CSCI 2120

Homework One – Class ComplexNumber

As you know we have data types for basic numbers; we have integers and we have decimal numbers (doubles and floats). We can accomplish quite a bit with these numbers, but what if we were engineers? Engineers often have to use complex numbers in their calculations to build airplanes, bridges, buildings, power plants, etc. Complex numbers do not exist as a primitive type in Java, but we can add them to our set of types. For this assignment, you will be writing class that models complex numbers.

Submission:

Create a new repository on gitlab for this assignments for this course. Add your instructor as a developer to your repository. Create a new directory within your repository. Name this directory HW1. As you write the program, add, commit, and push your files to the remote repository on the GitLab server. When you have finished the assignment, make sure you've uploaded the most up to date version of your files (check the website to see).

Assignment Statement:

1. Write a class that represents a complex number (i.e. a number with both real and imaginary parts). Also write a main method that "exercises" your complex number class and shows correct results for adding, subtracting, multiplying, and dividing multiple complex numbers. Tip: when testing your program to make sure your calculations are correct, use www.wolframalpha.com to verify your calculations.

Writing class ComplexNumber:

First, the class that defines a complex number: See below for a refresher on complex numbers. Your class will have a constructor that takes two floats as parameters. The two floats will represent the *real (a)* and *imaginary (b)* parts of the complex number.

Your class will have eight methods. Four of the methods will implement complex number addition, subtraction, multiplication, and division. Each will take a parameter: the other complex number to be added to, subtracted from, multiplied by, or divided into this complex number. Each of these methods will return a new complex number that is the result of the operation.

You will also need two getter methods to return the *a* and *b* parts of the complex number. You will provide an **equals** method by overriding Object's equals method. The last method will be a **toString** method that will return a string representing the complex number.

Here is a sketch of the class with the method for addition completed for you (You're welcome!).

```
class ComplexNumber{
     // The instance variables a and b representing
     // the real parts of the complex number
     private float a;
     private float b;
     public ComplexNumber(float _a, float _b){
          /* You fill in here! */
     }

     public ComplexNumber add(ComplexNumber otherNumber){
          ComplexNumber newComplex;
          float newA = a + otherNumber.getA();
          float newB = b + otherNumber.getB();
          newComplex = new ComplexNumber(newA, newB);
          return newComplex;
     }

     //...
     /* You will fill in the rest of the methods */
     //...
}
```

About Complex Numbers:

Recall that a complex number is a number

$$a + bi$$

where $i = \sqrt{-1}$

and a and b are real numbers.

To add two complex numbers:

$$(a + bi) + (c + di) = (a + c) + (b + d)i$$

Similarly, to subtract two complex numbers:

$$(a + bi) - (c + di) = (a - c) + (b - d)i$$

To multiply two complex numbers:

$$(a + bi)(c + di) = (ac - bd) + (bc + ad)i$$

Finally, to divide two complex numbers:

$$\frac{a + bi}{c + di} = \left(\frac{ac + bd}{c^2 + d^2}\right) + \left(\frac{bc - ad}{c^2 + d^2}\right)i$$

Due Date

You need to complete this assignment and have it pushed to the remote repository on the Git server by the date/time specified on Moodle.

There should be a README.txt file stating exactly how to compile, run, and test your code. Bonuses, if you chose to do them, should be separate (in subdirectories labeled bonus1, etc.) and fully complete implementations beyond the required one (in other words, start working from a fully implemented original copy in the subdirectory to do the bonus).

BONUS 1 (20 points): (Required for Honors Students) - Define a class RealNumber (a number without the imaginary part) and define methods in ComplexNumber that allow all the above operations on mixtures of RealNumbers and ComplexNumbers. Comment and test as above.

BONUS 2 (10 points): Can code duplication be reduced by using inheritance and having a common abstract superclass and/or interface? Implement and test this....