****** What is Method Overriding ?**

Some imporatnt points :

1) MOL is possible in the same class as well as super & sub class
2) MOR is only possible with Inheritance [Without inheritance Overriding is not possible]
3) Method Signature [Method Name along with method parameter]
4) In Overriding we will verify
   a) Method Signature must be same
   b) return type must be compaitable
5) Method Overriding is only possible with **non static method**
6) Method Hiding is only possible with **static method**

Defination of Method Overriding :

* Re-defining the super class non static method in the sub class in such a way that method signature must be same and return type must be compaitable is called Method Overriding.

* Method Overriding is not possible without inheritance.

Generally we can't change the return type of the method while overriding a method (compatibility issue) but from JDK 1.5v there is a concept called Co-variant (In same direction) through which we can change the return type of the method.

Example :

```
class Super
{
    public void m1()
    {
    }
}
class Sub extends Super
{
    public void m1() //Overridden Method
    {
      //Here Redefining the super class method body
    }
}
```

Why we will override a method:

* If we want to modify the super class method implementation, wants to provide our own implementation then we should override a method.

Advantage of Method Overriding :

The advantage of Method Overriding is, each sub class is specifying its own specific behavior.

Example :

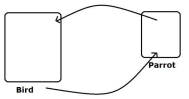
```
class Bird
{
    public void fly()
    {
        IO.println("Generic Bird is flying");
    }
}
class Parrot extends Bird
{
    public void fly()
    {
        IO.println("Parrot Bird is flying");
    }
}
class Sparrow extends Bird
{
    public void fly()
    {
        IO.println("Sparrow Bird is flying");
    }
}
```

Parrot & Sparrow both are specifying own specific behavior

What is Upcasting & Down Casting :

Upcasting :

* Assigning **sub class object to super class reference variable is called upcasting.**



* Parrot Object we can assign to Bird
* Bird Object we cannot assign to Parrot

All parrots are bird. //Valid
All birds are parrot. //Invalid

Bird bird = new Parrot(); //Upcasting

Downcasting :

* In order to call specific method we should use down-casting.

* Down casting is not possible without up-casting.

* It is a technique to assign to **sub class object to sub class reference** by using super class reference variable.

[Dog dog = (Dog) new Animal(); Never say It is down casting, It is not possible in Java]

Example :

```
Animal animal = new Dog();
animal.sleep();

//To call specific method we should use down-casting

Dog dog = (Dog) animal;  //Downcasting
dog.bark();
```

Rule for calling static & non static method by using upcasting and IS-A (normal Inheritance):

Different Cases	Non static method	static Method
Animal a = new Animal(); a.sleep();	sleep() is NSM, Here compiler will search the method in the Animal class & JVM will also execute from Animal class.	sleep() is SM, Here compiler will search the method in the Animal class & JVM will also execute from Animal class.
Dog d = new Dog(); d.bark();	bark() is NSM, Here compiler will search the method in the Dog class & JVM will also execute from Dog class.	bark() is SM, Here compiler will search the method in the Dog class & JVM will also execute from Dog class.
Animal a = new Dog(); a.sleep();	sleep() is NSM, Here compiler will search the method in the Animal class & JVM will **start** executing from Dog class.	sleep() is SM, Here compiler will search the method in the Animal class & JVM will **also** execute from Animal class. **{Static methods are not overridden It is Method Hiding}**