INLOOP Softwaretechnologie, Classes and Objects - Taxi

Henry Haustein

vollständiger Code

Datei `Human.java`

```java
public class Human {
  private String forname;
  private String name;

  public Human(String forname, String name) {
    this.forname = forname;
    this.name = name;
  }

  public String getName() {
    return name;
  }

  public String getForename() {
    return forname;
  }

  public String toString() {
    return forname + " " + name;
  }
}
```

Datei `Taxi.java`

```java
public class Taxi {
  private Human driver;
  private Human[] passengers;
  private int numberGuests;

  public Taxi(Human driver) {
    this.driver = driver;
    this.passengers = new Human[4];
    this.numberGuests = 0;
  }

  public String getDriverName() {
```

```
13      return driver.getForename() + " " + driver.getName();
14    }
15
16    public void add(Human passenger) {
17      if(numberGuests >= 4) {
18        System.out.println("We are sorry, " + passenger + ". The
              taxi is full.");
19      }
20      else {
21        for(int i = 0; i <= numberGuests; i++) {
22          if(passengers[i] == null) passengers[i] = passenger;
23        }
24        numberGuests++;
25        System.out.println(passenger + " gets in.");
26      }
27    }
28
29    public String toString() {
30      if(numberGuests == 0) {
31        return "This is the taxi of " + driver + ". He takes nobody
              along.";
32      }
33      if(numberGuests == 1) {
34        return "This is the taxi of " + driver + ". He takes " +
              passengers[0] + " along.";
35      }
36      if(numberGuests == 2) {
37        return "This is the taxi of " + driver + ". He takes " +
              passengers[0] + " and " + passengers[1] + " along.";
38      }
39      if(numberGuests == 3) {
40        return "This is the taxi of " + driver + ". He takes " +
              passengers[0] + ", " + passengers[1] + " and " +
              passengers[2] + " along.";
41      }
42      else {
43        return "This is the taxi of " + driver + ". He takes " +
              passengers[0] + ", " + passengers[1] + ", " + passengers
              [2] + " and " + passengers[3] + " along.";
44      }
45    }
46
47    public Human[] allGetOut() {
48      if(numberGuests == 0) {
49        Human[] returnArray = new Human[0];
50        numberGuests = 0;
51        return returnArray;
52      }
53      else {
54        Human[] oldpassengers = new Human[numberGuests];
55        for(int i = 0; i < numberGuests; i++) {
```

```
56            oldpassengers[i] = passengers[i];
57          }
58          numberGuests = 0;
59          return oldpassengers;
60        }
61      }
62  }
```

Erklärung

Die Klasse `Human` ist ziemlich einfach, an der Klasse `Taxi` habe ich mir am Anfang ziemlich den Kopf zerbrochen[1]. Ich initialisiere das Array `passengers` einfach schon mit 4 Plätzen, die standardmäßig mit `null` belegt werden. Gleichzeitig speichere ich die Anzahl der Gäste im Taxi ab.

Das Hinzufügen von neuen Gästen hab ich wieder kompliziert gelöst, indem ich mit einer for-Schleife nach dem ersten freien Platz im Array gesucht habe, aber eigentlich müsste das auch so gehen:

```
1  public void add(Human passenger) {
2          if(numberGuests >= 4) {
3                  System.out.println("We are sorry, " + passenger +
                       ". The taxi is full.");
4          }
5          else {
6                  passengers[numberGuests] = passenger;
7                  numberGuests++;
8                  System.out.println(passenger + " gets in.");
9          }
10 }
```

Auch die Funktion `toString()` hätte man sicherlich schöner lösen können. Aber da muss man aufpassen, wann man ein Komma für die Aufzählung ausgibt und an welche Stelle das "and" muss. Das war mir dann doch zu viel Aufwand, sowass ich hier einfach eine Reihe an if-Anweisungen hingeschrieben habe. Beim Erstellen dieses Dokuments ist mir dann noch eingefallen, dass es dafür eigentlich die `switch`-Verzweigung gibt...

Besonders schlecht im Programmierstil ist dann die `allGetOut()`-Funktion. Hätte ich das `passenger`-Array am Anfang versucht dynamisch zu implementieren, sodass man zuerst mit einem Array der Größe 0 startet und bei jedem weiteren einsteigenden Gast dass Array um eins vergrößert, hätte man hier nicht solche Verrenkungen machen müssen. Aber egal, wenn im Taxi keine Gäste sind, so erstellte ich einfach ein Array der Größe 0 und gebe das zurück. Wenn Gäste im Taxi sitzen, so erstellte ich ein neues Array `oldpassengers`, was ich auch gleich in der richtigen Größe initialisieren kann. Ich iteriere mich dann durch dass `passengers`-Array durch und kopiere die Einträge dieses Arrays an die entsprechende Stelle im `oldpassengers`-Array[2]. Zum Schluss setze ich noch den Gäste-Counter wieder auf 0 und gebe `oldpassengers` zurück.

[1] Ich programmiere gerne in Python, und dort kann man Array dynamisch in der Größe ändern. Das ist sehr sinnvoll, insbesondere dann wenn die Größe des Arrays sich ständig ändern soll. Das ist auch bei dieser Aufgabe so. Man kann das anders lösen, aber ich habe mir dann einen einfacheren, aber leider auch einige Verrenkungen im Quelltext auslösenden, Ansatz überlegt.

[2] Die offensichtliche Lösung, `oldpassengers = passengers`, funktioniert nicht, da die Arrays in den meisten Fällen nicht die selbe Größe haben, z.B. dann, wenn weniger als 4 Gäste im Taxi sitzen.