Hochschule Furtwangen
Fakultät Informatik
Prof. Dr. Lothar Piepmeyer

Praktikum ‚Objektorientierte Programmierung'

Aufgabenblatt 10

Aufgabe 1:

a. Definieren Sie eine Klasse `Person` mit den (privaten) Attributen `id` vom Typ `int`, sowie `last_name` und `first_name` vom Typ `string`. Entwickeln Sie einen geeigneten Konstruktor und definieren Sie Methoden zum Lesen der Attribute. Benötigen Sie weitere Konstruktoren oder Operatoren?

b. Leiten Sie aus `Person` eine Klasse `Guest` mit den zusätzlichen (privaten) Attributen `days` (vom Typ `int`) und `room_rate` (vom Typ `int`) ab. Entwickeln Sie einen geeigneten Konstruktor und definieren Sie auch für die neuen Attribute die öffentlichen Methoden zum Lesen der Daten.

c. Definieren Sie für die Klasse `Guest` eine öffentliche Methode `check` (im Sinne von ‚Rechnung'), die den Wert `days*room_rate` als Rechnungsbetrag liefert.

d. Testen Sie *alle* Methoden der Klasse `Guest`.

e. Können Sie ein Objekt vom Typ `Guest` einer Variablen vom Typ `Person` zuweisen? Ist es überhaupt sinnvoll den Typ `Guest` von `Person` abzuleiten?

Aufgabe 2:

Ergänzen Sie Ihre Lösung aus Aufgabe 1 wie folgt:

a. Leiten Sie von der Klasse `Person` die Klassen `Employee` ab. Die Klasse `Employee` enthält die öffentliche Methode `salary`, die einfach `0` zurückgibt.

b. Leiten Sie von der Klasse `Employee` die Klasse `Worker` ab. Die Klasse `Worker` enthält die (privaten) ganzzahligen Attribute `hourly_rate` und `hours_worked` sowie die öffentliche Methode `salary`, die das Ergebnis `hourly_rate*hours_worked` liefert.

c. Das Gehalt eines Verkäufers setzt sich oft aus einem festen Gehalt (`pay`) und einem erfolgsbezogenen Anteil (`commission`) zusammen. Leiten Sie von der Klasse `Employee` die Klasse `Seller` ab. Die Klasse `Seller` enthält die (privaten) ganzzahligen Attribute `pay` und `commission`, sowie die öffentliche Methode `salary`, die `pay+commission` zurückgibt.
Entwickeln Sie für alle Klassen einen geeigneten Konstruktor und definieren Sie Methoden zum Lesen der Attribute.

d. Ist es überhaupt sinnvoll die Klassenhierarche so zu definieren?

e. Testen Sie diese drei Klassen. Prüfen Sie auch welche verschiedenen Typen zuweisungskompatibel sind. Prüfen Sie in den Fällen, in denen das möglich ist, das Ergebnis der Methode `salary`. Wenn die Variable `employee` vom Typ `Employee` und die Variable `worker` vom Typ `Worker` ist, dann ist beispielsweise die folgende Zuweisung korrekt:

```
employee=worker
```

Hier stellt sich die Frage, welche der beiden `salary` Methode dann bei

`employee.salary()`
aufgerufen wird. Entspricht die aufgerufene Methode Ihren Erwartungen?