Loops Control Structure (*for* Loop)

Rab Nawaz Jadoon

Assistant Professor

COMSATS IIT, Abbottabad

Pakistan

Department of Computer Science

DCS

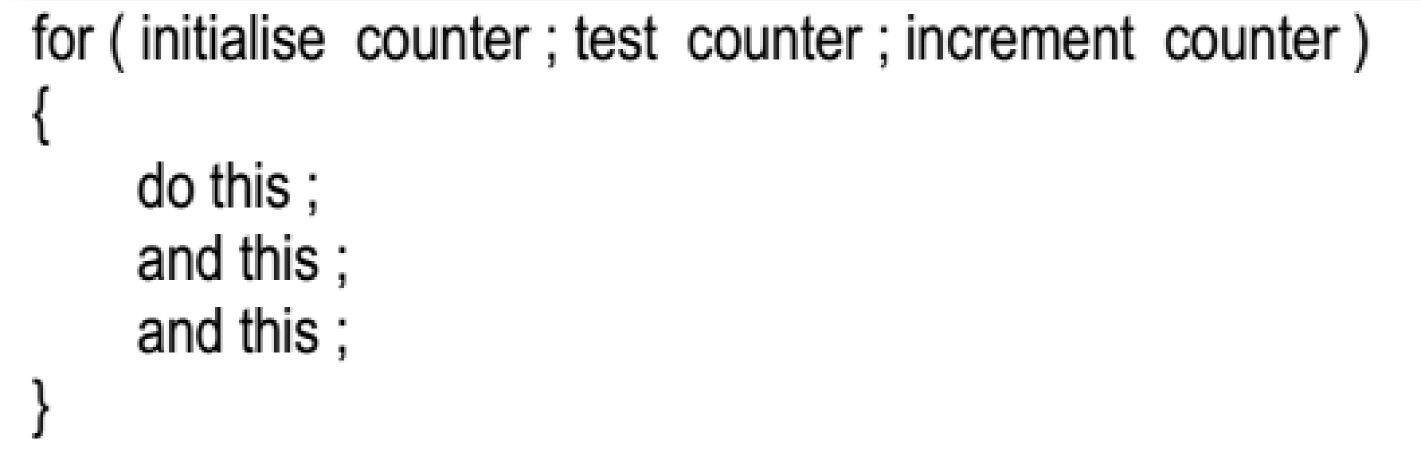
COMSATS Institute of Information Technology

Introduction to Computer Programming (ICP)

For Loop

- The for allows us to specify three things about a loop in a single line,
  - Setting a loop counter to an initial value.
  - Testing the loop counter to determine whether its value has reached the number of repetition desired
  - Increasing the value of loop counter each time the program segment within the loop has been executed.

General form



```
for ( initialise  counter ; test  counter ; increment  counter )
{
        do this ;
        and this ;
        and this ;
}
```

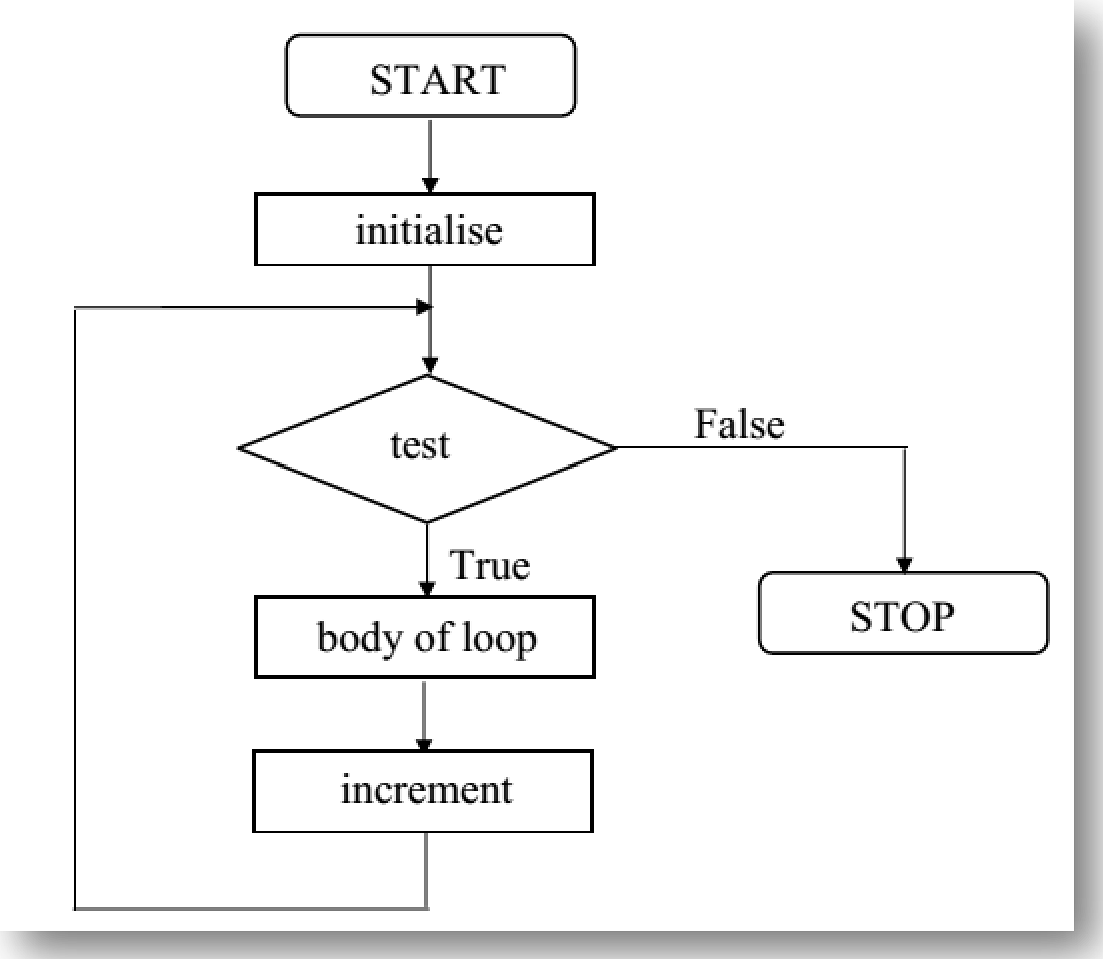
General Understanding about for Loop

- When the for statement is executed for the first time,
    - The value of count is set to an initial value 1.
    - Now the condition count <= 3 is tested. Since count is 1 the condition is satisfied and the body of the loop is executed for the first time.
    - Upon reaching the closing brace of for, control is sent back to the for statement, where the value of count gets incremented by 1.
    - Again the test is performed to check whether the new value of count exceeds 3.

General Understanding about for Loop

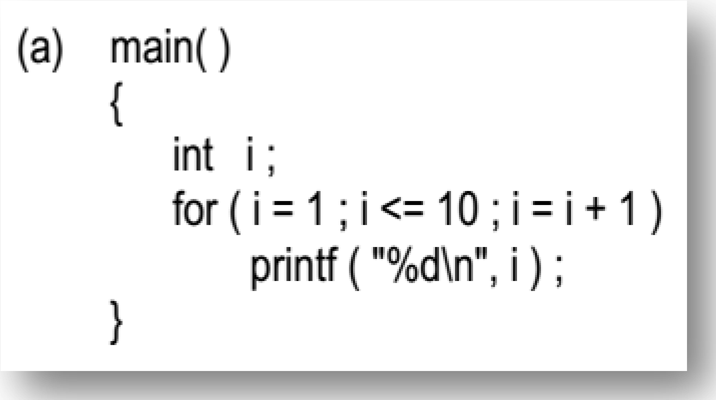
- If the value of count is still within the range 1 to 3, the statements within the braces of for are executed again.
- The body of the for loop continues to get executed till count doesn’t exceed the final value 3.
- When count reaches the value 4 the control exits from the loop and is transferred to the statement (if any) immediately after the body of for.

Flow chart

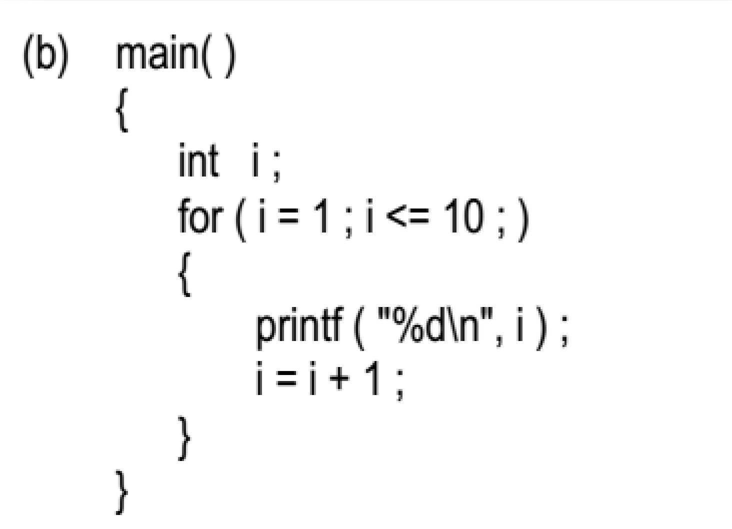

START
initialise
test
False
True
STOP
body of loop
increment

For loop (variations)

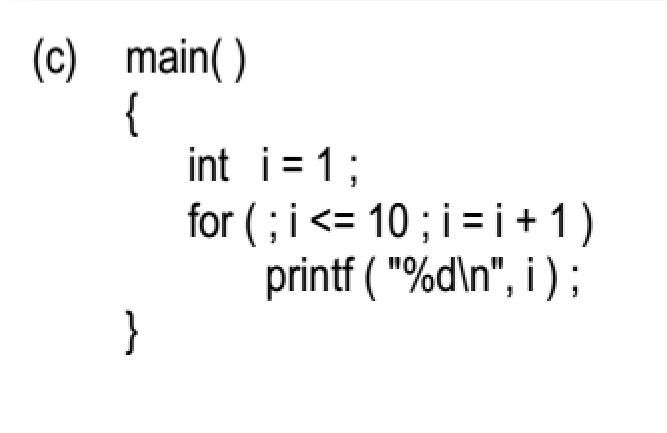
- Let us now write down the program to print numbers from 1 to 10 in different ways. This time we would use a for loop instead of a while loop.



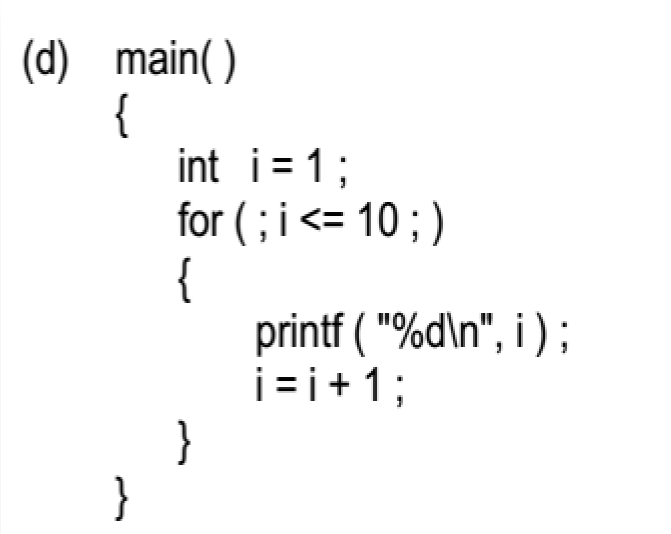
```
(a)   main( )
      {
          int  i ;
          for ( i = 1 ; i <= 10 ; i = i + 1 )
              printf ( "%d\n", i ) ;
      }
```



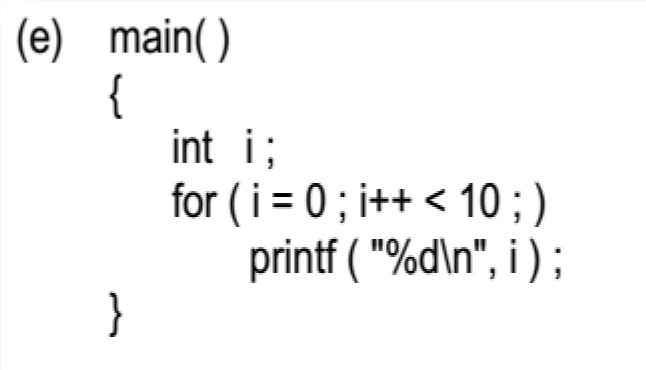
```
(b)   main( )
      {
          int  i ;
          for ( i = 1 ; i <= 10 ; )
          {
              printf ( "%d\n", i ) ;
              i = i + 1 ;
          }
      }
```



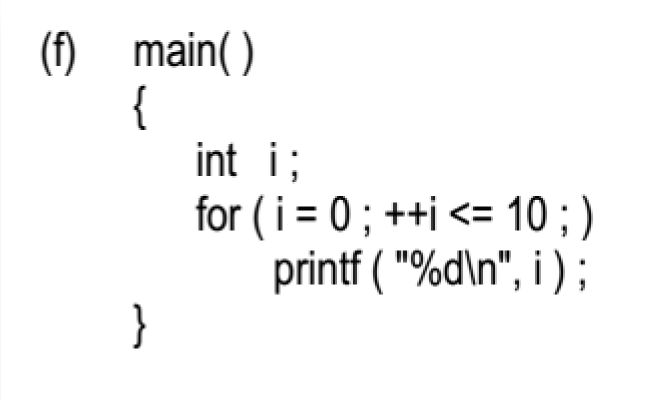
```
(c)  main( )
     {
         int  i = 1 ;
         for ( ; i <= 10 ; i = i + 1 )
             printf ( "%d\n", i ) ;
     }
```



```
(d)  main( )
     {
         int  i = 1 ;
         for ( ; i <= 10 ; )
         {
             printf ( "%d\n", i ) ;
             i = i + 1 ;
         }
     }
```



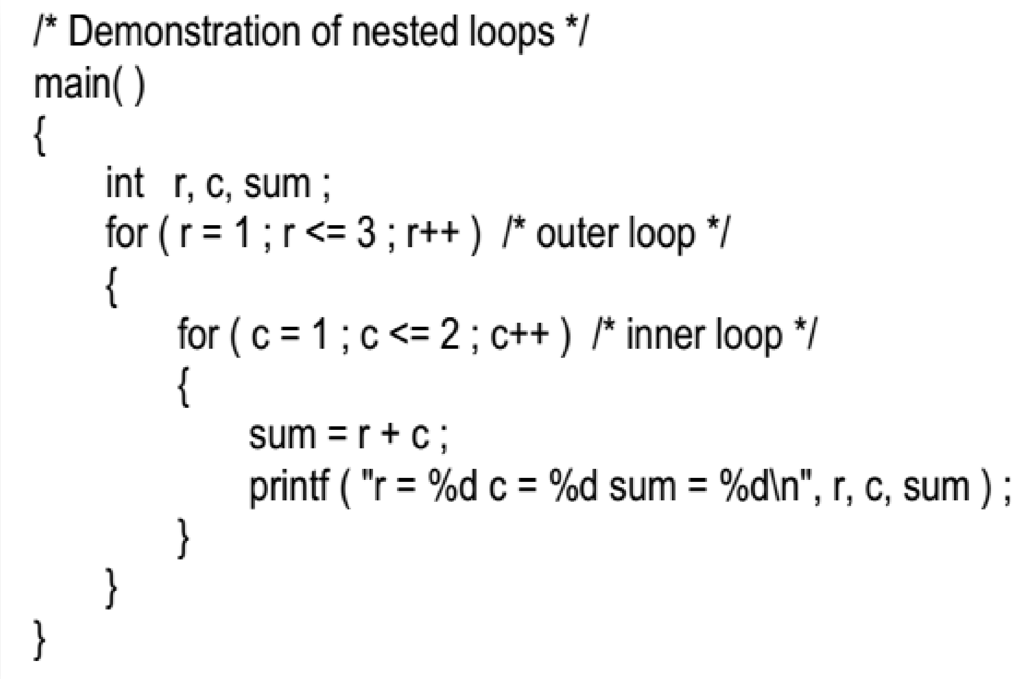
```
(e)  main( )
     {
         int  i ;
         for ( i = 0 ; i++ < 10 ; )
             printf ( "%d\n", i ) ;
     }
```



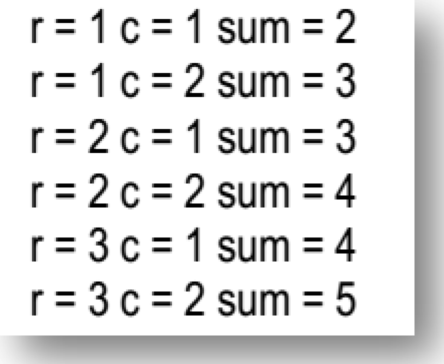
```
(f)  main( )
     {
         int  i ;
         for ( i = 0 ; ++i <= 10 ; )
             printf ( "%d\n", i ) ;
     }
```

Nesting of loops

- The way if statements can be nested, similarly whiles and fors can also be nested.
  - To understand how nested loops work, look at the program given below:



```
/* Demonstration of nested loops */
main( )
{
    int  r, c, sum ;
    for ( r = 1 ; r <= 3 ; r++ )  /* outer loop */
    {
        for ( c = 1 ; c <= 2 ; c++ )  /* inner loop */
        {
            sum = r + c ;
            printf ( "r = %d c = %d sum = %d\n", r, c, sum ) ;
        }
    }
}
```



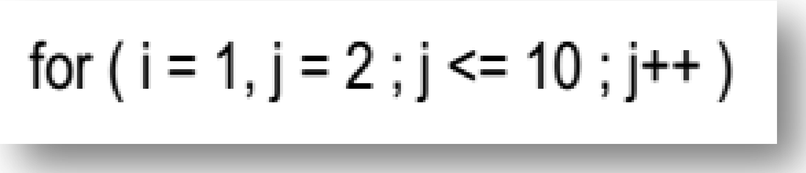
```
r = 1 c = 1 sum = 2
r = 1 c = 2 sum = 3
r = 2 c = 1 sum = 3
r = 2 c = 2 sum = 4
r = 3 c = 1 sum = 4
r = 3 c = 2 sum = 5
```

Program working

- Here, for each value of r the inner loop is cycled through twice, with the variable c taking values from 1 to 2.
- The inner loop terminates when the value of c exceeds 2, and the outer loop terminates when the value of r exceeds 3.
- Two while loops can also be nested. Not only this, a for loop can occur within a while loop, or a while within a for.

Multiple initialization in a for loop

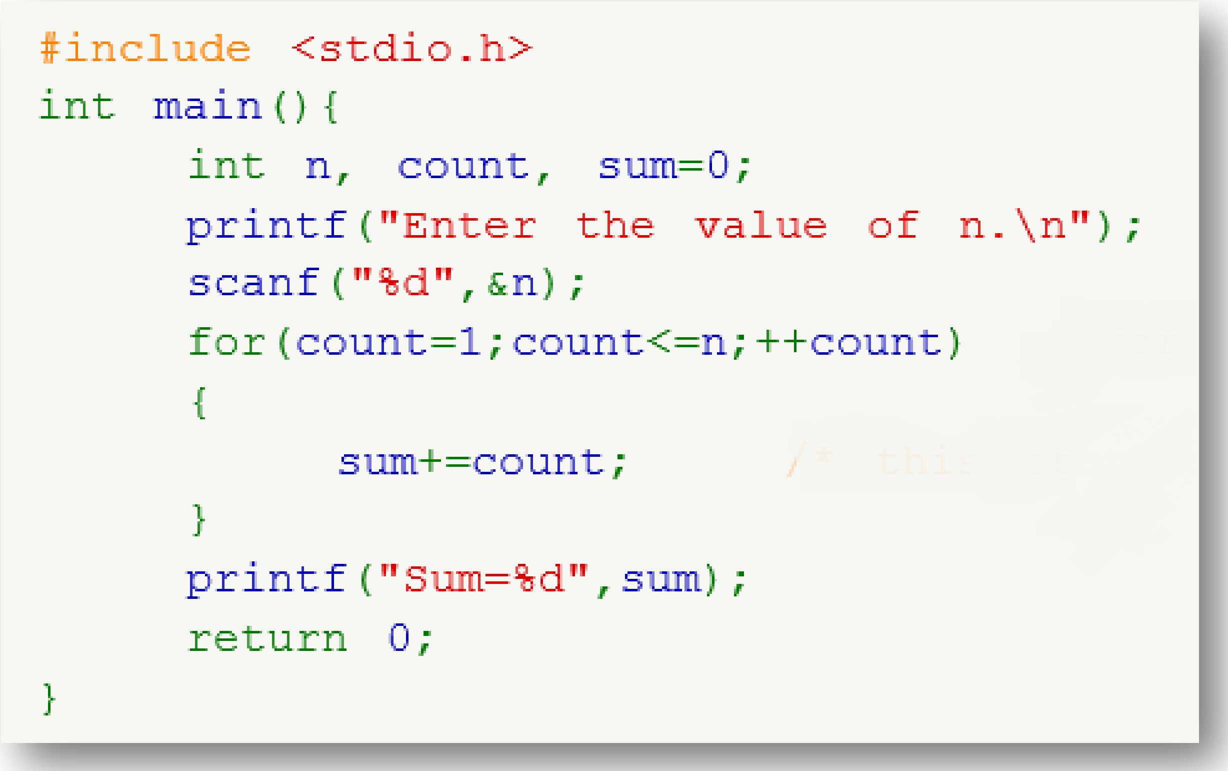
- The initialization expression of the for loop can contain more than one statement separated by a comma. For example,



```
for ( i = 1, j = 2 ; j <= 10 ; j++ )
```

Program 1

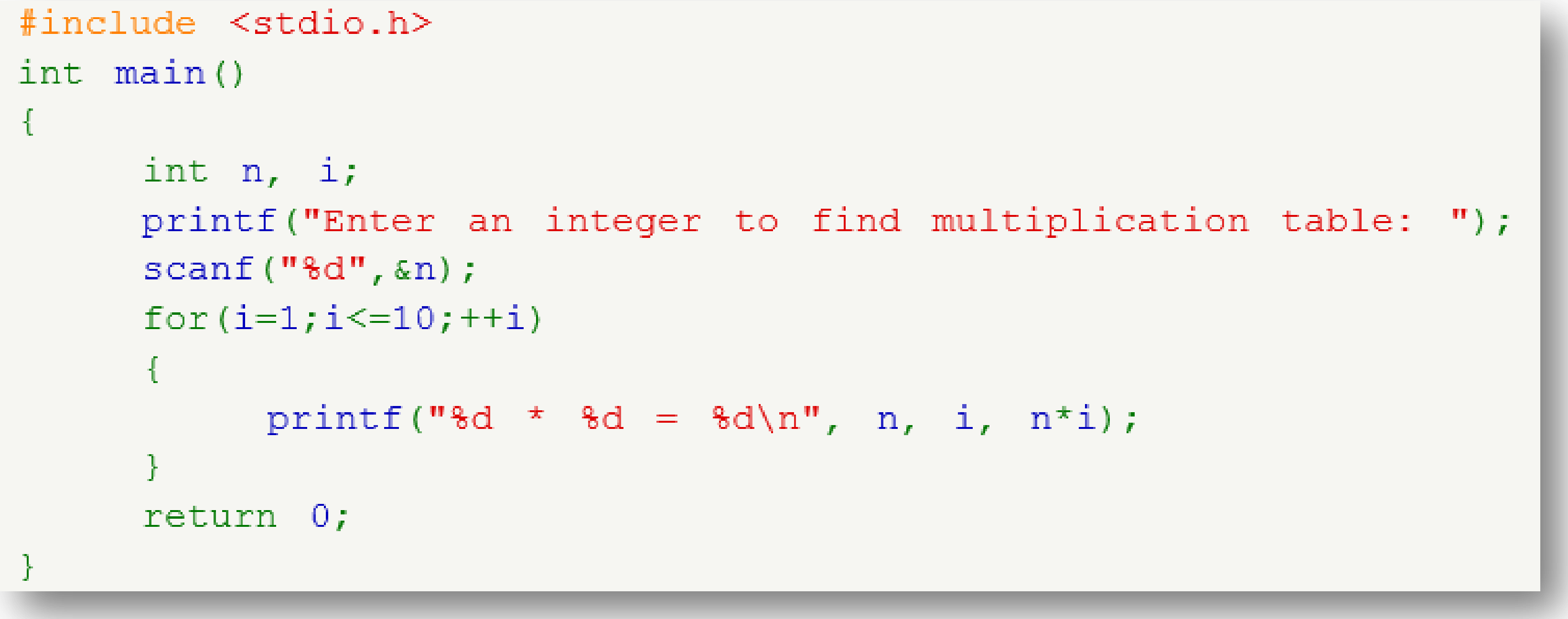
- Write a program to find the sum of first n natural numbers where n is entered by user.



```c
#include <stdio.h>
int main(){
      int n, count, sum=0;
      printf("Enter the value of n.\n");
      scanf("%d",&n);
      for(count=1;count<=n;++count)   //for
      {
            sum+=count;       /* this stat
      }
      printf("Sum=%d",sum);
      return 0;
}
```

Program 2

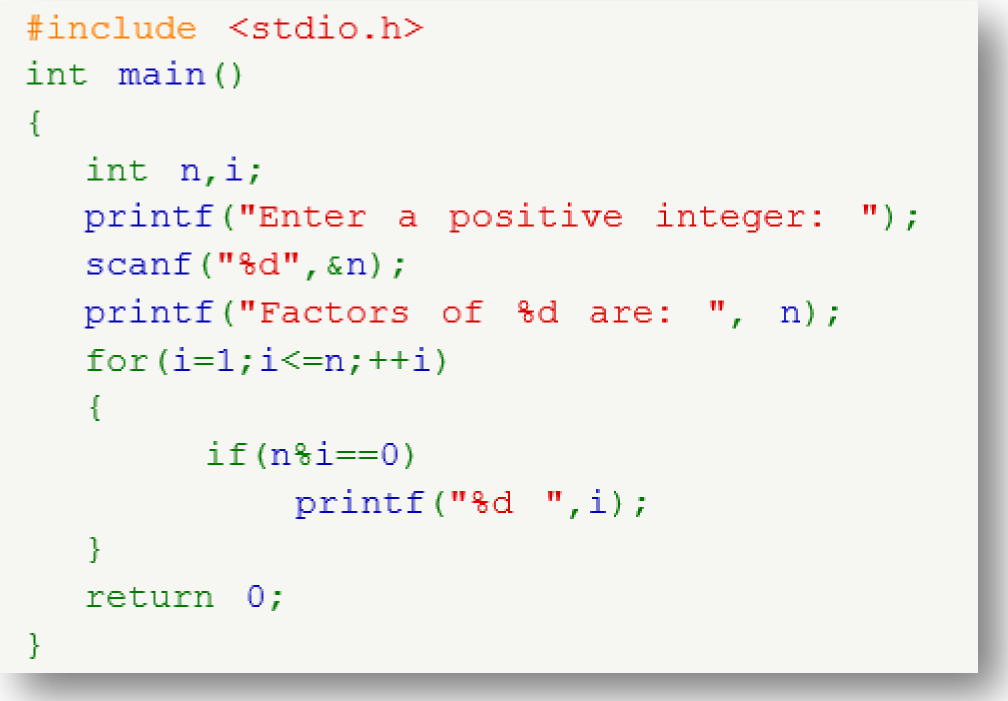
- Multiplication table using for loop



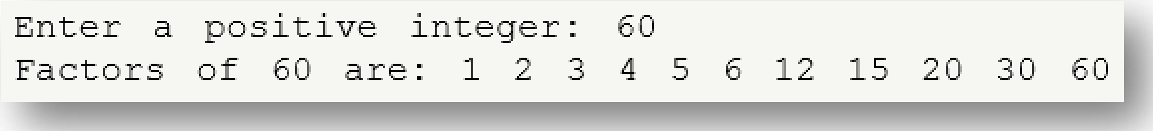
```c
#include <stdio.h>
int main()
{
      int n, i;
      printf("Enter an integer to find multiplication table: ");
      scanf("%d",&n);
      for(i=1;i<=10;++i)
      {
            printf("%d * %d = %d\n", n, i, n*i);
      }
      return 0;
}
```

Program 3

- Write a program using for loop that will display all the factors of a given number?



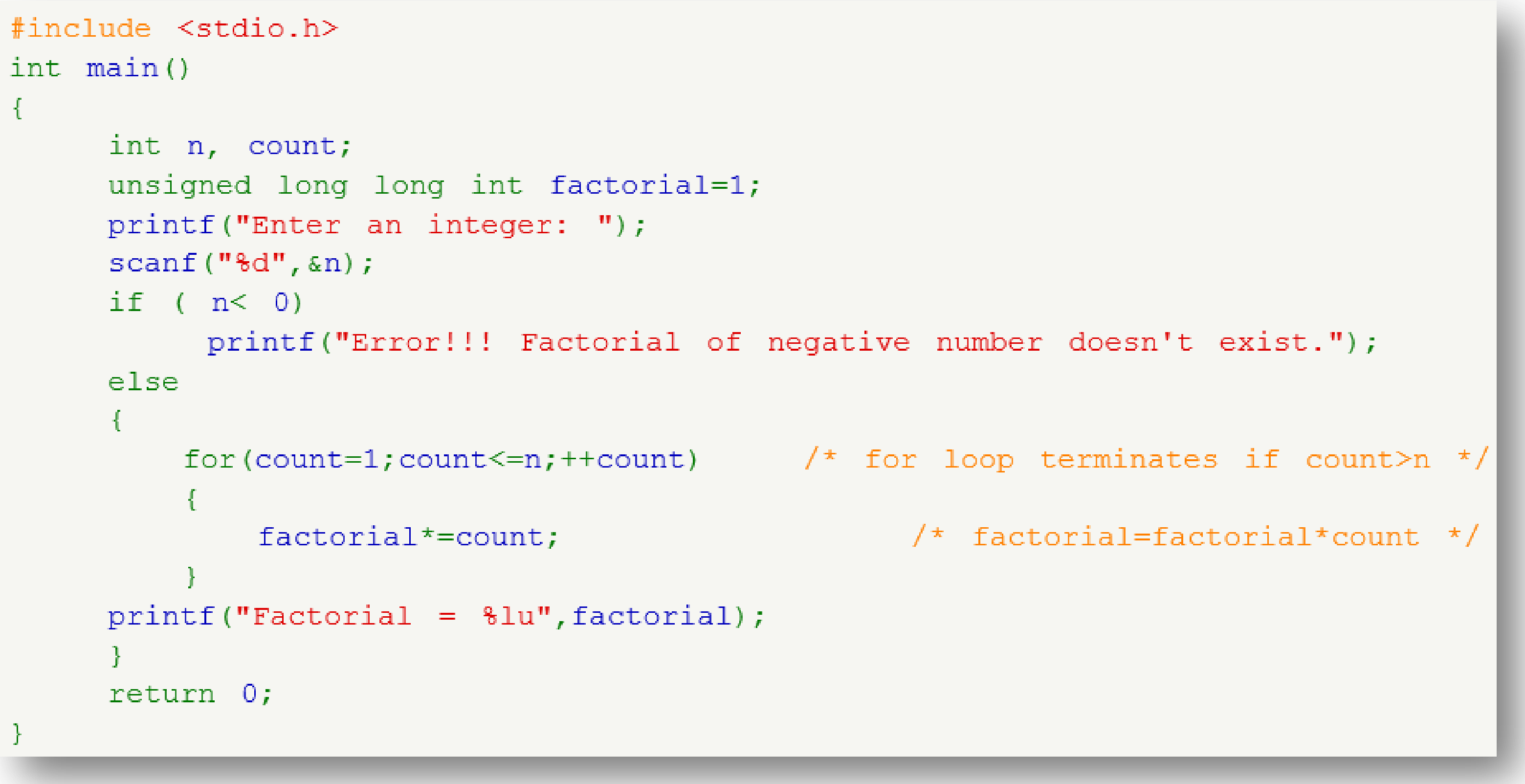
```c
#include <stdio.h>
int main()
{
   int n,i;
   printf("Enter a positive integer: ");
   scanf("%d",&n);
   printf("Factors of %d are: ", n);
   for(i=1;i<=n;++i)
   {
         if(n%i==0)
              printf("%d ",i);
   }
   return 0;
}
```



```
Enter a positive integer: 60
Factors of 60 are: 1 2 3 4 5 6 12 15 20 30 60
```

Program 4

- Write a Program using for loop that finds the factorial of the given number???



```
#include <stdio.h>
int main()
{
    int n, count;
    unsigned long long int factorial=1;
    printf("Enter an integer: ");
    scanf("%d",&n);
    if ( n< 0)
         printf("Error!!! Factorial of negative number doesn't exist.");
    else
    {
        for(count=1;count<=n;++count)      /* for loop terminates if count>n */
        {
            factorial*=count;                    /* factorial=factorial*count */
        }
    printf("Factorial = %lu",factorial);
    }
    return 0;
}
```

THANKS