|  |  |  |
| --- | --- | --- |
| American University of SharjahCollege of Engineering Department of Computer Science & Engineering  P. O. Box 26666  Sharjah, UAE |  | Instructor: Dr. Aliaa Moualla **Lab Instructor**: Eng. Sameer Alawnah  **Office**: EB1-0012C  **Phone**: 971-6-515-4940  **e-mail**: salawnah@aus.edu  **Semester**: Spring 2024 |

**CMP 220L – Programming II**

**Lab 7: Operator overloading**

A complex number is a number with two components, namely, the real part and the imaginary part. For example the complex number (a+bi) has **a** as the real part and **b** as the imaginary part. In developing such a class, assume that both of a and b to be **floats**. The symbol i represent

**Develop and test** thoroughly the class **Complex** containing proper constructor function and the following overloaded operator and member functions:

* isEqual member function, that return true of the input Complex object is equal to the calling Complex object.
* Add member function, that add the input Complex to the calling object and return the result in a new Complex.
* == operator as non-member function, this operator should make use of the isEqual function.
* != operator as member function, this operator should make use of the isEqual function.
* The addition operator **(+)** as non-member function to add two Complex number
* The input member function to read the real and imaginary parts from an input stream.
* The output member function to print the Complex number in the format (real +/- imaginary i)
* The \* operator as member function to multiply two complex numbers

In the main() you should continuously read complex numbers from the user till the user enter 0+0i, then you should stop reading and print the summation and multiplication of all entered numbers.

**Hint**

* Addition of two complex numbers: (a+bi) +(c+di) = (a+c) + (b+d) i

The neutral value is 0 + 0i

* Two complex numbers are equal if the real parts of both are equal and the imaginary parts of both are equal
* Multiplication of two complex numbers: (a+bi)(c+di) = (ac-bd) + (ad+bc)i

The neutral value is 1 +0i

**A sample run is given below:**

Enter number 0 : 1.1 3.2

Enter number 1 : 4.5 6.1

Enter number 2 : 2.3 4.1

Enter number 3 : 10.2 6.9

Enter number 4 : 14.1 1.1

Enter number 5 : 0 0

The sum is 32.2+21.4i

The product is -15142.5-14551.5i

Good Luck ☺

#include <iostream>

#include <cmath>

#include <istream>

**using** **namespace** std;

**class** Complex{

**private**:

**float** rnumber;

**float** inumber;

**public**:

Complex(**float** rnumber = 0.0, **float** inumber = 0.0): rnumber(rnumber), inumber(inumber) {}

**bool** isEqual(**const** Complex& other)**const**

{

**return** ((rnumber == other.rnumber) && (inumber == other.inumber));

}

Complex add(**const** Complex& other)**const**

{

**return** Complex(rnumber + other.rnumber, inumber + other.inumber);

}

**bool** **operator** !=(**const** Complex& other)

{

**return** !isEqual(other);

}

Complex **operator**\*(**const** Complex& other)**const**{

Complex result;

result.rnumber = (rnumber \* other.rnumber) - (inumber \* other.inumber);

result.inumber = (rnumber \* other.inumber) + (inumber \* other.rnumber);

**return** result;

}

**friend** Complex **operator**+(**const** Complex& n1, **const** Complex& n2)

{

**return** n1.add(n2);

}

ostream& output(ostream& os)

{

**if**(inumber > 0){

os << rnumber << " + " << inumber;

}

**else**

os << rnumber << inumber; //correct

**return** os;

}

istream& input(istream& in)

{

in >> rnumber >> inumber;

**return** in;

}

};

**bool** **operator**==(**const** Complex& n1, **const** Complex& n2)

{

**return** n1.isEqual(n2);

}

**int** main(){

Complex com;

Complex zero(0,0);

Complex sum(0,0);

Complex prod(1,0);

**int** count = 0;

cout << "Enter number " << count << " ";

com.input(cin);

**while**(com!=zero)

{

sum = sum + com;

prod = prod \* com;

count++;

cout <<"Enter number " << count << " ";

com.input(cin);

}

cout << "The sum is ";

sum.output(cout);

cout << endl;

cout <<"The product is ";

prod.output(cout);

}

