CS161 - Programming Assignment 5 - Functions (FunFun)

The purpose of this assignment is to get some experience working with functions.

Overview

The previous assignment (LoopFun) had 4 distinct parts, all of which were implemented in one function (main). This was a horrible, horrible idea, undertaken only because we didn't know about functions, and didn't know that functions should do only one thing. But now we know… so we're going to correct our mistake, and then never speak of it again.

For this assignment you'll take your last program (LoopFun) and break it into multiple functions to implement the different parts. The functions will be as follows:

> **int getInteger(int low, int high);**
> Gets an integer value from the user between the low and high value (inclusive).
> While the value entered by the user is out of bounds, it prints an error message and gets a new value. This continues until a valid value is entered, which is then returned.
>
> **void printRangeAndAvg(int low, int high);**
> Prints the range of numbers from low to high (inclusive) and then computes and prints the average of the numbers.
>
> **void printBox(int height, int width);**
> Prints a hollow box of * on the screen of the given height and width. (see output section for an example of the output)
>
> **void printWedge(int height);**
> Prints a wedge of * on the screen of the given height, with two * per row. See output section below for an example of the output.

main() will then be a few variable declarations and a series of function calls, like this:

```
cout << "Enter box height (between 3 and 10): ";
boxHeight = getInteger(3, 10);
cout << "Enter box width (between " << boxHeight << " and 20): ";
boxWidth = getInteger(boxHeight + 1, 20);

printRangeAndAverage( ….. );
printBox( …. );
printWedge( … );
```

Output should be as shown in the output section below. (which is identical to the last assignment)

Note: as a result of using good function names, someone can easily see the big picture of what your program does by simply reading the list of functions main calls. This is an example of "self-documenting code", and should always be your goal.

Details- identical to the previous assignment details

Part 1 - Get two integer values (a box height and width) from the user.

- The box height should be between between 3 and 10 (inclusive). If the user enters a number that is out of bounds (less than 3 or greater than 10), continue to prompt them until they enter an appropriate number (use a while loop).
- The box width should be between the height and 20. If the user enters a number that is out of bounds (<= the first number or greater than 20), continue to prompt them until they enter an appropriate number (use a while loop).

Part 2 - Use a for loop to print out all the numbers between the height and width (inclusive), and then print the average (mean) of those numbers. Print them all on one line, space delimited (see example below).

Part 3 - Print a hollow box made of * as shown in the example below. (this will require multiple loops, or one loop and use of some formatting functions covered in chapter 3) The width and height of the box should be the values entered by the user in part 1.

Part 4 - Use nested loops to print out the triangular shape shown in the example below. It's height will be the number entered for the box height. The first row of the triangle is two asterisks, and each subsequent row should be two asterisks bigger than the previous row. (see example below)

Sample Output

Output should look similar to this::

```
Enter a box height(between 3 and 10): 5

Enter a box width (between 6 and 20): 42
That number is out of bounds: Try again: 3
That number is out of bounds: Try again: 12
```

```
The integers between 5 and 12 are:
     5 6 7 8 9 10 11 12
and the average of those numbers is 8.5

************
*          *
*          *
*          *
************

**
****
******
********
**********
```

As with all programs written in this course, maintainability is as important as functionality, so your the code should be clear and easy to follow. Make sure it follows the class coding standards (verify it against this checklist before submitting).