OOP LABORATORY 12

Name: **ANIRBAN HAZRA**
Section: **B-12**
Roll : **2005643**

1. Write a program to overload <<,>> operator for complex class. Overload any operator of your choice to find modulus of a complex number.

PROGRAM CODE:

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

class Complex {
  private:
    int r;
    int i;

  public:
     Complex()
     {
        r=0;
        i=0;
     }
    Complex(int a, int b) {
      r=a;
      i=b;
    }
    friend ostream &operator<<( ostream &output, const Complex &D ) {
      output<< D.r<< "+" << D.i<<"i" ;
      return output;
    }

    friend istream &operator>>( istream  &input, Complex &D ) {
      input >> D.r >> D.i;
      return input;
    }
   void operator++()
   {
         float mod;
         mod=sqrt(pow(r,2)+pow(i,2));
         cout<<mod;
         }
```

```
};

int main()
{
  Complex  D1;

  cout << "Enter the real and imaginary part of 1st complex number : " << endl;
  cin >> D1;
  cout << "First Complex : " << D1 << endl;

 cout << "\nThe modulus of Complex Number is ";
  ++D1;
   return 0;
}
```

OUTPUT:

```
Enter the real and imaginary part of 1st complex number :
5 8
First Complex : 5+8i

The modulus of Complex Number is 9.43398
```

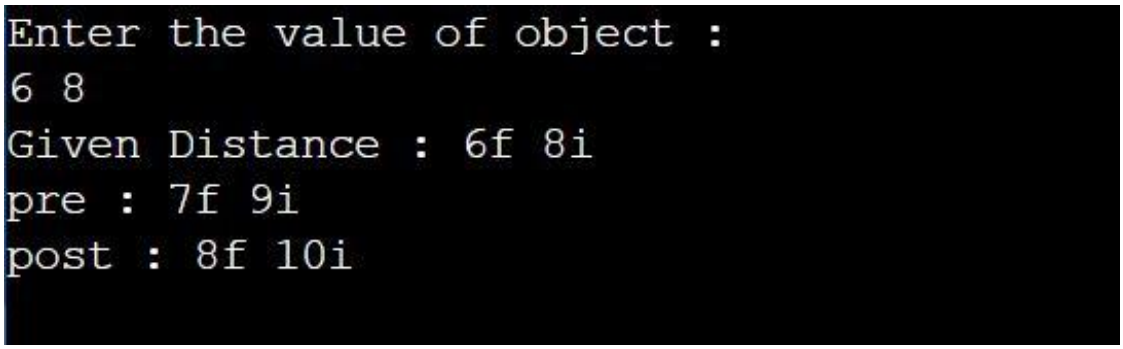
2. <u>WAP to overload ++ operator (post and pre) for distance class.</u>

PROGRAM CODE:

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

class Distance {
  private:
    int feet;
    int inches;

  public:

    Distance()
    {
      feet = 0;
      inches = 0;
    }

    friend ostream &operator<<( ostream &output, const Distance &D )
    {
      output<< D.feet<< "f " << D.inches  << "i ";
      return output;
    }
```

```
    friend istream &operator>>( istream  &input, Distance &D )
    {
      input >> D.feet >> D.inches;
      return input;
    }
    Distance operator++()
    {
       ++feet;
       ++inches;
    }
    Distance operator++(int)
    {
       feet++;
       inches++;
    }

};

int main() {
  Distance  D1;

  cout << "Enter the value of object : " << endl;
  cin >> D1;

  cout << "Given Distance : " << D1 << endl;

  ++D1;

   cout << "pre : " << D1 << endl;

  D1++;

   cout << "post : " << D1 << endl;

   return 0;
}
```

OUTPUT:



3. WAP to overload assignment operator for a class which stores an integer array dynamically. The size of the array is entered by the user for each object. Overload assignment and decrement (--operator), both pre and post format for the class.

PROGRAM CODE;

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

class num
{
   int c;
   int *arr;

   public:
   num()
   {
      arr = new int();
   }

   friend istream &operator>>(istream  &input, num &obj )
   {
      cout<<"Enter no. of elements in the array :";
      input>>obj.c;
      obj.arr = new int(obj.c);
      cout<<"Enter "<<obj.c<<" elements->\n";
      for(int i = 0; i<obj.c; i++)
      {
         cout<<"Item "<<i+1<<" :";
         input>>obj.arr[i];
      }
      return input;
   }

   friend ostream &operator<<(ostream &output, const num &obj)
   {
      cout<<"Elements->\n";
      for(int i = 0; i<obj.c; i++)
      {
         output<<obj.arr[i]<<" ";
      }
      return output;
   }

   num operator--()
   {
      num n;
      int x;
```

```
        for(int i = 0; i<c; i++)
        {
            x = --arr[i];
            n.arr[i] = x;
        }
        return n;
    }

    num operator--(int)
    {
        num n;
        int x;
        for(int i = 0; i<c; i++)
        {
            x = arr[i]--;
            n.arr[i] = x;
        }
        return n;
    }
};

int main()
{
    num n1;
    cin>>n1;
    cout<<n1;
    cout<<"\nPre decrement : ";
    --n1;
    cout<<n1;
    cout<<"\nPost decrement : ";
    n1--;
    cout<<n1;
    return 0;
}
```

OUTPUT:

```
Enter no. of elements in the array :4
Enter 4 elements->
Item 1 :3
Item 2 :5
Item 3 :7
Item 4 :9
Elements->
3 5 7 9
Pre decrement : Elements->
2 4 6 8
Post decrement : Elements->
1 3 5 7
```

4. **Create a class which stores the x and y coordinates of a point. Overload any operator of your choice to find the distance between two points (objects). When only one object is provided, find the distance between the object and the origin.**

PROGRAM CODE:

```
#include<iostream>
#include<math.h>
using namespace std;
class point
{
   float x,y;
   float s;


public:
   point()
   {
      x=0;
      y=0;
   }
   friend ostream &operator<<( ostream &output, const point &D ) {
       output<<"("<< D.x<< "," << D.y<<")" ;
       return output;
   }

   friend istream &operator>>( istream  &input, point &D ) {
       input >> D.x >> D.y;
       return input;
   }
   point operator+ (point p)
   {
      float s2,a,b;
      point t;
      a=pow(p.x-x,2);
      b=pow(p.y-y,2);
      s2=a+b;

      t.s=sqrt(s2);
      cout<<t.s;

      return t;
   }


};

int main()
{
```

```
    point p1,p2,s;

    cout<<"enter the 1st point= ";
    cin>>p1;

    cout<<"enter the 2nd point= ";
    cin>>p2;
    cout<<"distance=  ";
     s=p1+p2;

    return 0;
}
```

OUTPUT: