16.3.1 'this' and 'super' keyword

🔹 1. Introduction

In Java, `this` and `super` are **reference keywords** used inside a class or subclass to refer to current and parent class members, respectively.

Keyword	Refers to
`this`	Current class
`super`	Immediate superclass

These keywords help manage **inheritance**, **constructor calls**, and **variable/method ambiguity**.

🔹 2. The `this` Keyword in Java

✅ 2.1. `this` to Refer to Current Class Instance Variables

Used when **instance variable names** conflict with **constructor or method parameters**.

🔸 **Example:**

```java
class Student {
    int id;
    String name;

    Student(int id, String name) {
        this.id = id;        // 'this' refers to current object
        this.name = name;
    }

    void display() {
        System.out.println(id + " " + name);
    }
}
```

✅ 2.2. `this` to Invoke Current Class Methods

Can be used to **explicitly call another method** from the same class.

🔸 **Example:**

```java
class Demo {
    void show() {
        System.out.println("Show method called");
    }

    void display() {
        this.show();  // same as just calling show();
    }
}
```

✅ 2.3. `this()` to Call Another Constructor (Constructor Chaining)

Used to call **another constructor of the same class**. Must be the **first statement** in the constructor.

🔸 **Example:**

```java
class Book {
    String title;
    int pages;

    Book() {
        this("Unknown", 0);  // Calls parameterized constructor
    }

    Book(String title, int pages) {
        this.title = title;
        this.pages = pages;
    }

    void display() {
        System.out.println(title + " - " + pages + " pages");
    }
}
```

🔹 3. The `super` Keyword in Java

Used to refer to the **immediate parent class** of the current object. Useful when a subclass inherits from a superclass.

✅ 3.1. `super` to Refer to Parent Class Variables

Used when **subclass variables** hide **superclass variables** (name conflict).

🔸 **Example:**

```java
class Parent {
    int x = 100;
}

class Child extends Parent {
    int x = 200;

    void print() {
        System.out.println("Child x: " + x);
        System.out.println("Parent x: " + super.x);
    }
}
```

✅ 3.2. `super` to Call Parent Class Methods

Used to **invoke an overridden method** of the parent class from the subclass.

🔸 **Example:**

```java
class Animal {
    void sound() {
        System.out.println("Animal makes sound");
    }
}

class Dog extends Animal {
    void sound() {
        System.out.println("Dog barks");
    }

    void callParentSound() {
        super.sound();  // Calls Animal's sound()
    }
}
```

✅ 3.3. `super()` to Call Parent Class Constructor

Used to explicitly call **parent class constructor**. Must be the **first line** of the subclass constructor.

🔸 **Example:**

```java
class Vehicle {
    Vehicle() {
        System.out.println("Vehicle constructor");
    }
}

class Car extends Vehicle {
    Car() {
        super();  // calls Vehicle()
        System.out.println("Car constructor");
    }
}
```

🔄 4. Constructor Chaining in Java

Constructor chaining means **calling one constructor from another**, either in the **same class (** `this()` **)** or **parent class (** `super()` **)**.

🔹 4.1. Using `this()` — Same Class Constructor Chaining

```java
class Employee {
    String name;
    int id;

    Employee() {
        this("Default Name", 0);  // Calls parameterized constructor
    }

    Employee(String name, int id) {
        this.name = name;
        this.id = id;
    }

    void display() {
        System.out.println(name + " - " + id);
    }
}
```

🔹 **4.2. Using `super()` — Parent Class Constructor Chaining**

```java
class Person {
    Person() {
        System.out.println("Person constructor");
    }
}

class Teacher extends Person {
    Teacher() {
        super();  // Calls Person()
        System.out.println("Teacher constructor");
    }
}
```

⚠️ 5. Important Rules & Points

🔸 `this` **keyword**

- Must be the **first statement** if used in a constructor.
- Used for **method**, **variable**, and **constructor** access within the same class.

🔸 `super` **keyword**

- Can be used to access **hidden fields**, **overridden methods**, and **parent constructors**.
- If not explicitly called, the compiler **automatically inserts** `super()` as the first line in a subclass constructor (if no `this()` is used).

📚 6. Summary Table

Feature	`this`	`super`
Refers to	Current class	Parent class

Accesses	Variables, methods, constructors	Parent variables, methods, constructors
Used in	Same class	Subclass extending superclass
Must be first statement?	Yes (if calling constructor)	Yes (if calling constructor)
Constructor Chaining	Within same class (`this()`)	Parent class constructor (`super()`)

📝 7. Practice Questions

1. Create a class with two constructors and use `this()` to chain them.
2. Create a parent and child class, both with constructors. Use `super()` in the child class.
3. Demonstrate the use of `this` and `super` with method and variable name conflicts.
4. Explain what happens if you use both `this()` and `super()` in the same constructor.

Document by **Suyash** 😇