OOPS Concept:

1. Class
2. Object
3. Inheritance (IS A Relationship)
4. Abstraction
5. Polymorphism(Method Overloading and Method Overriding)
6. Encapsulation
7. Association (Mapping Concepts, See in Hibernate)
8. Aggregation (HAS A Relationship) //best for code reusability….
9. Composition (HAS A Relationship) //best for code reusability….

Class:

Class is a **Core** of Java Applications.
Class is a **Keyword**.
Class is a **Blue Print or Template** for creating the Objects.
A Class Contains **Fields, Methods** to define the **State and Behavior** of its object.

```
Class Hello{

}
```

Object:

Creating **Instance** of a Class is nothing but **Object Creation** of a Particular Class.
Real world **Entities** are known as Objects.

```
Object o = new Object();
```

Inheritance:

The process of acquiring **properties** from one class to other class is called Inheritance.
We have **extends** keyword to connect from one class to other.
The class which is giving **properties** to another class is called **Parent Class/Super Class.**
The class which is getting **properties** from another class is called **Child Class/Sub Class.**
Inheritance Concept is also known as **IS-A-Relationship**.

Is-A Relationship

In **Is-A Relationship** one class is obtaining the features of another class by using **Inheritance** concept with **extends** keywords.
It means, that the child class is a type of parent class.

HAS-A-Relationship

In **Has-A Relationship** an object of one class is created as **Data Member** in another class.
The relationship between these two classes is Has-A.

Types of Inheritance

1.Single Level Inheritance
2.Multi Level
3.Hirarchial Inheritance
4.Multiple Inheritance
5.Hybrid Inheritance

Single Level Inheritance →

In Single Level Inheritance the **properties** extending from **Single Parent Class** to **Single Child Class** is called Single Level Inheritance.

Class **A**
Class **B**

Class **A** <- Class **B** **{ Class B is derived from One Base Class A}**

Multilevel Inheritance →

EX:

A, **B** and **C** are the Classes.

C can be a sub class of **B** which is a sub class of **A**.

C inherits all the aspects of **B** and **A**.

Class **C** -→ Class **B** → Class **A** **{ Class C is derived from One Base Class B}**
{ Class B is derived from One Base Class A}

Multiple Inheritance

Ex:

A, **B** and **C** are the Classes.

Class **B** → Class **A**

Class **C** → Class **B**

Class **C** → Class **A**, **B** (not possible)

1. Multiple Inheritance is **not supported** by Java because of **ambiguity** problems.
2. But, we have support **Interface.**

Hierarchical Inheritance: →

A, **B**, **C** and **D** are the Classes.

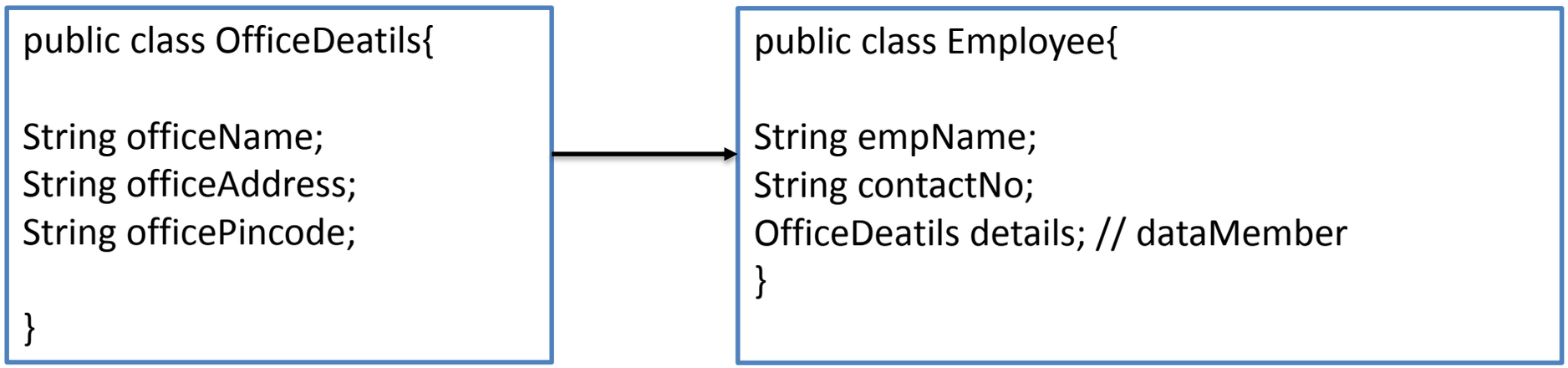
Class **B** → Class **A { Class B is derived from One Base Class A}**
Class **C** → Class **A { Class C is derived from One Base Class A}**

Getting **Properties** from **One Parent Class** to **Many Child Class** is Possible.
When two or more Classes Inherits same class.

Hybrid Inheritance: Combination of **Multilevel Inheritance** and **Hierarchical Inheritance.**

A, B, C and D are the Classes.
Class B → Class A
Class C → Class B
Class D → Class C

Class B → Class A
Class C → Class A
Class D → Class A


public class OfficeDeatils{
String officeName;
String officeAddress;
String officePincode;
}
public class Employee{
String empName;
String contactNo;
OfficeDeatils details; // dataMember
}

```java
package com.dl.singlelevel;

public class Parent{

public void m1() {
System.out.println("M1 Method");
}

public static void m2() {
System.out.println("M2 Method");
}
}
```

```java
package com.dl.singlelevel;

public class Child extends Parent {

public static void main(String[] args) {

Child child = new Child();
child.m1();
Child.m2();
}
}
```

M1 Method
M2 Method

```java
package com.dl.multilevel;

public class GrandParent {

public void m1() {
System.out.println("M1 Method");
}
}
```

```java
package com.dl.multilevel;

public class Parent extends GrandParent{

public void m2() {
System.out.println("M2 Method");
}
}
```

```java
package com.dl.multilevel;

public class Child extends Parent {

public void m3() {
System.out.println("M3 Method");
}

public static void main(String[] args) {

Child child = new Child();
child.m3();
child.m2();
child.m1();

}
}
```

```
M3 Method
M2 Method
M1 Method
```

```
package com.dl.multiple;

public class GrandParent {

}
```

```
package com.dl.multiple;

public class Parent extends GrandParent{

}
```

```
package com.dl.multiple;

public class Child extends Parent, GrandParent{

}
```

Exception in thread "main" java.lang.Error:
Unresolved compilation problems:

Multiple Inheritance is not supported in Java Class`s, But Multiple Inheritance is supported in Java Interfaces

```
package com.dl.multiple.interfaces;

public interface GrandParent {

}
```

```
package com.dl.multiple.interfaces;

public interface Child extends Parent, GrandParent{

}
```

```
package com.dl.multiple.interfaces;

public interface Parent {

}
```

```
public class Address {

String city;
String state;
String country;

public Address(String city, String state, String country) {
this.city = city;
this.state = state;
this.country = country;
}
}
```

```
public class Emp {

int id;
String name;
Address address; // data member, has-a-relationship

public Emp(int id, String name, Address address) {
this.id = id;
this.name = name;
this.address = address;
}

public void display() {
System.out.println(id + " " + name + " " + address.city + " " + address.state + " " + address.country);
}

public static void main(String[] args) {
Address address1 = new Address("Hyd", "TG", "India");
Emp e1 = new Emp(111, "Sai Kiran", address1);
e1.display();

Address address2 = new Address("Hyd", "TG", "India");
Emp e2 = new Emp(222, "Sai Kiran", address2);
e2.display();
}
}
```

```
111 Sai Kiran Hyd TG India
222 Sai Kiran Hyd TG India
```