OOP Assignment-1

21118

classmate
Date
Page

Problem Statement:
Implement a class complex which represents a complex number data type. Implement the following
- Constructor (including a default constructor which creates the complex number 0+0i
- Overload operator + to add two complex numbers
- Overload operator * to multiply two complex numbers.
- Overload operators << & >> to print & read complex numbers.

Objectives
To learn concept of constructor, default constructor, operator overloading using member function & friend function.

Outcomes
Implement operator overloading using CH OOP concepts.

Software Requirements:
Operating System : Windows 10 home single language
Ide: Code blocks version 20.03
compiler: g++ (version 10.0.0)

Hardware Requirement:
Manufacturer & model: Acer Swift 3 (Intel i5 8th gen)
Installed memory: 8GB RAM, 512 GB SSD.
Architecture: 64-bit

Theory:
operator overloading: It is a specific case of polymorphism where different operators are used to handle user defined data types.

Structure of class:

```
class Complex {
private:
    float real, img;
public:
    Complex() {  // constructor
        //code
    }
    inline Complex operator+(Complex C) {  // overloading + operator for addition of complex numbers
        //code
    }
    inline Complex operator-(Complex C) {  // Substraction of complex numbers
        //code
    }
    inline Complex operator*(Complex C) {  // Multiplication of complex no's
        //code
    }
    inline Complex operator/(Complex C) {  // Division of complex no's
        //code
    }
    friend ostream& operator<<(ostream& out, Complex& C);
    friend ostream& operator>>(istream& in, Complex& C);
};
```

insertion & extraction operators are overloaded to get input & to print complex number.

Pseudo codes:

```
Algorithm operator+ (Complex C1, Complex C2)
    Complex res;
    res.real ← C1.real + C2.real;
    res.img ← C1.img + C2.img;
```

return res;

}

Same pseudo code will be applicable for substraction & multiplication operation.

Algorithm operator / (Complex c1, Complex c2)

float divfact ← (c2.real * c2.real) + (c2.img * c2.img);

Complex res.

res.real ← ((c1.real * c2.real) + (c1.img * c2.img)) / divfact;

res.img ← (- (c1.real * c2.img) + (c1.img * c2.real)) / divfact;

return res;

}

Test cases

a = (1+2i) b = (2+0i)

Sr.No.	Test case	Expected output	Actual Output
1)	Addition	(3+2i)	(3+2i)
2)	Substraction	(-1+2i)	(-1+2i)
3)	Multiplication	(2+4i)	(2+4i)
4)	Division	(0.5+1i)	(0.5+1i)

Conclusion:

Hence, we have studied concept of operator overloading.

Input:

```
Enter the values of a and b for first complex number :
1 2

Enter the values of a and b for second complex number :
2 0


Choose Operation :
```

Addition:

```
Choose Operation :
0 to Exit
1 for addition
2 for substraction
3 for multiplication
4 for division
Enter Here : 1
Addition is : 3 + i(2)
```

Subtraction:

```
Choose Operation :
0 to Exit
1 for addition
2 for substraction
3 for multiplication
4 for division
Enter Here : 2
Subtraction is : -1 + i(2)
```

Multiplication:

```
Choose Operation :
0 to Exit
1 for addition
2 for substraction
3 for multiplication
4 for division
Enter Here : 3
Multiplication is : 2 + i(4)
```

Division:

```
Choose Operation :
0 to Exit
1 for addition
2 for substraction
3 for multiplication
4 for division
Enter Here : 4
Division is : 0.5 + i(1)
```