University of Ontario Institute of Technology

INFR 2140: Object Oriented Programming

Assignment 1

Date Posted: July 9th, 2018

Due Date: July 19th, 2018 (11:59pm) via Blackboard.

Instructions:

Not following these instructions will make you lose 10 points.

- This is an INDIVIDUAL assignment.
- You MUST have separate compilation files for each class you include in your program (header files and cpp files).
- Provide sufficient amount of comment.
- All your files need to have your name and student ID commented above the code.
- Submit a single zipped file containing the files.
- Name the zipped file with your student ID.
- Submit your assignment on Blackboard; No other mean of submission (e.g., email, printouts, etc.) is accepted.

(100 Points) - (Complex Number Class)

1) Define a class for complex numbers. A complex number is a number that can be expressed in the form ***a* + *bi***, where *a* and *b* are real numbers. For the complex number *a* + *bi*, *a* is called the ***real part***, and *b* is called the ***imaginary part***. Represent complex numbers as two values of type double, one for the real part (i.e., a) and one for the imaginary (i.e., b). Call the class Complex (Complex.h).

2) Include a constructor with two arguments that can be used to set the member variables of an object to any legitimate values. The constructor should contain default values in case no initializers are.

3) Include getter and setter functions for both data members.

4) Include a static data member which keeps track of the number of complex number objects created, ComplexNo.

5) Include a member function that returns the value ComplexNo data member and call the function, countNumber.

6) Overload the input and output operators >> (stream insertion operator) and << (stream extraction operator). The stream extraction operator prints the number as ***a* + *bi*** (e.g., 12 + 3i) and the stream insertion operator receives the user input as ***a* + *bi*** and extract the real and imaginary parts from that.

7) Overload + (addition), - (subtract), * (multiplication), == (equality), > (greater than), and += (addition assignment) operators for this class.

8) You also need to overload the prefix increment operator as well as and the postfix decrement operator. For these two operators, your program applies increment/decrement to the real part (i.e., a) of the complex number.

Include implementation of all of the above function definitions in a file named (Complex.cpp).

Write a test program (main.cpp) to test your class. In this program, you first instantiate 5 different complex numbers. Then, prompt user for two complex numbers and then apply different operations (i.e., +, -, *, >) on these two numbers and print the result of each operation. At the end of the program call then countNumber member function and print the total number of complex numbers instantiated in your program.