JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY–NOIDA
CI311- Object-Oriented Programming
Tutorial / LAB – Week 8

Lab Test will be conducted from 6th to 11th October of 20 marks.

1. Define a class String. Write the constructors, destructor, the copy constructor. Overload =, = =, ! (empty string), <, >, !=, +=, () (has two parameters int, int and returns a substring), [](subscript), << and >>.

2. Define a class queue that contains elements of type integer. Define two operators on queue, '+' to insert an element in it and '–' to remove and element from it. Use the friend function approach first and then the one without friends.

3. Create a C++ class PolyLink to represent and manipulate univariate (single variable) polynomials using a single linked list. Provide member functions that use polynomial objects for read, display, copy, assign, add, subtract, multiply behaviour and evaluate the polynomial at a given value of x. Define two operators in this class, '+' to add two polynomial objects and '=' to assign one polynomial object to another. Write a main function to test the functionality of the two operators on the objects of class Polynomial. Your program should be designed to handle multiple assignments. You are required to increment coefficient of each term by value given by user at runtime. Use the friend function approach to handle both situations.

Polynomial P;
P+5;
5+P;

4.Design a class Numdays. The class's purpose is to store a value that represents a number of work hours and convert it to a number of days. For example 8 hours would be converted into 1 day, , 8 hours would be converted into 1.5 day, 18 hours would be converted into 2.25 days. The class should have a constructor that accepts a number of hours, as well as member function for storing and retrieving the hours and days. The class should also have following overloaded operators:-

+ addition operator:-when two numdays object are added together , the overloaded + operator should return the sum of the two objects hours members.
-subtraction operators:- :-when one numdays object subtracted from another, , the overloaded - operator should return the difference of the two objects hours members.
++prefix and postfix increment operators:-these operator should increment the number of hours stored in the object. When incremented , the number of days should be automatically recalculated.
--prefix and postfix increment operators:- these operator should decrement the number of hours stored in the object. When decremented , the number of days should be automatically recalculated.

5.Find the output of the following :-

(a).

```
 #include<iostream.h>
using namespace std;
class space{
int x,y,z;
public:
void getdata(int a,int b,int c);
void display(void);
void operator-();
};
void space :: getdata(int a,int b,int c)
{
x=a;
y=;
```

```
z=c;
}
void space :: display(void)
{
cout<< x<<" ";
cout<< y<<" ";
cout<< z<<"\n ";
}
void space :: operator-()
{
x=-x;
y=-y;
z=-z;
}
int main()
{
space s;
s.getdata(10,-20,30);
cout<< "s :";
s.display();
-s;
cout<<" s :";
s.display();
return 0;
}
```

(b).

```
#include<iostream.h>
using namespace std;
class complex
{
float x,y;
public:
complex(){  }
complex(float real, float img)
{
x=real;
y=img;
}
complex operator+(complx);
void display(void);
};
complex complex :: operator+(complx  c)
{
complx t;
t.x= x+c.x;
t.y= y + c.y;
return(t);
}
void complex :: display(void)
{
cout << x << "+j" <<y <<"\n";
}
int main()
{
complex c1,c2,c3;
```

```
c1= complex(2.5,3.5);
c2 = complex(1.6,2.7);
c3 = c1+c2;
cout<< “c1 = “;c1.display();
cout<< “c2 = “;c2.display();
cout<< “c3 = “;c3.display();
return 0;
}
```

(c).

```
#include<iostream.h>
using namespace std;
template <class t>
void roots(t a,t b t )
{
t d= b*b – 4*a*c;
if( d==0)
cout<<” r1=r2 =” <<-b/(2*a) <<endl;
else if(d>0)
{
cout<<” roots are real”;
float r= sqrt(d);
float  r1 = (-b+r)/(2*a);
float r2 = (-b-r)/(2*a);
cout<<” r1 = “<<r1<<”and”;
cout<<”r2 =”<<r2;
}
else
{
cout<<”rots are complex”;
float r1=-b/(2*a);
float r2= sqrt(-d)/(2*a);
cout<<” real part= “<<r1;
cout<<”complex part =”<< r2;
}
}
int main()
{
cout<<”\n integer coefficient”;
roots(1,-5,6);
cout<<” float coefficient”;
roots(1.5,3.6,5.0);
return 0;
}
```

(d).

```
#include<iostream.h>
using namespace std;
class class_2;
class class_1
{
int value1;
public:
void indata(int a)
{
value1= a;
}
```

```
void display(void)
{
cout<< value1<<"\n";
}
friend void exchange(class_1 &, class_2 &);
};
class class_2
{
int value2;
void indata(int a)
{
value2= a;
}
void display(void)
{
cout<< value2<<"\n";
}
friend void exchange(class_1 &, class_2 &);
};
void exchange(class_1 &, class_2 &)
{
int temp = x.value1;
x.value1 = y.value2;
y.value2 = temp;
}
int main()
{
class_1 c1;
class_2 c2;
c1.indata(100);
c2.indata(200);
cout<<" values before exchange";
c1.display();
c2.display();
exchange(c1,c2);
cout<<" values after exchange";
c1.display();
c2.display();
return 0;
}
```

5. Find the error in the following code:-

```
(a).class point
{
private:
int x1 ,y1;
public:
point(int x, int y)
{
x1=x;
y1=y;}
void operator+(const &point right)
{
x1 +=right.x1;
y1 +=right.y1;
```

```
}
};
(b).class box
{
private:
float wi, le, he;
public:
box(float w,l,h)
{
wi=w;
le=l;
he=h;
}
void operator++()
{
++wi;
++le;}
void operator++()
{
wi++;
le++;}
};
(c).
class yard{
private:
float le;
public:
yard(float l)
{le=l;}
void operator float()
{
return le;}
};
```