NAME	**SURAJ VASANTRAO WARBHE**
ROLL NO.	**231066**
GR NO.	**21910631**
SUBJECT	**OOP**
ASSIGNMENT NO.	**02(B)**
DATE	**04 September 2020**

****

ASSIGNMENT NO. : 2(B)

Design a class ‘Complex ‘with data members for real and imaginary part. Provide default and parameterized constructors. Write a program to perform Arithmetic operations of two complex numbers using operator overloading. Multiplication and division using member functions.

AIM : To design a class ‘Complex ‘with data members for real and imaginary part. Provide default and parameterized constructors and write a program to perform Arithmetic operations of two complex numbers using operator overloading like Multiplication and division using member functions.

OBJECTIVE : To understand and implement the concept of

1. Class and Objects
2. Operator overloading

THEORY :

1. OPERATOR OVERLOADING :

An overloaded declaration is a declaration that is declared with the same name as a previously declared declaration in the same scope, except that both declarations have different arguments and different definition (implementation). In C++, we can change the way operators work for user-defined types like objects and structures. This is known as operator overloading.

SYNTAX :

To overload an operator, we use a special operator function.

```
class className
{
 ... .. ...
    Public :
    returnType operator symbol (arguments)
       {
          ... .. ...
}   ... .. ...
};
```

Here,

a. return Type is the return type of the function.

b. operator is a keyword.

c. symbol is the operator we want to overload. Like: +, <, -, ++, etc.

d. arguments is the arguments passed to the function.

PROGRAM CODE :

```
#include <iostream>
using namespace std;
class complex
{
        float real;
        float imag;
  public:
        complex()
        {
                real = 0;
                imag = 0;
        }
        void ACCEPT()
        {
                cout << "\nReal Part : ";
                cin >> real;
                cout << "Imaginary Part : ";
                cin >> imag;
        }
        complex operator*(complex r);
        complex operator/(complex q);

        void DISPLAY()
        {
```

```
        cout << real << "+" << imag << "i"
            << "\n";
    }
};

complex complex::operator*(complex c1)
{
    complex c;
    c.real = c1.real * real - c1.imag * imag;
    c.imag = c1.imag * real + c1.real * imag;
    return c;
}

complex complex::operator/(complex c2)
{
    float m, n, p;
    complex c;
    m = (real * c2.real) + (imag * c2.imag);
    n = (c2.real * c2.real) + (c2.imag * c2.imag);
    p = (imag * c2.real) - (real * c2.imag);
    c.real = m / n;
    c.imag = p / n;
    return c;
}

int main()
```

```
{
	complex c1, c2, c3;
	int choice = 0;

	cout << "\nEnter First Complex Number";
	c1.ACCEPT();
	c1.DISPLAY();
	cout << "\nEnter Second Complex Number";
	c2.ACCEPT();
	c2.DISPLAY();

	do
	{
		cout << "\n********MENU********\n\n(1) MULTIPLICATION\n(2) DIVISION\n(3) EXIT\n\n***********************\n\n";
		cout << "Enter your choice : ";
		cin >> choice;

		switch (choice)
		{
		case 1:
			c3 = c1 * c2;
			c3.DISPLAY();
			break;

		case 2:
```

```
            c3 = c1 / c2;
            c3.DISPLAY();
            break;
        }
    } while (choice != 3);
    return 0;
}
```

<u>OUTPUT</u> :

```
Enter First Complex Number

Real Part : 1

Imaginary Part : 4

1+4i

Enter Second Complex Number

Real Part : 4

Imaginary Part : 3

4+3i


********MENU********

(1) MULTIPLICATION

(2) DIVISION

(3) EXIT

**********************

Enter your choice : 1

-8+19i


*********MENU*********

(1) MULTIPLICATION

(2) DIVISION

(3) EXIT

***********************

Enter your choice : 2
```

```
0.64+0.52i


*********MENU*********

(1) MULTIPLICATION

(2) DIVISION

(3) EXIT

***********************

Enter your choice : 3

[Program finished]
```

CONCLUSION : This Assignment helps us to learn basic concepts of class, objects and operator overloading in the 'C++' Programming language. We got an insight about how to overload the operators, accept and display it.

*** **