Aufgabe 06.02.2025 – Mitarbeiterverwaltung

Ein Unternehmen verwaltet seine Mitarbeiter in einer **hierarchischen Struktur**. Es gibt eine **Hauptklasse Mitarbeiter**, von der verschiedene Spezialisierungen erben:

- **Manager** → Verantwortlich für ein Budget
- **Entwickler** → Hat eine Programmiersprache als Spezialisierung
- **Praktikant** → Hat ein Ausbildungsjahr

Ziel ist es, ein **Mitarbeitersystem** zu entwickeln, das polymorph arbeitet. Methoden werden überschrieben und überladen, und wir nutzen instanceof, um sicherzustellen, dass spezielle Methoden von Unterklassen verwendet werden können.

1. Aufgabe: die Mitarbeiter-Basisklasse

🔹 **Anforderungen:**

- Die Klasse **Mitarbeiter** soll **abstract** sein.
- Enthält allgemeine Attribute für alle Mitarbeiter:
  - protected String name
  - protected double gehalt
  - protected String abteilung
- Erstelle ein Konstruktor, der alle Attribute entgegennimmt
- Enthält folgende Methoden:
  - public void anzeigen() → Gibt allgemeine Informationen aus.
  - public abstract void arbeiten(); → Muss von Unterklassen überschrieben werden.
  - **Überladene Methode** public void anzeigen(boolean detailliert) → Gibt je nach Parameter mehr Infos aus.

2. Aufgabe: Unterklassen Manager, Entwickler, Praktikant

🔹 **Klasse Manager erbt von Mitarbeiter**

Zusätzliche Attribute:

- private double budget;

Methoden:

- **Überschreibe den Konstruktor**→ verwende den super() und erweitere ihn
- **Überschreibe arbeiten()** → "Manager verwaltet das Budget von XYZ EUR"
- **Überschreibe anzeigen()**, um das Budget mit anzuzeigen
- **Zusätzliche Methode** public void calcPaycheck(Mitarbeiter m) → Gebe Je nachdem was der Mitarbeiter ist (Manager, Programmierer oder Praktikant) das Gehalt, den Namen und den Beruf in der Konsole aus.
- **Überlegung:** Was muss zusätzlich noch gemacht werden, um den Namen des Mitarbeiters herauszufinden, wenn der name private wäre?

🔹 **Klasse Entwickler erbt von Mitarbeiter**

Zusätzliche Attribute:

- private String programmiersprache;

Methoden:

- **Überschreibe den Konstruktor**→ verwende den super() und erweitere ihn
- **Überschreibe arbeiten()** → "Entwickler programmiert in XYZ"
- **Überschreibe anzeigen()** → erweitere die Programmiersprache
- **Zusätzliche Methode** public void debuggen(String project) → "Entwickler debuggt Projekt XYZ"

🔹 **Klasse Praktikant erbt von Mitarbeiter**

Zusätzliche Attribute:

- private int ausbildungsjahr;

Methoden:

- **Überschreibe den Konstruktor**→ verwende den super() und erweitere ihn
- **Überschreibe arbeiten()** → "Praktikant lernt neue Aufgaben"
- **Überschreibe anzeigen()** → erweitere durch das Ausbildungsjahr
- **Zusätzliche Methode** public void lernen() → "Praktikant besucht ein Seminar"

3. Aufgabe: Verwaltung der Mitarbeiter

Erstelle eine Firma-Klasse mit einer main()-Methode:

- Erstelle ein Array mit gemischten Mitarbeiter-Objekten
- Gehe das Array mit einer Schleife durch
- Wende auf jedes Objekt anzeigen() und arbeiten() an
- Prüfe, ob das Objekt ein Manager, ein Programmierer oder ein Praktikant ist. Wende je nachdem die passende Methode an (z.B. Praktikant hat die Methode lernen() usw.).
- Erstelle eine zweite Schleife und lasse alle Objekte im Array über einen Manager und die Methode calcPaycheck(Mitarbeiter m) laufen, welche die dementsprechenden Infos nochmal ausgibt.