Computer Science 181

Project 2 – A Complex Number Calculator

Write a C++ class that can be used to represent complex numbers, and that overloads several operations. Then, write a main program that uses your class to implement a simple complex number calculator.

Recall from elementary algebra that a complex number is a number of the form, where and are real numbers, and . The following rules are used to perform arithmetic on the complex numbers and .

* , for

Your complex class should represent the real and imaginary parts of the number as double instance variables. Give the class a single constructor that takes the values of the real and imaginary parts as parameters. Use default parameter values to 0.0 for both. *Do not provide any getters or setters for the instance variables.*

Overload the following binary operations on Complex objects: +, -, \*, /, ==. In addition, your class should overload the insertion and extraction operators << and >> so that Complex objects can be both read from the user and printed to the screen.

When overloading the << operator, your overloaded function should print the Complex number in the form a,b. When overloading the >> operator, you should assume that the user enters the complex number using the same form a,b. Note that there is no space between the real and imaginary parts and the comma in either the input or the output.

When you are overloading the == operator, recall that double values cannot be compared exactly in the computer. Consider two doubles and to be equal if .

After you have writen your Complex class, write a main program that operates as a simple calculator on complex numbers. When executed, your calculator should read a single complex number, a char representing one of the arithmetic operations, and a second complex number from the user, and then print out the result. Your program should use only the operators << and >> for IO.

The calculator must check to make sure it is not asked to perform division by zero. A complex number is equal to zero when both the real and imaginary parts are zero. Use the == operator to test for this in the main program before performing the division.

Here is an example of what your program **must** look like when it is executed. User input is shown in bold.

Enter a complex number problem: **1.0,-2.5 – 1.0,3.14**

Result: 0,-5.64

**What to turn in:** When you are ready to submit your program, upload all three files to blackboard.