CS368 Programming Assignment 4

Section 1, Fall 2015

Due by 11:59 pm on Monday, November 30

In this page: Announcements | Overview | Specifications | Handing in Related pages: Assignments

Announcements

Check here periodically.

11/20/2015 Program released. To ask questions about the homework and see questions posed by other students and their answers, go to: http://www.piazza.com/wisc/fall2015/cs3681 and sign-in using your wisc.edu account.

Overview

Why are we doing this program?

Description

For this assignment you are to implement the `ComplexNumber` class that corresponds to the interface (i.e., the header file) that we've provided. This requires overloading arithmetic, relational, assignment, and I/O operators. You are also to implement a separate `main` function (in the file `main.cpp`) that tests your `ComplexNumber` class.

Goals

The goals of this assignment are to give you experience:

- overloading operators.
- using a class with overloaded operators.

Specifications

What are the program requirements?

The Complex Numbers

Complex numbers are abstract quantities having a real and an imaginary part and are expressed in the form *a* + *b*i where *a* and *b* are real numbers (i.e., doubles) and *i* is the imaginary unit that results from taking the square root of -1. Arithmetic operations are defined for complex numbers such as:

- **Addition:** `(A + Bi) + (C + Di) = (A + C) + (B + D)i`
- **Subtraction:** `(A + Bi) - (C + Di) = (A - C) + (B - D)i`
- **Multiplication:** `(A + Bi) * (C + Di) = (A*C - B*D) + (A*D + B*C)i`

For a further review of complex numbers, take a look at this quick overview of complex numbers.

The `ComplexNumber` class

The `ComplexNumber` class is used to represent complex numbers and to define operations on complex

numbers. The interface for the `ComplexNumber` class is given in the header file `ComplexNumber.h`. You can copy this file from this location:

```
~cs368-1/public/html/assignments/p4/files/ComplexNumber.h
```

Complete the `ComplexNumber` class by writing all of the functions given in the header file. To do this you will be overloading several kinds of operators:

- assignment operators (=, +=, -=, *=)
- arithmetic operators (+, - , *)
- relational operators (==, !=)
- I/O operators (<<, >>)

The `main` Function

Write a `main` function to test the overloaded operators of the `ComplexNumber` class. Your `main` function should read in two complex numbers (using the overloaded input operator) and use the overloaded operators to compute (and display the results of) the following (in this order):

- `C1 + C2`
- `C1 - C2`
- `C1 * C2`
- `C1 == C2`
- `C1 != C2`
- `C1 += C2` (then reset `C1` to its original value)
- `C1 -= C2` (then reset `C1` to its original value)
- `C1 *= C2`

An example of how your `main` function should run and the output it should produce is given below:

```
Enter a complex number C1:
4.2 + 8.3i
Enter a complex number C2:
3.1 - 9.2i
For C1 = 4.2 + 8.3i and C2 = 3.1 - 9.2i :
C1 + C2 = 7.3 - 0.9i
C1 - C2 = 1.1 + 17.5i
C1 * C2 = 89.38 - 12.91i
C1 == C2 is false
C1 != C2 is true
After C1 += C2, C1 = 7.3 - 0.9i
After C1 -= C2, C1 = 1.1 + 17.5i
After C1 *= C2, C1 = 89.38 - 12.91i
```

You may assume that the user will always enter both the real and the imaginary parts of a complex number, e.g., the user will enter "`5 + 0i`" (and not "`5`") or "`0 - 6.2i`" (and not "`-6.2i`").

Handing in

What should be handed in?

Make sure your code follows the style and commenting standards used in CS 302 and CS 367. Note: the commenting standards use javadoc comments for class, method, and constructor headers. You do not need to use javadoc comments in your programs for CS 368; however, your comments should include the same

information as the javadoc comments. For example, your function header comments should include a description of what the function does, the name and a short description of each parameter, and a description of the return value.

Electronically submit the following files to your **`In`** "handin" directory by the due date and time (or refer to the late policy):

- **`main.cpp`** containing your `main` function,
- **`ComplexNumber.cpp`** containing your source code for the `ComplexNumber` class.

- Pair Programming students must also submit a `README` file: All students working in pairs must read the collaboration policy and submit a `README` file. Each student working in a pair must hand-in this file to his/her own hand-in directory. Copy, complete, and hand-in the file that is linked here:
  - `README.txt`

Do not turn in `ComplexNumber.h` that we've provided. You should **not** modify this file in any way. We will use the original version to execute your code for grading.

Please turn in only the file named above. Extra files clutter up the "handin" directories.

Last Updated: 11/16/2015 © 2015 CS368 Instructors