Assignment 3: Input and Nested If Statements

This assignment is more exercise with conditional statements and code to make your program more robust. It builds on the previous assignment in that you must now use else conditions and conditional blocks. The attached documents contain reminders to help you remember how to use if, else, blocks, and nested if statements in case you need a refresher.

The exercise is designed in a special way to show you how you should write large programs by adding a little bit at a time. Please do these steps one at a time. The practice you will follow is called "refactoring."

To show your work increments, implement each step as a separate function in its own cpp file, called step1.cpp, step2.cpp, etc. Each time, copy the code from the previous step, rename the function, and then add the new modifications to the new function. Be aware that you will not always be adding to the end of the new function, and may even have to rearrange the order of the previous tests, in order to satisfy the new expectations presented in one or more of the steps.

To test your code, call the function for whichever step you are working on from main(). Note that in C++ you must provide "prototypes" of any functions that are called in the same file as the calls. At the top of the file with main, put the function prototypes. A prototype is a function's signature, but no body: void step1();

step1. Write the code that prompts the user to enter an integer between 1 and 20 (including 1 and 20), reads the value using cin, and then prints the value that they entered in a statement that begins with "You entered a ".

step2. Add more code or modify your code so that if the user enters a value that is less than 1 or greater than 20, it prints out "The value you entered is not between 1 and 20." Otherwise, and only if the value is between 1 and 20, it prints the statement as before.

step3. Add more code or modify your code so that if the user enters the values 8, 11, or 18, the statement that is printed is "You entered an" instead of "You entered a".

step4. Add more code or modify your code so that if the user enters something that is not a number, i.e. includes letters, your program does not print any of the above lines, but instead prints out "Idiot! Your input could not be read as a number. Get a life!". As mentioned in the previous assignment, you can test to see if the previous cin failed to read a value by calling cin.fail().

Include comments in your code to explain what each of the 'if' statements is testing.

Zip the cpp file(s) (with main plus the 4 steps) in a single zip file and submit the zip file.
You may discuss assignments with other students, but you must do your work separately. Do not submit a copy of the same work as another student.