Java_Assignment-04

1️⃣ Inheritance - Social Media Influencer System

🎯 **Scenario:** Create a system where different types of **content creators** inherit properties from a base **Influencer** class.

🔷 **Base class:** `Influencer`

- Properties: `name`, `followers`, `platform` (Instagram, YouTube, TikTok)
- Method: `showDetails()` to display influencer info

🔷 **Subclasses:**

- `YouTuber`: Additional property `subscribers` and method `createVideo()`
- `TikToker`: Additional property `viralVideos` and method `doTrendyChallenge()`

📝 **Task:** Create objects for a YouTuber and a TikToker, set values, and call their respective methods.

2️⃣ Method Overriding - AI Chatbot Personalities

🎯 **Scenario:** Build a **Chatbot system** where different AI personalities override a general response method.

🔷 **Base class:** `Chatbot`

- Method `respond()` → `"Hello! How can I assist you?"`

🔷 **Subclasses:**

- `SassyBot` → Overrides `respond()` to give funny replies
- `MotivationBot` → Overrides `respond()` to give motivational responses
- `TechBot` → Overrides `respond()` to provide tech news

📝 **Task:** Create objects for each chatbot and call `respond()`.

3️⃣ Method Overloading - Music Streaming App

🎯 **Scenario:** Simulate a **music player** that can play songs based on different parameters.

🔷 **Class:** `MusicPlayer`

- Overload the `playSong()` method:
  - `playSong(String songName)` → Plays song by name
  - `playSong(String songName, String artistName)` → Plays song with artist info
  - `playSong(int durationInSeconds)` → Plays a random song for a given duration

📝 **Task:** Create an object of `MusicPlayer`, call all overloaded `playSong()` methods, and observe the behavior.

4️⃣ Multi-Level Inheritance - Streaming Subscription Service

🎯 **Scenario:** Model a **Streaming Service Subscription System** with multi-level inheritance.

🔷 **Base class:** `Subscription`

- Properties: `userName`, `subscriptionType` (Basic, Premium, VIP)
- Method `showSubscription()`

🔷 **Intermediate class:** `PremiumSubscription` (inherits Subscription)

- Adds feature: `downloadContent()`

🔷 **Final class:** `VIPSubscription` (inherits PremiumSubscription)

- Adds feature: `accessExclusiveShows()`

📝 **Task:** Create a **VIP user**, assign values, and call all available methods.

5️⃣ Abstraction - Gaming Console System

🎯 **Scenario:** Create an **Abstract Gaming Console** class where different gaming consoles extend it.

🔷 **Abstract class:** `GamingConsole`

- Abstract method `startGame()`
- Method `displayConsoleDetails()`

🔷 **Subclasses:**

- `PlayStation` → Implements `startGame()` with `"Starting game on PlayStation!"`

- `Xbox` → Implements `startGame()` with `"Loading game on Xbox!"`

📝 **Task:** Create objects for **PlayStation and Xbox**, call `startGame()` and `displayConsoleDetails()`.

🚀 Submission Guidelines:

- Implement each program separately and test it.
- Use meaningful variable and method names.
- Have fun coding, and feel free to ask doubts!
