Assignment - 27 **A Job Ready Bootcamp in C++, DSA and IOT** **MySirG**

Operator overloading and friend function

1. Define a class Complex with appropriate instance variables and member functions. Define following operators in the class:
   a. +
   b. -
   c. *
   d. ==
2. Write a C++ program to overload unary operators that is increment and decrement.
3. Write a C++ program to add two complex numbers using operator overloaded by a friend function.
4. Create a class Time which contains:
   - Hours
   - Minutes
   - Seconds

   Write a C++ program using operator overloading for the following:
   1. = = : To check whether two Times are the same or not.
   2. >> : To accept the time.
   3. << : To display the time.

   Output -

```
Enter First Time
--------------------
Enter Hours   :  24

Enter Minutes :  30

Enter Seconds :  40

First Time
Hours   :  24
Minutes :  30
Seconds :  40

Enter Second Time
--------------------
Enter Hours   :  24

Enter Minutes :  30

Enter Seconds :  40

Second Time
Hours   :  24
Minutes :  30
Seconds :  40

Times are Same
```

5. Consider following class Numbers

   class Numbers

```
{
        int x,y,z;
        public:
                // methods
};
```

Overload the operator unary minus (-) to negate the numbers.

6. Create a class CString to represent a string.

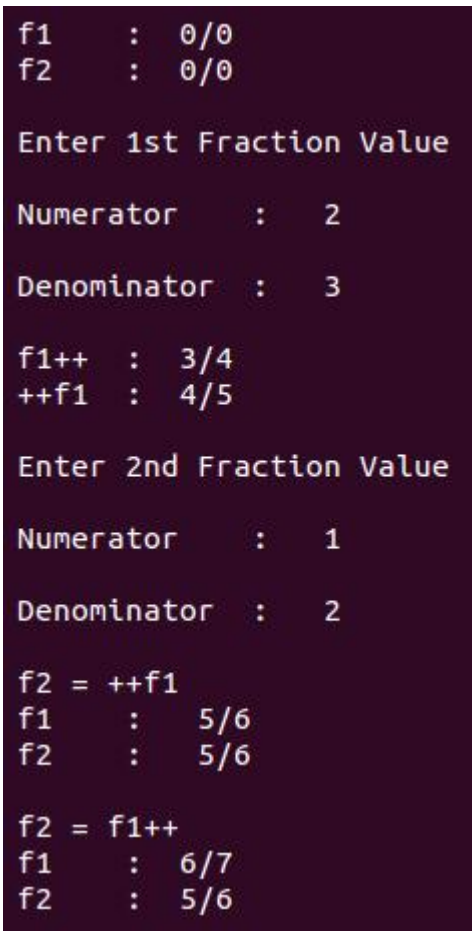
   a) Overload the + operator to concatenate two strings.
   b) == to compare 2 strings.

7. Define a C++ class fraction

```
class fraction
{
        long numerator;
        long denominator;
    Public:
        fraction (long n=0, long d=0);
}
```

Overload the following operators as member or friend:

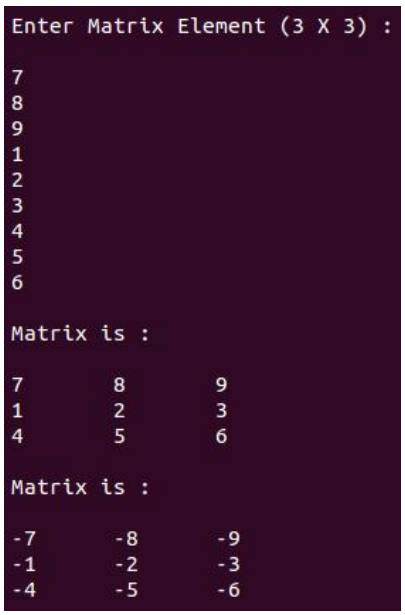
a) Unary ++ (pre and post both)
b) Overload as friend functions: operators << and >>.
Output-



8. Consider a class Matrix

```
Class Matrix
{
      int a[3][3];
   Public:
     //methods;
};
```

Overload the - (Unary) should negate the numbers stored in the object.
Output -



9. Consider the following class mystring

```
Class mystring
{
        char str [100];
   Public:
        // methods
};
```

Overload operator "!" to reverse the case of each alphabet in the string (Uppercase to Lowercase and vice versa).

10. *Class Matrix*

```
{
        int a[3][3];
   Public:
        //methods;
};
```

Let m1 and m2 are two matrices. Find out m3=m1+m2 (use operator overloading).
Output -

```
Enter Matrix Element (3 X 3) :
4 5 6 1 2 3 7 8 9

Enter Matrix Element (3 X 3) :
1 2 3 4 5 6 7 8 9

First Matrix :

4       5       6
1       2       3
7       8       9

Second Matrix :

1       2       3
4       5       6
7       8       9

Addition of Matrix :

5       7       9
5       7       9
14      16      18
```