Defining and Implementing Interfaces

1. **Smart Device Control Interface**
    - Scenario: All devices (lights, AC, TV) should have `turnOn()` and `turnOff()` methods.
    - Task: Create an interface and implement it in multiple classes.
2. **Multi-Vehicle Rental System**
    - Scenario: Cars, bikes, and buses share `rent()` and `returnVehicle()` methods.
    - Task: Use interface-based design.
3. **Digital Payment Interface**
    - Scenario: UPI, Credit Card, Wallet all must implement `pay()` method.
    - Task: Define and implement.

Functional Interfaces

1. **Temperature Alert System**
    - Scenario: Alert if temperature crosses threshold.
    - Task: Use `Predicate<Double>` functional interface.
2. **String Length Checker**
    - Scenario: Check if a message exceeds character limit.
    - Task: Use `Function<String, Integer>`.
3. **Background Job Execution**
    - Scenario: Execute tasks asynchronously.
    - Task: Use `Runnable` functional interface.

Static Methods in Interfaces

1. **Password Strength Validator**
    - Scenario: In an insurance portal, password policy rules are centrally defined.
    - Task: Create a static method in an interface `SecurityUtils` to check password strength.
2. **Unit Conversion Tool**
    - Scenario: Logistics software needs standard unit conversions (km to miles, kg to lbs).
    - Task: Implement conversions as static interface methods.
3. **Date Format Utility**
    - Scenario: An invoice generator must format dates in multiple formats.
    - Task: Use a static interface method to format dates.

Default Methods in Interfaces

1. **Payment Gateway Integration**
    - Scenario: Multiple payment providers integrate with your app. A new refund method needs to be added without breaking old providers.
    - Task: Add a default `refund()` method in the `PaymentProcessor` interface.
2. **Data Export Feature**
    - Scenario: A reporting module can export in CSV and PDF. Later, JSON support was added.
    - Task: Add a default method `exportToJSON()` to avoid code changes in all implementers.
3. **Smart Vehicle Dashboard**
    - Scenario: All vehicles have a method `displaySpeed()`, but electric vehicles also display battery percentage.
    - Task: Use default methods to add this new feature.

Marker Interfaces

1. **Data Serialization for Backup**
   - Scenario: Mark certain classes as `Serializable` for backup storage.
   - Task: Implement marker interface for backup processing.
2. **Cloning Prototype Objects**
   - Scenario: Clone a predefined object model.
   - Task: Use `Cloneable` marker interface.
3. **Sensitive Data Tagging**
   - Scenario: Mark sensitive data classes for encryption.
   - Task: Create a custom marker interface.