Aufgabe / Übung

Voraussetzung:
Kenntnisse von OOP mit Python (Teil 1 und 2)

1. Erstellen Sie ein UML, das folgenden Sachverhalt modelliert:

   Mitarbeiter sollen verwaltet werden mit Namen und Gehalt. Dazu soll übergreifend die Gesamtzahl der Mitarbeiter ermittelt werden können. Auf ein Mitarbeiter-Objekt wird zugegriffen mit den Methoden *setName*, *getName*, *setGehalt*, *getGehalt* und *zeigeGehalt*. Der Name soll als *String*, das Gehalt als *float* abgespeichert werden.
   Arbeiter erbt von Mitarbeiter und kann Werkzeug besitzen.
   Tragen Sie ein bis zwei sinnvolle eigene Attribute und Methoden für Arbeiter ein.

2. Implementieren Sie dieses Modell in Python (TigerJython) mit entsprechenden Testobjekten. Erläutern Sie in ihren Kommentaren im Quelltext die Besonderheiten bei der Umsetzung einer Assoziation von UML in Python-Code am Beispiel der „hat“-Beziehung von Arbeiter zu Werkzeug.

3. Das UML-Diagramm soll ergänzt werden: Eine neue Klasse Abteilungsleiter erbt von Mitarbeiter. Ein Abteilungsleiter hat zusätzlich zum Mitarbeiter einen Umsatz als Eigenschaft. Zwei eigene Methoden des Abteilungsleiters sind *setProvisionsanteil* und *getProvisionsbetrag*. Damit kann in Abhängigkeit des Umsatzes eine Provision berechnet werden. Der Provisionsbetrag ist der prozentuale Provisionsanteil des Umsatzes. Beispiel: beträgt der Provisionsanteil 1,6 (Prozent), dann ergibt sich bei einem Umsatz von 98.000 (Euro) ein Provisionsbetrag von 1.568 (Euro). Jeder Abteilungsleiter hat ein Büro. Ein Büro wird durch eine Büronummer identifiziert und hat eine Größe in Quadratmetern. Ergänzen Sie das UML-Diagramm den obigen Vorgaben entsprechend. Geben Sie bei den Klassendefinitionen auch die notwendigen Datentypen und Parameter an.

4. Implementieren Sie ihr neues UML-Diagramm in Python (TigerJython) ebenfalls mit sinnvollen Testobjekten.