Aufgabe 04.02.2025 Fahrzeug-Verwaltungssystem

Aufgabenstellung:
Entwickle ein **Fahrzeugverwaltungssystem**, das verschiedene Fahrzeuge wie Autos und Motorräder verwaltet. Dabei sollen **Interfaces**, **Default-Methoden**, **Static-Methoden** sowie **Methodenüberladung** zum Einsatz kommen. Das System soll Fahrzeuge verwalten und deren Daten in einer Flotte speichern können.

1. Definiere ein Interface Fahrzeug

Erstelle ein Interface Fahrzeug, das folgende Methoden enthält:

- void starten(); → Startet das Fahrzeug
- void stoppen(); → Stoppt das Fahrzeug
- void beschleunigen(int geschwindigkeit); → Erhöht die Geschwindigkeit
- **Default-Methode:**
    - default void statusAnzeigen() → Zeigt den aktuellen Status des Fahrzeugs an (simpler String output)
- **Static-Methode:**
    - static void allgemeineInfo() → Gibt eine allgemeine Information zu Fahrzeugen aus (simpler String output)

Überlege:

- Wieso ist eine default-Methode hier sinnvoll?
- Warum kann eine static-Methode nicht auf Instanzvariablen zugreifen?

2. Erstelle eine abstrakte Klasse AbstraktesFahrzeug

Da alle Fahrzeuge einige gemeinsame Eigenschaften besitzen (z. B. **Modellname**, **Geschwindigkeit**, **Kennzeichen**), soll eine abstrakte Klasse AbstraktesFahrzeug erstellt werden, welche von dem Interface erbt.

- Diese Klasse soll die grundlegenden Eigenschaften und Methoden enthalten
    - protected String modell;
    - protected int maxGeschwindigkeit;
    - protected String kennzeichen;

- Konstruktor: AbstraktesFahrzeug(String modell, String kennzeichen, int maxGeschwindigkeit)
- void detailsAnzeigen() → Zeigt Fahrzeugdetails

3. Implementiere Auto als konkrete Klasse

Erstelle eine Klasse Auto, die sowohl das Interface Fahrzeug implementiert als auch von AbstraktesFahrzeug erbt.

- Auto soll zusätzlich eine Eigenschaft **Anzahl der Türen** haben.
- Implementiere die Methoden aus Fahrzeug und passe sie an. Es reichen simple String outputs als Nachricht in der Konsole.
- Überlade beschleunigen():
  - beschleunigen(int geschwindigkeit, int sekunden) → Gibt aus, wie viel km/h das Auto in der Zeit schafft.

4. Implementiere Motorrad als konkrete Klasse

Erstelle eine Klasse Motorrad, die ebenfalls Fahrzeug implementiert und AbstraktesFahrzeug erweitert.

- Motorräder haben zusätzlich eine Eigenschaft **Helmpflicht** (boolean).
- Implementiere beschleunigen():
  - Überlade die Methode so, dass auch ein **Gang** als Parameter übergeben werden kann.

5. Entwickle eine Klasse Flotte für das Management mehrerer Fahrzeuge

Erstelle eine Klasse Flotte, die eine **Liste von Fahrzeugen** speichert.

- Methode void fahrzeugHinzufuegen(Fahrzeug f), um Fahrzeuge zu einer Liste hinzuzufügen.
- Methode void alleFahrzeugeAnzeigen(), um alle gespeicherten Fahrzeuge auszugeben.
- Methode void gesamtAnzahlFahrzeuge(), um die Anzahl der gespeicherten Fahrzeuge auszugeben.

6. Schreibe die Main-Klasse zur Ausführung

In der main-Methode sollen folgende Schritte durchgeführt werden:

1. Erstelle eine Flotte.
2. Erzeuge mindestens zwei Autos und zwei Motorräder mit unterschiedlichen Werten.
3. Füge die Fahrzeuge zur Flotte hinzu.
4. Zeige alle Fahrzeuge an.
5. Beschleunige einige Fahrzeuge mit unterschiedlichen Methodenüberladungen.
6. Teste die statusAnzeigen()-Methode der Fahrzeuge.
7. Rufe die allgemeineInfo()-Methode des Interfaces auf.
8. Gib die Anzahl der gespeicherten Fahrzeuge aus.