Hirerachy -> Reusablity

1. Association
    - has-a relationship between 2 entities
2. Inheritance
    - is-a relationship between 2 entities

Association (has-a)
1. Composition
2. Aggegration

Human has-a Heart (Composition)

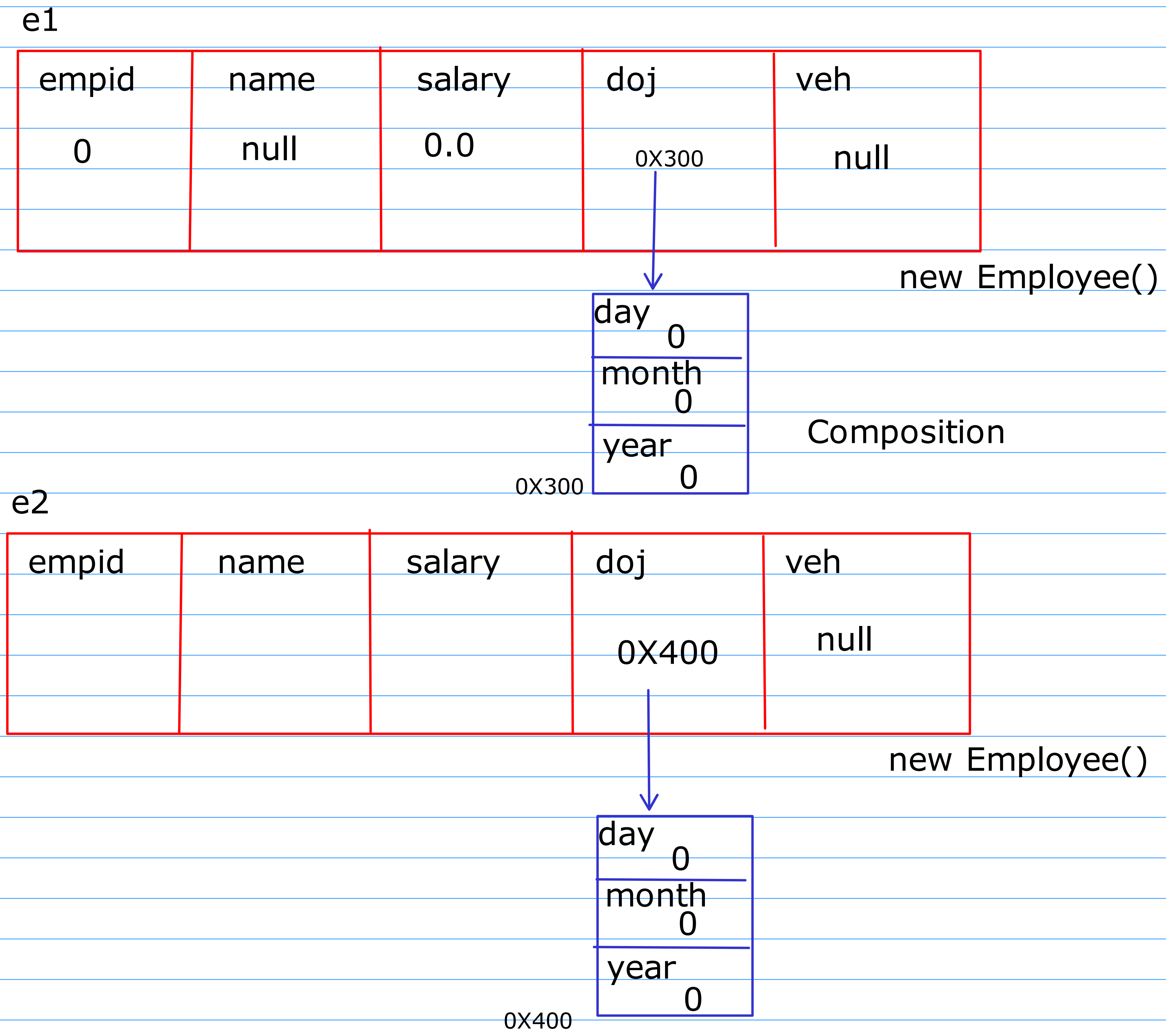
Dependent     Dependency
Car has-a Engine (Composition)

Car has-a  MusicPlayer(Aggegration)

```
Depenedent                 Dependency
Employee{                  Date{
id                         day,month,year
name                       }
salary
Date doj; // Association
}
```

Employee has-a DateofJoining

```
Customer{
cid
name
dob
}
```



is-a relationship (inheritance)

eg-> Apple is-a Fruit
Employee is-a Person
Mobile is-a  Device
Car is-a Vehicle
Bike is-a Vehicle

Child is-a Parent
SubType  is-a SuperType

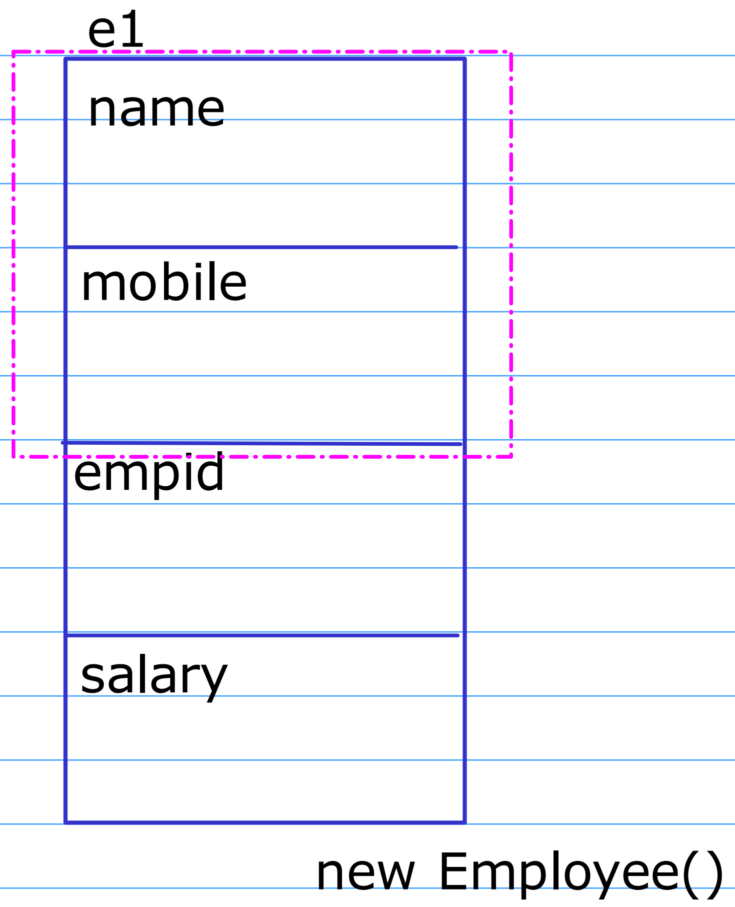
Dependent has-a Dependeny
SubType is-a SuperType

```
class Person{
String name;
String mobile;
}
```

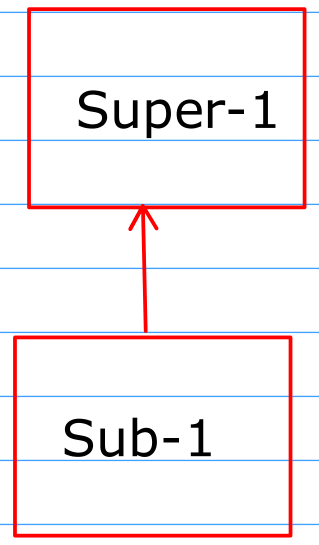
```
class Employee extends Person{
int id;
double salary;

}
```

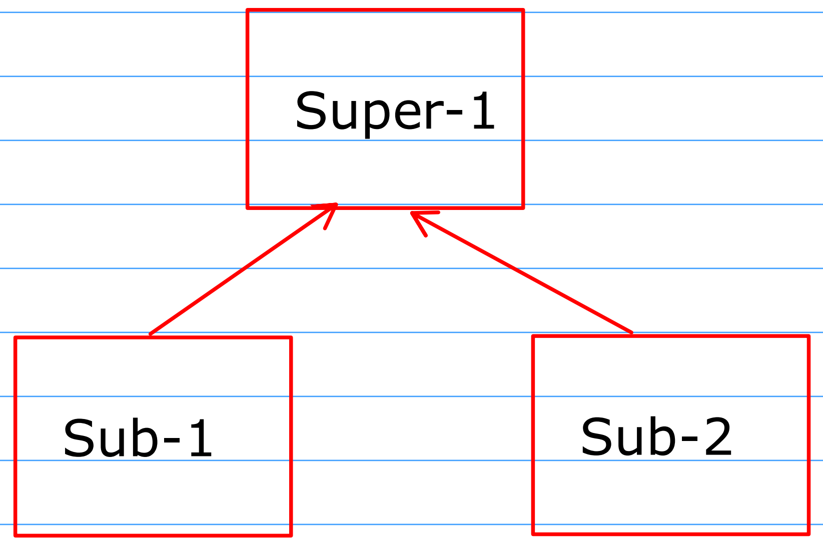
```
class Student{
int rollno;
double marks;
}
```



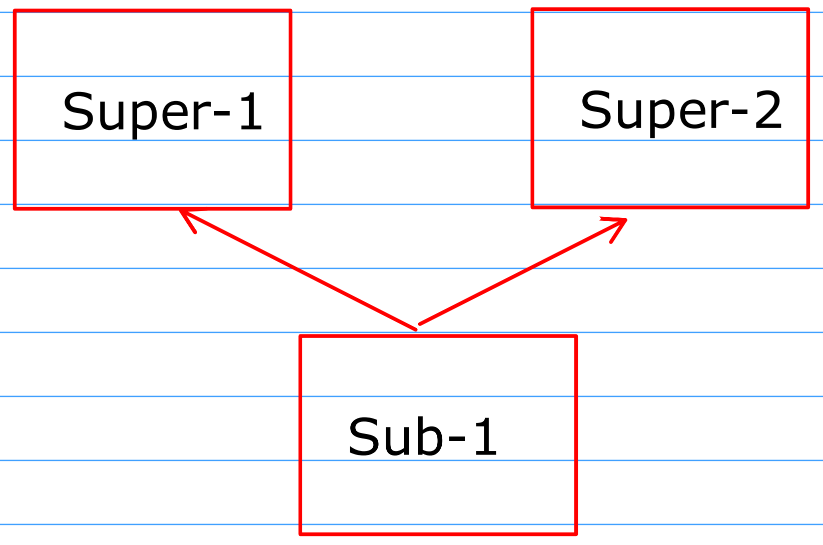
Types of Inheritance



Single Inheritance

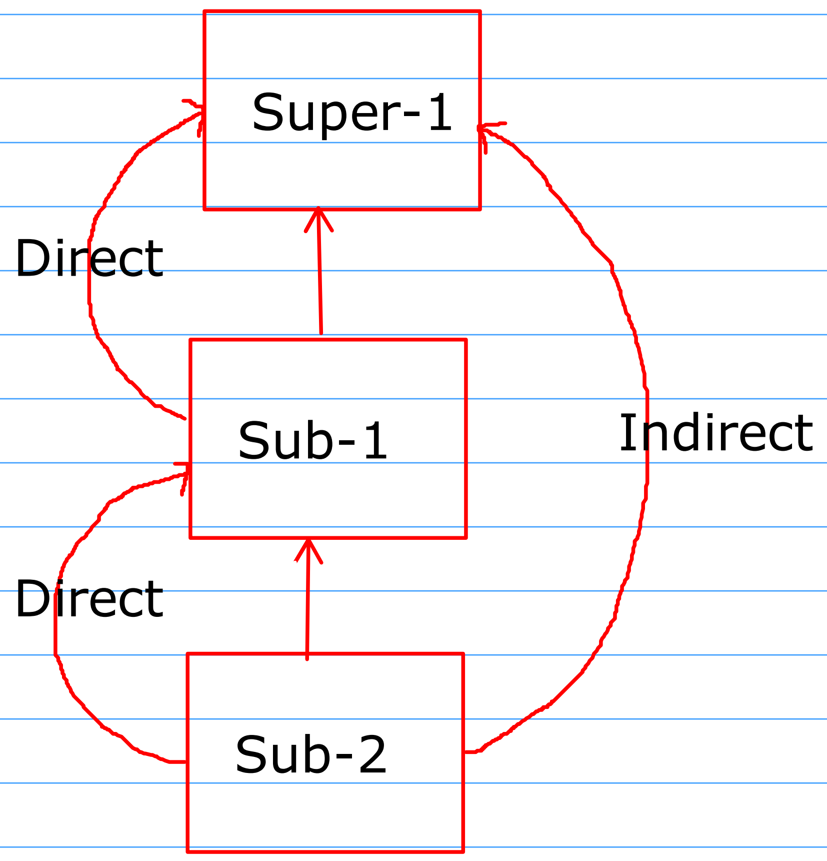


Hirerachical Inheritance

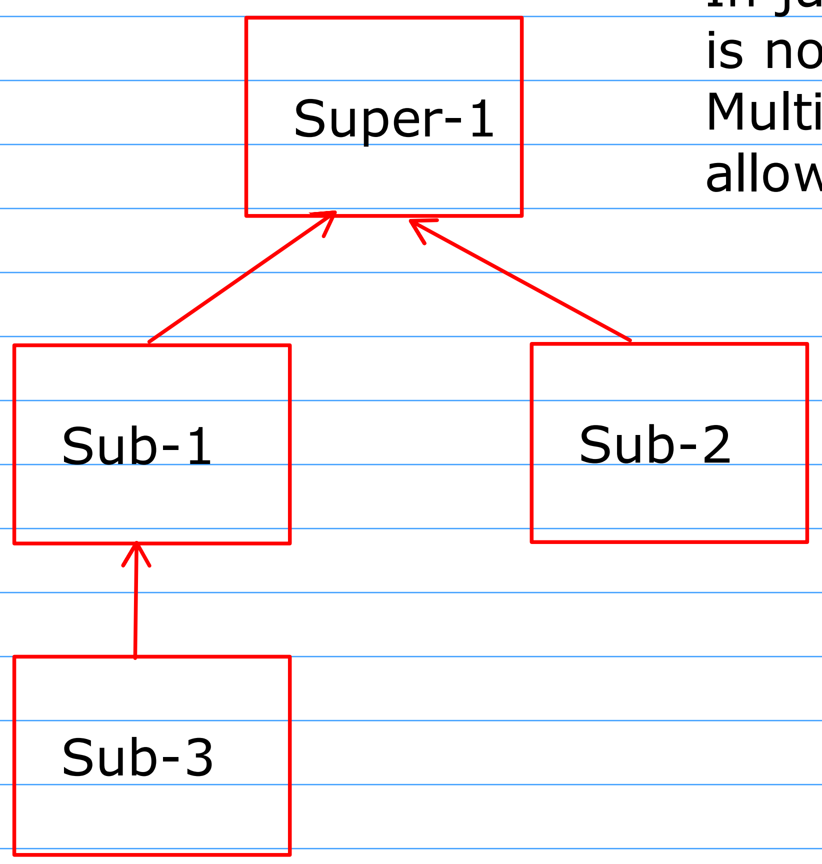


Multiple Inheritance

In java multiple class inheritance
is not allowed
Multiple Interface inheritance is
allowed



Multilevel Inheritance



Hybrid Inheritance

Method Overriding
- Why to perform method overriding
1. If the implementation of the super class method is partial complete
2. If the implementation of the super class method is 100% incomplete
3. If the implementation of the sub class method we want totally different from super class method

Rules of Method Overriding
1. Name and signature of the method of super class and sub class should be same
2. The visibibility modifer should be same or should be wider type
3. The return type of sub class method can be same as that of return type of the super class metho
or ot can be its sub type
4. The execption list in the sub class method should be same as that of super class method or
it should be its sub set.

Polymorphism