Grundlagen der Programmierung

Marcel Lüthi
Andreas Morel-Forster
Universität Basel
Fachbereich Informatik
HS 23

Übung 10

Voraussetzung

- Ein JDK ist installiert.
- Installierte IDE, Visual Studio Code sowie die Plugins für Java und Gradle
- Wenn Sie die Vorlesung verpasst haben, dann empfehlen wir Ihnen die Unterlagen anzuschauen.
- Die Zip-Datei, die auch dieses Übungsblatt enthält, muss entpackt werden. Es enthält die gesamte Übungsumgebung. Schreiben Sie ihre Lösungen in die dafür vorgesehenen Dateien, wie in der jeweiligen Übungsaufgabe angegeben.

Wichtiger Hinweis

- *Achten Sie auf guten Programmierstil.*

Aufgabe 10.1 (Kassenbon, 4 Punkte)
In dieser Aufgabe werden Sie ein Programm schreiben, welches für einen Einkauf einen Kassenzettel erstellt. Sie finden im Verzeichnis `src/main/java/kassenbon` die Hauptklasse `Kasse`. An dieser Datei sollten Sie nichts ändern. Ziel ist es nun, die benötigen Klassen `Kassenbon`, `Artikel` und `Adresse` zu erstellen, um folgende Ausgabe zu erhalten:

```
|==========================|
|    Herbstmesse Basel     |
|        Uni Basel         |
|      Petersplatz 1       |
|        4001 Basel        |
|==========================|

Marroni       2 x  5.40
                        10.80
Magebrot      5 x  1.10
                         5.50
Glühwein      2 x  6.00
                        12.00

----------------------------
Total                   28.30
============================
```

- Erstellen Sie die Dateien für die Klassen `Kassenbon`, `Artikel` und `Adresse`.
- Leiten Sie aus der Hauptklasse ab, welche Felder Sie in den jeweiligen Klassen benötigen.
- Schreiben Sie in jeder Klasse einen Konstruktor, der diese mit den übergebenen Werten füllt.
- Fügen Sie der Klasse `Kassenbon` eine Liste `artikelliste` vom generischen Typen `ArrayList` hinzu, die Artikel halten kann. Sie müssen dazu folgenden Teil der API importieren: `import java.util.ArrayList;`.
- Fügen Sie der Klasse `Kassenbon` eine Methode `add` hinzu, um der `artikelliste` einen zusätzlichen `Artikel` hinzuzufügen. Suchen Sie die benötigte Methode, um diesen `Artikel` in der `ArrayList` hinzuzufügen, in der API-Dokumentation.
- Fügen Sie den Klassen `Kassenbon`, `Artikel` und `Adresse` je eine Methode `print` hinzu, diese darf noch leer sein.
- Ihr Gesamt-Programm sollte nun bereits kompilieren, sie müssen dazu nur das Kompilieren der Hauptklasse ausführen.
- Fügen Sie der Klasse `Artikel` eine Methode `getPrice` hinzu, welche den Preis dieses Postens zurückgibt.
- Schreiben Sie nun die `print`-Methoden für `Artikel` und `Adresse` - achten Sie in einem ersten Schritt nur auf den Inhalt, die Formatierung erfolgt später.
- Schreiben Sie nun auch die `print`-Methode für `Kassenbon` diese soll die `print`-Methoden für `Artikel` und `Adresse` aufrufen und das Total berechnen.
- Die Ausgabe sollte nun inhaltlich gleich sein wie die obige Vorlage.
- Versuchen Sie nun die Formatierung der Vorlage anzupassen. Hilfreich ist dazu die Funktion `String.format`.

Hinweis: Für diese Aufgabe testen die automatisierten Tests nur, dass alle Methoden vorhanden sind, und nicht ob diese richtig implementiert sind..

Aufgabe 10.2 (Generische Datenstrukturen, 3 + 2 + 1 Punkte)

In dieser Aufgabe sollen generische Datenstrukturen implementiert werden.
Als erstes soll eine generische, sortierte verkettete Liste geschrieben werden. Die Liste soll folgende Eigenschaften haben:

- Die Klasse soll in der Datei `src/main/java/generics/SortedList.java` geschrieben werden.
- Die Klasse `SortedList` soll mittels Java-generics Elemente von generischem Typ `T` verwalten.
- Elemente vom verwalteten Typ `T` müssen das Interface `Comparable` implementieren. (Das Comparable Interface ist Teil von der Java Standardbibliothek.)
- Die Klasse soll eine öffentliche Methode `insert` besitzen, welches ein Element vom Typ `T` entgegennimmt und an der richtigen Stelle in der sortierten Liste einfügt.
- Die Elemente sollten in aufsteigender Reihenfolge sortiert werden.
- Die Klasse soll eine öffentliche Methode `print` anbieten, welche alle Elemente auf der Konsole (mit `System.out.println`) ausgibt.
- Die Klasse soll eine Methode `size` haben, welche die Anzahl Elemente in der Liste ausgibt.
- Die Klasse muss eine Methode `toArray` anbieten, welche die Elemente der Liste in ein zu übergebendes Array vom Typ `T[]` der richtigen Grösse kopiert.
- Die Tests sollen erfolgreich durchlaufen, wenn Sie den Testcode in `src/test/java/generics/SortedListTests.java` einkommentieren.

Schreiben Sie dann eine Klasse `Pair`, welche Paare beliebigen Typs speichern kann.

- Die Klasse soll in der Datei `src/main/java/generics/Pair.java` geschrieben werden.
- Die Klasse *Pair* soll generisch sein. Das Erste Element, mit den Namen *first* soll vom generischen Type `T` und das zweite, mit den Name *second* vom Typ `U` sein.
- Die Klasse soll die `toString` Methode überschreiben, welche das Tupel zu enem String der Form `(first, second)` konvertiert.
- Implementieren Sie das Interface Comparable. Dabei sollen für zwei gegebene Paare `p1` und `p2` gelten, dass `p1` $<$ `p2` wenn `p1.first` $<$ `p2.first` oder `p1.first = p2.first` und `p2.second` $<$ `p2.second` gilt.

Schreiben Sie ein Testprogramm.

- Das Testprogramm soll in die Klasse `src/main/java/generics/Main.java` geschrieben werden.
- Es soll eine sortierte Liste von Paaren vom Typ `Pair<String, Integer` erstellen.
- In dieser Liste sollen die Paare `("aac", 5), ("aac", 3)` und `("aaa", 3"` gespeichert und richtig sortiert werden.

Abgabe Erstellen Sie eine Zip-Datei der gesamten Übungsumgebung (also des Verzeichnisses `uebung010`) und laden Sie dieses auf Adam hoch.