STATIC IN JAVA

What is `static`?

`static` is a keyword in Java used for memory management.

It belongs to the **class**, not to the object.

If something is static, it is shared among all objects of the class.

Why Static is Needed?

As shown in your Page 4 memory diagram

class delivery notes 05

, Java memory is divided into:

- Heap Memory
- Stack Memory
- Method Area (Class Area)

Static members are stored in the **Method Area**.

Memory Representation

Explanation:

- Instance variables → Stored in Heap (separate copy for each object)
- Static variables → Stored once in Method Area
- All objects share the same static variable

Types of Static Members

1. Static Variable
2. Static Method
3. Static Block
4. Static Nested Class

---

1. Static Variable

Definition

A variable declared using the `static` keyword.
It is shared among all objects.

Example

```
class Student {
  int rollNo;
  static String college = "ABC College";

  Student(int r) {
     rollNo = r;
  }

  void display() {
     System.out.println(rollNo + " " + college);
  }

  public static void main(String[] args) {
     Student s1 = new Student(1);
     Student s2 = new Student(2);

     s1.display();
     s2.display();
  }
}
```

Output:

1 ABC College

2 ABC College

Only one copy of `college` exists in memory.

If 1000 students are created:

- rollNo → 1000 copies
- college → 1 copy

---

2. Static Method

Definition

A method declared using `static`.
It can be called using the class name without creating an object.

Example

```
class Demo {

  static void show() {
     System.out.println("Static Method Called");
  }

  public static void main(String[] args) {
     Demo.show();
  }
}
```

Important Rules

- Static methods can access only static data directly.
- They cannot use the `this` keyword.
- They cannot directly access non-static variables.

---

3. Static Block

Definition

Used to initialize static variables.
Executed only once when the class is loaded.

Example

```
class Test {
   static int x;

   static {
      x = 100;
      System.out.println("Static Block Executed");
   }

   public static void main(String[] args) {
      System.out.println(x);
   }
}
```

Output:

```
Static Block Executed
100
```

---

4. Static Nested Class

```
class Outer {
   static class Inner {
      void show() {
         System.out.println("Static Nested Class");
      }
   }
}
```

FINAL KEYWORD IN JAVA

Unified Definition

The `final` keyword in Java is used to restrict modification.
It can be applied to variables, methods, and classes to prevent changes after declaration.

Depending on where it is used, `final` prevents:

- Value modification (Variable)
- Method overriding (Method)
- Inheritance (Class)

---

1. Final Variable

Definition

A final variable is a variable whose value can be assigned only once.
After it is initialized, its value cannot be changed.

Key Points

- Must be initialized:
    - At the time of declaration, or
    - Inside a constructor (for instance variables)
- Reassignment is not allowed.
- Often used to create constants.

Example

```
class Demo {
  final int x = 10;

  void change() {
     // x = 20;  // Compile-time error
  }
```

```
}
```

Blank Final Variable

```
class Student {
  final int rollNo;

  Student(int r) {
      rollNo = r;  // assigned once
  }
}
```

---

2. Final Method

Definition

A final method is a method that cannot be overridden by a subclass.

Key Points

- Prevents method overriding.
- Used to maintain original implementation.

Example

```
class Parent {
  final void show() {
      System.out.println("Parent Method");
  }
}

class Child extends Parent {
  // void show() {}  // Error: cannot override
}
```

---

3. Final Class

Definition

A final class is a class that cannot be inherited.

Key Points

- No other class can extend it.
- Used for security and immutability.
- Example: The String class is final in Java.

Example

```
final class Vehicle {
}

// class Car extends Vehicle { }  // Error
```

Final Keyword Summary Table

Applied To	Restriction	Prevents
Variable	Value change	Reassignment
Method	Overriding	Method overriding
Class	Inheritance	Subclass creation

2-Mark Exam Definition

The final keyword in Java is used to restrict modification.
When applied to a variable, it prevents value change;
when applied to a method, it prevents overriding;
when applied to a class, it prevents inheritance.