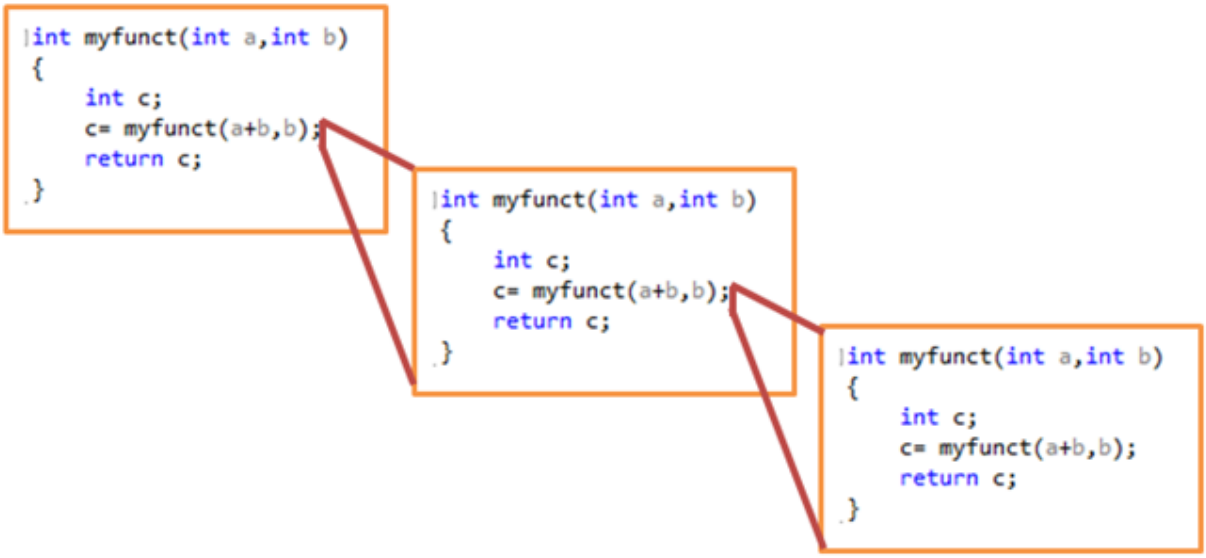
Lab 6 - wb 02/12/13

This week we are looking at recursion. The idea of a recursive function is a function that calls itself. When a function calls itself it is calling a copy of itself that runs separately.Think of putting two mirrors together, one sees the other, that sees the other…etc. The diagram below may help.



Obviously you wouldn't want to run this program from above as it would never stop running, it would just keep creating a new instance of itself. To fix this we tend to have a break clause of some sort. Normally some kind of counter, so if you wanted it to loop N times, pass the first function N, and then when you call the function again pass it N-1. Then when N=1 return something that isn't calling the function. See example 2 for this.

A Simple recursion -

A very simple example of recursion is to call the main() function to restart the program if the user doesn't want to exit. Please note this may not work in code blocks. Visual studio allows this but other programs don't, so this is for example purposes only!

```
#include <iostream>
#include <stdlib.h>
 using namespace std;

int main()
{
    cout<<"Hello this is my program!";// A very short program it is...
    cout<<endl;//New line
    cout<<"Would you like to exit? Y/N :";//Ask if they want to exit
    char answer; // Declare somewhere to store their answer
    cin>>answer; //Get the users input
    answer = toupper(answer);//convert to uppercase so we dont have to compare y,Y,n,N
    system("cls");//Clear the Screen

    if(answer=='N') main(); //If they dont want to exit we can recusivly start again
    return 0;
}
```

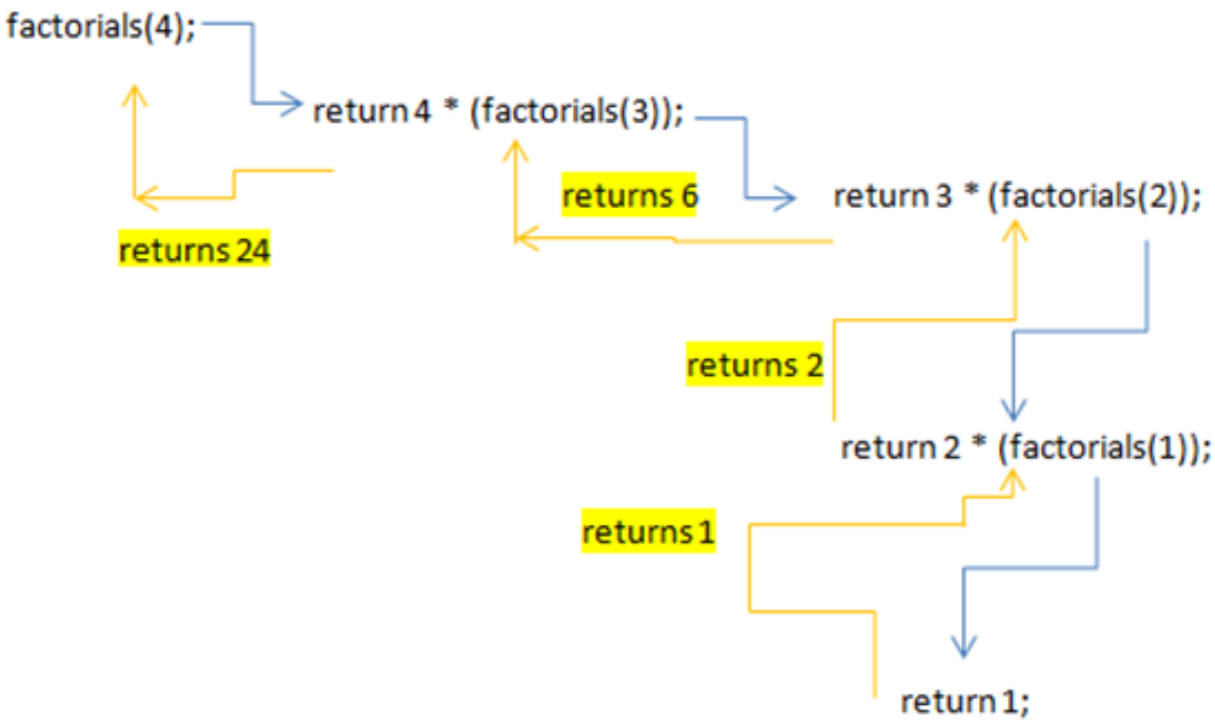
The breakout part of this recursive function is if the user answers yes, then it just returns 0, instead of calling main() again.

A slightly more advanced recursive function.

Recursion is more likely to be used for somthing mathematical. for example N! (N factorial) is the idea of multiplying the first number by the next number, by the next number… so on.

```
#include <iostream>
#include <stdlib.h>
using namespace std;


int factorials(int n)
{
    if (n==1) return 1; //if at the end of the recursion return 1 to break out

    return n*(factorials(n-1));// else return n * the factoral for the previous numbers
}


int main()
{
    cout<<"Choose a number and I will work out the factorial up to that number!";
    cout<<endl;//New line
    int N;
    cin>>N;

    cout<<"The factorials of that number is : "<<factorials(N); //call the function and output the result

    cout<<endl;//New line
    cout<<"Would you like to exit? Y/N :";//Ask if they want to exit
    char answer; // Declare somewhere to store their answer
    cin>>answer; //Get the users input
    answer = toupper(answer);//convert to uppercase so we dont have to compare y,Y,n,N
    system("cls");//Clear the Screen

    if(answer=='N') main(); //If they dont want to exit we can recusivly start again
    return 0;
}
```

So if i wanted to know the factorial of 4! it would look a bit like this.



Have A Go - Test Your Knowledge!

Challenge 1

Write a program that asks the user for a number. Using a recursive function output “hello”,the amount of times the user asked for. Do not do this with loops, please use recursion!

Challenge 2

Create a recursive function that allows the user to input a number and outputs all the numbers before it added up. For example if I input 8, the program will do 8+7+6+5+4+3+2+1+0
Again make sure you use recursion for this and not loops!

Next Week - Get Ahead!

Next week we are looking at pointers. If you want a head start then you can look up c++ pointers on google. Or have a look at the following link. If you don’t understand it, Don’t worry, we will go through it next week!
http://www.cprogramming.com/tutorial/lesson6.html