# Programming In C++

Course 2: Lecture 5, Recursion Function

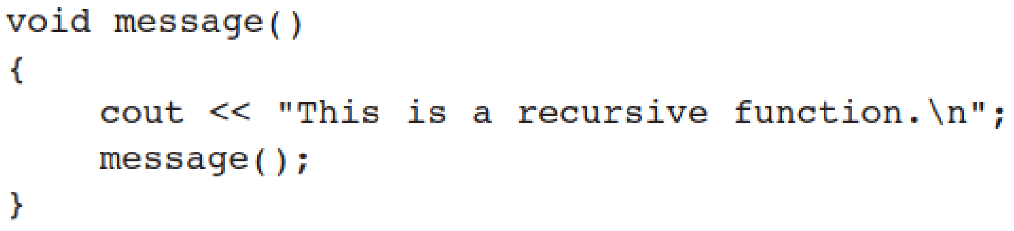
Prepared By Dr. Ali Mohsin

Nineveh University- Faculty of IT- department of software

# Recursion Functions

We have seen how functions calling other functions. Function A can call function B, which can then call function C.

It’s also possible for a function to call itself. A function that calls itself is a  recursive function.



```
void message()
{
    cout << "This is a recursive function.\n";
    message();
}
```

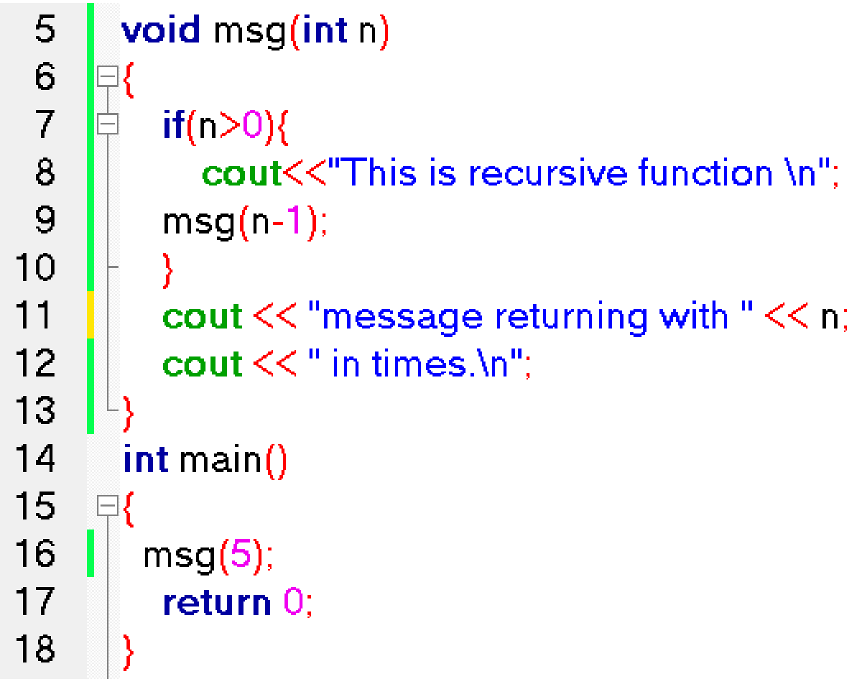
This function displays the string “This is a recursive function.\n”, and then calls itself. Each time it calls itself, the cycle is repeated.

The problem of This function is like an infinite loop because there is no code to stop it from repeating.

# Recursion Functions

Like a loop, a recursive function must have some method to control the number of times it repeats. The following is a modification of the  message function.

It passes an integer argument, that holds the number of times the function is to call itself.

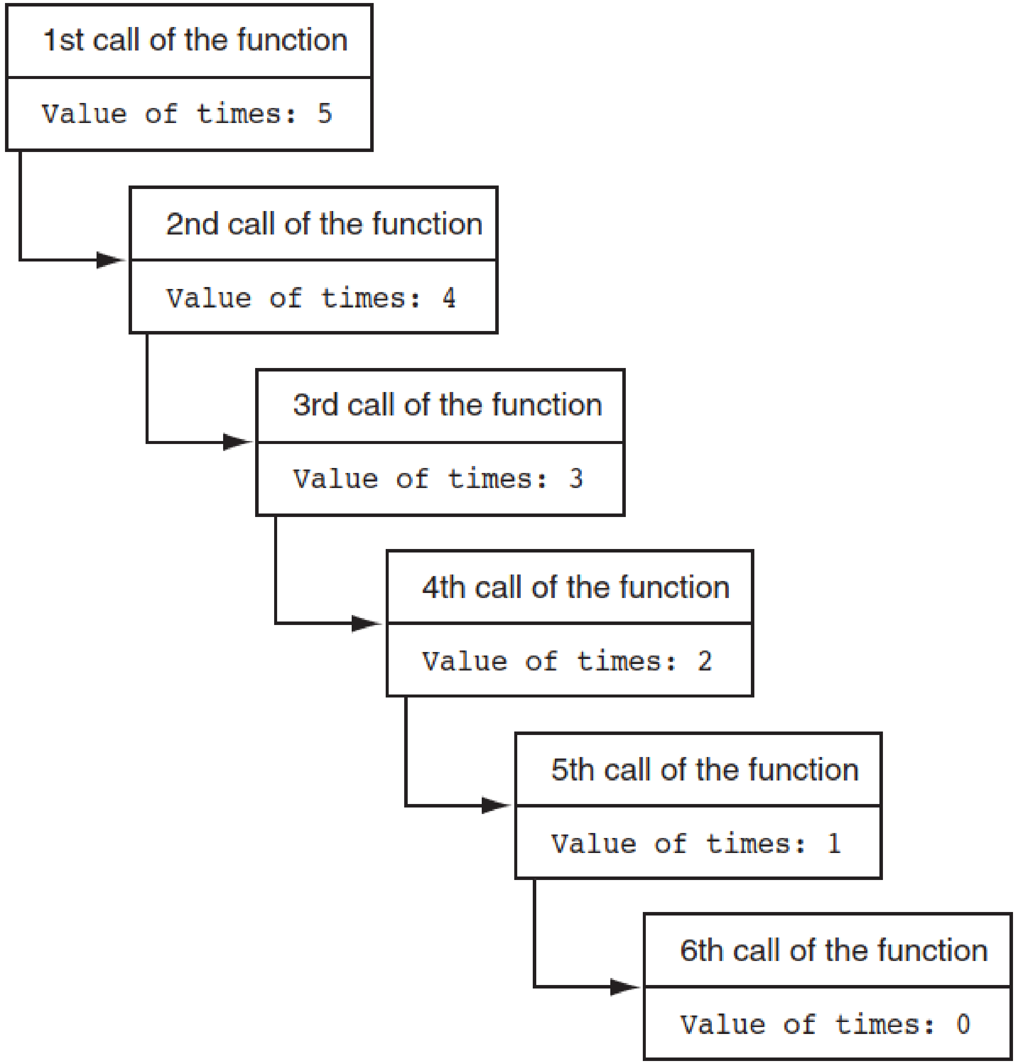


```
void msg(int n)
{
    if(n>0){
        cout<<"This is recursive function \n";
    msg(n-1);
    }
    cout << "message returning with " << n;
    cout << " in times.\n";
}
int main()
{
   msg(5);
    return 0;
}
```

This function contains an  if statement that controls the repetition.

 As long as the  times argument is greater than zero, it will display the message and call itself again.

# Recursion Functions



1st call of the function

Value of times: 5

2nd call of the function

Value of times: 4

3rd call of the function

Value of times: 3

4th call of the function

Value of times: 2

5th call of the function

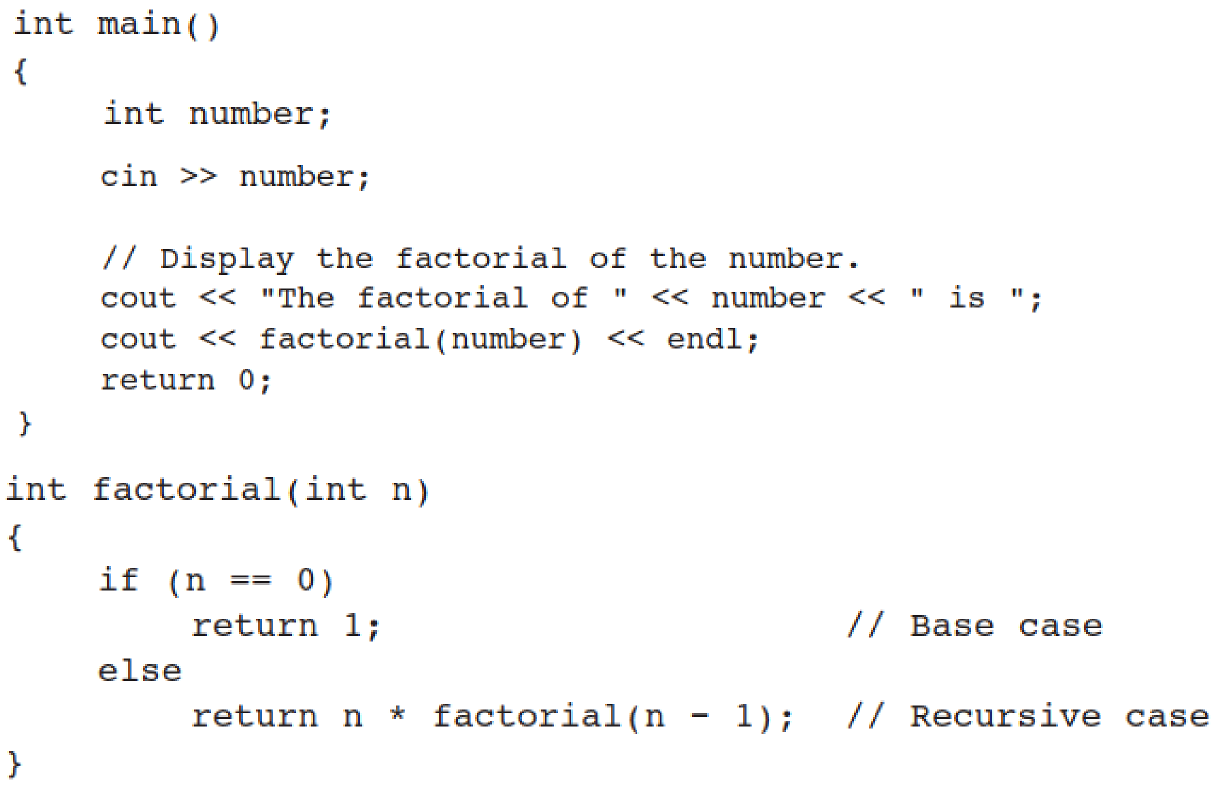
Value of times: 1

6th call of the function

Value of times: 0

# Recursion Functions

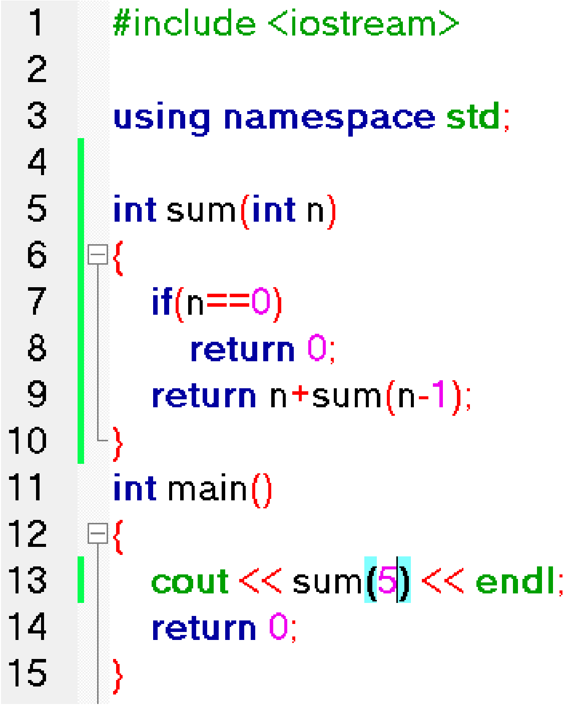
Using Recursion to Calculate the Factorial of a Number



```
int main()
{
    int number;

    cin >> number;

    // Display the factorial of the number.
    cout << "The factorial of " << number << " is ";
    cout << factorial(number) << endl;
    return 0;
}

int factorial(int n)
{
    if (n == 0)
        return 1;                       // Base case
    else
        return n * factorial(n - 1);   // Recursive case
}
```

# Recursion Functions

Using Recursion to Calculate the Summation of a Number



```
#include <iostream>

using namespace std;

int sum(int n)
{
    if(n==0)
        return 0;
    return n+sum(n-1);
}
int main()
{
    cout << sum(5) << endl;
    return 0;
}
```

# The End