ACCESS CONTROL IN JAVA PACKAGES

You are developing a **university management system** where different departments need to access student records. The system consists of multiple packages:

1. `university` (Main package)
2. `university.students` (Handles student details)
3. `university.faculty` (Handles faculty details)
4. `external` (For external organizations collaborating with the university)

Each class in these packages needs to access **Student** details with different access levels (Private, Protected, Public, and No Modifier).

Design a Java program that demonstrates **access control** for a `Student` class, considering different **visibility rules** across the following:

Same Class** (Inside `Student` class itself)
Same Package Subclass** (A subclass inside the same package as `Student`)
Same Package Non-Subclass** (Another class in the same package but not a subclass)
Different Package Subclass** (A subclass in a different package)
Different Package Non-Subclass** (A non-subclass in a different package)

1. **Create a `Student` class** inside `university.students` with the following:
   - **Private variable**: `private int studentID`
   - **No Modifier variable**: `String studentName`
   - **Protected variable**: `protected double GPA`
   - **Public variable**: `public String department`

2. **Access `Student` members from:**
   - Another class in the **same package** (both subclass and non-subclass).
   - A subclass in a **different package**.
   - A non-subclass in a **different package**.

3. **Test access levels** for each variable type (`private`, `protected`, `public`, and no modifier).

SPOT QUESTION

Implementing Multiple Inheritance Using Interfaces in Java

You are working for a **Smart Device Development Company** that builds **AI-powered Smart Homes**. Your team is designing a **Smart Home System** that integrates different functionalities like **Voice Control** and **Remote Control** into a single device. Design a Java program to implement the **SmartDevice** system using **multiple interfaces**.

1. **Create an interface named `VoiceControl`**
   - This interface should define a method:
     void activateVoiceCommand(String command);
   - The method should take a voice command as input and simulate a response.

2. **Create another interface named `RemoteControl`**
   - This interface should define a method:
     void pressButton(String button);
   - The method should take a button input (like "Power" or "Volume Up") and simulate the corresponding action.

3. **Create a class `SmartHomeDevice` that implements both `VoiceControl` and ` RemoteControl`**
   - Implement both methods to display appropriate messages when a voice command is given or a remote button is pressed.

4. **Write a main program in `SmartHomeTest` to test the functionality**
   - Create an instance of `SmartHomeDevice` and call both methods to simulate user interactions.

SPOT QUESTION