EECS402 Winter 2017 "Mini-Project" 1

This mini programming assignment will ensure you are able to write a program, compile it, run it, test it, and submit it. Since I'm trying to keep this at a minimal level, it will be quite short – in no way does this project represent the complexity of future projects, which will vary significantly. Despite this, especially if you don't have much or any programming experience, you should start it right away in case you run into problems with compiling, etc…

For this project, you will develop a simple number guessing game. We will use this program to practice input, output, functions, loops, and selection. As described in lecture, your final grade depends on more than just correct behavior, so be sure to follow the specs exactly, and also write easy to read and understand code, use the best types of loops, etc., as appropriate. In addition to correctness, you grade will also consider all other aspects of your implementation, including style, organization, readability, etc.

Submission and Due Date

You will submit your program using an email-based submission system by attaching *one C++ source file* only (named exactly "project1.cpp"). Do not attach any other files with your submission – specifically, do *not* include your compiled executable, etc. The due date for the project is **Friday, January 27, 2017 at 4:00pm**. Early submission bonus and late submission penalties will be applied as described in the course syllabus.

No submissions will be accepted after the late submission final deadline. As discussed in lecture, please double check that you have submitted the correct file (the correct version of your *source code* file named exactly as specified above). Submission of the "wrong file" will not be grounds for an "extension" or late submission of the correct file, and will result in a score of 0.

Game Description

This will be a two player game. The first player will enter a value between 1 and 100 (inclusive) and the second player will attempt to guess the number. For each incorrect guess, the second player will be told whether the number is higher or lower and will be allowed to continue guessing until the correct value is guessed. After the player guesses the value, he or she is told how many guesses it took to get the correct number.

Requirements

You will implement four functions, whose *exact specifications* are provided below. You must implement exactly these functions, as described, with the same name, parameter types and names, return types, etc. Do not implement any additional or fewer functions, or change the given specifications in ***any*** way.

`int getIntInRange(int minValue, int maxValue)`

This function will prompt the player to input a value between the specified minValue and maxValue (inclusive), using this prompt (for this example, assume minValue == 1 and maxValue == 100):

```
"Enter a number from 1 to 100 (inclusive): "
```

If the player enters a valid value, it is returned as the integer return value of the function. If the value entered is out of range, the following message is printed on a single line:

```
"ERROR: Value is not within valid range. Try again."
```

In that case, the player will be re-prompted (using the same prompt). This will continue until the player enters a value in the valid range specified via the input parameters (inclusive). Note: You are NOT required to handle the case where the player enters a non-integer value. For this project, you can assume the player will enter an integer, but that integer may be out of the specified range.

`int getSecretNumber(int minValue, int maxValue)`

This function will get a number from a player in the range specified via the minValue and maxValue parameters (inclusive) . Immediately after getting a valid value, the string:

```
"Clearing screen!"
```

will be printed ***exactly*** 30 times, one per line (to effectively clear the screen so the next player can't see the number that was entered). The function then returns the integer value that was entered.

`bool getGuessAndCheck(int minValue, int maxValue, int secretNum)`

This function will get a number in the range specified via the minValue and maxValue parameters from a player. The input parameter named "secretNum" will be the number that the player is trying to guess (as entered by the other player). Immediately after getting a valid value, one of the following three possible strings is printed as a single line:

```
"Congratulations! You guessed it!"
"The secret number is LOWER than your guess"
"The secret number is HIGHER than your guess"
```

Regardless of the outcome, only a single guess is processed, and the function returns true if the player guessed the correct value, or false if they did not.

`int main()`

This function is the main "driver" of the game. First, the function prints this message to player 1 as a single line:

```
"Player 1: You will enter a secret number for the other player to
try to guess"
```

The function will then get a valid secret number from player 1 ("clearing the screen" too as described) using the function getSecretNumber. Next, the following message will be printed to player 2:

```
"Player 2: You will try to guess the secret number"
```

Then, the function getGuessAndCheck is used to get a single guess from player 2 and determine the result. If the guess was not correct, then the above message is printed and getGuessAndCheck is called again. That process continues until finally the guess was correct, at which time the following is printed:

```
"It took you 3 guesses to guess it!"
```

Where the number 3 is replaced by the actual number of guesses it took to get the correct value.

Additional Information

- For this project, you should assume the game is played using a range of values from 1 to 100, inclusive
- Remember, “magic numbers” are bad and need to be avoided
- Remember, “duplicated code” is bad and needs to be avoided
- Remember, identifier naming is important. Name your variables, constants, etc., using a descriptive name using the style described in lecture
- To compile and execute on the Linux command line, refer to the “gPlusPlus” lecture posted on the Canvas site, which walks you through it step-by-step
    - If you want to use an IDE to build your project, that’s ok, but you must try out the Linux command line interface as well
    - Some future projects will have a small part of the project tied to using the command line interface, so getting a little experience with it would be good, even if you choose to do most of your development in an IDE
- As mentioned above, for this project, you are required to implement the project exactly as described here using the exact output strings provided. Make sure you implement and utilize the exact functions specified and do not add any additional functions, parameters, etc. You will have more “freedom” in your design and implementation as the course progresses.