Complex Number Class I

Assignment

Define a class for complex numbers, name it `myComplex`. In addition to the member functions that sets and gets the member attributes, define following member functions:

1. Constructors
   a. Default constructor
   b. Another constructor uses default parameters.
   c. Copy constructor
2. Assessors: setReal(), setImaginary()
3. Mutators: getReal(), getImaginary()
4. Overload output operators (<<) for `myComplex` class.
5. Overload input operator (>>) for `myComplex` class.
6. Overload assignment operator (=) for `myComplex` class

Write a driver program using `myComplex` class. Every member functions and operators should be called or activated to verify that they are working properly.
** Use UML to design the class (see Ch 13.15)

Objectives

The objective of this lab is to define a class with constructors and overload input/output operators.

- Define and implement `myComplex` class
- Define and implement constructors of `mComplex` class
- Overload operators: >>, <<, =

An Example of Program Output

This is the input/output screen using the given driver program:

```
+-------------------------------------------+
|  This program tests complex class operations |
+-------------------------------------------+
----Results of using constructors----
c1 = 0 + 0i
c2 = 3 - 2i
c3 = 3 - 2i

----using input operator-------------
Enter a complex number: 3 -2
c1 = 3 - 2i

----using assignment operator-------------
c1 = 3 - 2i
c2 = 3 - 2i

Press any key to continue . . .
```

Complex Number Class I

Design / Define a class for complex numbers

To design a complex class, we will list the information (i.e. data) of a complex number and actions that a complex number can do.

A complex number:
A complex number has the form of `a + bi`, where `a` and `b` are real numbers and `i` is the square root of – 1. We refer to `a` as the real part and `b` as the imaginary part of the complex number. For example: 2.5 + 3 i is a complex number. 2.5 is the real part and 3 is the imaginary part of 2.5 + 3 i.

Member attributes:
The class should have two member attributes to represent the real and imaginary parts of a complex number.

Overload input and output operators

Input operator (>>) is defined as a friend function of class complex. By doing so, the input operator can modify private member attributes. The function prototype is:

```
friend std::istream& operator >> (std::istream &ins, myComplex &c);
```

The output operator (<<) is defined as a non-member function of class complex. The function prototype takes the similar form as that of the input operator.

Submissions

Submit following 3 files, including your test result:

1. UML of complex class design
2. myComplex.h – header file
   with constructors, and input/output operator overloaded
3. myComplex.cpp – implementation file
4. Your driver program (lab04.cpp file)
5. your test result: result.txt
