

Übung zur Vorlesung *Einführung in die Informatik 2 für Ingenieure (MSE)*

Alexander van Renen (renen@in.tum.de)

<http://db.in.tum.de/teaching/ss20/ei2/>

Blatt Nr. 4

Dieses Blatt wird am Montag, den 18. Mai 2020 besprochen.

Aufgabe 1: Stack mit verketteter Liste

Implementieren Sie einen Stack für `int`-Werte mithilfe einer verketteten Liste. Der Stack sollte dabei die Methoden `push()` und `pop()` anbieten, mit denen Elemente auf den Stack gelegt bzw. von ihm heruntergenommen werden. Eine mögliche Modellierung ist in Abbildung 1 gezeigt. Wie könnte man diesen Stack generischer machen?

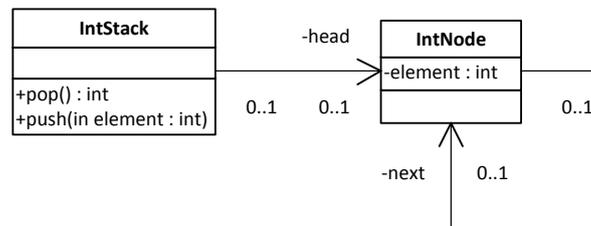


Abbildung 1: Stack mit verketteter Liste

Aufgabe 2: Don't repeat yourself (DRY)

Auf der Webseite¹ finden Sie die Klasse `Warenhaus.java`.

- Nennen Sie mindestens zwei Gründe, warum die Modellierung nicht optimal ist – beispielsweise, wenn sich der Mehrwertsteuersatz mal wieder ändern sollte.
- Implementieren Sie den Aufzählungstyp `Mehrwertsteuersatz`, der zwei Werte für den normalen und den vergünstigten Mehrwertsteuersatz hat.
- Ziehen Sie die Gemeinsamkeiten der Produkte in eine gemeinsame Oberklasse `Produkt`. Führen Sie dafür eine Methode ein, die für ein Produkt angibt, ob die vergünstigte Mehrwertsteuer anwendbar ist. Verwenden Sie diese Methode bei der Preisberechnung.
- Führen Sie die Klasse `Einkaufskorb` ein, die eine Menge von Produkten verwaltet und deren Gesamtpreis bestimmen kann. Verwenden Sie diese Klasse in der `main`-Methode des `Warenhauses`.

¹Hier: <http://db.in.tum.de/teaching/ss20/ei2>

Aufgabe 3: Werte vs. Objekte

In dieser Aufgabe schauen wir uns noch einmal den Unterschied zwischen Werten und Objekten an. Überlegen Sie sich, was die Ausgabe sein sollte und warum.

```
1 class PersonA {
2     int alter;
3     public PersonA(int alter) {
4         this.alter = alter;
5     }
6 }
7
8 class PersonB {
9     Alter alter;
10    public PersonB(Alter alter) {
11        this.alter = alter;
12    }
13 }
14
15 class Alter {
16    public int jahre;
17    public Alter(int jahre) {
18        this.jahre = jahre;
19    }
20 }
21
22 class WerteVsObjekte {
23    public static void main(String[] args) {
24        PersonA mickeyA = new PersonA(50);
25        PersonA donaldA = new PersonA(55);
26
27        donaldA.alter = mickeyA.alter;
28        mickeyA.alter = 51;
29        System.out.println("MickeyA_ist_" + mickeyA.alter);
30        System.out.println("DonaldA_ist_" + donaldA.alter);
31
32        System.out.println("==");
33
34        PersonB mickeyB = new PersonB(new Alter(50));
35        PersonB donaldB = new PersonB(new Alter(55));
36
37        donaldB.alter = mickeyB.alter;
38        mickeyB.alter.jahre = 51;
39        System.out.println("MickeyB_ist_" + mickeyB.alter.jahre);
40        System.out.println("DonaldB_ist_" + donaldB.alter.jahre);
41    }
42 }
```

Aufgabe 4: Operationen

- (a) Welche drei Gruppen von Operationen haben wir eingeführt und welche Eigenschaften machen diese Gruppen aus?
- (b) Warum sollte eine Beobachterfunktion das Objekt nicht verändern?
- (c) Welcher Rückgabebetyp macht bei einer Beobachterfunktion keinen Sinn?
- (d) Welche Access Modifier gibt es und was bedeuten sie?