Java Aufgaben zu Anonymen Klassen

Aufgabe 1: Künstler mit Anonymer Klasse

- Erstellen Sie eine **abstrakte Klasse** `Künstler` mit einer **abstrakten Methode** `kunstErstellen()`.
- In der `main`-Methode erzeugen Sie **zwei Objekte** vom Typ `Künstler` mit Hilfe **anonymer Klassen**.
- Überschreiben Sie die Methode `kunstErstellen()`:
  - Ein Maler soll auf der Konsole ausgeben: `"Ich male ein Bild"`.
  - Ein Sänger soll auf der Konsole ausgeben: `"Ich singe einen Song"`.
- Rufen Sie für beide Objekte die Methode auf.

Aufgabe 2: Taschenrechner mit Anonymen Klassen

- Erstellen Sie eine **Klasse** `Calculator` mit einer Methode `calculate(int a, int b, Operation op)`, die das Ergebnis zurückgibt.
- Erstellen Sie ein **Interface** `Operation` mit der Methode `execute(int a, int b)`.
- Verwenden Sie **anonyme Klassen**, um für folgende vier Operationen ein Objekt zu erstellen:
  - Addition
  - Subtraktion
  - Multiplikation
  - Division
- In der `main`-Methode übergeben Sie die jeweiligen anonymen Klassen an die `calculate`-Methode und geben das Ergebnis aus.

Aufgabe 3: Druckbare Objekte mit Anonymer Klasse

- Erstellen Sie ein **Interface** `Druckbar` mit der Methode `getInfo(): String`.
- Erstellen Sie eine **Klasse** `Kunde` mit dem Attribut `name`, das über den Konstruktor gesetzt wird.
- `Kunde` implementiert `Druckbar` und gibt in `getInfo()` den Namen zurück.
- Schreiben Sie in der Main-Klasse eine **statische Methode** `drucken(Druckbar... druckbar)`:
  - Die Methode gibt für jedes Objekt das Ergebnis von `getInfo()` aus.
- In der `main`-Methode:
  - Erstellen Sie zwei `Kunde`-Objekte.
  - Erstellen Sie eine **anonyme Klasse**, die `Druckbar` implementiert und in `getInfo()` nur `">"` zurückgibt.
  - Rufen Sie `drucken()` mit den beiden Kunden und dem anonymen Objekt dazwischen auf, um zusätzliche Trennung im Output zu erreichen.

Aufgabe 4: StringConverter mit Anonymer Klasse

- Erstellen Sie eine **abstrakte generische Klasse** `StringConverter<T>` mit:
  - `abstract String toString(T obj)`
  - `abstract T fromString(String str)`
- Erstellen Sie eine **Model-Klasse** `Mitarbeiter` mit:

    - `String name`, `int nummer`, beides über Konstruktor gesetzt.
    - `List<Mitarbeiter> liste` als **statische öffentliche Liste**.
    - Getter für `name`.
- Fügen Sie in `Mitarbeiter` ein **statisches Feld** `StringConverter<Mitarbeiter> converter` hinzu und instanziieren Sie eine **anonyme Klasse**:
    - `toString()` gibt die Nummer als String zurück.
    - `fromString()` durchsucht die Liste nach einem passenden Mitarbeiter anhand der Nummer. Falls keiner gefunden wird, gibt die Methode `null` zurück.
- In der `main`-Methode:
    - Erstellen Sie vier `Mitarbeiter` und fügen diese zur Liste hinzu.
    - Lassen Sie über `toString()` die Nummern ausgeben.
    - Fragen Sie eine Nummer über Konsole ab und geben über `fromString()` den entsprechenden Namen aus.