

# OOP with C++

---

## Lab work - 08

Lab Date - 20th March 2021

Name - Rishabh

Regno. - 201800631

Semester - 4th

GitHub - <https://github.com/rishabh-live/oop-w-cpp-4-sem/tree/main/Labs>

---

1) Write a program to overload + and - operators using member functions.

### Source Code

```
// using member function
#include <bits/stdc++.h>

using namespace std;
class myclass {
    int a;
    public:
        void getdata(int x) {
            a = x;
        }
        myclass operator - (myclass o) {
            myclass temp;
            temp.a = a - o.a;
            return (temp);
        }
        myclass operator + (myclass o1) {
            myclass temp;
            temp.a = a + o1.a;
            return (temp);
        }
        void display(void) {
            cout << "A = " << a << "\n";
        }
};

int main() {
    myclass obj, obj1, obj2;
    obj.getdata(2);
    obj1.getdata(3);
    cout << "Before Overloading Initial Value\n";
    obj.display();
```

```

obj1.display();
obj2 = obj.operator + (obj1); // or obj2 = obj + obj1;
cout << "After overloading sum value\n";
obj2.display();
obj2 = obj.operator - (obj1); // or obj2 = obj - obj1;
cout << "After overloading subtracted value\n";
obj2.display();
return 0;
}

```

## Output

```

rishabh@DESKTOP-AUG0508U: ~/Desktop/cpp/OOP with CPP/Labs/Lab 8
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 7$ ./a.out
Enter Two Numbers:2 8
Sum of two numbers is 10
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 7$ cd ..
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs$ cd Lab/ 8
bash: cd: too many arguments
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs$ cd Lab\ 8
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$ ls
q1.cpp
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$ g++ q1.cpp -o q1
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$ ./q1
Before Overloading Initial Value
A = 2
A = 3
After overloading sum value
A = 5
After overloading subtracted value
A = -1
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$

```

2) Write a program Write a program to overload \* and / operators using friend functions.

## Source Code

```

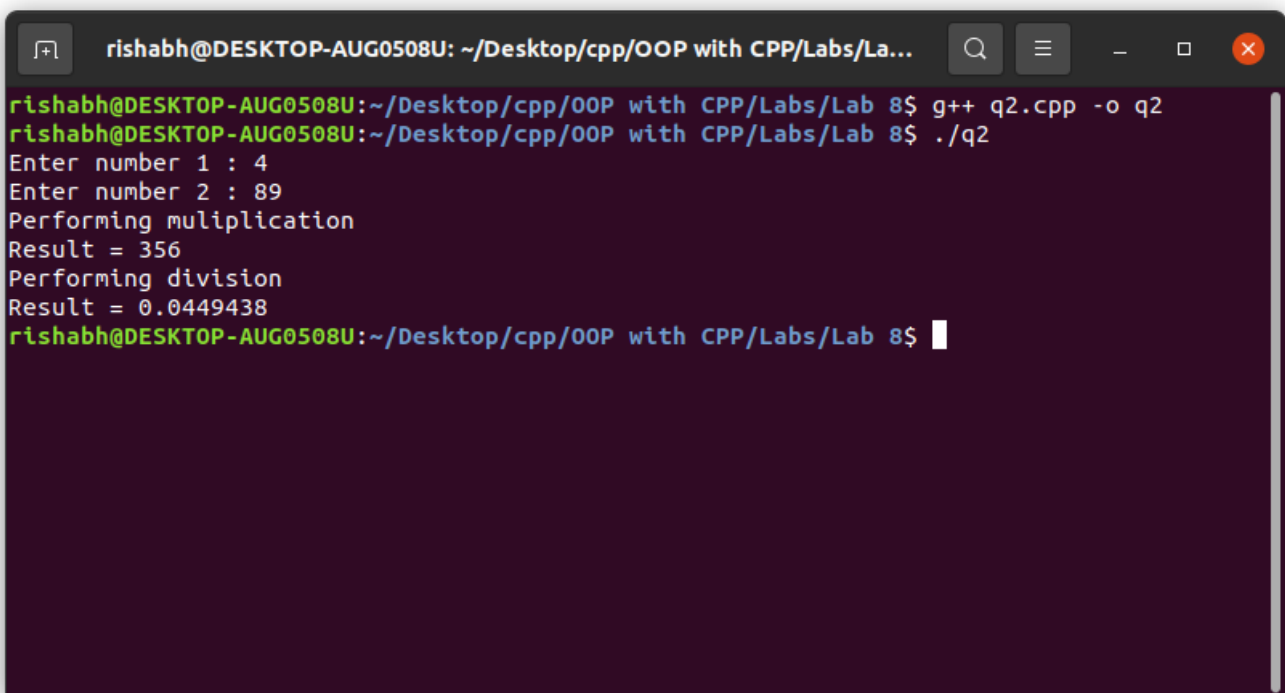
// using friend function
#include <bits/stdc++.h>

using namespace std;
int i = 1;
class myclass {
    float a;
public:
    void getdata(void) {
        cout << "Enter number " << i << " : ";
        cin >> a;
        i++;
    }
}

```

```
    }  
    friend myclass operator / (myclass a, myclass b) {  
        myclass temp;  
        temp.a = a.a / b.a;  
        return (temp);  
    }  
    friend myclass operator * (myclass a, myclass b) {  
        myclass temp;  
        temp.a = a.a * b.a;  
        return (temp);  
    }  
    void display(void) {  
        cout << "Result = " << a << "\n";  
    }  
};  
int main() {  
    myclass c1, c2, c3;  
    c1.getdata();  
    c2.getdata();  
    cout << "Performing muliultiplication\n";  
    c3 = c1 * c2;  
    c3.display();  
    cout << "Performing division\n";  
    c3 = c1 / c2;  
    c3.display();  
    return 0;  
}
```

### Output



```
rishabh@DESKTOP-AUG0508U: ~/Desktop/cpp/OOP with CPP/Labs/La...  
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$ g++ q2.cpp -o q2  
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$ ./q2  
Enter number 1 : 4  
Enter number 2 : 89  
Performing muliultiplication  
Result = 356  
Performing division  
Result = 0.0449438  
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$
```

3) Create a class 'COMPLEX' to hold a complex number. Write a friend function to add, subtract and multiply two complex numbers. Also implement the following operator overloading functions for COMPLEX numbers. (a) >> operator to take input of a complex number (b) << operator to display a complex number in the form of a+ib (c) + operator to add two complex number. – operator to subtract one from other complex number – operator to multiply two complex number (d) == to compare two complex number.

### Source Code

```
#include <bits/stdc++.h>

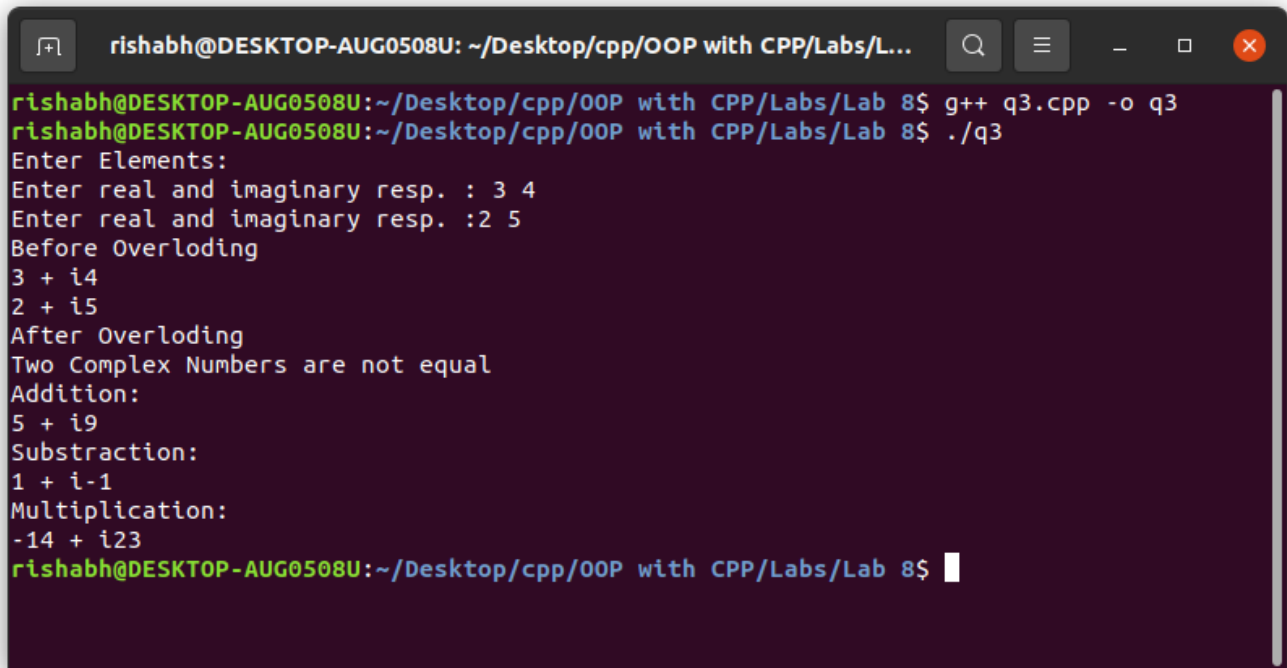
using namespace std;
class complex1 {
    float x, y;
public:
    friend void operator == (complex1 o, complex1 o1) // to compare
    {
        if ((o.x == o1.x) && (o.y == o1.y))
            cout << "Two Complex Numbers are equal\n";
        else
            cout << "Two Complex Numbers are not equal\n";
    }
    friend complex1 operator + (complex1 o, complex1 o1) // to add
    {
        complex1 temp;
        temp.x = o.x + o1.x;
        temp.y = o.y + o1.y;
        return (temp);
    }
    friend complex1 operator - (complex1 o, complex1 o1) // to subtract
    {
        complex1 temp1;
        temp1.x = o.x - o1.x;
        temp1.y = o.y - o1.y;
        return (temp1);
    }
    friend complex1 operator * (complex1 o, complex1 o1) // to multiply
    {
        float prod1, prod2, prod3;
        complex1 temp1;
        prod1 = o.x * o1.x;
        prod2 = o.y * o1.y;
        prod3 = (o.x + o.y) * (o1.x + o1.y);
        temp1.x = prod1 - prod2;
        temp1.y = prod3 - (prod1 + prod2);
        return (temp1);
    }
    // overloading >> , << operators
    friend istream & operator >> (istream & din, complex1 & v) // to take
    input
    {
```

```

    cout << "Enter real and imaginary resp. :";
    din >> v.x >> v.y;
    return (din);
}
friend ostream & operator << (ostream & dout, complex1 & v1) // to
display output
{
    dout << v1.x << " + i" << v1.y << "\n";
    return (dout);
}
};
int main() {
    complex1 c1, c2, c3, c4, c5; // objects declaration
    cout << "Enter Elements:\n"; // input overloading
    cin >> c1;
    cin >> c2;
    cout << "Before Overloading\n"; // output overloading
    cout << c1;
    cout << c2;
    cout << "After Overloading\n";
    c1 == c2;
    // == operator overloading
    cout << "Addition:\n"; // + operator overloading
    c3 = c1 + c2;
    cout << c3;
    // << operator overloading
    cout << "Substraction:\n"; // - operator overloading
    c4 = c1 - c2;
    cout << c4;
    // << operator overloading
    cout << "Multiplication:\n"; // * operator overloading
    c5 = c1 * c2;
    cout << c5;
    // << operator overloading
    return 0;
}

```

### Output



```
rishabh@DESKTOP-AUG0508U: ~/Desktop/cpp/OOP with CPP/Labs/Lab 8$ g++ q3.cpp -o q3
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$ ./q3
Enter Elements:
Enter real and imaginary resp. : 3 4
Enter real and imaginary resp. :2 5
Before Overloading
3 + i4
2 + i5
After Overloading
Two Complex Numbers are not equal
Addition:
5 + i9
Substraction:
1 + i-1
Multiplication:
-14 + i23
rishabh@DESKTOP-AUG0508U:~/Desktop/cpp/OOP with CPP/Labs/Lab 8$
```