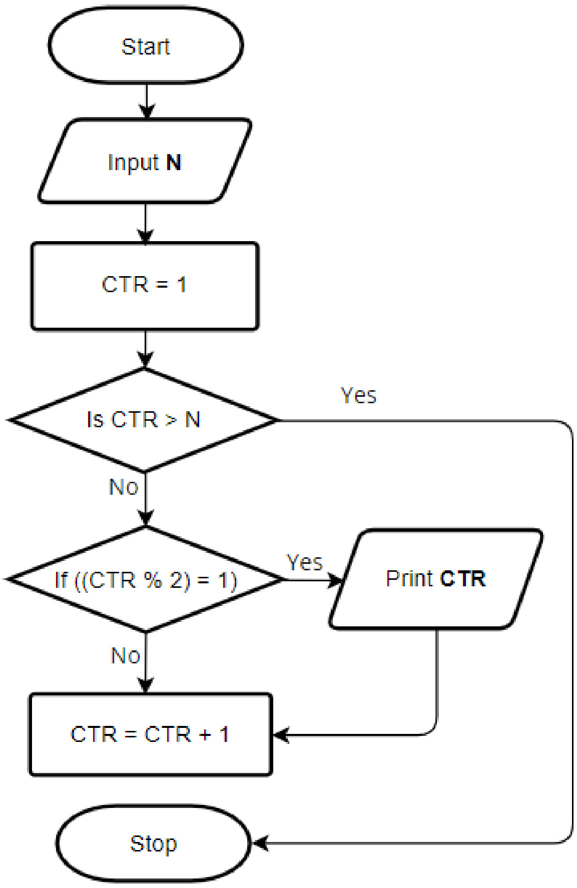
Problem: Find the EVEN number between 1 to 50

```
Step 1   Start

Step 2   CTR = 1

Step 3   IF (CTR > 50) THEN
             GOTO Step 7
         ENDIF

Step 4   IF ((ctr % 2 ) = 0) THEN
             Display CTR
         ENDIF

Step 5   CTR = CTR + 1

Step 6   GOTO Step 3

Step 7   Stop
```



Problem: Find the ODD number between 1 to N

```
Step 1    Start
Step 2    Input value for N
Step 3    CTR = 1
Step 3    IF (CTR > N) THEN
              GOTO Step 7
          ENDIF
Step 4    IF ((ctr % 2 ) = 1) THEN
              Display CTR
          ENDIF
Step 5    CTR = CTR + 1
Step 6    GOTO Step 3
Step 7    Stop
```



Start
Input N
CTR = 1
Is CTR > N
Yes
No
If ((CTR % 2) = 1)
Yes
Print CTR
No
CTR = CTR + 1
Stop