Overriding

Overriding:- **It has same name with same argument. It is not possible one class. It is possible only in inheritance.**

Example:-

```
class Base
{
public void show()
{
System.out.print("this is show() in Base class\n");
}
}
class Derive extends Base
{
public void show()
{
super.show(); //used to invoke base class
                methods from derive class overriding
System.out.print("this is show() in derived class");
}
}
class aman1
{
```

```
public static void main(String agr[])
{
Derive d=new Derive();
d.show();
}
}
```

Output:-

```
this is show() in Base class
this is show() in derived class
```

Abstract class and abstract method:- It is fellow with help of abstract keyword.

Example:-

```
abstract class Base              //abstract class
{
public abstract void show();        //abstract method
public void show1()
{
System.out.print("this is show1() in Base class\n");
```

```
}
}
class Derive extends Base
{
public void show()
{
System.out.print("this is show() in derived class\n");
}
public void show2()
{
System.out.print("this is show2() in derived class\n");
}
}
class aman1
{
public static void main(String agr[])
{
Derive d=new Derive();
d.show();
d.show1();
```

```
d.show2();
}
}
```

Output:-

```
this is show in derived class
this is show1() in base class
this is show2() in derived class
```

Abstract keyword:- **It is use to create abstract class as well as abstract method.**

Abstract class:- **It is follow with the help of abstract keyword. It contains abstract method as well as normal method. Abstract class can not initialize their object but we can create reference variable.**

Abstract method:- **It is no body structure. It is work as do nothing methods. It must be redefine derive class.**

Interface:- **it is same as abstract class but it contains abstract method only it is inharit into derive class using implements keyword.**

Example:-

```
interface Base
{
public abstract void show();
}
class derive implements Base
{
public void show()
{
System.out.print("this is show() in derive class");
}
}
class mainclass
{
public static void main(String arg[])
{
derive d=new derive();
d.show();
}
}
```

Output:-

```
this is show() in derive class
```

Partial multiple inheritance:-

Example:-

```
interface Base1
{
public abstract void show();
}
class Base2
{
public void show()
{
        System.out.print("This is show() in Base2 class");
}
}
class derive extends Base2 implements Base1
{
public void show()
{ super.show();
System.out.print("\nThis is show() in derive class");
```

```
}
}
class mainclass
{
public static void main(String arg[])
{
derive d=new derive();
d.show();
}
}
```

Solution:-

```
This is show() in Base2 class
This is show() in derive class
```

Created by Ajay Kumar Verma