Algorithm characteristics

1. **It should have finite number of steps**. No one can be expected to execute infinite number of steps.
2. The steps must be in order and simple
3. **Each step should be defined clearly stated i.e. without un-ambiguity (without doubtfulness)**
4. Must include all required information
5. Should exhibit at least one output

Algorithm	Flowchart	Program
An algorithm is defined as **sequence of steps to solve a problem (task)**.	**A flowchart is pictorial (graphical) representation of an algorithm**.	Set of instructions. Instruction is a command to the computer to do some task.
Algorithm can also be defined as a plan to solve a problem and represents its logic.	A picture is worth of 1000 words. We can understand more from picture than words.	Implementation of Algorithm or flowchart

Different algorithms have different performance characteristics to solve the same problem. Some algorithms are fast. Some are slow. Some occupy more memory space. Some occupy less memory space. Some are complex and some algorithms are simple.

Logically algorithm, flowchart and program are the same.

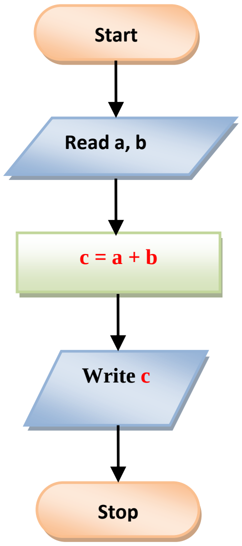
Examples of Algorithms and Flowcharts (with C code)

1. To find sum of two numbers

Algorithm

1. Start
2. Read a, b
3. c = a + b
4. Print or display c
5. Stop

Flowchart



Program

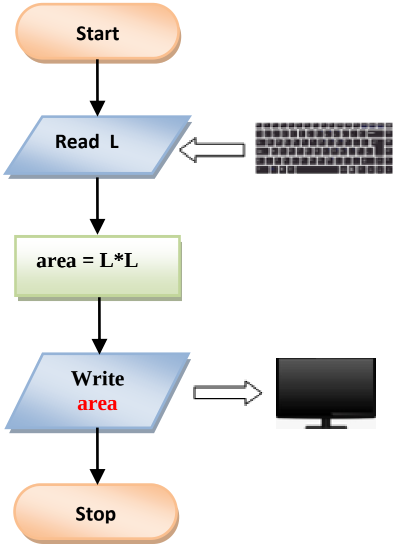
```
#include<stdio.h>

int main()
{
    int a, b, c;

    printf("Enter value of a: ");
    scanf("%d", &a);

    printf("Enter value of b: ");
    scanf("%d", &b);
    c = a+b;

    printf("Sum of given two numbers is: %d", c);

return 0;
}
```

2. Finding Area of the square

Algorithm

1. Start
2. Read length, L
3. area = L*L
4. Print or display area
5. Stop

Flowchart



Program

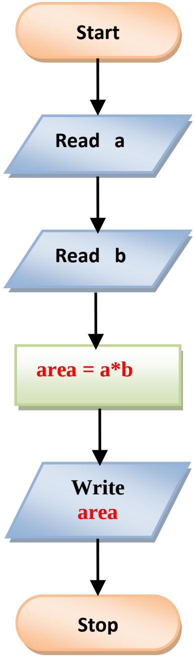
```
#include<stdio.h>

int main()
{

    int L, area;

    printf("Enter length of square L: ");
    scanf("%d", &L);

    area = L*L;

    printf("Area of square is: %d", area);

    return 0;
}
```

3. Finding Area of the rectangle

Algorithm

1. Start
2. Read side length, a
3. Read side length b
4. area = a*b
5. Print or display area
6. Stop

Flowchart



Program

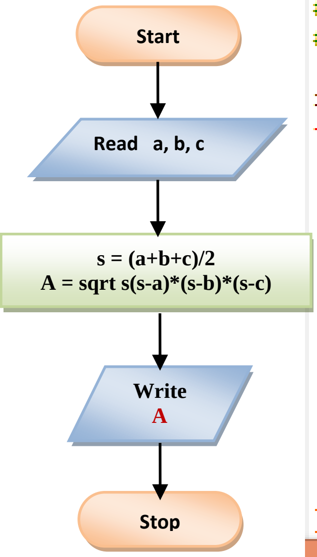
```
#include<stdio.h>

int main()
{

    int a, b, area;
    printf("Enter side length a: \n");
    scanf("%d", &a);

    printf("Enter side length b: \n");
    scanf("%d", &b);

    area = a*b;

    printf("Area of rectangle is: %d ", area);

    return 0;
}
```

4. Area of a triangle where three sides are given

Algorithm

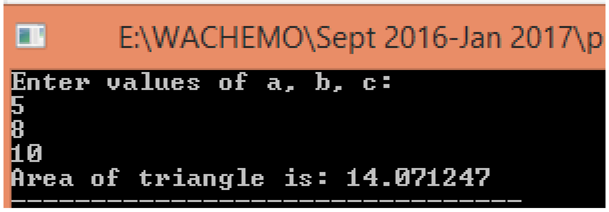
1. Start
2. Read a, b, c
3. s = (a+b+c)/2
4. A=sqrt (s *(s-a)*(s-b)*(s-c))
5. Print or display A
6. Stop

Flowchart



Program

```
#include<stdio.h>
#include<math.h>

int main()
{
    int a, b, c;
    float s, A;

    printf("Enter values of a, b, c: \n");
    scanf("%d %d %d", &a, &b, &c);

    s = (a+b+c)/2;
    A = sqrt (s*(s-a)*(s-b)*(s-c));

    printf("Area of triangle is: %f", A);

    return 0;
}
```



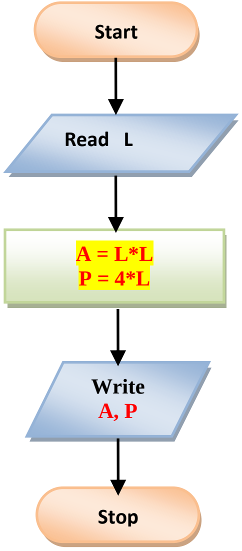
```
Enter values of a, b, c:
5
8
10
Area of triangle is: 14.071247
```

5. Find the area & perimeter of a square

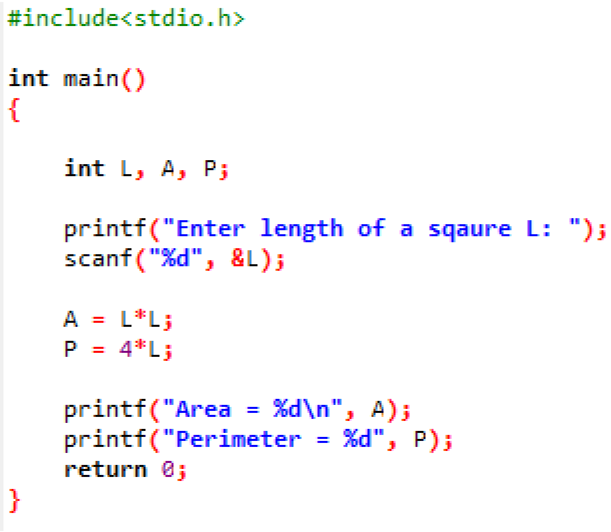
Algorithm

1. Start
2. Read length L
3. Area A = L*L
4. Perimeter P = 4*L
5. Print or display A,P
6. Stop

Flowchart



Program



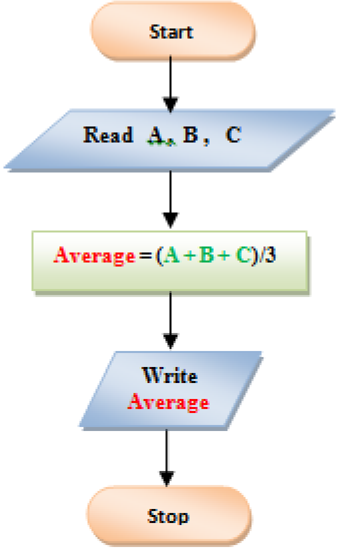
```
#include<stdio.h>

int main()
{

    int L, A, P;

    printf("Enter length of a sqaure L: ");
    scanf("%d", &L);

    A = L*L;
    P = 4*L;

    printf("Area = %d\n", A);
    printf("Perimeter = %d", P);
    return 0;
}
```

6. Calculating the average for 3 numbers

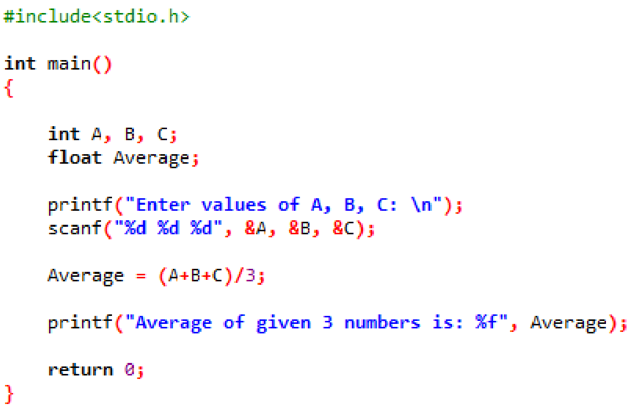
Algorithm

1. Start
2. Read 3 numbers A, B, C
3. Calculate the average by the equation:
   Average = (A + B + C)/3
4. Print average
5. Stop

Flowchart



Program



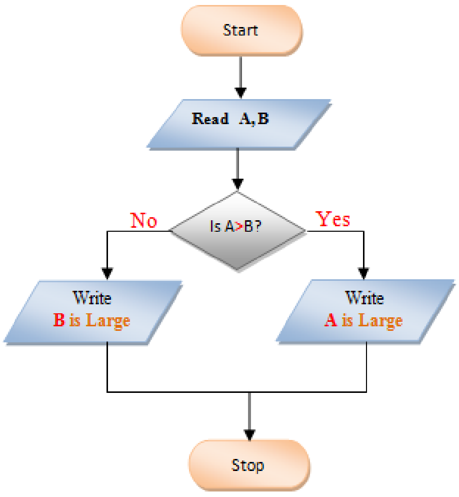
```
#include<stdio.h>

int main()
{

    int A, B, C;
    float Average;

    printf("Enter values of A, B, C: \n");
    scanf("%d %d %d", &A, &B, &C);

    Average = (A+B+C)/3;

    printf("Average of given 3 numbers is: %f", Average);

    return 0;
}
```

7. Greatest of two numbers

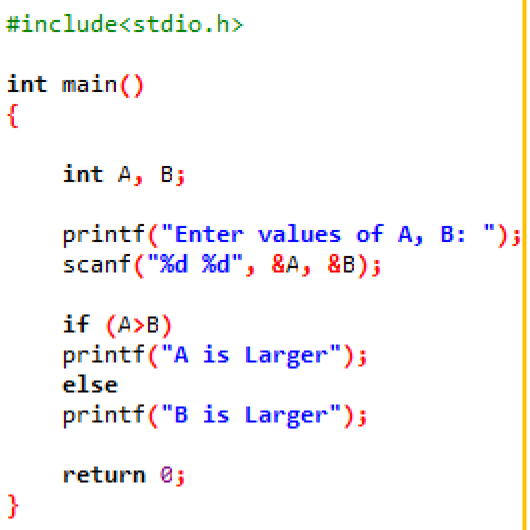
Algorithm

```
1. Start
2. Read A,B
3. If A > B then
           Print A is large
   else
           Print B is large
4. Stop
```

Flowchart



Program



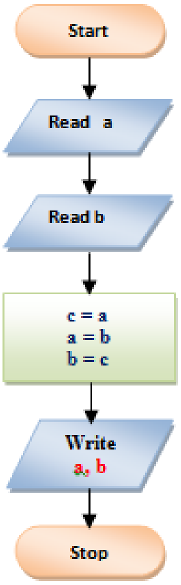
```
#include<stdio.h>

int main()
{

    int A, B;

    printf("Enter values of A, B: ");
    scanf("%d %d", &A, &B);

    if (A>B)
    printf("A is Larger");
    else
    printf("B is Larger");

    return 0;
}
```

8. Interchange the value of two numbers

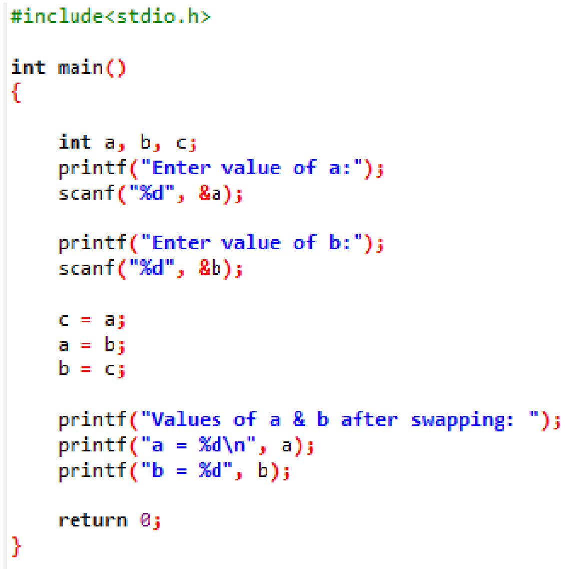
Algorithm

```
1. Start
2. Read two values into two variables a, b
3. Declare third variable, c
           c = a
           a = b
           b = c
4. Print or display a, b
5. Stop
```

Flowchart



Program



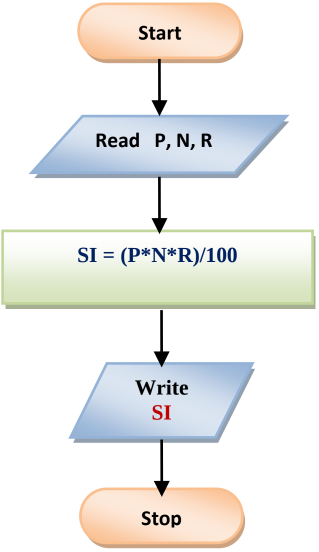
```
#include<stdio.h>

int main()
{

    int a, b, c;
    printf("Enter value of a:");
    scanf("%d", &a);

    printf("Enter value of b:");
    scanf("%d", &b);

    c = a;
    a = b;
    b = c;

    printf("Values of a & b after swapping: ");
    printf("a = %d\n", a);
    printf("b = %d", b);

    return 0;
}
```

9. Calculate simple interest using the expression (SI=PNR/100)

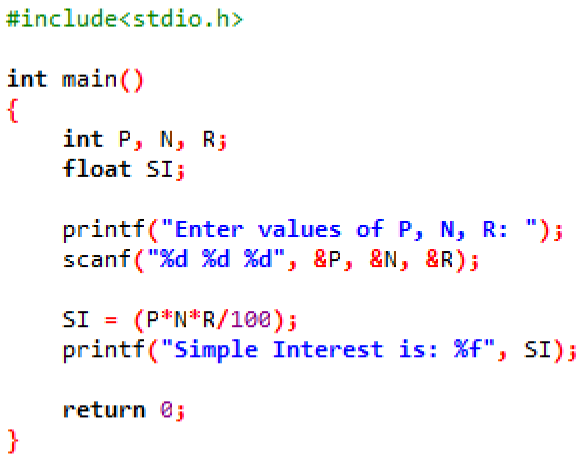
Algorithm

1. Start
2. Read P, N, R
3. SI=(PNR)/100
4. Print SI
5. Stop

Flowchart



Program



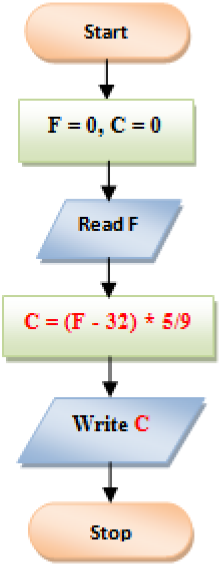
```
#include<stdio.h>

int main()
{
    int P, N, R;
    float SI;

    printf("Enter values of P, N, R: ");
    scanf("%d %d %d", &P, &N, &R);

    SI = (P*N*R/100);
    printf("Simple Interest is: %f", SI);

    return 0;
}
```

10. Convert temperature from Fahrenheit to Celsius

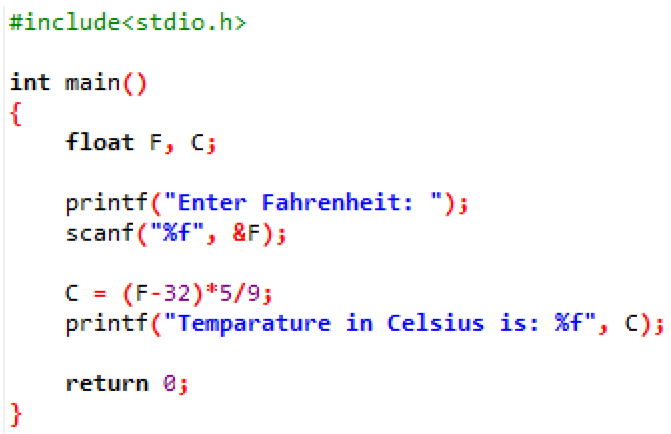
Algorithm

1. Start
2. Initialize F = 0, C = 0
3. Read F
4. C = (F-32) * 5/9
5. Write C
6. Stop

Flowchart



Program



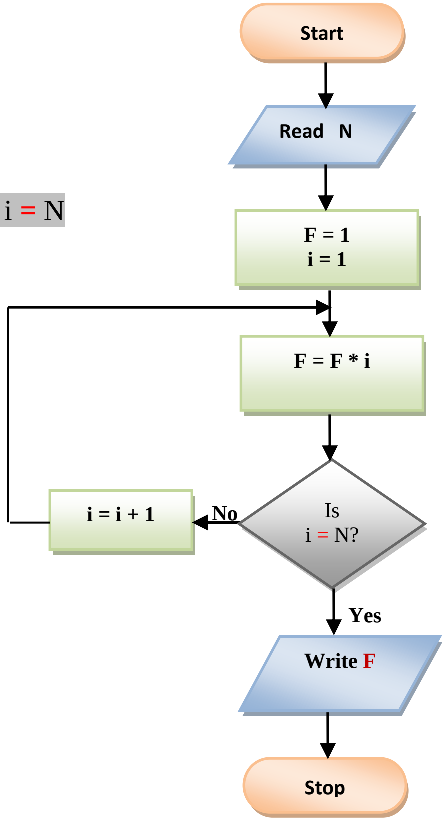
```
#include<stdio.h>

int main()
{
    float F, C;

    printf("Enter Fahrenheit: ");
    scanf("%f", &F);

    C = (F-32)*5/9;
    printf("Temparature in Celsius is: %f", C);

    return 0;
}
```

11. Draw a flowchart for computing factorial N, where N! = 1 * 2 * 3 * …… N

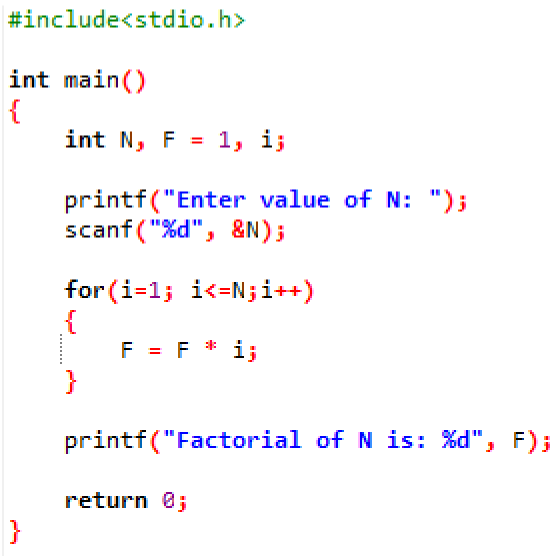
Algorithm

1. Start
2. Read N
3. Initialize F =1, i = 1
4. F = F * i;
5. Increment i by 1
6. Repeat step 4 & 5 until i = N
7. Print F
8. Stop

Flowchart



Program



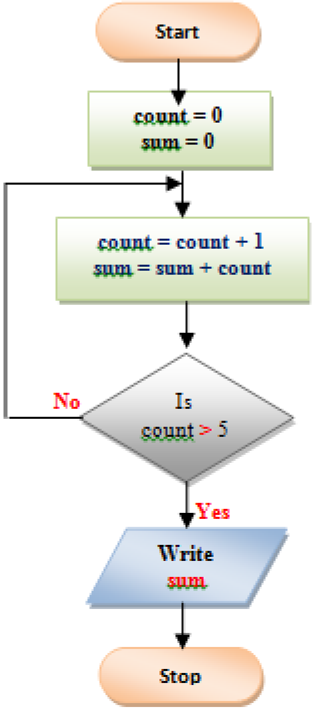
```
#include<stdio.h>

int main()
{
    int N, F = 1, i;

    printf("Enter value of N: ");
    scanf("%d", &N);

    for(i=1; i<=N;i++)
    {
        F = F * i;
    }

    printf("Factorial of N is: %d", F);

    return 0;
}
```

12. Find the Sum of First Five Natural Numbers

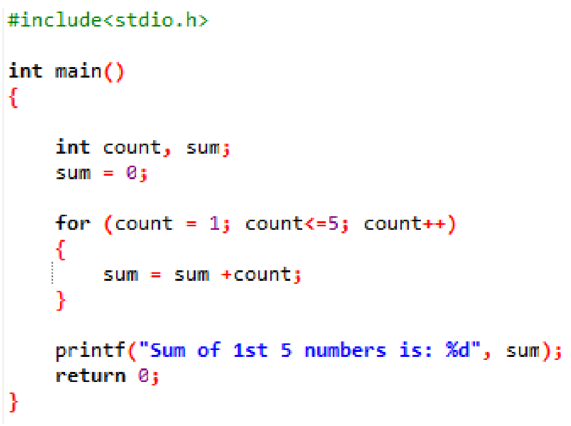
Algorithm

1. Start
2. Initialize count = 0, sum = 0
3. count = count + 1
4. sum = sum + count
5. Repeat steps 3,4 until count > 5
6. Print sum
7. Stop

Flowchart



Program



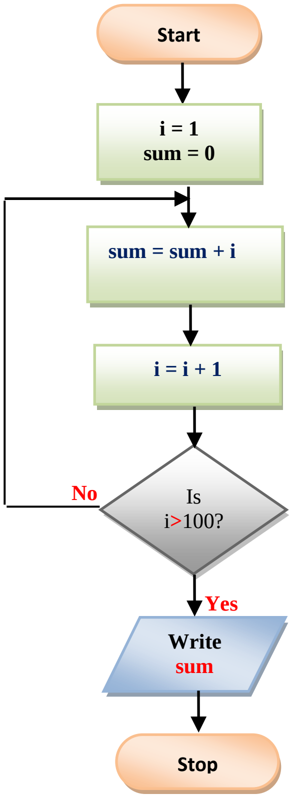
```
#include<stdio.h>

int main()
{

    int count, sum;
    sum = 0;

    for (count = 1; count<=5; count++)
    {
        sum = sum +count;
    }

    printf("Sum of 1st 5 numbers is: %d", sum);
    return 0;
}
```

13. Calculating sum of integers 1 to 100

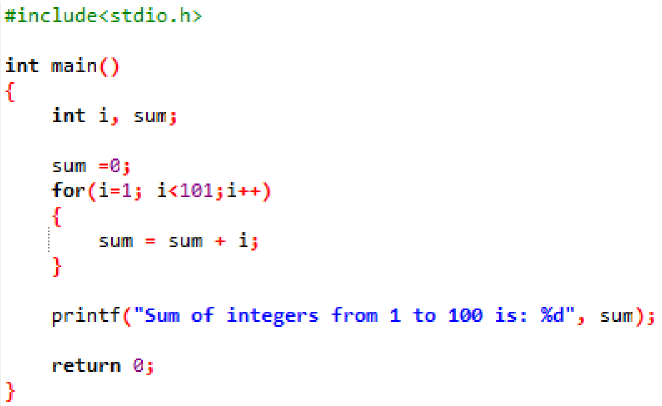
Algorithm

1. Start
2. Initialize count i = 1, sum = 0
3. sum = sum + i
4. Increment i by 1
5. Repeat steps 3 & 4 until i > 100
6. Print sum
7. Stop

Flowchart



Program



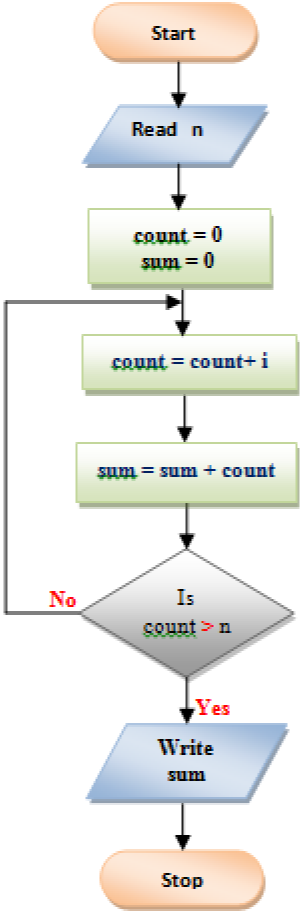
```
#include<stdio.h>

int main()
{
    int i, sum;

    sum =0;
    for(i=1; i<101;i++)
    {
        sum = sum + i;
    }

    printf("Sum of integers from 1 to 100 is: %d", sum);

    return 0;
}
```

14. To find the sum of n natural Numbers

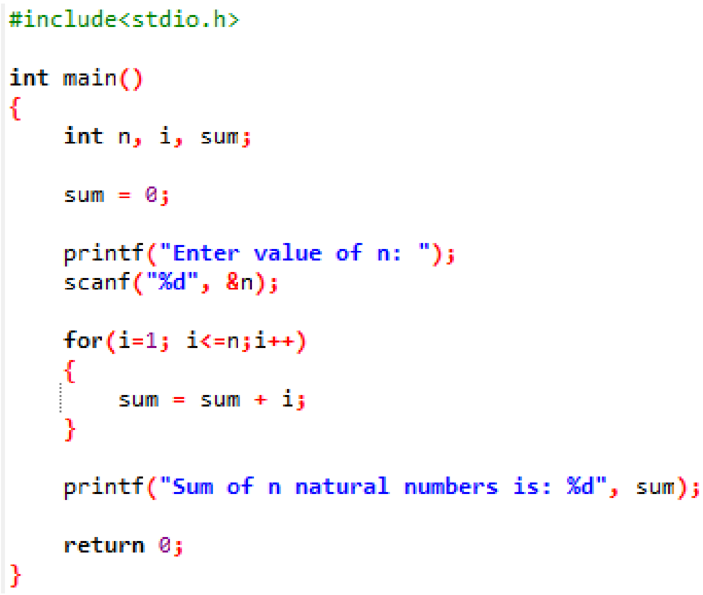
Algorithm

1. Start
2. Read n
3. count=0
4. sum=0
5. count = count + 1
6. sum = sum + count
7. Repeat steps 5 & 6 until count > n
8. Print sum
9. Stop

Flowchart



Program



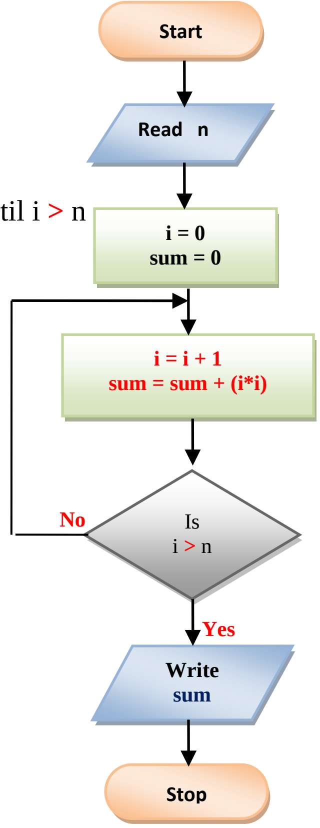
```
#include<stdio.h>

int main()
{
    int n, i, sum;

    sum = 0;

    printf("Enter value of n: ");
    scanf("%d", &n);

    for(i=1; i<=n;i++)
    {
        sum = sum + i;
    }

    printf("Sum of n natural numbers is: %d", sum);

    return 0;
}
```

15. Sum of squares of n natural numbers

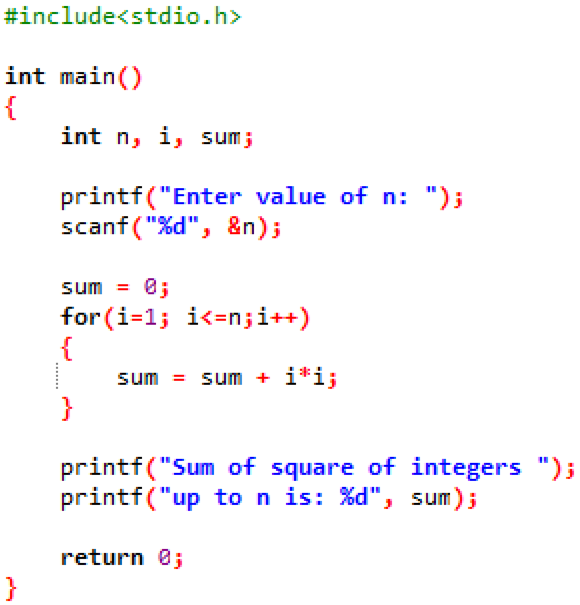
Algorithm

1. Start
2. Read n
3. i = 0, sum = 0
4. i = i + 1
5. sum = sum + (i*i)
6. Repeat steps 4 and 5 until i > n
7. Print sum
8. Stop

Flowchart



Program



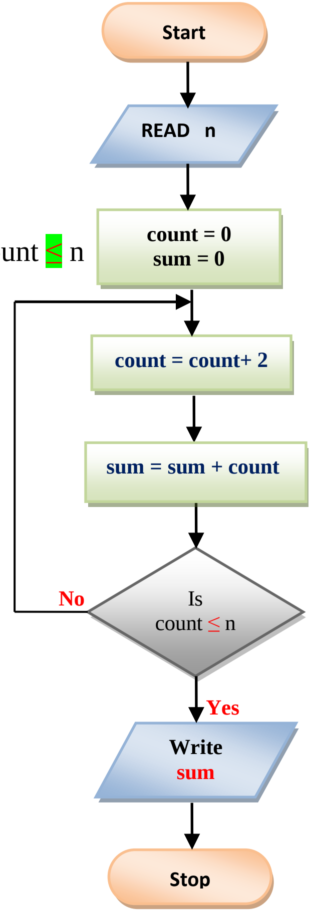
```
#include<stdio.h>

int main()
{
    int n, i, sum;

    printf("Enter value of n: ");
    scanf("%d", &n);

    sum = 0;
    for(i=1; i<=n;i++)
    {
        sum = sum + i*i;
    }

    printf("Sum of square of integers ");
    printf("up to n is: %d", sum);

    return 0;
}
```

16. To find the sum of all even numbers up to ‘n’

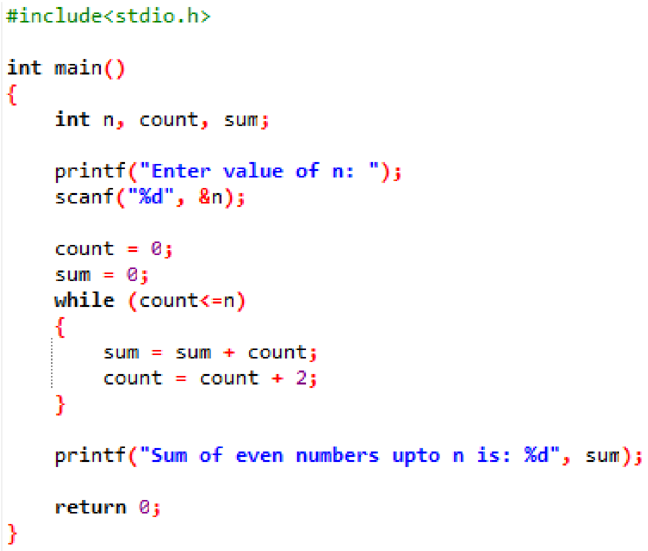
Algorithm

1. Start
2. Read n
3. count=0
4. sum=0
5. count = count + 2
6. sum = sum + count
7. Repeat steps 5 & 6 until count ≤ n
8. Print sum
9. Stop

Flowchart



Program



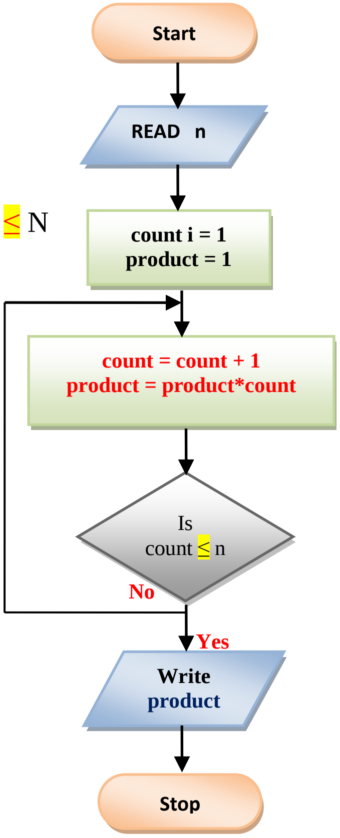
```
#include<stdio.h>

int main()
{
    int n, count, sum;

    printf("Enter value of n: ");
    scanf("%d", &n);

    count = 0;
    sum = 0;
    while (count<=n)
    {
        sum = sum + count;
        count = count + 2;
    }

    printf("Sum of even numbers upto n is: %d", sum);

    return 0;
}
```

17. To find Product of numbers up to N

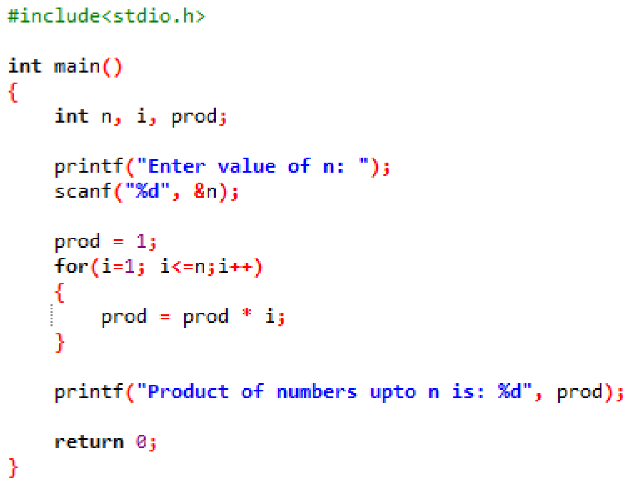
Algorithm

1. Start
2. Read n
3. count i = 1
4. product = 1
5. product=count*product
6. count = count + 1
7. Repeat steps 5,6 until count ≤ N
8. Print product
9. Stop

Flowchart



Program



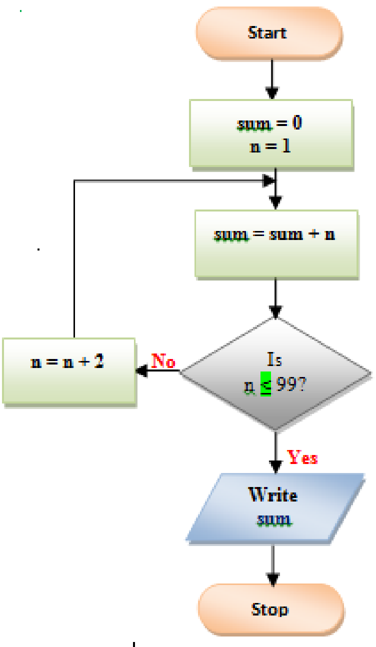
```
#include<stdio.h>

int main()
{
    int n, i, prod;

    printf("Enter value of n: ");
    scanf("%d", &n);

    prod = 1;
    for(i=1; i<=n;i++)
    {
        prod = prod * i;
    }

    printf("Product of numbers upto n is: %d", prod);

    return 0;
}
```

18. Sum of first 50 odd numbers

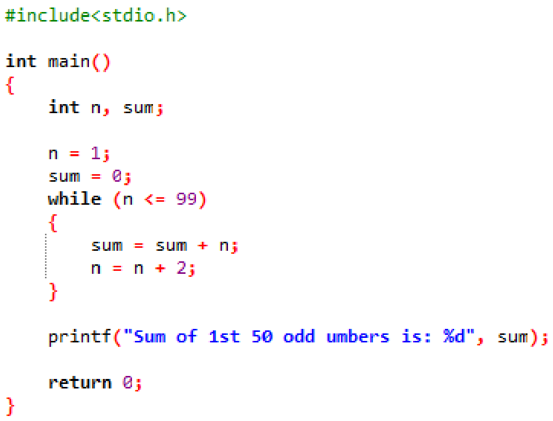
Algorithm

1. Start
2. sum=0, n = 1
3. sum=sum + n
4. n = n + 2
5. Repeat steps 4 and 5 until n ≤ 99
6. Print sum
7. Stop

Flowchart



Program



```
#include<stdio.h>

int main()
{
    int n, sum;

    n = 1;
    sum = 0;
    while (n <= 99)
    {
        sum = sum + n;
        n = n + 2;
    }

    printf("Sum of 1st 50 odd umbers is: %d", sum);

    return 0;
}
```

Current sum

Previous :

Initial count = 0, sum = 0	
count	sum = sum + cou
1	sum = 0 + 1 = 1
2	sum = 1 + 2 = 3
3	sum = 3 + 2 = 5
4	sum = 5+ 2 = 7
5	sum = 7 + 2 = 9
Up to n<=99	