# Method Overriding

1) If the child class not satisfy the parent class method implementation then it is possible to override that method in the child class based on child class requirement.
2) If we want to achieve method overriding we need two class(child and parent).
3) In the overriding concept the child class and parent class method signatures must be same otherwise we are getting compilation error.

4) There must be an IS-A relationship (inheritance).

The parent class method is called------------overridden method

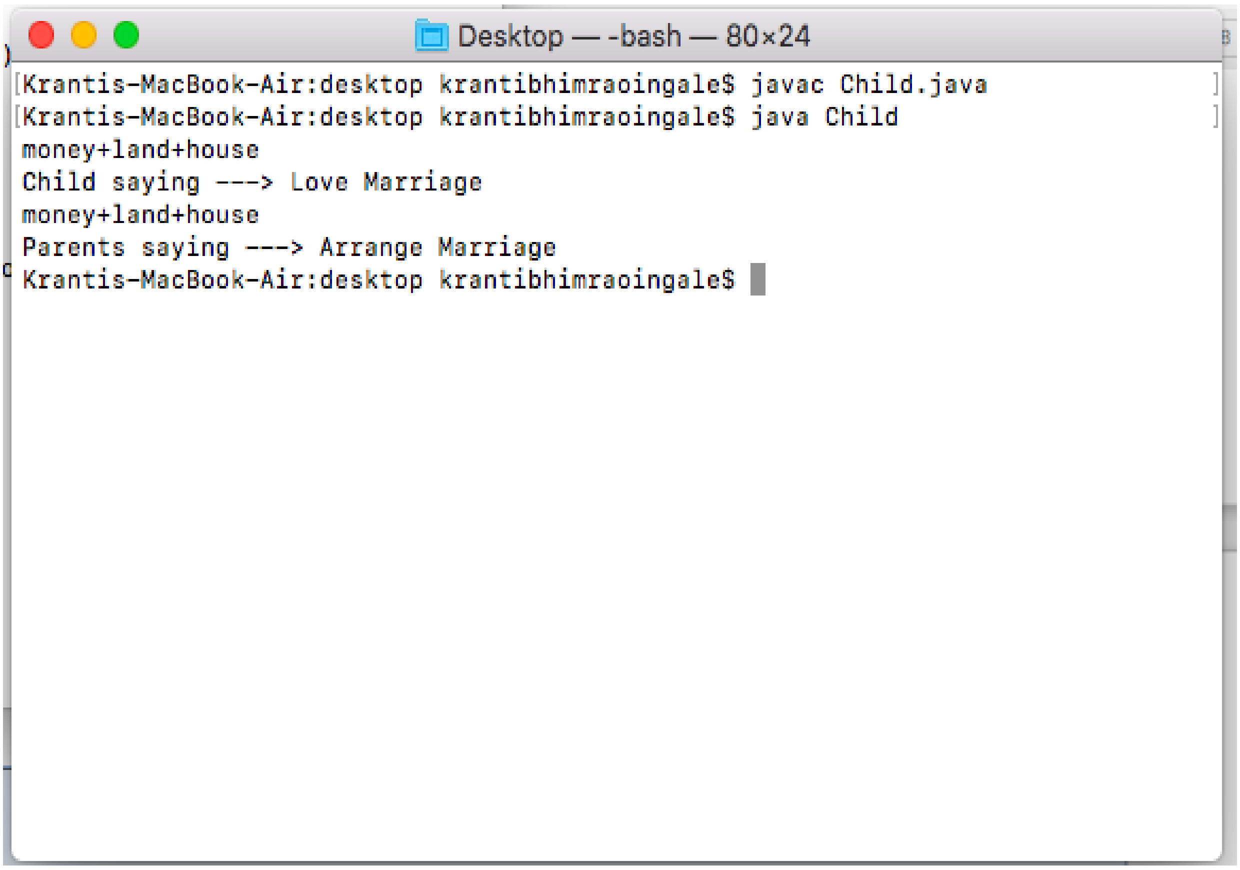
The child class method is called--------------overriding method

Ex:-methodOverriding

```
class Parent
{

void property()
{
System.out.println("money+land+house");
}
void marry()
{
System.out.println("Parents saying ---> Arrange Marriage"); //Overridden method

}

}

class Child extends Parent
{

void marry()
{
System.out.println("Child saying ---> Love Marriage"); //overriding method

}
public static void main(String[] args)
{
Child c=new Child(); // call default constructor of Child , Child called ---> Parent's Default
c.property();               // constructor as they are in inheritance
c.marry();
Parent p=new Parent();
```

```
        p.property();   // This will call parent class method property()  and marry () as p is Parent
        p.marry();     // class object
        }

        }
```

**Output :**



```
Krantis-MacBook-Air:desktop krantibhimraoingale$ javac Child.java
Krantis-MacBook-Air:desktop krantibhimraoingale$ java Child
money+land+house
Child saying ---> Love Marriage
money+land+house
Parents saying ---> Arrange Marriage
Krantis-MacBook-Air:desktop krantibhimraoingale$ 
```

## Method Overloading vs Overriding

| **Method overloading** | **Method overriding** |
|---|---|
| Method overloading is performed **within class.** | Method overriding occurs in two classes that have IS-A (inheritance) relationship. |
| In case of method overloading, parameter must be different. | In case of method overriding, parameter must be same. |
| Method overloading is the example of **compile time polymorphism.** | Method overriding is the example of **run time polymorphism.** |
| 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. |
| 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** in method overriding. |