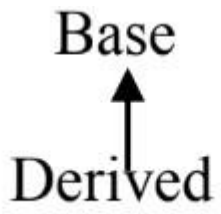
OOPS LAB
ASSIGNMENT – 3

NAME : Swarnendu Banerjee

ROLL : 002311001016

DEPARTMENT : IT – A1

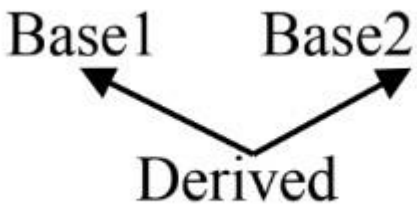
Q28.Write empty class declarations for the following class hierarchy.



CODE:

```
#include<bits/stdc++.h>
using namespace std;
class base{
public :
};
class derived : public base {
public :
};
```

29.Write empty class declarations for the following class hierarchy.



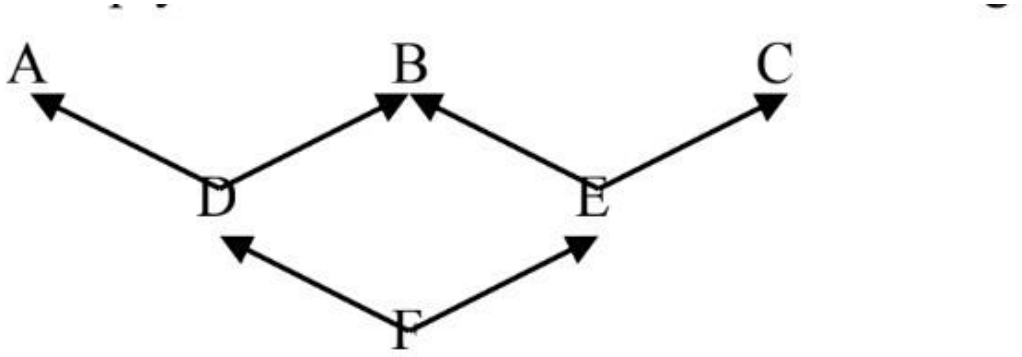
CODE:

```
#include<bits/stdc++.h>
using namespace std;
class base1 {};
```

class base2{};

class derived : public base1, public base2
{};

30.Write empty class declarations for the following class hierarchy.



CODE:

#include<iostream> using
namespace std;

class A
{

};

class B
{

```
};

class C {

};

class D : public A,public B
{

};

class E : public B,public C
{

};

class F : public D,public E
{

}; int
main() {
return 0;
 }
```

31.Write a class Person having data member name, age, height etc. Write proper constructors, methods to get/set them and a method printDetails() that prints all information of a person. Now write another class Student from Person and add data members roll, year of admission etc. Write constructors, methods to get/set them and a override printDetails(). Now create a Person and a Student object and call printDetails() function on them to display their information. Now Create an array of pointers to Person and store addresses of two Persons and two Students. Call printDetails() on all elements (a loop may be used). Are you getting output which is supposed to come? Make printDetails() function virtual in the base class and check the result.

CODE:

```
#include<iostream>

using namespace std;



class Person{

string name;

int age,height;

public:

Person(string name,int age,int
height):name(name),age(age),height(height){}


virtual       void set_details(string x,int y,int z){

name=x;

age=y;
```

```
height=z;}

virtual        void printDetails(){
cout<<"name = "<<name<<endl;
cout<<"age = "<<age<<endl;
cout<<"height = "<<height<<" cm"<<endl;}
};
class Student:public Person{
int roll;
int yoa;
public:
Student(string name,int age,int height,int roll,int
yoa):Person(name,age,height),roll(roll),yoa(yoa){}
void set_details(string n,int a,int h,int r,int y){
Person::set_details(n,a,h);
roll=r;
yoa=y;}

void printDetails(){
Person::printDetails();
cout<<"Roll no.:"<<roll<<endl;
cout<<"Year of admission :"<<yoa<<endl;}
};
int main(){
Person p("Swarnendu",19,181);
```

```
Student s("Swarnendu",19,181,16,2023);
p.printDetails();
s.printDetails();
cout<<"creating array of pointers and printing details:\n";
Person *arr[4];
for(int i=0;i<2;i++){
string n;
int a;
int h;
cout<<"Enter name, age and height: ";
cin>>n>>a>>h;
arr[i]=new Person(n,a,h);}
for(int i=2;i<4;i++){
string n;
int h,a,r,y;
cout<<"Enter name, age, height, roll no. and year of admission :";
cin>>n>>a>>h>>r>>y;
arr[i]=new Student(n,a,h,r,y);}
for(int i=0;i<4;i++)arr[i]->printDetails();
return 0;}
}
```

32.Write a class Employee having data member name, salary etc. Write proper constructors, methods to get/set them and a virtual method printDetails() that prints all information of a person. Now write two classes Manager and Clerk from Employee. Add ‘type’ and ‘allowance’ in the manager and Clerk respectively. Write constructors, methods to get/set them and a override printDetails(). Now create a Manager and a Clerk object and call printDetails() function on them to display their information. Now Create an array of pointers to Employee and store addresses of two Employee, two Managers and two Clerks. Call printDetails() on all elements (a loop may be used). Also find the total salary drawn by all employees.

Code:

```
#include<bits/stdc++.h>
using namespace std;
class Employee
{
	string name ;
	double salary;
	public :
	void setName (string name )
	{
		this->name = name ;
	}
	void setSalary ( double salary)
	{
		this->salary = salary ;
	}
```

```
        Employee(string name, double salary)
        {
                setName(name);
                setSalary(salary);
        }

        virtual void printDetails()
        {
                cout<<" Name of employee : "<<name<<endl;
                cout<<" Salary : " <<salary<<endl;
        }
};


class Manager : public Employee
{
        int type ; double allowance ;
        public:
        Manager(int t , double a) : Employee("",0.0)
        {
                this->type = t;
                this->allowance = a;
        }

        void printDetails(){
```

```
        Employee::printDetails();
        cout<<"TYPE: "<<endl;
        cout<<"Allowance: "<<endl;}


};
    class Clerk : public Employee
    {

        int type ; double allowance ;
        public:
        Clerk(int t, double a):Employee("",0.0)
        {
            this->type = t;
            this->allowance = a;
        }
        void printDetails()
        {
            Employee::printDetails();
            cout<<" TYPE : "<<type<<endl;
            cout<<" Allowance : "<<allowance<<endl;

        }

    };
```

```
int main()
{

    Employee* e[2];

    for(int i=0;i<2;i++)
    {
        cout<<" enter name salary type allowance for Clerk "<<i+1<<endl;
        string name; double sal,all;int t;
        cin>>name>>sal>>t>>all;

        e[i] = new Clerk(t,all);
        e[i]->setName(name);
        e[i]->setSalary(sal);
        e[i]->printDetails();
      delete e[i];}
        for(int i =0 ;i<2;i++)
        {
            cout<<"enter name salary type allowance for Manager "<<i+1<<endl;
            string name ; double sal,all;int t;
```

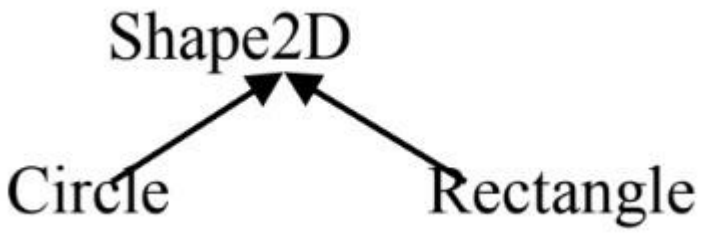
```
            cin>>name>>sal>>t>>all;

            e[i]=new Manager(t,all);
            e[i]->setName(name);
            e[i]->setSalary(sal);

            e[i]->printDetails();
        }


        return 0;
    }
```

33.Write class definitions for the following class hierarchy



The Shape2D class represents two dimensional shapes that should have pure virtual functions area(), perimeter() etc. Implement these functions in Circle and Rectangle. Also write proper constructor(s) and other functions you think appropriate in the Circle and Rectangle class. Now create an array of 5 Shape2D pointers. Create 3 Circle and 2 Rectangles objects and place their addresses in that array. Use a loop to print area and perimeter of all shapes on this array.

Code:

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

class Shape2D{
public:
 virtual void  area()=0;
 virtual void  perimeter()=0;
};

class Circle:public Shape2D{
        float r;
        public:
        Circle(float r){
                this->r=r;}
        void area(){
                cout<<"area of circle is :"<<3.14*r*r<<endl;}
        void perimeter(){
                cout<<"perimeter of circle is "<<2*3.14*r<<endl;}
};
class Rectangle:public Shape2D{
        float l,b;
        public:
        Rectangle(float l,float b){
                this->l=l;
```

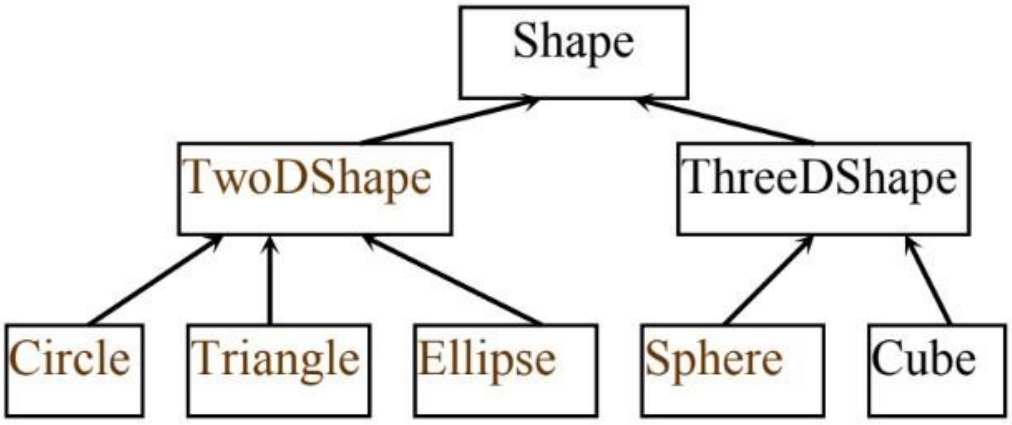
```
        this->b=b;
    }
    void area(){
        cout<<"area of rectangle  is :"<<l*b<<endl;}
    void perimeter(){
        cout<<"perimeter of circle is: "<<2.0*(l+b)<<endl;}
};

int main(){

    Shape2D* ptr[5];
    for(int i=0;i<5;i++){
        if(i<3){
            int r;
            cout<<"enter the radius of circle : ";
            cin>>r;
            ptr[i]=new Circle(r);
        }
        else{
            int l,b;
            cout<<"enter the length and breadth of rectangle : ";
            cin>>l>>b;
            ptr[i]=new Rectangle(l,b);
        }}
        for(int i=0;i<5;i++){
```

```
            ptr[i]->perimeter();
            ptr[i]->area();}
        return 0;}
```

34.Implement the Shape hierarchy as shown in the figure. Each TwoDShape should contain function getArea to calculate the area of two-dimensional shape. Each ThreeDShape should have member functions getArea and getVolume to calculate the surface area and volume of the three-dimensional shape respectively. Create a program that uses Vector of Shape pointers to objects of each concrete class in the hierarchy. Now write a program that processes all the shapes in the Vector such that if the shape is a TwoDShape it prints name of shape and its area while it prints name of shape, its area and volume if the shape is a ThreeDShape.



Code:

```
#include<bits/stdc++.h>
using namespace std;
class shape{
    public:
        virtual void area()=0;
```

```
        virtual void volume()=0;
};
class twodshape:public shape{
    public:
        void area(){
            cout<<"2D";
        }
};
class threedshape:public shape{
    public:
        void area(){
            cout<<"3D";}
        };
        class circle:public twodshape{
            int r;
            public:
            circle(float r){
                this->r=r;
            }
            void area(){
                cout<<"It is a circle with area
:"<<3.14*r*r<<endl;}
            void volume(){
                return; }
        };
```

```
class triangle:public twodshape{
        float l,b;
        public:
        triangle(float l,float b){
                this->l=l;
                this->b=b;

        }

        void area(){
                cout<<"It is a triangle with area
:"<<0.5*l*b<<endl;}
        void volume(){return ;}
};
class ellipse:public twodshape{
        float l,b;
        public:
        ellipse(float l,float b){
                this->l=l;
                this->b=b;
        }

        void area(){
                cout<<"It is an ellipse with area
:"<<3.14*l*b<<endl;}
```

```
        void volume(){return ;}
    };
    class sphere:public threedshape{
        float r;
        public:
        sphere(float r){
            this->r=r;
        }

        void area(){
            cout<<"It is a sphere  with surface area
:"<<4*3.14*r*r<<endl;}
        void volume(){
            cout<<" its volume is
:"<<4/3*3.14*r*r*r<<endl;}
    };
    class cube:public threedshape{
        float a;
        public:
        cube(float a){
            this->a=a;
        }

        void area(){
            cout<<"It is a cube  with surface area
:"<<6*a*a<<endl;}
```

```
        void volume(){
            cout<<" its volume is :"<<a*a*a<<endl;}
    };

    int main(){


        vector<shape*>ptr;
        int i=0;
        while(1){
            cout<<" choose 1.circle 2.triangle 3.ellipse
4.cube 5.sphere 6.exit "<<endl;
            int  t;
            cin>>t;
            if(t==1){
                cout<<"enter the radius of circle :";
                float r;
                cin>>r;
                ptr.push_back(new circle(r));
                ptr[i]->area();
            }
            if(t==2){
                cout<<"enter the length and breadth of
rectangle :";
                float l,b;
```

```
                cin>>l>>b;
                ptr.push_back(new triangle(l,b));
                ptr[i]->area();
            }
            if(t==3){
                cout<<"enter the major  and minor axis's
lengths :";
                float b,h;
                cin>>b>>h;
                ptr.push_back(new ellipse(b,h));
                ptr[i]->area();
            }
            if(t==4){
                cout<<"enter the side of cube :";
                int r;
                cin>>r;
                ptr.push_back(new cube(r));
                ptr[i]->area();
                ptr[i]->volume();
            }
            if(t==5){
                cout<<"enter the radius of sphere:";
                int r;
                cin>>r;
                ptr.push_back(new sphere(r));
```

```
            ptr[i]->area();
            ptr[i]->volume();
        }
        if(t==6)break;
        i++;}
    return 0;}
```

35.Write a program to illustrate the role of virtual destructor.

Code:

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

class B{
public:
B(){
cout<<"B const";
}
virtual ~B(){
cout<<"B dest ";}};

class c:public B{
public:
c(){
cout<<"c const";}

~c(){
cout<<" c dest";}
};
int main(){
```

```
B* ptr=new c();
delete(ptr);}
```