INHERITANCE

(1) Inheritance:

- Inheritance is one of the key features of OOPS that allow user to create a new class from an existing class. The new class that is created is known as subclass(child or derived class) and existing class from where the child class is derived is known as superclass(parent or base class).

  Ex-

```
package org.example.opps_concept.inheritance;

public class Vehicle {
    int countWheel;
    String model;

    void start(){
        System.out.println("vehicle is starting..");
    }
}
```

```
package org.example.opps_concept.inheritance;

public class Car extends Vehicle{
    public static void main(String[] args) {
        Car car = new Car();
        car.countWheel=4;
        car.model="I10";
        car.start();
        System.out.println(car.countWheel);
    }
}
```

(2) Method overriding:

- If a subclass provides the specific implementation of the method that has been declared by one of its parent class , it is known as method overriding.
  Method overriding is also known as runtime polymorphism. Hence, we can achieve polymorphism in java with the help of inheritance .

  Ex-

```
package org.example.opps_concept.inheritance;

public class Vehicle {
    int countWheel;
    String model;

    void start(){
        System.out.println("vehicle is starting..");
```

```
    }
}
```

```
package org.example.opps_concept.inheritance;

public class Car extends Vehicle{

    @Override
    void start() {
        System.out.println("Car is starting.");
    }

    public static void main(String[] args) {
        Car car = new Car();
        car.start(); // car is starting.
    }
}
```

(3) Super keyword:

- Super is a special keyword in java that is used to refer to the instance of the immediate parent class.
- Use of super keyword in java:

  (a) It is used to refer to an instance variable of the immediate parent class.

  (b) It is used to invoke a method of the immediate parent class.

  (c) It is used to invoke a constructor of the immediate parent class.

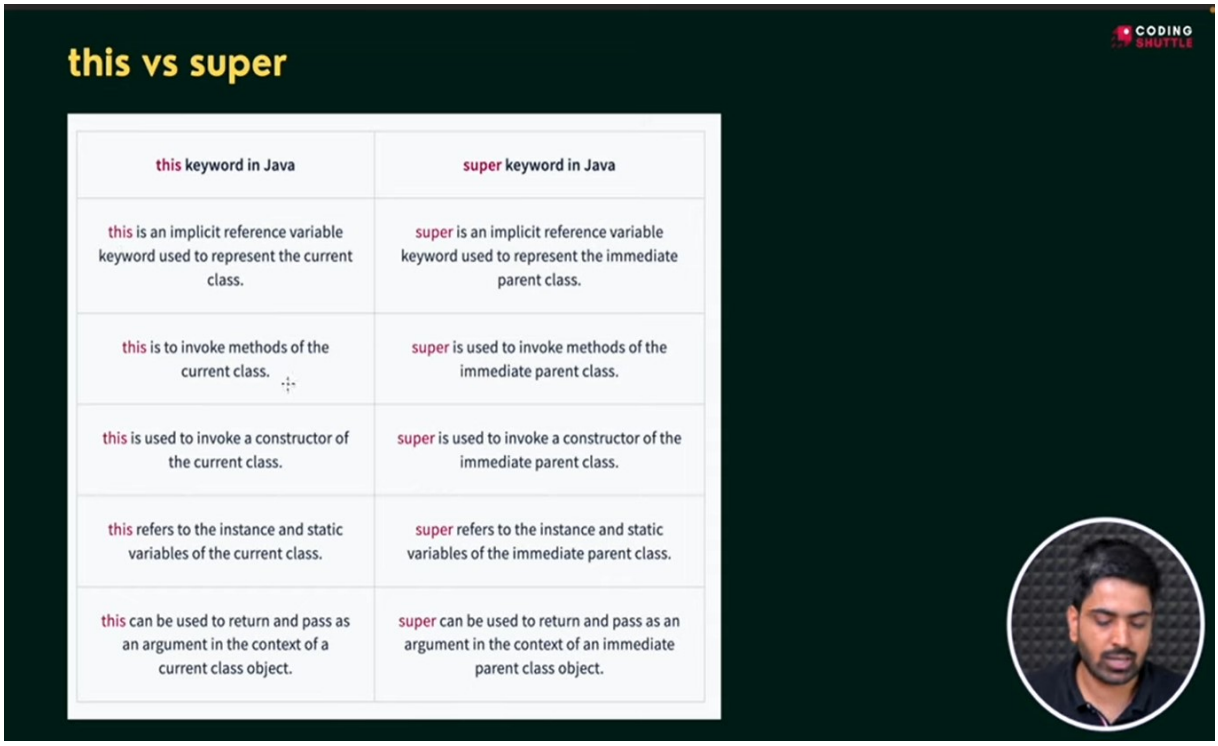
  Ex-

```
package org.example.opps_concept.inheritance;

public class Vehicle {
    int countWheel;
    String model;

    public Vehicle(){
        System.out.println("parent constructor");
    }
    void start(){
        System.out.println("vehicle is starting..");
    }
}
```

```
package org.example.opps_concept.inheritance;

public class Car extends Vehicle{

    // parent constructor call
```

```
    // parent properties used
    Car(int wheelCount,String model){
        super();
        this.countWheel=wheelCount;
        this.model=model;
    }

    // child constructor to parent method
    // - called
    Car(int i){
        super.start();

    }


    // child method to parent method
    // - called
    void test(){
        super.start();

    }

    public static void main(String[] args) {
        Car car = new Car(56,"maruti");
        System.out.println(car.countWheel);
        System.out.println(car.model);
    }
}
```

(4) Difference between this and super?



(5) Final keyword: final keyword is a non-access modifier that is used to define entities that can not be changed or modified.

- Final Variable: variable with final keyword can not be assigned again.
- Final Method: method with final keyword can not be overridden by it's subclasses.
- Final class: class with final keyword can not be extended or inherited from other classes.