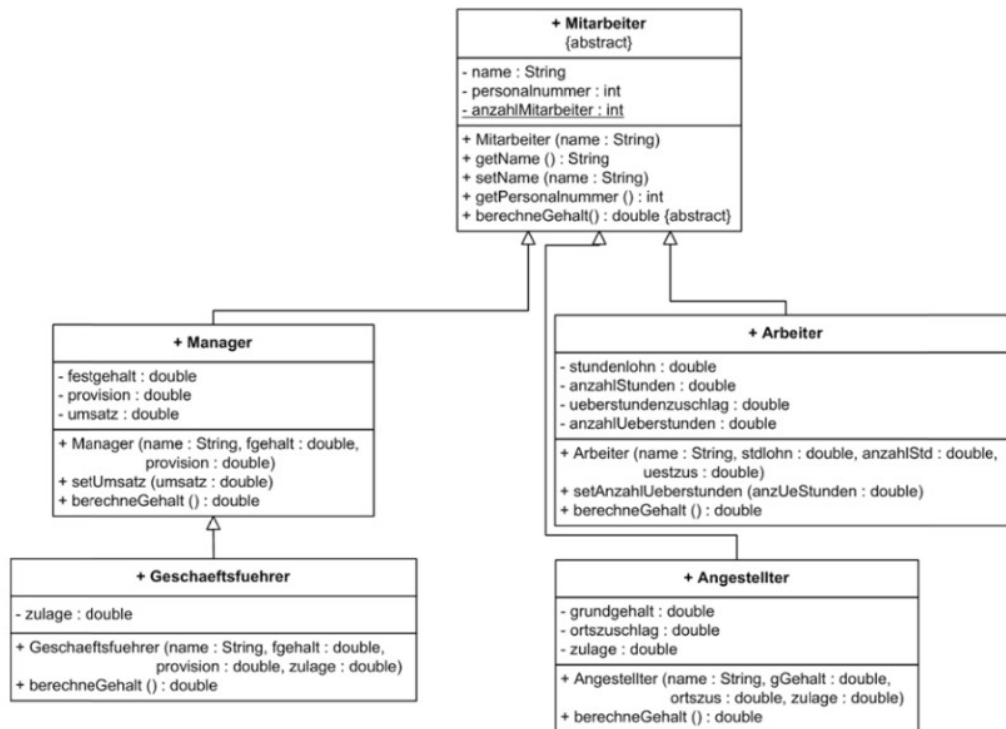


Für eine Mitarbeiterverwaltung wurde folgendes UML-Klassendiagramm entworfen:



Dieses soll teilweise in ein Java-Programm umgesetzt werden.

- Programmieren Sie die Klassen `Mitarbeiter`, `Manager` und `Geschäftsfuehrer` in Java. Jeder `Mitarbeiter` soll dabei eine fortlaufende Personalnummer erhalten, die sich aus der Anzahl der Mitarbeiter ergibt.
- Programmieren Sie eine Java-Klasse `MitarbeiterTest` mit einer `main`-Methode, die einen `Manager` und einen `Geschäftsfuehrer` erzeugt, deren Namen und Personalnummern sowie deren Gehalt auf der Konsole ausgibt.

Aufgabe 5 – BVB 09 und Co.

In dieser Aufgabe soll eine Verwaltung für Mannschaften aus der Fußball-Bundesliga realisiert werden. Für die Mannschaftsmitglieder muss zunächst eine Klassenhierarchie entwickelt und in Java implementiert werden. Anschließend sollen Testdaten erstellt und auf der Konsole ausgegeben werden. Letztendlich werden die Mitglieder zu einer Mannschaft zusammengefasst, und es soll dem Benutzer ermöglicht werden, weitere Mitglieder per Konsolen-Eingabe anzulegen.

a) Implementierung der Mannschaftsmitglieder

Mitglieder sind entweder Ärzte, Spieler oder Trainer. Alle Mitglieder sollen die Eigenschaften Name und Vorname besitzen. Die Spieler erhalten zusätzliche Angaben über ihre Spielposition und die Anzahl der Spieleinsätze pro Saison. Ärzte hingegen besitzen nur eine Angabe über ihre Fachrichtung. Dem Trainer soll ein bereits zuvor angelegter Spieler als Lieblingsspieler zugewiesen werden können. Jedes Mitglied soll zusätzlich über eine Methode verfügen, die sein Jahresgehalt berechnet. Ärzte sollen generell mit 10.000 €, Trainer mit 165.000 € und Spieler mit 180.000 € monatlich entlohnt werden. Ein Spieler erhält zudem einen Bonus in Höhe von 5.000 € pro Spieleinsatz.

Überlegen Sie, welche Klassen für diese Problemstellung benötigt werden, und implementieren Sie diese. Nutzen Sie hierzu die Konzepte der Vererbung, und beachten Sie das Geheimnisprinzip. Des Weiteren sollen Sie sicherstellen, dass nur Objekte vom Typ Arzt, Spieler und Trainer erzeugt werden können.

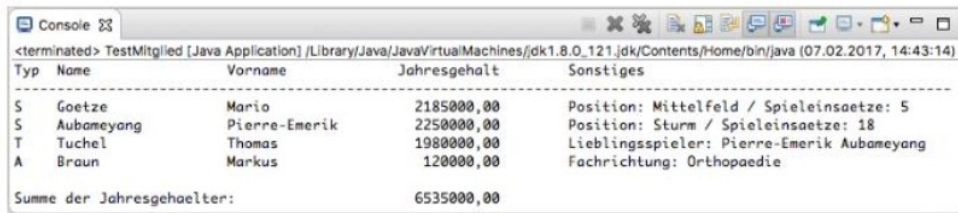
Tipp: Skizzieren Sie Ihre Klassenhierarchie zunächst in Form eines UML-Klassendiagramms.

b) Objekterzeugung und Konsolen-Ausgabe

Erstellen Sie eine Testklasse, in deren `main`-Methode die folgenden Objekte erzeugt werden:

| Typ | Name | Vorname | Spezifische Eigenschaften |
|---------|------------|---------------|--|
| Spieler | Götze | Mario | Position: Mittelfeld / Spieleinsätze: 5 |
| Spieler | Aubameyang | Pierre-Emerik | Position: Sturm / Spieleinsätze: 18 |
| Trainer | Tuchel | Thomas | Lieblingsspieler: Pierre-Emerik Aubameyang |
| Arzt | Braun | Markus | Fachrichtung: Orthopädie |

Anschließend sollen diese Objekte in Form einer Tabelle auf der Konsole ausgegeben werden:



Console [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/bin/java (07.02.2017, 14:43:14)

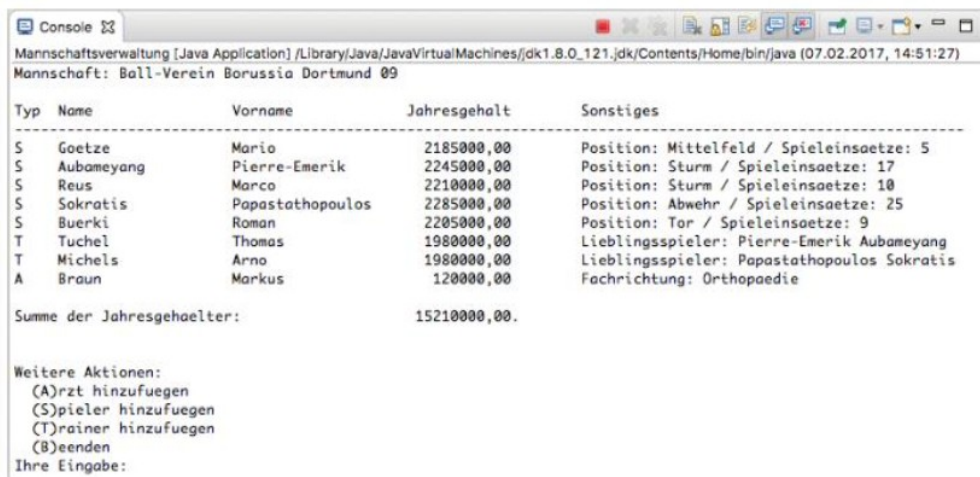
| Typ | Name | Vorname | Jahresgehalt | Sonstiges |
|----------------------------|------------|---------------|--------------|--|
| S | Goetze | Mario | 2185000,00 | Position: Mittelfeld / Spieleinsätze: 5 |
| S | Aubameyang | Pierre-Emerik | 2250000,00 | Position: Sturm / Spieleinsätze: 18 |
| T | Tuchel | Thomas | 1980000,00 | Lieblingsspieler: Pierre-Emerik Aubameyang |
| A | Braun | Markus | 120000,00 | Fachrichtung: Orthopaedie |
| Summe der Jahresgehaelter: | | | 6535000,00 | |

Tipp: Für die Ausgabe der Tabelle ist die Verwendung von `String.format(...)` bzw. `System.out.printf(...)` hilfreich.

c) Mannschaftsverwaltung und Erzeugung von Mitgliedern über die Konsole

Erweitern Sie das Programm um eine Klasse, die eine Mannschaft repräsentiert. Die Mannschaft soll einen Namen und bis zu 30 Mitglieder besitzen. Erzeugen Sie anschließend in der Testklasse ein Mannschafts-Objekt und fügen hier die bereits erzeugten Mitglieder hinzu. Darüber hinaus soll die Konsolen-Ausgabe um den Namen der Mannschaft erweitert werden.

Ermöglichen Sie dem Benutzer per Konsole weitere Mitglieder einzugeben:



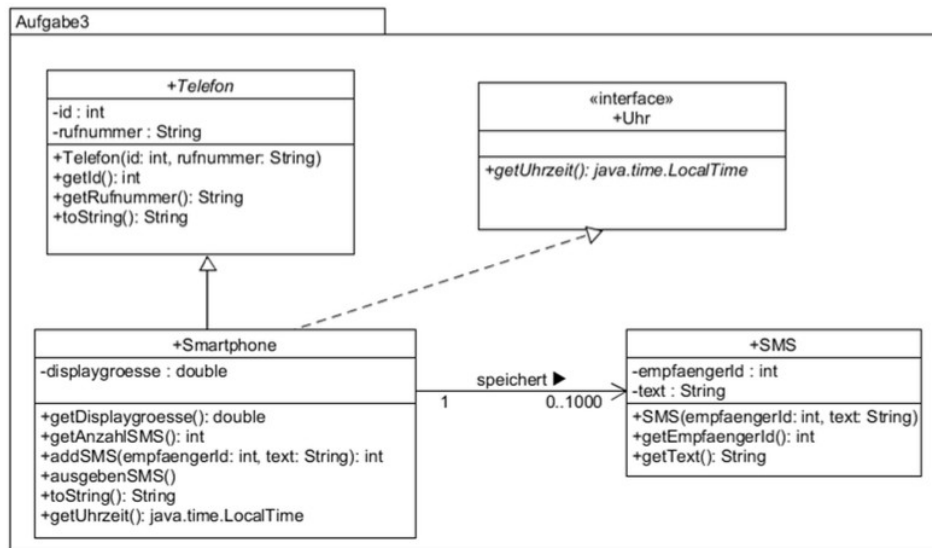
Mannschaftsverwaltung [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/bin/java (07.02.2017, 14:51:27)

Mannschaft: Ball-Verein Borussia Dortmund 09

| Typ | Name | Vorname | Jahresgehalt | Sonstiges |
|----------------------------|------------|------------------|--------------|---|
| S | Goetze | Mario | 2185000,00 | Position: Mittelfeld / Spieleinsätze: 5 |
| S | Aubameyang | Pierre-Emerik | 2245000,00 | Position: Sturm / Spieleinsätze: 17 |
| S | Reus | Marco | 2210000,00 | Position: Sturm / Spieleinsätze: 10 |
| S | Sokratis | Papastathopoulos | 2285000,00 | Position: Abwehr / Spieleinsätze: 25 |
| S | Buerki | Roman | 2205000,00 | Position: Tor / Spieleinsätze: 9 |
| T | Tuchel | Thomas | 1980000,00 | Lieblingsspieler: Pierre-Emerik Aubameyang |
| T | Michels | Arno | 1980000,00 | Lieblingsspieler: Papastathopoulos Sokratis |
| A | Braun | Markus | 120000,00 | Fachrichtung: Orthopaedie |
| Summe der Jahresgehaelter: | | | 15210000,00. | |

Weitere Aktionen:
 (A)rzt hinzufuegen
 (S)pieler hinzufuegen
 (T)rainer hinzufuegen
 (B)enden
 Ihre Eingabe:

Gegeben sei das folgende Klassendiagramm.



Erstellen Sie in Eclipse ein neues Projekt mit dem Namen `EidP_VL14`.

- a) Übertragen Sie das UML-Klassendiagramm in Java-Klassen und Schnittstellen. Ergänzen Sie die Klasse `Smartphone` um einen sinnvollen Konstruktor und Attribute zur Verwaltung der Assoziation.

Die Methoden der Klasse `Smartphone` sollen folgendes leisten:

- `getDisplaygroesse`: gibt die Displaygröße als Ergebnis zurück
- `getAnzahlSMS`: gibt die Anzahl gespeicherter SMS als Ergebnis zurück
- `addSMS`: fügt eine SMS an einer freien Position ein. Falls eingefügt werden konnte, ist der Rückgabewert die Position, an der eingefügt wurde, sonst -1.
- `ausgebenSMS`: gibt für die gespeicherten SMS jeweils die EmpfaengerId und den Text auf der Konsole aus.
- `toString`: fasst alle Attributwerte (außer zur Verwaltung der Assoziation) durch Leerzeichen getrennt zu einer Zeichenkette zusammen.
- `getUhrzeit`: gibt die aktuelle Uhrzeit als Ergebnis zurück.

Hinweise:

- Die Methode `now` der Klasse `LocalTime` im Paket `java.time` liefert die aktuelle Uhrzeit und ist wie folgt deklariert: `static LocalTime now()`.
- Außer `String` und `LocalTime` dürfen keine Klassen der Java-Klassenbibliothek verwendet werden!

- b) Programmieren Sie eine Test-Klasse mit einer `main`-Methode. Es sollen mindestens zwei `Smartphone`-Objekte angelegt werden, denen mehrere SMS zugeordnet sind. Geben Sie anschließend die Attributwerte der `Smartphone`-Objekte, die gespeicherten SMS sowie die Uhrzeit auf der Konsole aus.