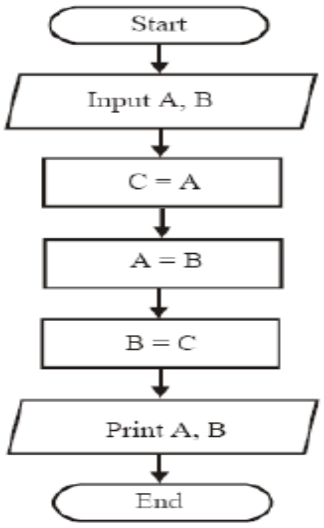
Example 7:

Write algorithm for swapping contents of two variables.

Algorithm:

1. Start
2. Read value of A
3. Read value of B
4. C=A
5. A=B
6. B=C
7. Print A
8. Print B
9. End



Example 8:

Write an algorithm to find the summation of numbers from 0 to 15.

1. Start
2. I=0
3. Sum=0
4. Sum=Sum+I
5. if I =15 then
   Goto 6
   else
   I=I+1
   Goto 4
   endif
6. print Sum
7. stop

Example 9:

Write an algorithm to sum the odd numbers between 10 and 200.

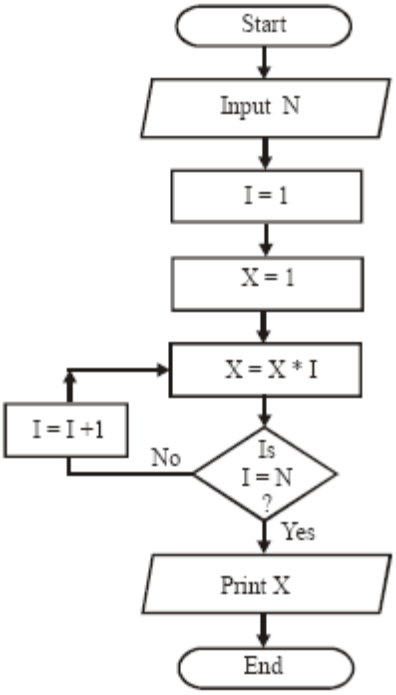
1. Start
2. I=11
3. Sum=0
4. Sum=Sum+I
5. if I = 199 then
   Goto 6
   else
   I=I+2
   Goto 4
   endif
6. print Sum
7. stop

Note: Can find the even numbers in the same way.

Example 10:

Write an algorithm for calculating and printing factorial (!) of a given number.
(n!=1*2*3.....*n)

1. Start
2. Input n
3. I=1
4. X=1
5. X=X*I
6. if I = n then
   Goto 6
   else
   I=I+1
   Goto 5
   endif
6. print X
7. stop



Example 11:

Write an algorithm to find S=x/3+x^2/5+x^3/7+.x^n/2n+1

1. Start
2. Input x, n
3. I=1
4. S=0
5. S=S+ X^I / 2*I+1
6. if I = n then
   Goto 6
   else
   I=I+1
   Goto 5
   endif
6. print S
7. stop

H.W: Write an algorithm to calculate T=X!+10

H.W: Write an algorithm to calculate M=1/x!+2/(x!)^2+3/(x!)^3….n/(x!)^n

Example 12:

Write an algorithm to find out the number is odd or even.

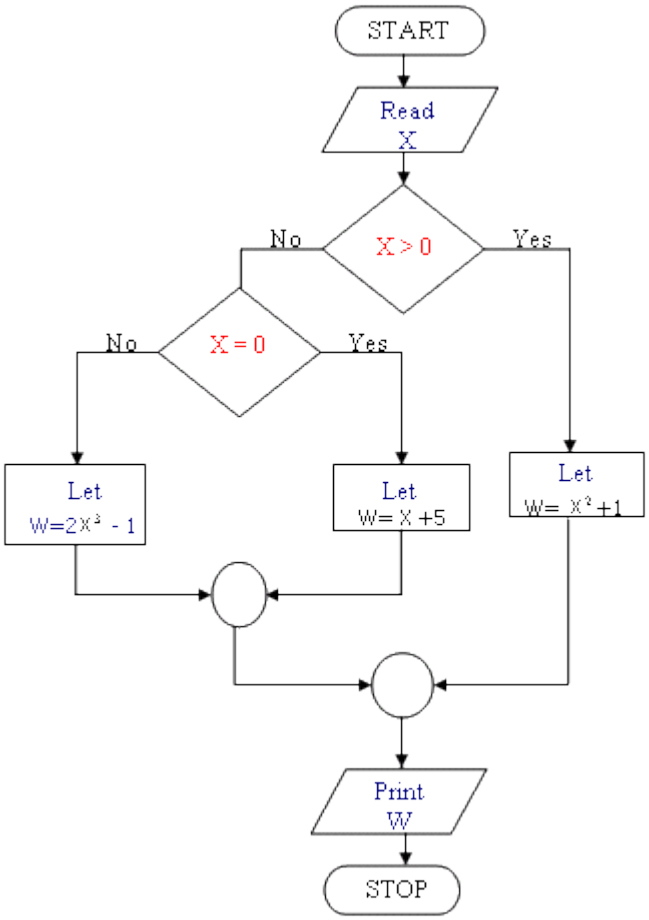
1. Start
2. Input x
3. D = X/2
4. If D = 0 then

Print “Even”
Else
Print “Odd”
5. Stop

Example 13:

Draw the flowchart to find W:

$$W=\begin{cases} X^2+1 ; X>0 \\ X+5 ; X=0 \\ 2X^3-1 ; X<0 \end{cases}$$



Conclusion:

***Algorithm:**

- A step-by-step procedure for solving a problem in a finite amount of time.
- Algorithms can be represented using Flow Charts.

*** Characteristics of an algorithm:**

1. Algorithms always have a definite starting point and an end point. These points are generally marked with the words like Start, Begin, End, Stop etc.
2. They consist of finite number of steps.
3. They always relate to a specific problem or you can say that they are written for a given problem.
4. They serve as *foundation stone for programming*.
5. They are written in easy language.

*** Problems with algorithms:**

As mentioned earlier, algorithms are written in English like sentences. Sentences are always subject to misinterpretation. If the sentence is complex, different readers may interpret it differently. This definitely leads to problems at the time of coding. To overcome this problem, often the solution of the problem is provided in **pictorial** form. Pictures carry more meaning and don't lead to ambiguity. Such type of step-by-step solution, provided in pictorial form is called flowchart.