CS 315 Homework 1

Due Thursday, September 13, 2018 by class time

You need to **STUDY** section 7.3 (pages 241 to 250) in the text for this class before tackling the following problem. Follow the PATTERN on how to define a Rational class. Come to see me if you are even the slightest bit confused.

A complex number is a number in the form a + bi, where a and b are real numbers and i is $\sqrt{-1}$. The numbers a and b are known as the real part and imaginary part of the complex number, respectively. You can perform addition, subtraction, multiplication, and division on complex numbers using the following formulas:

(a + bi) + (c + di) = (a + c) + (b + d)i
(a + bi) - (c + di) = (a - c) + (b - d)i
(a + bi)*(c + di) = (ac - bd) + (bc + ad)i
(a + bi)/(c + di) = (ac + bd)/(c^2 + d^2) + (bc - ad)i/(c^2 + d^2)

You can also obtain the absolute value of a complex number using the following formula:

$|a + bi| = \sqrt{a^2 + b^2}$

Design a class named **Complex** for representing complex numbers. The class should define two instance variables, **a** and **b**, of type double. (You do not have to define an **i**. That is implied by b.) Define methods **add**, **subtract**, **multiply**, **divide**, and **abs** for performing complex number operations, and an overridden **toString** method for returning a string representation for a complex number. The **toString** method returns a string looking like **(a + bi)** with **a** and **b** being replaced by their actual values. For example: if the object has 1.2 in a and -3.4 in b, then the toString method should return a string looking like: (1.2 - 3.4i)
Note that in the example, I didn't put +-3.4i. You should be able to figure out how to put a plus sign when the imaginary part is positive but a minus sign when the imaginary part is negative.

Provide three constructors **Complex()** and **Complex(a, b)**.
Complex() creates a **Complex** object with a and b set to **0** and **Complex(a, b)** creates a **Complex** object with the a and b instance variables set to the values in the parameter list.
Also provide **getRealPart()** and **getImaginaryPart()** methods for returning the real and imaginary part of the complex number, respectively.

Design another class called **Real** that extends Complex. A Real number is a complex number that has **b** always set to 0. This class should have a default constructor that sets a to 0 in the superclass. It should also have an overloaded constructor with one parameter that sets the value of a in the superclass to that parameter.

You will have to create add, subtract, multiply, and divide methods that work on Real objects only so that two Real objects produce a Real result and not a Complex result. But, a Real and a Complex should produce a Complex result by automatically using the Complex methods. (See my Point2D and Point3D classes for examples.)

The toString method for this class should override the toString method of the Complex class so that only the value in a is returned as a string. That is, if the real number is 3.14, then the toString method of the Real class should return "3.14" and NOT "(3.14 + 0i)".

Write a test class called Arithmetic that will do Complex and Real number arithmetic. The program should ask the user to enter **two** strings. Each string will have one of the following formats:

```
R value
or
C value value
```

This lets the program know if the user entered a real number (R) or a complex number (C).
Once the user enters the two numbers the program will ask what the user wants to do with them, add, subtract, multiply, divide,

or find absolute value. The program should then display the results of the operation based on user feedback as follows:

num1 operator num2 = result

where num1 and num2 are the numbers to be operated on and operator is one of the operators add, subtract, multiply, and divide. For absolute value have the program display the answer in the format:

|num1| = answer1
|num2| = answer2

Examples:

User enters:

R 22.3
C 5 -1

Then, if the user lets the program know (you determine how) to add them, the result should be displayed as:

22.3 + (5 – 1i) = (27.3 – 1i)

If the user chose to find absolute values instead, then the result would be:

|22.3| = 22.3
|(5 – 1i)| = 5.09... (dots indicate more digits)

Test your program thoroughly.