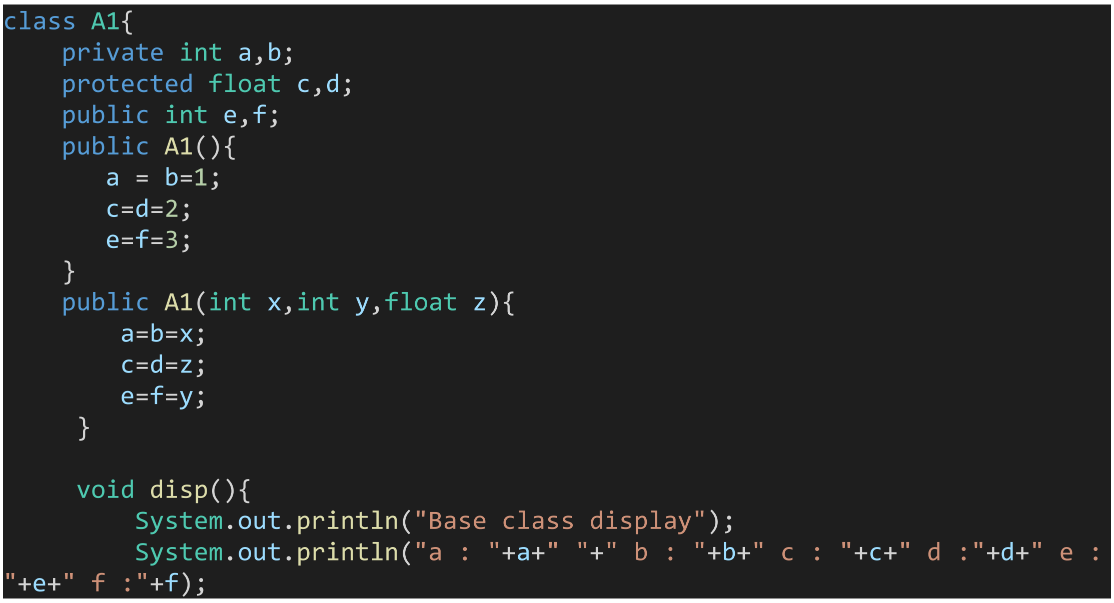
Lab-4

2a. Identify the type of inheritance in the given diagram. Create a class A with two integer member
variables that are private, two float variables that are protected and two integer variables that are public.
Let class B inherit class A and class C and Class D are inherited from class B. Write appropriate
methods to illustrate the following
i) Usage of super keyword
ii) Function overriding
iii) Default constructors
iv) Parameterized constructors
v) How to we make a method not to be over ridden and a class not be inherited further

Code:



```
class A1{
    private int a,b;
    protected float c,d;
    public int e,f;
    public A1(){
       a = b=1;
       c=d=2;
       e=f=3;
    }
    public A1(int x,int y,float z){
        a=b=x;
        c=d=z;
        e=f=y;
     }

     void disp(){
         System.out.println("Base class display");
         System.out.println("a : "+a+" "+" b : "+b+" c : "+c+" d :"+d+" e :
"+e+" f :"+f);
```

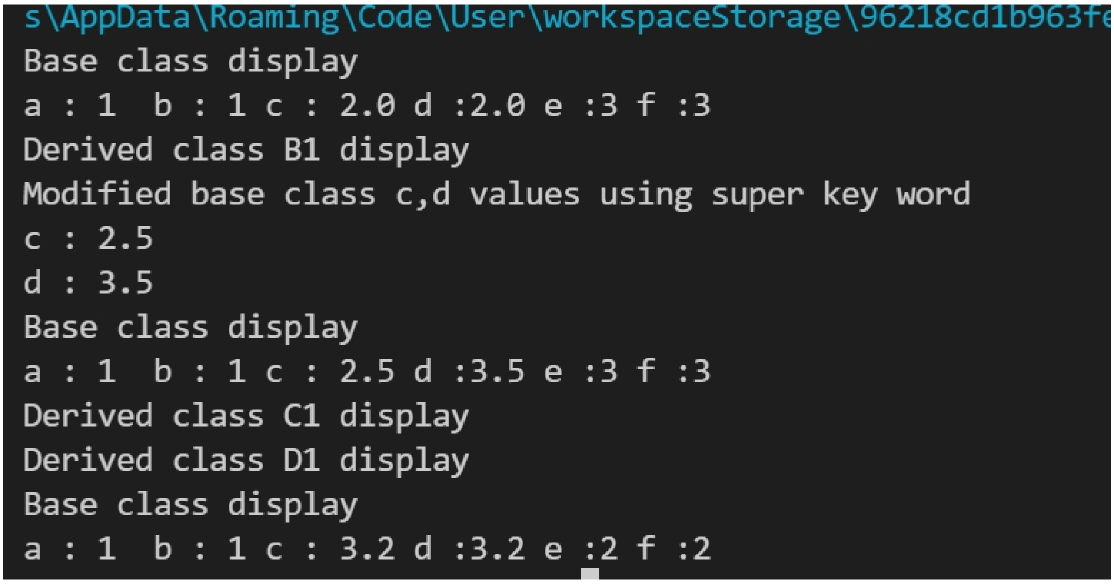
```
     }
```

```
}

class B1 extends A1{
    int x,y,z;
    public B1(){
       super();
    //    System.out.println("Deriver class");
    }

    void disp(){
        super.c = 2.5f;
        super.d = 3.5f;
        System.out.println("Derived class B1 display");
        System.out.println("Modified base class c,d values using super keyword \nc : "+super.c+"\nd : "+super.d);

        super.disp();

    }

}
final class C1 extends B1
{
  final void disp(){
    System.out.println("Derived class C1 display");
    // super.disp();

}
}
final class D1 extends B1
{
  final void disp(){
    System.out.println("Derived class D1 display");
    // super.disp();

}
}
```

```
public class inherit {
    public static void main(String[] args) {
        A1 a = new A1();
        B1 b = new B1();
        C1 c = new C1();
        D1 d = new D1();

        a.disp();
        b.disp();
        c.disp();
        d.disp();

        A1 a1 = new A1(1 ,2,3.2f);

        a1.disp();
```

```
    }
}
```

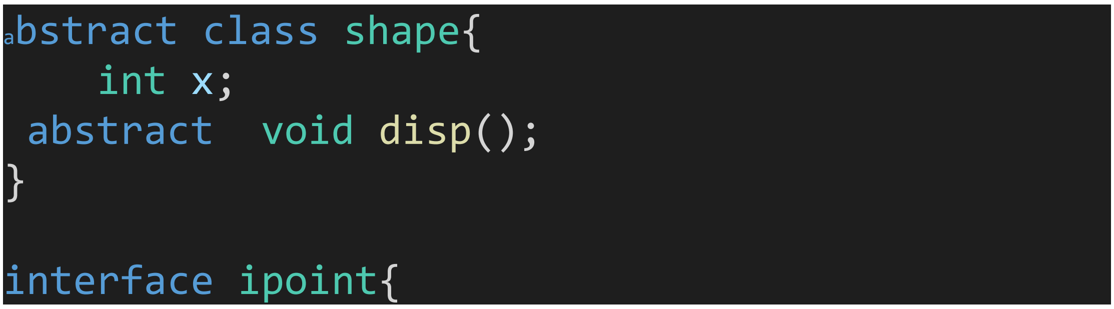
Output:



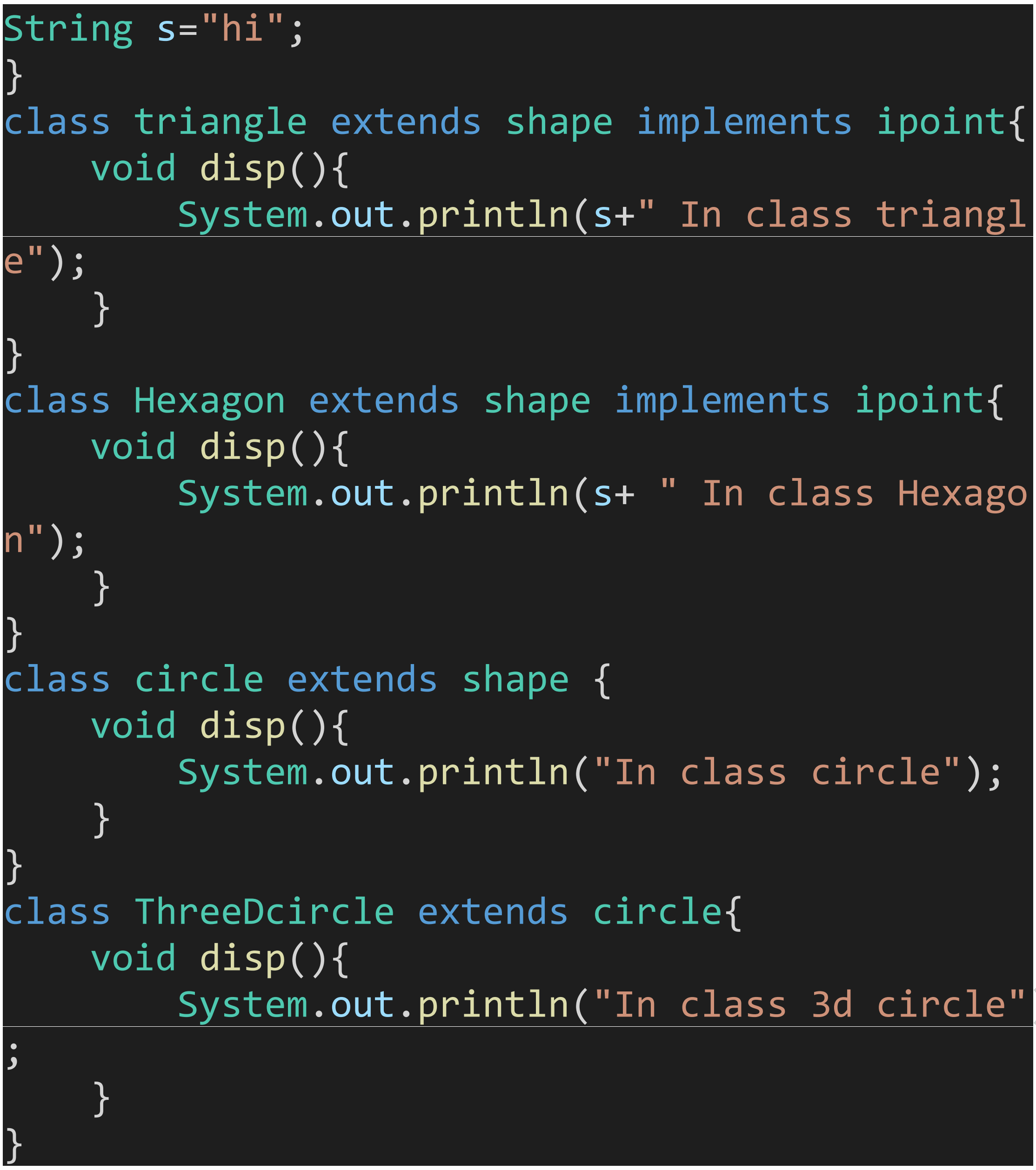
```
s\AppData\Roaming\Code\User\workspaceStorage\96218cd1b963f
Base class display
a : 1  b : 1 c : 2.0 d :2.0 e :3 f :3
Derived class B1 display
Modified base class c,d values using super key word
c : 2.5
d : 3.5
Base class display
a : 1  b : 1 c : 2.5 d :3.5 e :3 f :3
Derived class C1 display
Derived class D1 display
Base class display
a : 1  b : 1 c : 3.2 d :3.2 e :2 f :2
```

2c. Design the given model in java

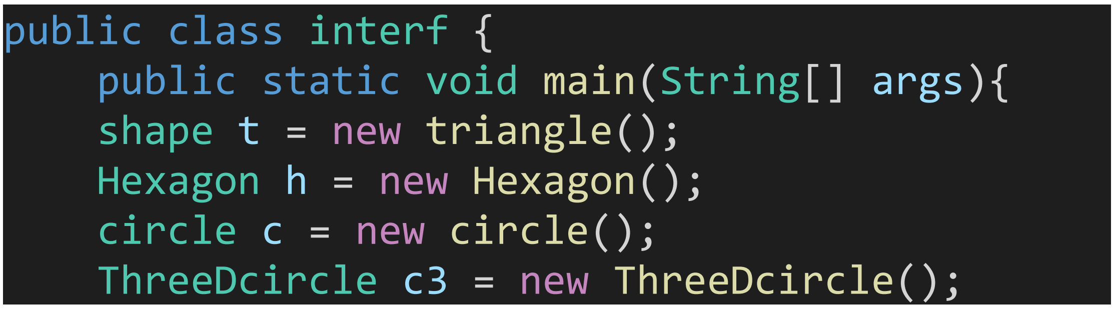
Code:



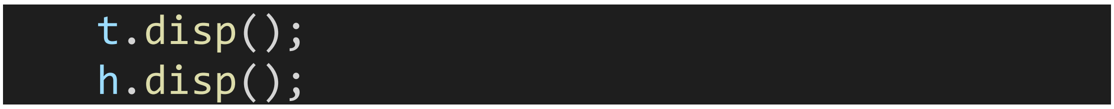
```
abstract class shape{
    int x;
 abstract  void disp();
}

interface ipoint{
```



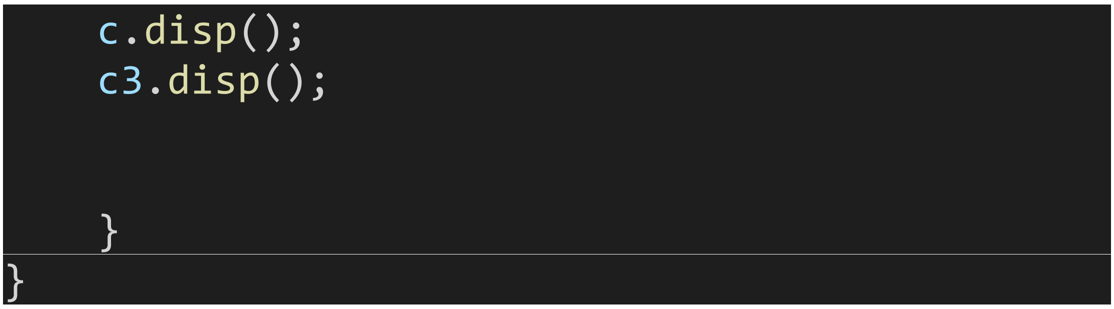
```java
String s="hi";
}
class triangle extends shape implements ipoint{
    void disp(){
        System.out.println(s+" In class triangle");
    }
}
class Hexagon extends shape implements ipoint{
    void disp(){
        System.out.println(s+ " In class Hexagon");
    }
}
class circle extends shape {
    void disp(){
        System.out.println("In class circle");
    }
}
class ThreeDcircle extends circle{
    void disp(){
        System.out.println("In class 3d circle");
    }
}
```



```java
public class interf {
    public static void main(String[] args){
    shape t = new triangle();
    Hexagon h = new Hexagon();
    circle c = new circle();
    ThreeDcircle c3 = new ThreeDcircle();
```

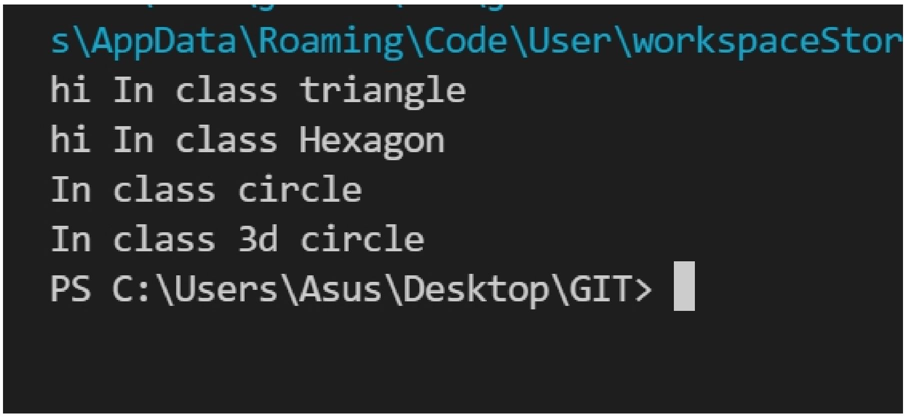


```java
    t.disp();
    h.disp();
```



```
    c.disp();
    c3.disp();


    }
}
```

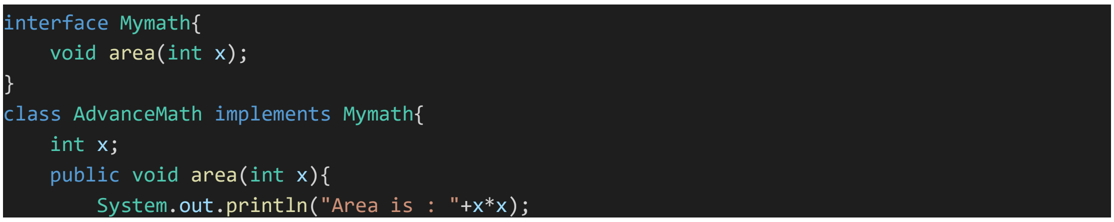
Output:



```
s\AppData\Roaming\Code\User\workspaceStor
hi In class triangle
hi In class Hexagon
In class circle
In class 3d circle
PS C:\Users\Asus\Desktop\GIT>
```

Write an interface, called MyMath, for your own Math Library functions.
Create a new class called AdvancedMath that inherits standard Math class and implement MyMath interface.

Code:

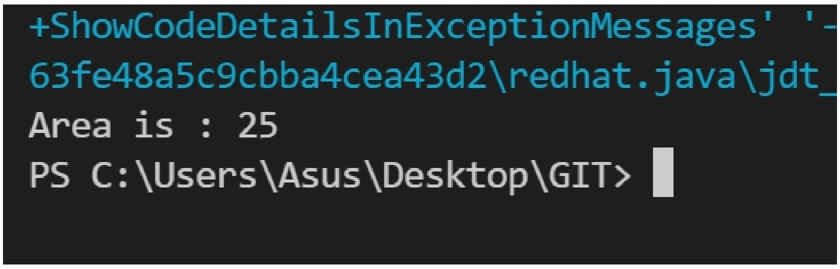


```
interface Mymath{
    void area(int x);
}
class AdvanceMath implements Mymath{
    int x;
    public void area(int x){
        System.out.println("Area is : "+x*x);
    }

}
```

```
public class mathMain {
    public static void main(String[] args){
    AdvanceMath a = new AdvanceMath();
    a.area(5);
```

```
    }
}
```

Output:



```
+ShowCodeDetailsInExceptionMessages' '-
63fe48a5c9cbba4cea43d2\redhat.java\jdt_
Area is : 25
PS C:\Users\Asus\Desktop\GIT>
```