Method Overriding in Java

1. Understanding problem without method overriding
2. Can we override the static method
3. method overloading vs method overriding

If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding in java**.

In other words, If subclass provides the specific implementation of the method that has been provided by one of its parent class, it is known as method overriding.

Usage of Java Method Overriding

- Method overriding is used to provide specific implementation of a method that is already provided by its super class.
- Method overriding is used for runtime polymorphism

Rules for Java Method Overriding

1. method must have same name as in the parent class
2. method must have same parameter as in the parent class.
3. must be IS-A relationship (inheritance).

Understanding the problem without method overriding

Let's understand the problem that we may face in the program if we don't use method overriding.

```
1. class Vehicle{
2.   void run(){System.out.println("Vehicle is running");}
3. }
4. class Bike extends Vehicle{
5.
6.   public static void main(String args[]){
7.   Bike obj = new Bike();
8.   obj.run();
9.   }
10.     }
```

Problem is that I have to provide a specific implementation of run() method in subclass that is why we use method overriding.

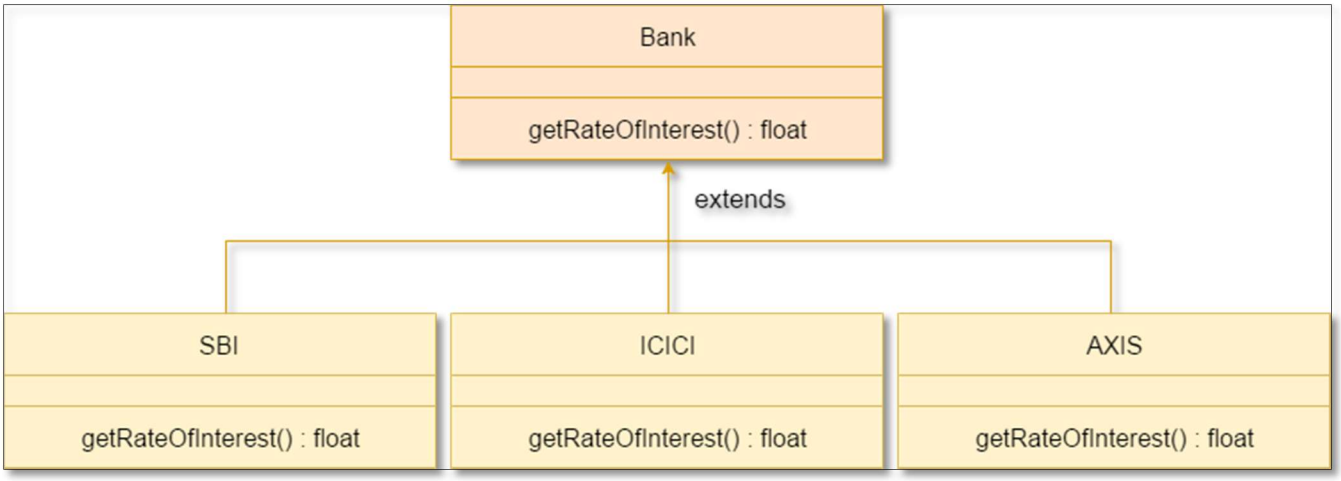
Example of method overriding

In this example, we have defined the run method in the subclass as defined in the parent class but it has some specific implementation. The name and parameter of the method is same and there is IS-A relationship between the classes, so there is method overriding.

```
1. class Vehicle{
2. void run(){System.out.println("Vehicle is running");}
3. }
4. class Bike2 extends Vehicle{
5. void run(){System.out.println("Bike is running safely");}
6.
7. public static void main(String args[]){
8. Bike2 obj = new Bike2();
9. obj.run();
10.     }
```

Real example of Java Method Overriding

Consider a scenario, Bank is a class that provides functionality to get rate of interest. But, rate of interest varies according to banks. For example, SBI, ICICI and AXIS banks could provide 8%, 7% and 9% rate of interest.




```
1. class Bank{
2. int getRateOfInterest(){return 0;}
3. }
4.
5. class SBI extends Bank{
6. int getRateOfInterest(){return 8;}
7. }
8.
9. class ICICI extends Bank{
10.     int getRateOfInterest(){return 7;}
11.     }
12.     class AXIS extends Bank{
13.     int getRateOfInterest(){return 9;}
14.     }
15.
16.     class Test2{
17.     public static void main(String args[]){
```

```
18.     SBI s=new SBI();
19.     ICICI i=new ICICI();
20.     AXIS a=new AXIS();
21.     System.out.println("SBI Rate of Interest: "+s.getRateOfInterest());
22.     System.out.println("ICICI Rate of Interest: "+i.getRateOfInterest());
23.     System.out.println("AXIS Rate of Interest: "+a.getRateOfInterest());
24.     }
25.     }
```

Can we override static method?

No, static method cannot be overridden. It can be proved by runtime polymorphism, so we will learn it later.

Why we cannot override static method?

Because static method is bound with class whereas instance method is bound with object. Static belongs to class area and instance belongs to heap area.

Can we override java main method?

No, because main is a static method.

Difference between method Overloading and Method Overriding in java

There are many differences between method overloading and method overriding in java. A list of differences between method overloading and method overriding are given below:

No.	Method Overloading	Method Overriding
1)	Method overloading is used *to increase the readability* of the program.	Method overriding is used *to provide the specific implementation* of the method that is already provided by its super class.
2)	Method overloading is performed *within class*.	Method overriding occurs *in two classes* that have IS-A (inheritance) relationship.
3)	In case of method overloading, *parameter must be different*.	In case of method overriding, *parameter must be same*.
4)	Method overloading is the example of *compile time polymorphism*.	Method overriding is the example of *run time polymorphism*.

5)	In java, method overloading can't be performed by changing return type of the method only. *Return type can be same or different* in method overloading. But you must have to change the parameter.	*Return type must be same or covariant* in method overriding.

Java Method Overloading example

```
1. class OverloadingExample{
2. static int add(int a,int b){return a+b;}
3. static int add(int a,int b,int c){return a+b+c;}
4. }
```

Java Method Overriding example

```
1. class Animal{
2. void eat(){System.out.println("eating...");}
3. }
4. class Dog extends Animal{
5. void eat(){System.out.println("eating bread...");}
6. }
```