LAB - 11

Q1. Write a program to overload unary(-) operator

```
#include <iostream>
using namespace std;

class NUM
{
private:
    int n;

public:
    void getNum(int x)
    {
        n = x;
    }
    void dispNum(void)
    {
        cout << "value of n is: " << n <<
    endl; }
    void operator-(void)
    {
        n = -n;
    }
};

int main()
{
    NUM num;
    num.getNum(10);
    num.dispNum(); -
    num;
    num.dispNum();
    cout << endl;
    return 0;
}
```

```
value of n is: 10
value of n is: -10
```

Q2. Write a program to overload unary(-) operator using friend function.

```
#include <iostream>
using namespace std;
class NUM
{
    int a, b, c;

public:
    void
    getNum(){
        cout << "Values of A, B & C\n";
        cout << "a ";
        cin >> a;
        cout << "b ";
        cin >> b;
        cout << "c ";
        cin >> c;
    }
    void
    dispNum(){
        cout << a << "\n"
             << b << "\n"
             << c << "\n"
             << endl;
    }
    void friend operator-(NUM
&x); };
void operator (NUM &x)
```

```
void operator-(NUM
&x){
    a= -x.a;
    x.b = -x.b;
    x.c = -x.c;
}
int main()
{
    NUM x1;
    x1.getNum();
    cout << "Before Overloading\n";
    x1.dispNum();
    cout << "After Overloading \n";
    -x1;
    x1.dispNum();
    return 0;
}
```

```
Values of A, B & C
a 10
b 12
c -20
Before Overloading
10
12
-20

After Overloading
-10
-12
20
```

Q3. Write a program to overload unary(++) operator and unary (--) operator.

```
#include <iostream>
using namespace std;
class Int
{
private:
    int i;

public:
    Int(int i =
    0){
        this->i = i;
    }
    Int operator+
    +(){
        Int temp;
        temp.i = ++i;
        return temp;
    }
    Int operator+
    +(int){
        Int temp;
        temp.i = i++;
        return temp;
    }
    Int operator--
    (){
        Int temp;
        temp.i = --i;
        return temp;
    }
    Int operator--
    (int){
        Int temp;
        temp.i = i--;
        return temp;
    }
    void
    display(){
        cout << "i = " << i << endl;
    }
```

```
    }
};
int main()
{
    Int i1(3), i4(3), i5(3), i7(3);
    cout << "Before increment: ";
    i1.display();
    Int i2 = ++i1;
    cout << "After pre increment: ";
    i2.display();
    Int i3 = i4++;
    cout << "After post increment: ";
    i3.display();
    cout << "_____" << endl;
    cout << "Before Decrement: ";
    i1.display();
    Int i6 = --i5;
    cout << "After pre Decrement: ";
    i6.display();
    Int i8 = i7--;
    cout << "After post Decrement: ";
    i8.display();
}
```

```
Before increment: i = 3
After pre increment: i = 4
After post increment: i = 3
------------------------
Before Decrement: i = 4
After pre Decrement: i = 2
After post Decrement: i = 3
```

Q4. Write a program to overload unary(++) operator and unary (--) operator using friend function.

```
#include <iostream>
using namespace std;

class opr
{
    int x, y;

public:
    void
    input(){
        cout << "Enter the values of x and y : ";
        cin >> x >> y;
    }

    friend void operator--(opr &o);
    friend void operator++(opr &o);

    void display()
    {
        cout << "\n x : " << x; cout
    << "\n y : " << y;
    }
};
void operator--(opr &o)
{
    o.x--;
    o.y--;
}
void operator++(opr &o)
{
    o.x++;
    o.y++;
}

int main()
{
    opr obj;
    obj.input
```

```
    obj.input();

    --obj;
    cout << "\n After Decrementing : ";
    obj.display();

    ++obj;
    cout << "\n After Incrementing : ";
    obj.display();

    return 0;
}
```

```
Enter the values of x and y : 10 12

 After Decrementing :
 x : 9
 y : 11
 After Incrementing :
 x : 11
 y : 13
```

Q5 Write a program to design a class representing complex numbers and having functionality of performing addition and multiplication of two complex numbers using operator overloading.

```
#include <iostream>
#include <conio.h>
using namespace std;

class Complex
{
public:
    int real, img;
    void add(Complex c1, Complex
    c2){
        int x, y;
        x = c1.real + c2.real;
        y = c1.img + c2.img;
        cout << "\n(" << c1.real << "+" << c1.img << "i)+(" << c2.real << "+"
             << c2.img << "i)=(" << x << "+" << y << "i)";
    }
    void multiply(Complex c1, Complex
    c2){
        int x, y;
        x = c1.real * c2.real - c1.img * c2.img;
        y = c1.real * c2.img + c1.img * c2.real;
        cout << "\n(" << c1.real << "+" << c1.img << "i)*(" << c2.real << "+"
             << c2.img << "i)=(" << x << "+" << y << "i)";
    }
};

int main()
{
    Complex a, b, c, d, e;
    cout << "\nEnter real and imaginary part of first complex number:";
    cin >> a.real >> a.img;
    cout << "\nEnter real and imaginary part of second complex number:";
    cin >> b.real >> b.img;
    c.add(a, b);
    d.multiply(a, b);
}
```

```
Enter real and imaginary part of first complex number:10 12

Enter real and imaginary part of second complex number:20 18

(10+12i)+(20+18i)=(30+30i)
(10+12i)*(20+18i)=(-16+420i)
```

Q6. Wap add two string using binary operator overloading.

```
#include <iostream>
```

```
#include <iostream>
#include <string.h>
using namespace std;

class String
{
public:
    char str[20];

public:
    void
    accept_string(){
        cout << "\n Enter String: ";
        cin >> str;
    }
    void display_string(){
        cout << str;
    }
    String operator+(String
    x){

        String s;
        strcat(str, x.str);
        strcpy(s.str, str);
        return s;
    }
};
int main()
{
    String str1, str2, str3;

    str1.accept_string();
    str2.accept_string();
    cout << "\n ----------------------------------------";
    cout << "\n\n First String is  : ";
    str1.display_string();
    cout << "\n\n Second String is : ";
    str2.display_string();
    cout << "\n ----------------------------------------";
    str3 = str1 + str2;
    cout << "\n\n Adding String    : ";
    str3.display_string();
    return 0;
}
```

```
Enter String: Anupam

Enter String: Moharana

------------------------------------------------

First String is  :  Anupam

Second String is :  Moharana
------------------------------------------------

Adding String    :  AnupamMoharana
```