*C++ requirements*

* The numbers need to be read in as type double. The calculations need to be done using type double as well.
* You must properly handle end of file conditions.
* You must properly open and close any files.

Failure to follow the C++ requirements could reduce the points received from passing the tests.

*General overview*

In this program you will be reading numbers from a file. You will validate the numbers and calculate the average of all of the valid numbers.

Your program will read in a file with numbers. The numbers will be of type double.

The numbers should be in the range from 0 to 110 (inclusive). You need to count all of the numbers between 0 and 110. You also need to calculate the average of these numbers.

If a number is not valid (that is, it is less than 0 or greater than 110) you need to count it (as a count of invalid values) and you need to write out the number to a file called "invalid-numbers.txt". Values written to file *invalid-numbers.txt* should be in fixed format with four digits to the right of the decimal point.

As you did in lab lesson 7 part 1 you need to read in the input file name using cin.

You need to properly handle end of file conditions. See lab lesson 7 part 1 for details on reading files. Also see Demo of file input and output of data in this unit.

The output from your program will be written to cout. The output must contain the file being processed, the total number of values read in from the file, the number of invalid values read in, the number of valid values read in.

The last thing you need to output is either the average of the valid values or an error message. The average must have three digits of precision to the right of the decimal point and must be in fixed format. If there is not valid average you should output the message:

An average cannot be calculated

In what case would you display this message?

If the input file cannot be opened, you will need to output a message. Assume the input file name is *badinput.txt* and it cannot be opened. You will display the following error message to cout

File "badinput.txt" could not be opened

Here is an example of a working program:

Assume the file name read in from cin is:

input.txt

and that *input.txt* contains:

-12

0

98.5

100

105.5

93.5

88

75

-3

111

89

-12

Your program would output the following:

Reading from file "input.txt"

Total values: 12

Invalid values: 4

Valid values: 8

Average of valid values: 81.188

The contents written out to file *invalid-numbers.txt* are:

-12.0000

-3.0000

111.0000

-12.0000

You are reading from an input file and you are writing to an output file. Make sure you close both files after you are finished using them. You must do this in your program, you cannot just let the operating system close the files for you.

For tests where there is output written to an output file the contents of the output file will determine if you passed that test or not. For cases where you have written out to cout the tests will check the output sent to cout. In some cases output will be written to a file and to cout. When this is the case the test will be run twice with the same input. Once to test cout and once to test the contents of the output file. An example of this would be tests 2 and 3. Both use the same input file. Test 2 check the output written to cout and test 3 checks the output written to the file *invalid-numbers.txt*.

Failure to follow the requirements for lab lessons can result in deductions to your points, even if you pass the validation tests. Logic errors, where you are not actually implementing the correct behavior, can result in reductions even if the test cases happen to return valid answers. This will be true for this and all future lab lessons.

*Expected output*

There are eight tests. Some tests are parts 1 and 2 of one test. Test 1 uses an input file that contains only valid values. Test 2 tests an input file that does not exist. Test 3 and 4 each have two parts. The first part tests the output to cout and the second part tests the contents of the output file invalid-numbers.txt. The final two tests have just one part.

You will get yellow highlighted text when you run the tests if your output is not what is expected. This can be because you are not getting the correct result. It could also be because your formatting does not match what is required. The checking that zyBooks does is very exacting and you must match it exactly. More information about what the yellow highlighting means can be found in course "How to use zyBooks" - especially section "1.4 zyLab basics".

Finally, do not include a system("pause"); statement in your program. This will cause your verification steps to fail.

*Note:* that the system("pause"); command runs the pause command on the computer where the program is running. The pause command is a Windows command. Your program will be run on a server in the cloud. The cloud server may be running a different operating system (such as Linux).