Don Bosco Institute of Technology, Kurla(W)
Department of Electronics and Tele-Communication Engineering
ECL304 - Skill Lab: C++ and Java Programming
Sem III
2021-22

Lab Number:	**5**
Student Name:	**Simran Santosh Koparkar**
Roll No :	**41**

Title:

To perform Operator Overloading using C++ for

- adding 2 complex numbers
- adding matrices

Learning Objective:

- Students will be able to perform user-defined overloading of built-in operators.

Learning Outcome:

- Understanding the overloading concept on built-in operators.

Course Outcome:

ECL304.2	Comprehend building blocks of OOPs language, inheritance, package and interfaces

Theory:

Explain about operator overloading with respect to:

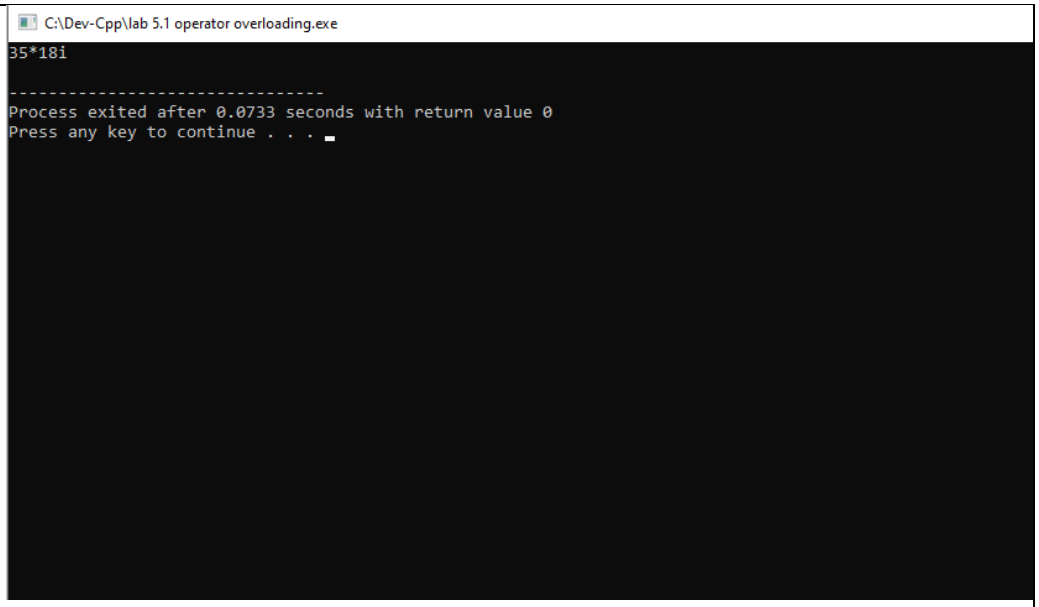
- constructor,
- methods and
- operators

We can change the way operators work for user-defined types like objects and structures. This is known as operator overloading.

Program :	`#include<iostream>` `using namespace std ;` `class complexno` `{` `public :`

```
                int real, imag;
                complexno()
                {
                        real = 0;
                        imag = 0;

                }
                complexno(int r, int i)
                {
                        real = r;
                        imag = i;
                }
                void display()
                {
                        cout << real << "*" << imag <<
"i" << endl;
                   }
                complexno operator *(complexno c)
                {
                        complexno temp;
                        temp.real = real * c.real;
                        temp.imag =     imag * c.imag;

                        return temp;
                }
};

int main()
{
        complexno c3;
        complexno c1(7,2);
        complexno c2(5,9);
        c3 = c1*c2;
        c3.display();
        return 0;
}
```

Input given:	-

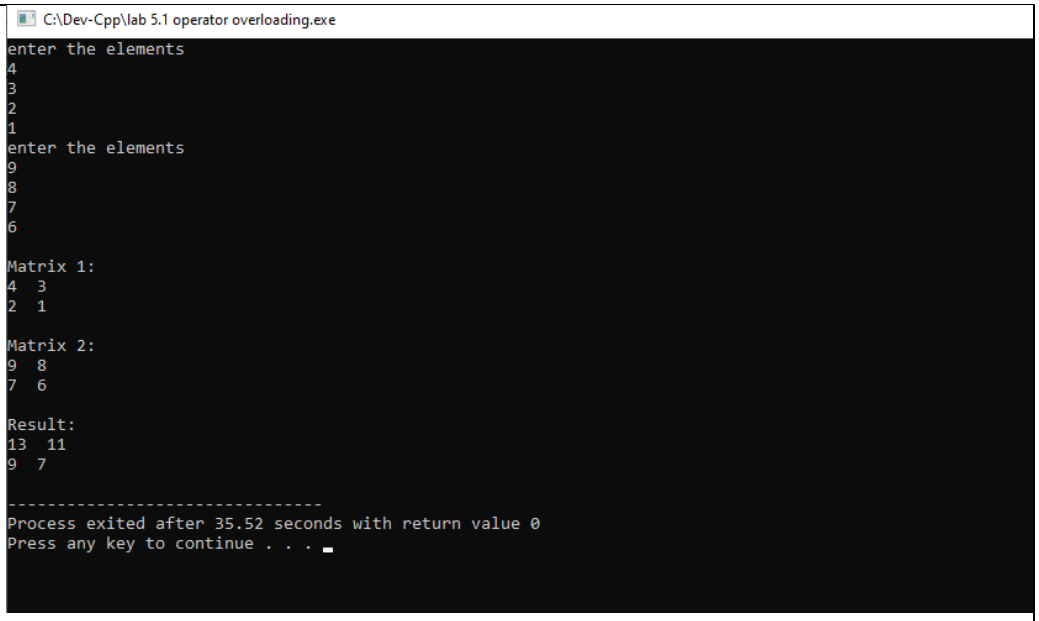
Output Screenshot:	

Program:	# include<iostream> using namespace std; class matrices { int a[2][2]; int b[2][2]; int c[2][2]; public:

```
                void get_elements();  //take numbers
from user

                matrices operator +(matrices m2);
        //operator overloading

                void display();       //print the
result

};



//functions outside class, using scope resolution



void matrices::get_elements()

{

        cout<<"enter the elements";

        for(int i=0;i<2;i++)  //for row

        {

                for(int j=0;j<2;j++)  //for columns

                        cin>>a[i][j];

        }

}



void matrices:: display()

{

        for(int i=0;i<2;i++)
```

```
        {
                for(int j=0;j<2;j++)
                        cout<<a[i][j]<<"  ";
                cout<<endl;
        }
}

matrices matrices::operator+(matrices m2)
{
        matrices m3;
        for(int i=0;i<2;i++)
        {
                for(int j=0;j<2;j++)
                        m3.a[i][j]=a[i][j]+m2.a[i][j];
        }
        return(m3);
}

int main()
{
        matrices ob1,ob2;
```

	`` ob1.get_elements(); ob2.get_elements(); cout<<"\nMatrix 1:\n"; ob1.display(); cout<<"\nMatrix 2:\n"; ob2.display(); ob1=ob1+ob2; cout<<"\nResult:\n"; ob1.display(); } ```
Input given:	**4** **3** **2** **1** **9** **8** **7** **6**

Output Screenshot:	

Don Bosco Institute of Technology, Kurla(W)
Department of Electronics and Tele-Communication Engineering
ECL304 - Skill Lab: C++ and Java Programming
Sem III
2021-22

Faculty: Ms. Deepali Kayande