Inheritance in Java

Definition:

Inheritance in Java is a mechanism where one class (called subclass or child class) acquires the properties and behaviors (fields and methods) of another class (called superclass or parent class). It promotes code reusability and method overriding.

Types of Inheritance in Java:

1. Single Inheritance: one child class inherits from one parent class
Eg . Dog inherits from animal

2. Multilevel Inheritance: A class inherits from a class which itself inherited from another class. .
Eg., BabyDog->Dog->Animal

3. Hierarchical Inheritance: Multiple child classes inherits from more than one interface.
Eg., Dog, Cat, Cow all inherit from animal.

4. (Multiple Inheritance) – A class inherits from more than one interface

5. (Hybrid Inheritance) - Achieved through interfaces only

Syntax:

```
class Parent {
   void display() {
      System.out.println("This is the parent class.");
   }
}

class Child extends Parent {
   void show() {
      System.out.println("This is the child class.");
   }
}
```

Example:

```
class Animal {
   void eat() {
      System.out.println("This animal eats food.");
   }
}

class Dog extends Animal {
   void bark() {
```

```
        System.out.println("The dog barks.");
    }
}

public class TestInheritance {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.eat();   // Inherited method
        d.bark();  // Child method
    }
}
```

Output:

```
This animal eats food.
The dog barks.
```

SUPER_KEYWORD

What is super keyword?

The super keyword in Java is used inside a subclass to refer to its immediate parent class.

It helps access parent class methods, constructors, and variables when they are overridden or hidden by the child class.

Uses of super keyword:

1. Access parent class variable:

If the child class has a variable with the same name as the parent, super can be used to access the parent version.

```
class Parent {
    int a = 10;
}

class Child extends Parent {
    int a = 20;

    void show() {
        System.out.println(super.a); // prints 10 (from Parent)
    }
}
```

2. Call parent class method:

If a method is overridden in the child class, super.method() calls the parent version.

```
class Parent {
   void display() {
      System.out.println("Parent method");
   }
}

class Child extends Parent {
   void display() {
      super.display();  // calls Parent's display
      System.out.println("Child method");
   }
}
```

Output:

```
Parent method
Child method
```

3. Call parent class constructor:

super() is used to call the constructor of the parent class from the child class constructor.

```
class Parent {
   Parent() {
      System.out.println("Parent Constructor");
   }
}

class Child extends Parent {
   Child() {
      super(); // calls Parent()
      System.out.println("Child Constructor");
   }
}
```

Output:

```
Parent Constructor
Child Constructor
```