

Recursion

In mathematics there are many functions that are defined recursively.

A recursive function is one that relies on previous terms to find another.

Where most functions you can easily find the value of the 100th term by substituting 100 in for the variable, in a recursive function to find term 100 you will need to find all 99 terms before it in order to be able to calculate 100.

A good example of this is the factorial.

The factorial of a number is represented as $n!$.

To find the value of a number factorial you must first know that:

$0!$ is 1

and

$n!$ is $n * (n - 1)!$

Therefore the value $4!$ is:

$$4 * 3 * 2 * 1 = 24$$

This mathematical function has many uses in topics such as probability.

We're not as concerned about the mathematical applications of this type of function but how it would need to be coded in C++.

The way to deal with functions that are recursive in C++ is to call the function within itself.

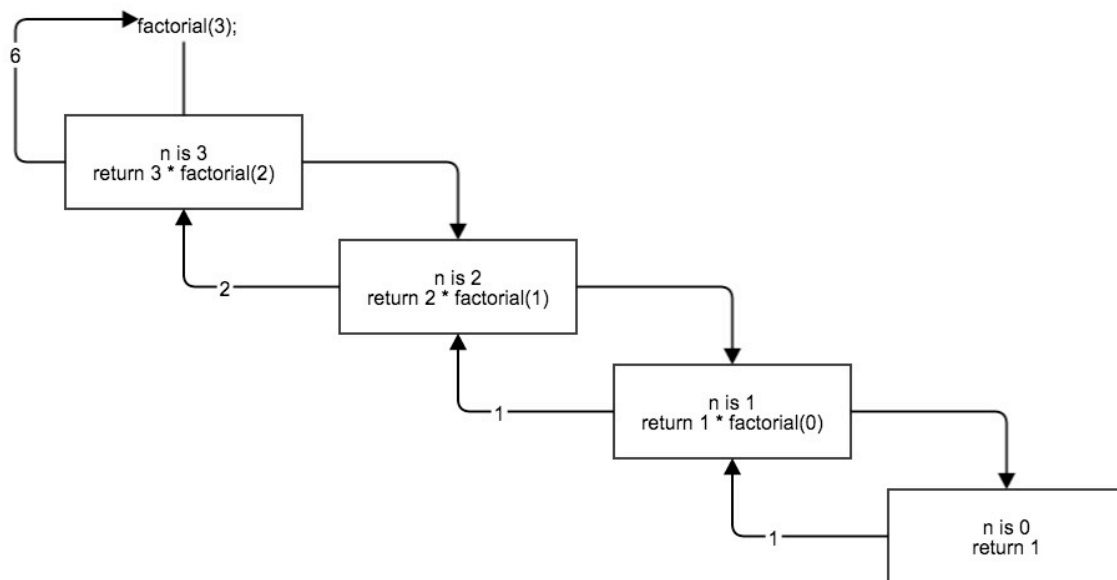
For example a program to calculate factorial may have a line like:

```
return n * factorial (n-1);
```

This means that the result of the current call (argument n) is determined by multiplying the result of the next call (argument $n - 1$) by n .

```
int factorial(int n)
{
    if (n <= 0)
    {
        return 1;
    }
    else
    {
        return n * factorial(n-1);
    }
}
```

The recursive implementation of the above function looks like this:



This is not the only way to calculate the value of a factorial but it is a great example of calling a function within a function.