Einstieg in Java und OOP

Christian Silberbauer

Übungsblatt 2

Das Unternehmen Staubsauger AG hat deutschlandweit Vertriebsniederlassungen. Es möchte über Umsätze der einzelnen Bereiche Statistiken erstellen. Die Umsätze gestalten sich wie folgt:

Bereich	Umsatz
Bremen	5.000,00 €
Dresden	9.000,00 €
Frankfurt	12.000,00 €
Hamburg	1.000,00 €
Hannover	8.000,00 €
Köln	3.000,00 €
Leipzig	3.000,00 €
München	4.000,00 €
Potsdam	5.000,00 €
Stuttgart	8.000,00 €

Aufgabe 1

Definieren Sie einen Array `umsaetze` und schreiben Sie Javacode, um die Einträge der obigen Tabelle dem Array elementweise hinzuzufügen. Die einzelnen Elemente sind vom Typ `BereichsUmsatz`, dessen Implementierung nachfolgend abgebildet ist:

```
class BereichsUmsatz {
      String bereich;
      double umsatz;
}
```

Aufgabe 2

Geben Sie Code an, um die Bereichsumsätze wie folgt auf der Konsole auszugeben:

```
Bremen (5000.0)
Dresden (9000.0)
Frankfurt (12000.0)
Hamburg (1000.0)
Hannover (8000.0)
Köln (3000.0)
Leipzig (3000.0)
München (4000.0)
Potsdam (5000.0)
Stuttgart (8000.0)
```

Aufgabe 3

Implementieren Sie die Methode `avgUmsatz()`, welche den durchschnittlichen Umsatz über alle Bereiche ermittelt.

```
public static double avgUmsatz(BereichsUmsatz[] umsaetze) {
```

Aufgabe 4

Geben Sie Code an, um den durchschnittlichen Umsatz unter zu Hilfenahme der Methode `avgUmsatz()` auszugeben. Es soll dabei der in Aufgabe 1 definierte Array `umsaetze` verwendet werden und zu folgender Ausgabe führen:

```
Durchschnittlicher Umsatz: 5800.0
```

Aufgabe 5

Implementieren Sie die Methode `switchPosition()`, welche den `BereichsUmsatz` an Position `pos1` mit dem `BereichsUmsatz` an Position `pos2` vertauscht.

```
public static void switchPosition(BereichsUmsatz[] umsaetze,
                                  int pos1, int pos2) {
```

Aufgabe 6

Implementieren Sie die Methode `indexOfMinUmsatz()`, die den Index des Unternehmensbereiches mit dem kleinsten Umsatz ermittelt. Ignorieren Sie dabei, dass es möglicherweise mehrere Bereiche mit gleichem minimalen Umsatz geben könnte.

```
public static int indexOfMinUmsatz(BereichsUmsatz[] umsaetze) {
```

Aufgabe 7

Implementieren Sie die Methode `setFlop()`, welche im Array `umsaetze` den Eintrag mit dem höchsten Index mit demjenigen vertauscht, der den geringsten Umsatz hat. (Hinweis: die Methoden `switchPosition()` und `indexOfMinUmsatz()` sind hilfreich.)

```
public static void setFlop(BereichsUmsatz[] umsaetze) {
```

Aufgabe 8

Geben Sie Code an, der die Methode `setFlop()` anwendet und dann den Bereich an letzter Position wie folgt ausgibt:

```
Letzter: Hamburg (1000.0)
```

Aufgabe 9

Implementieren Sie die Methode `indexOfMaxUmsatz()`, die im Array `umsaetze` im Indexbereich zwischen `von` und `bis` den Index des Unternehmensbereiches mit dem größten Umsatz ermittelt. Ignorieren Sie dabei, dass es möglicherweise mehrere Bereiche mit gleichem maximalen Umsatz geben könnte.

```
public static int indexOfMaxUmsatz(BereichsUmsatz[] umsaetze,
                                    int von, int bis) {
```

Aufgabe 10

Implementieren Sie die Methode `set3Tops()`, welche an den ersten drei Positionen des Arrays `umsaetze` der Reihe nach diejenigen Bereiche mit dem größten Umsatz hinvertauscht. (Hinweis: die Methoden `switchPosition()` und `indexOfMaxUmsatz()` sind hilfreich.) Darüber hinaus sollen weitere Bereiche auf den Folgeplätzen gesetzt werden, wenn deren Umsatz mit dem Drittplatzierten übereinstimmt. Der Rückgabewert gibt die Anzahl der relevanten Indexe an (im Normalfall also 3).

```
public static int set3Tops(BereichsUmsatz[] umsaetze) {
```

Aufgabe 11

Geben Sie Code an, der die Methode `set3Tops()` anwendet und dann die Bereiche der erstplatzierten ausgibt. Berücksichtigen Sie, dass möglicherweise mehr als drei Bereiche relevant sind (siehe Aufgabe 9). Der Code soll zu folgender Ausgabe führen (Stuttgart und Hannover dürfen auch vertauscht angezeigt werden):

```
1. Frankfurt (12000.0)
2. Dresden (9000.0)
3. Stuttgart (8000.0)
3. Hannover (8000.0)
```

Aufgabe 12

Setzen Sie die Attribute der Klasse `BereichsUmsatz` auf `private` und definieren Sie get- und set-Methoden für beide Attribute. Die set-Methode des `umsatz`-Attributs soll dabei verhindern, dass ein negativer Umsatz angegeben werden kann. Definieren Sie zudem einen Default-Konstruktor (ohne Parameter) und einen Initialisierungskonstruktor mit beiden Attributen als Parameter. Denken Sie bitte daran, dass auch über den Initialisierungskonstruktor kein negativer Umsatz gesetzt werden darf. Schreiben Sie zudem eine toString()-Methode, die den Bereich als String im folgenden Format zurückgibt:

<<bereich>> (<<umsatz>>)

Aufgabe 13

Ändern Sie die Implementierung der bisherigen Lösungen derartig ab, dass sie zur neuen `BereichsUmsatz`-Klasse kompatibel ist. Verwenden Sie zum Anlegen der Objekte den Initialisierungskonstruktor. Verwenden Sie zur Ausgabe die `toString()`-Methode. Nutzen Sie die Tatsache aus, dass die `toString()`-Methode von `System.out.println()` aufgerufen wird, wenn Sie lediglich die Objektreferenz angeben.

Aufgabe 14

Erklären Sie, warum in Aufgabe 11 die Bereiche „Stuttgart“ und „Hannover“ in anderer Reihenfolge aufgeführt sind als im ursprünglichen Array.