Lab 3: Recursion

UWYO COSC 2030

Introduction:

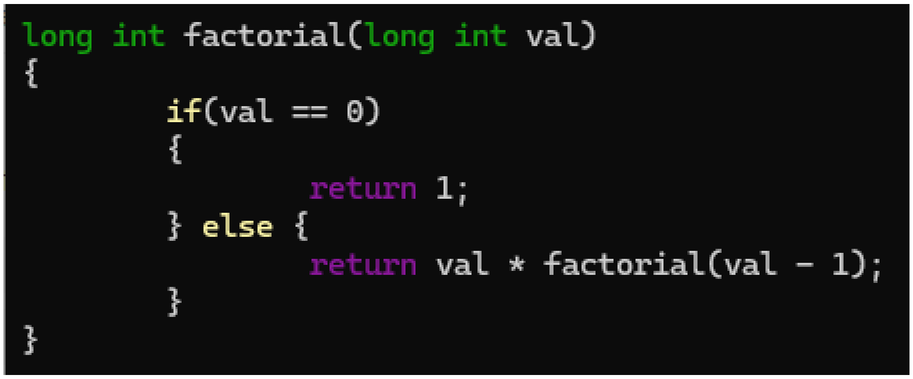
Unless you've had previous programming experience, recursion is likely a foreign concept to you. The easiest way that I can think of to describe it is by breaking it down into word fragments and defining them:

re-: again

curs: run

-**ion**: action or process

So, the literal definition of recursion is the action/process of running again. In the computer science world, we take this to mean that a particular given function will run again until some criterion is met. This is most intuitive when you're looking at a function that recursively finds the factorial of a given integer:



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

Notice how in the general return statement the function is called again, until the number 0 is reached, at which point the program will start working its way back up the return statements, eventually returning the final factorial value. If this is still confusing, I recommend the C++ Function Recursion page or emailing me.

Reminder about the README.md

We went over this in the last lab, but there were a few people who didn't follow the instructions, so I will give this one final heads-up before I'm required to start taking points. What I ask is that for this, as well as all the upcoming labs, you edit the README.md document to show:

> Your name
>
> Help given/received

So for me, this would look like:

> Michael Stoll
>
> Help given/received: I had help from Professor Ward

This might seem obnoxious, but for the sake of avoiding plagiarism allegations, please humor me.

Edited by Michael Stoll for UWYO COSC 2030

Lab 3:

First of all, you'll need to accept the lab at this link: https://classroom.github.com/a/vGN-R5fh.

When you do accept the assignment and clone down the repo, you'll notice a file called Lab3.cpp. Open it up in whatever environment you would like and take a look at the instructions. Don't mess with the code that is already written, just navigate to the findBinaryNotation declaration and start from there.

This part has less instructions than have been in the other labs because this is really going to be on an individual basis. This class isn't about having the best or fastest solution to every problem; rather, it's about building your problem-solving skills and figuring out how the code makes sense to you. Everybody approaches a coding problem with a different perspective, your job in this course is simply to find one that makes sense to you. The optimization will come later, trust me.

Submission:

For this week, you'll need to update your README.md and submit your completed Lab3.cpp file. Don't bother including your executables that you've used for testing, unless for some reason you want them in your repo for your sake. This lab will be due this Sunday, Feb 16th, at 11:59pm.