LAB 6

1. /*Create a base class called shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called triangle and rectangle from the base shape. Add to the baseclass, a member function get_data( ) to initialize base class data members and another member function display_area( ) to compute and display the areaof figure. Make display_area( ) as a virtual function and redefine this function in the derived classes to suit their requirements.*/

```
#include<iostream>
using namespace std;
class shape
{
        protected:
                double x , y;
        public:
                void get_data(int a , int b)
                {
                        x=a;
                        y=b;
                }
                virtual void display_area()=0;
};
class triangle: public shape
{
        public:
                void display_area()
                {
                        cout<<"The area of triangle is "<<(x*y/2);
                }
};
class rectangle: public shape
{
        public:
                void display_area()
                {
                        cout<<"The area of rectangle is "<<(x*y);
                }
};
int main()
{
        triangle t;
        shape *s = &t;
        s->get_data(5,6);
        t.display_area();

        cout<<endl<<endl;

        rectangle r;
        s = &r;
        s->get_data(8,9);
        r.display_area();
}
```

2. /*Extend the above program to display the area of circles. This requires addition of a new derived class circle that computes the area of a circle. Remember, for a circle we need only one value, its radius, but the get_data( ) function in the base class requires two values to be passed. (Hint: Make the second argument of get_data( ) as default one with zero value.)*/

```
#include<iostream>
#define PI 3.14
using namespace std;
class shape
{
        protected:
                double x , y;
        public:
                void get_data(int a , int b=0)
                {
                        x=a;
                        y=b;
                }
                virtual void display_area()=0;
};

class triangle: public shape
{
        public:
                void display_area()
                {
                    cout<<"The area of triangle is "<<(x*y/2);
                }
};

class rectangle: public shape
{
        public:
                void display_area()
                {
                        cout<<"The area of rectangle is "<<(x*y);
                }
};

class circle: public shape
{
        public:
                void display_area()
                {
                        cout<<"The area of circle is "<<(PI*x*x);
                }
};

int main()
{
        triangle t;
        shape *s = &t;
        s->get_data(5,6);
        t.display_area();

        cout<<endl<<endl;

        rectangle r;
        s = &r;
        s->get_data(8,9);
        r.display_area();

        cout<<endl<<endl;

        circle c;
        s = &c;
        s->get_data(7);
        c.display_area();
}
```

3. /*Create an abstract base class place with data members name, address and mobile and member function getdata( ) and display as pure virtual function. Based on this class, derive a new Classes KTM and MNR to inherit all features .*/

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

class place
```

```
{
        protected:
                char name[20], address[20], mobile[10];
        public:
                void getdata()
                {
                        cout<<"Enter the name : ";
                        cin>>name;
                        cout<<"Enter the address : ";
                        cin>>address;
                        cout<<"Enter the mobile number : ";
                        cin>>mobile;
                }
                virtual void display()=0;
};

class ktm: public place
{
        public:
                void display()
                {
                        cout<<"Kathmandu"<<endl;
                        cout<<"Name : "<<name<<endl;
                        cout<<"Address : "<<address<<endl;
                        cout<<"Mobile Number : "<<mobile<<endl;
                }
```

```
};

class mnr: public place
{
	public:
		void display()
		{
			cout<<"Mahendranagar"<<endl;
			cout<<"Name : "<<name<<endl;
			cout<<"Address : "<<address<<endl;
			cout<<"Mobile Number : "<<mobile<<endl;
		}
};

int main()
{
	cout<<"For Kathmandu"<<endl;
	ktm k;
	place *p = &k;
	p->getdata();

	cout<<endl;

	cout<<"For Mahendranagar"<<endl;
	mnr m;
	p = &m;
```

```
        p->getdata();

        cout<<endl<<endl;
        k.display();
        cout<<endl;
        m.display();

}
```

4. /*Write a program to swap the private data members of two different classes using friend function.*/

```
#include<iostream>
using namespace std;
class B;
class A
{
 int a,b;
 friend void swap(A&,B&);
 public:
 A()
 {
  a=1;
  b=2;
 }
 void display()
 {
      cout<<"a = "<<a<<endl;
  cout<<"b = "<<b<<endl;
 }
};
class B
{
  int c,d;
  friend void swap(A&,B&);
  public:
  B()
      {
      c=3;
    d=4;
  }
  void display(){
  cout<<"c = "<<c<<endl;
  cout<<"d = "<<d<<endl;
 }
};
int main()
{
  A A1;
  B B1;
  cout<<"----------BEFORE SWAP-----------"<<endl;
  A1.display();
  B1.display();
  swap(A1,B1);
  cout<<"----------AFTER SWAP-----------"<<endl;
  A1.display();
  B1.display();
  return 0;
}
void swap(A &obj1,B &obj2)
{
  int temp[2];
  temp[0]=obj1.a;
  temp[1]=obj1.b;
  obj1.a=obj2.c;
  obj1.b=obj2.d;
  obj2.c=temp[0];
  obj2.d=temp[1];
}
```

5. /*Write a program to find the average of the private data members of two different classes using friend function.*/

```
#include<iostream>
using namespace std;
class beta;
class alpha
{
        private:
                float data;
```

```
    public:
        void setdata(int d)
        {
            data=d;
        }
        friend float avg(alpha , beta);
};
class beta
{
    private:
        float data;
    public:
        void setdata(int d)
        {
            data = d;
        }
        friend float avg(alpha , beta);
};

float avg(alpha a , beta b)
{
    return (a.data+b.data)/2;
}

int main()
{
	alpha a;
	a.setdata(4);
	beta b;
	b.setdata(5);

	cout<<"Average : "<<avg(a,b);
	return 0;
}
```