Lab Exercise 03.01: Dynamic Premium Calculation Based on Applicant Attributes

Objective:
The goal of this lab is to leverage Drools rules to dynamically calculate insurance premiums based on various attributes of the `Application` object. Students will create individual rules for each attribute affecting the premium calculation, encouraging a hands-on understanding of decision logic in rule engines.

Background:
The `Application` object includes several attributes that can influence the cost of an insurance premium. These attributes are:

- Pre-existing Conditions (PEC) (`boolean`)
- Risk Level (`String`: "Low", "Medium", "High")
- Smoker Status (`boolean`)
- Body Mass Index (BMI) (`int`)
- Age of Client (`clientAge`)

Requirements:

- Basic understanding of Drools rule syntax, session management, and rule attributes.
- Access to the provided Java classes and repository setup.

Tasks:

1. Setup a new directory and session for the execution of your rules. Use what you havev learned so-far.
2. **Create a `Premium` Java Class**:
    - Define a new Java class named `Premium` in your project. This class should include at least one attribute, `basePremium`, which represents the starting calculation point for an insurance premium.
    - Implement methods to adjust (increase/decrease) the `basePremium` based on various factors.
3. **Rule Creation**:
    - Develop separate rules within a `PremiumCalculation.drl` file for each of the following attributes: PEC, Risk, Smoker Status, BMI, and Age. Each rule should adjust the `basePremium` based on the value of its respective attribute.
    - **Example Rule Structure** (for illustration only):
        - `rule "Calculate Premium for Smoker Status"`
        - `when`
        - `  Applicant is a smoker`
        - `then`
        - `  // Pseudocode for adjusting premium. Students will de fine the logic.`

- `Increase Base Premium by 50`

`end`

4. **Dynamic Premium Calculation**:
    - Use your own creativity in how the premium is adjusted for each attribute. For instance, higher risk or the presence of pre-existing conditions might significantly increase the premium, whereas a lower BMI could reduce it.
    - Consider implementing a final rule that aggregates all adjustments to insert the final premium calculation.
5. **Testing**:
    - Create test scenarios that apply various combinations of attributes to ensure the rules work as expected and the final premium is calculated accurately.

Expected Outcome:

- The insurance premium should dynamically adjust based on the applicant's attributes.
- The `Premium` class should reflect the final calculated premium after all rules have been applied.

Evaluation Criteria:

- Correct implementation and integration of the `Premium` class.
- Logical and efficient rule creation for premium calculation based on given attributes.
- Demonstrable proper functioning of rules and session in lab environment.

Note: This exercise not only tests the students' ability to work with Drools but also challenges them to think critically about how different factors affect insurance premium calculations, thus bridging the gap between technical skills and practical business logic.