Lecture Notes: Access Modifiers in Java

1. Introduction

Access Modifiers in Java define the **visibility** or **accessibility** of classes, methods, and variables.
They control **who can access what** in your code.

---

2. Types of Access Modifiers

Modifier	Within Same Class	Same Package	Subclass (Different Package)	Other Package (Non-subclass)
public	✅ Yes	✅ Yes	✅ Yes	✅ Yes
protected	✅ Yes	✅ Yes	✅ Yes	❌ No
no modifier (default)	✅ Yes	✅ Yes	❌ No	❌ No
private	✅ Yes	❌ No	❌ No	❌ No

---

3. Syntax

modifier dataType variableName;

modifier returnType methodName() { ... }

4. Examples

Example 1: All Modifiers in One Class

```
// File: AccessExample.java
package accessdemo;

public class AccessExample {
    public int publicVar = 10;
    protected int protectedVar = 20;
    int defaultVar = 30; // no modifier
    private int privateVar = 40;

    public void publicMethod() {
        System.out.println("Public Method");
    }

    protected void protectedMethod() {
        System.out.println("Protected Method");
    }

    void defaultMethod() {
        System.out.println("Default Method");
    }

    private void privateMethod() {
        System.out.println("Private Method");
    }

    public void displayAll() {
```

```
        // All accessible inside the same class
        System.out.println(publicVar);
        System.out.println(protectedVar);
        System.out.println(defaultVar);
        System.out.println(privateVar);

        publicMethod();
        protectedMethod();
        defaultMethod();
        privateMethod();
    }
}
```

Example 2: Access in the Same Package

```
// File: SamePackageTest.java
package accessdemo;
public class SamePackageTest {
    public static void main(String[] args) {
        AccessExample obj = new AccessExample();

        // Accessible: public, protected, default
        System.out.println(obj.publicVar);
        System.out.println(obj.protectedVar);
        System.out.println(obj.defaultVar);

        obj.publicMethod();
        obj.protectedMethod();
        obj.defaultMethod();

        // ❌ Not Accessible: private
        // System.out.println(obj.privateVar);
        // obj.privateMethod();
    }
}
```

Output:

```
10
20
30
Public Method
Protected Method
Default Method
```

Example 3: Access from Different Package – Subclass

```
// File: SubClassTest.java
package otherpackage;
import accessdemo.AccessExample;

public class SubClassTest extends AccessExample {
    public static void main(String[] args) {
        SubClassTest obj = new SubClassTest();

        System.out.println(obj.publicVar);      // ✅
        System.out.println(obj.protectedVar);   // ✅ (through inheritance)

        obj.publicMethod();
        obj.protectedMethod();

        // ❌ Not Accessible
        // System.out.println(obj.defaultVar);
        // System.out.println(obj.privateVar);
        // obj.defaultMethod();
        // obj.privateMethod();
    }
}
```

Output:

```
10
20
Public Method
Protected Method
```

Example 4: Access from Different Package – Non-subclass

```
// File: NonSubClassTest.java
package otherpackage;
import accessdemo.AccessExample;

public class NonSubClassTest {
    public static void main(String[] args) {
        AccessExample obj = new AccessExample();

        System.out.println(obj.publicVar); // ✅
        obj.publicMethod();              // ✅

        // ❌ Not Accessible
        // System.out.println(obj.protectedVar);
        // System.out.println(obj.defaultVar);
        // System.out.println(obj.privateVar);
        // obj.protectedMethod();
        // obj.defaultMethod();
        // obj.privateMethod();
    }
}
```

Output:

```
10
Public Method
```

5. Key Points

1. **private** → Most restrictive, visible only inside the same class.
2. **default** → Accessible inside the same package only.
3. **protected** → Accessible in the same package + subclasses (even if in different package).
4. **public** → Accessible everywhere.

Same Package

- Classes are in the **same package** if they share the **same package name** at the top of the file.
- They can access each other's **public**, **protected**, and **default (no modifier)** members.
- **private** members are still restricted to the same class only.

Example:

package mypackage; // same for both files

Different Package

- Classes are in **different packages** if their package names are different.
- They can only access **public** members directly.
- **protected** members are accessible only if the class is a **subclass**.
- **default** and **private** members are **not** accessible.

Example:

package anotherpackage; // different from mypackage