In objected oriented programing language starts from bottom to top because we write main method in bottom

In procedure-oriented programing language starts from top to bottom.

Types Of Inheritance:

1) Single Inheritance ⇨ One level of Inheritance [OneParent-One Child]
Every Java class by default exhibit single inheritance, because Object class is the parent Class for all the Classes by default.

2) Multi-Level Inheritance ⇨ In Multilevel inheritance a child class object should be able to access both parent class methods and grandparent class methods.

3) Heirarchal Inheritance ⇨

4) Hybrid Inheritance ⇨

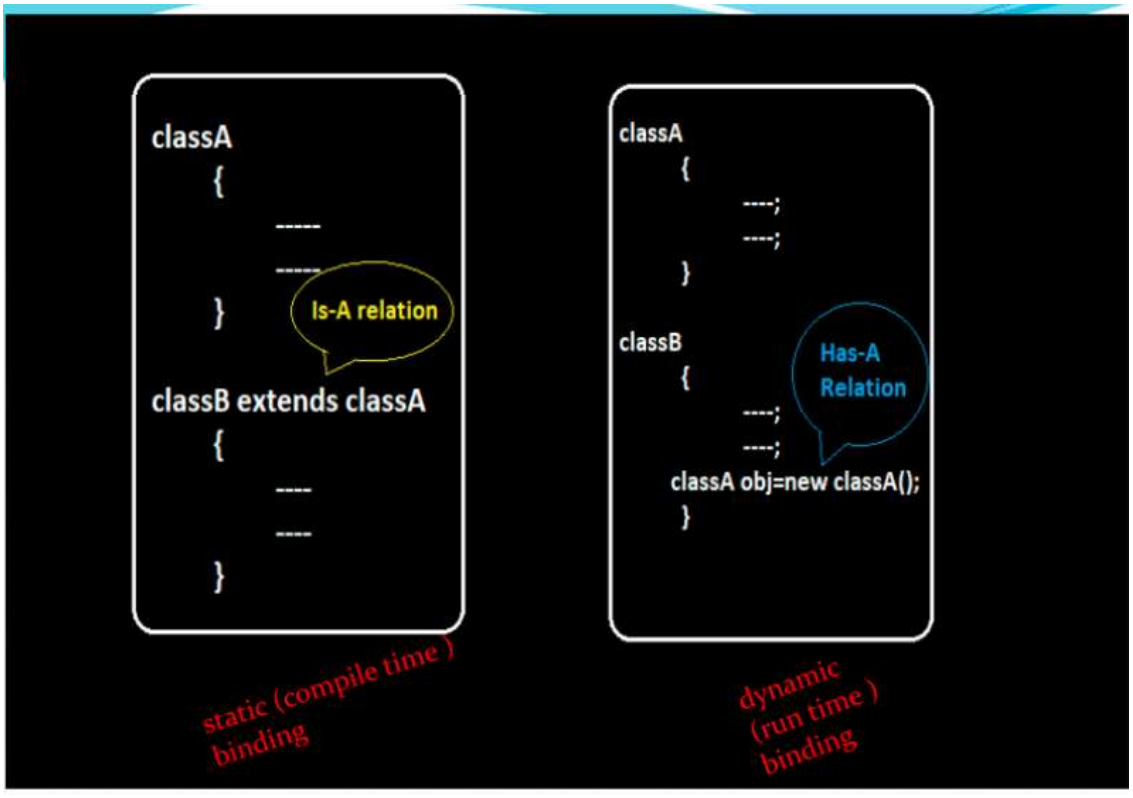
5) Multiple Inheritance ⇨

Understanding Inheritance

- Inheritance is the ability of a class inheriting data and behaviors from another class.
- It is also called "Is-A" relation.
- The main advantage of inheritance is code reusability.
- By using "extends" keyword we can implement IS-A relationship.
- After inheriting the complete functionality of super class Sub class can access the super class methods with its reference object.

Common Terminology:

Parent class	–	Child class
Base class	-	Derived class
Super class	-	Sub class



- All the data and methods which were present in parent class is by default available to child class, but the reverse is not applicable.
- Hence by using child class reference we can call both parent and child class methods.
- But by using parent reference we can call only methods available in the parent class and we can't call child class specific methods.
- Parent class reference can be used to hold child class object, but Child class reference cannot be used to hold parent class object.
- For all the java classes including predefined and user defined classes Object class acts as the super class to be precise **java.lang.Object** class is superclass of all classes.

- When a class extends another class, the subclass inherits all the public and protected members of the super class. The default members are inherited only in the same package.
- Constructors are not inherited in to the sub class during inheritance, we can call the constructors of super class by invoking super class object.

Types Of Inheritance:

- Java supports '3' types of inheritance.
    1. Single Inheritance
    2. Multi-Level Inheritance
    3. Hierarchal Inheritance

Inheritance: Inheritance means acquaring the properties of one class into another class.

Code Reusability — Inorder to acheive Inheritance we need to use either **extends** (or) **implements** keywords

```
public class ClassA
{
    void meth1()
    {
        System.out.println("Class A method");
    }
}
```

```
public class ClassB extends ClassA
{
    void meth2()
    {
        System.out.println("Class B method");
    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();

        ClassB bobj=new ClassB();
        bobj.meth1();
        bobj.meth2();
    }
}
```

ClassA	ClassB
Parent Class	Child Class
Super Class	Sub Class
Base Class	Derived Class

Keypoints

1) We can hold a parent class of Object with a parent class reference and with that reference we can call only parent class methods
   `ClassA aobj=new ClassA(); [P]`
2) We can hold a child class object with a parent class reference and with that reference we can call only parent class methods
   `ClassA aobj=new ClassB(); [P]`
3) We can hold a child class object with a child class reference and we can call **both** parent and child class methods
   `ClassB bobj=new ClassB(); [P & C]`
4) We cannot hold a parent class object with a child class reference will be getting an compile time error
   `ClassB bobj=new ClassA();` ⇨ **INVALID**

```
package com.pack1;

public class ClassA
{
    void meth1()
    {
        System.out.println("Class A method");
    }
}
```

```
package com.pack1;

public class ClassB
{
    void meth2()
    {
        System.out.println("Class B method");
    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();

        ClassB bobj=new ClassB();
        bobj.meth1();
        bobj.meth2();
    }
}
```

```
package com.pack1;

public class ClassA
{
    void meth1()
    {
        System.out.println("Class A method");
    }
}
```

```
package com.pack1;

public class ClassB extends ClassA
{
    void meth2()
    {
        System.out.println("Class B method");
    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();

        ClassB bobj=new ClassB();
        bobj.meth1();
        bobj.meth2();
    }
}
```

IS-A-Relationship is nothing but inheritance.

```java
package com.pack1;

public class ClassA
{
    void meth1()
    {
        System.out.println("Class A method");
    }
}
```

```java
package com.pack1;

public class ClassB extends ClassA
{
    void meth2()
    {
        System.out.println("Class B method");
    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA(); // HAS-A-Relation
        aobj.meth1();

        ClassB bobj=new ClassB(); // IS-A-Relation
        bobj.meth1();
        bobj.meth2();
    }
}

// ClassA ===> Parent Class
// ClassB ===> Child Class
```

Inheritance: Inheritance means acquaring the properties of one class into another class.

Code Reusability

Inorder to acheive Inheritance we need to use either **extends** (or) **implements** keywords

```java
public class ClassA
{
    void meth1()
    {
        System.out.println("Class A method");
    }
}
```

```java
public class ClassB extends ClassA
{
    void meth2()
    {
        System.out.println("Class B method");
    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();

        ClassB bobj=new ClassB();
        bobj.meth1();
        bobj.meth2();
    }
}
```

ClassA ClassB

Parent Class ⇨ Child Class
Super Class ⇨ Sub Class
Base Class ⇨ Derived Class

Keypoints

1) We can hold a parent class of Object with a parent class reference and with that reference we can call only parent class methods

ClassA aobj=new ClassA(); [P]

2) We can hold a child class object with a parent class reference and with that reference we can call only parent class methods

ClassA aobj=new ClassB(); [P]

3) We can hold a child class object with a child class reference and we can call **both** parent and child class methods

ClassB bobj=new ClassB(); [P & C]

4) We cannot hold a parent class object with a child class reference will be getting an compile time error

ClassB bobj=new ClassA(); ⇨ **INVALID**

```java
package com.pack1;

public class ClassA
{
    void meth1()
    {
        System.out.println("Class A
    }
}
```

```
Console ×
<terminated> ClassB [Java Application] C:\Program Files\Java\jdk-17\bin\javaw
Class A method
Class A method
Class A method
Class B method
```

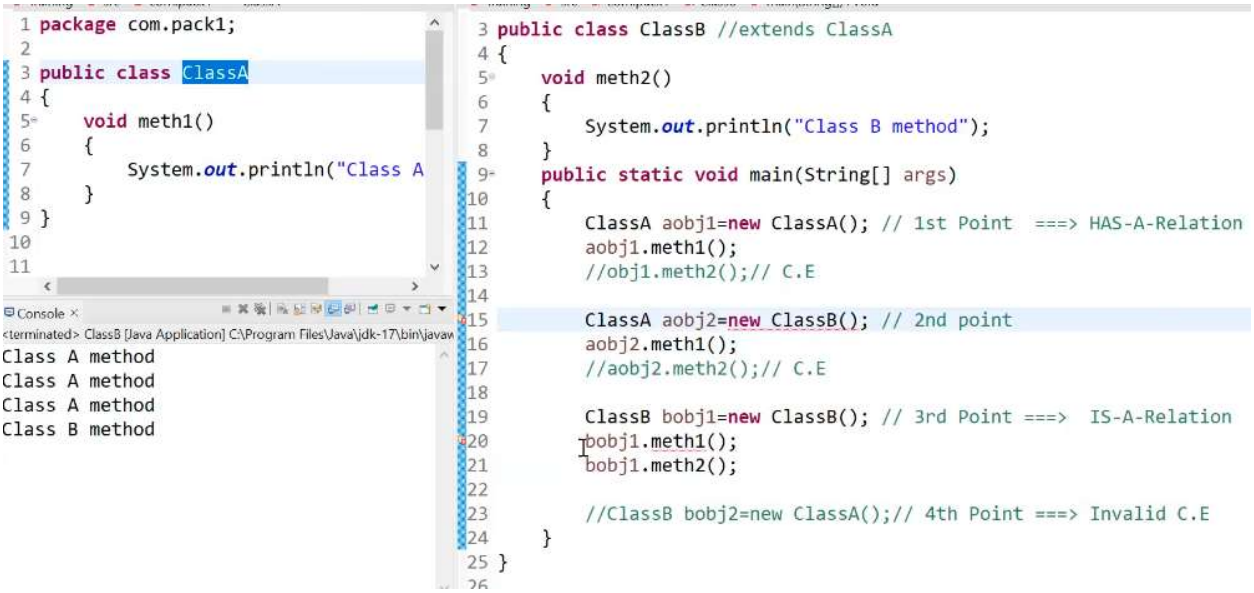
```java
package com.pack1;

public class ClassB extends ClassA
{
    void meth2()
    {
        System.out.println("Class B method");
    }
    public static void main(String[] args)
    {
        ClassA aobj1=new ClassA(); // 1st Point  ===> HAS-A-Relation
        aobj1.meth1();
        //obj1.meth2();// C.E

        ClassA aobj2=new ClassB(); // 2nd point
        aobj2.meth1();
        //aobj2.meth2();// C.E

        ClassB bobj1=new ClassB(); // 3rd Point ===>  IS-A-Relation
        bobj1.meth1();
        bobj1.meth2();

        //ClassB bobj2=new ClassA();// 4th Point ===> Invalid C.E
    }
```



```package com.pack1;

public class ClassA
{
    void meth1()
    {
        System.out.println("Class A
    }
}
```

```
Console ×
<terminated> ClassB [Java Application] C:\Program Files\Java\jdk-17\bin\javaw
Class A method
Class A method
Class A method
Class B method
```

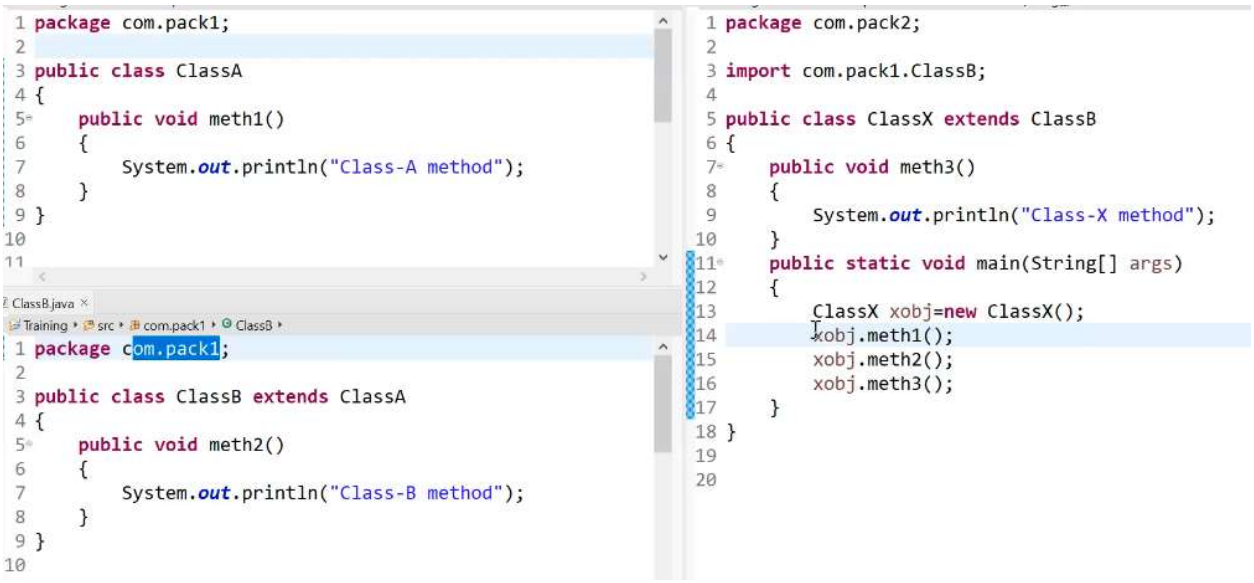
```public class ClassB //extends ClassA
{
    void meth2()
    {
        System.out.println("Class B method");
    }
    public static void main(String[] args)
    {
        ClassA aobj1=new ClassA(); // 1st Point  ===> HAS-A-Relation
        aobj1.meth1();
        //obj1.meth2();// C.E

        ClassA aobj2=new ClassB(); // 2nd point
        aobj2.meth1();
        //aobj2.meth2();// C.E

        ClassB bobj1=new ClassB(); // 3rd Point ===>  IS-A-Relation
        bobj1.meth1();
        bobj1.meth2();

        //ClassB bobj2=new ClassA();// 4th Point ===> Invalid C.E
    }
}
```

Multilevel inheritance



```package com.pack1;

public class ClassA
{
    public void meth1()
    {
        System.out.println("Class-A method");
    }
}
```

```ClassB.java ×
Training ▸ src ▸ com.pack1 ▸ ClassB ▸
package com.pack1;

public class ClassB extends ClassA
{
    public void meth2()
    {
        System.out.println("Class-B method");
    }
}
```

```package com.pack2;

import com.pack1.ClassB;

public class ClassX extends ClassB
{
    public void meth3()
    {
        System.out.println("Class-X method");
    }
    public static void main(String[] args)
    {
        ClassX xobj=new ClassX();
        xobj.meth1();
        xobj.meth2();
        xobj.meth3();
    }
}
```