CSCE 240 HOMEWORK 2.5 DUE: FRIDAY 02 MARCH 2018

Let's get a bit more practice with C++ and concepts of software development. Extend your class as indicated in the attached header file. Do note you may have more methods or members than indicated in the provided header. What I have given you is a minimum, not a maximum. Be mindful of making changes to the public interface. As in the last assignment, I will be using it for testing.

This assignment is worth 4 out of 3 Points—3 for compiling with no warnings and passing all tests with an additional point for style.

## Existing: class should maintain at least the following from 02hw for unit testing:

- **constructor 1**: The default constructor sets both the real and imaginary component members to 0.0.
- **constructor 2**: Accepts a double value which is assigned to the imaginary component member. The real component member is set to 0.0.
- **constructor 3**: Accepts two double values for the real and imaginary component members. The first parameter should be assigned to the real and the second parameter assigned to the imaginary member.
- **real**: The overloaded method should return the real portion of the Complex instance or set it if a double parameter is passed.
- **imag**: The overloaded method should return the imaginary portion of the Complex instance or set it if a double parameter is passed.

## New: In addition to the above methods, you must create and update the following portions of the public interface:

- **ToComplex**: This static method shall accept a string of the type described in ToString and return a Complex class instance. In this version of the method, you must detect and assert the validity of the string representation. A viable representation is, in order:
  1. An open parenthesis '('
  2. 0 or more space characters (no '\t' or '\n' characters)
  3. A possibly signed ($\pm$) integer or floating point value
  4. 0 or more space characters
  5. Exactly one of
     (a) A '+' or '$-$' character, signifying an imaginary number is present, followed by
        i. 0 or more space characters
        ii. an unsigned integer or floating point value
     (b) The character 'i', signifying that no real is present, or
     (c) A close parenthesis character ')', signifying no imaginary number given, the Complex is found, and the method is done.
  6. 0 or more space characters
  7. A close parenthesis
- **LT** and **operator<**: These comparison methods should implement comparison via the complex of a modulus number. For a given complex $v = (\alpha + \gamma i)$, the modulus or $\bar{v}$ is defined as $\bar{v} = \sqrt{\alpha^2 + \gamma^2}$. You will need to determine a means to compare Complex instances to ints and doubles using this logic.

- **operator>>**: This function (friend or not) shall accept an istream object and Complex object. It shall extract a string of the type described in ToString from the stream and return a complex number. As in **ToComplex**, the method should assert the validity of string representation. The function should correctly accept and return the istream object so that it behaves as cin.
- **IsComplex**: these static methods should examine the contents of their respective string and stream instances to ensure that a Complex can be extracted from the contained representation. Note that the istream must be maintained—that is after you have evaluated its contents, you must place those contents back on the stream. You may want to read the istream::putback documentation. It will require a modicum of creativity to get this correct for the overloaded stream method.

## Submission

: To get credit, you must upload a zipped (not tarred, gzipped, or 7zwhatever-itscalled) archive containing your submission files in the the directory tree:

- **username/hw/02_05/**.

The archive file should be named:

- **username.zip.**

In both cases, the username should be your Blackboard login name. Your two files should be named:

- **complex.h**, and
- **complex.cc**.

Late submissions will be handled as per syllabus with no extensions or penalty reductions applied. Plan your work and questions accordingly.

The End.