TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

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

Alexander van Renen (renen@in.tum.de)
http://db.in.tum.de/teaching/ss17/ei2/

Blatt Nr. 4
Dieses Blatt wird am Montag, den 29. Mai 2017 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.

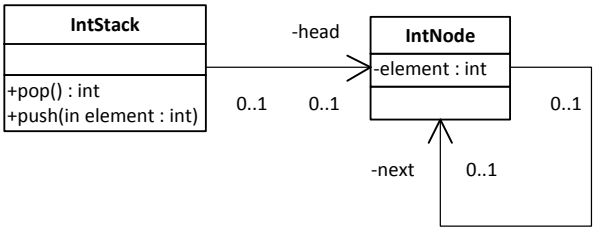

Abbildung 1: Stack mit verketteter Liste

Aufgabe 2: Türme von Hanoi

In dieser Aufgabe implementieren wir das bekannte Knobelspiel „Die Türme von Hanoi“.[1]. Dabei verwenden wir den Stack aus Aufgabe 1 um die drei Türme zu modellieren.

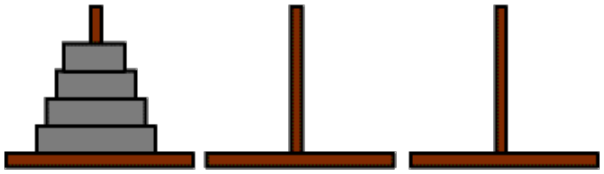

Abbildung 2: Die Türme von Hanoi mit vier Scheiben

Überlegen Sie sich einen rekursiven Algorithmus um alle Scheiben von Position 1 auf Position 3 zu verschieben. Dabei darf immer nur die oberste Scheibe eines Turms auf einen anderen Turm gelegt werden, wenn dessen oberste Scheibe größer ist (oder an der Zielposition gar keine Scheiben sind).

Tipp: Damit die unterste Scheibe zur Position 3 bewegt werden kann, muss der darüber liegende Turm erstmal auf Position 2 verschoben werden. Dann kann man die größte Scheibe verschieben und dann den kleineren Turm von Position 2 auf die große Scheibe verschieben.

[1] Für eine ausführliche Beschreibung siehe Wikipedia: `http://de.wikipedia.org/wiki/Türme_von_Hanoi`. Das ganze kann auch auf unzähligen Seiten ausprobiert werden, z.B.: `http://vornlocher.de/tower.html`

Überlegen Sie sich wie dieses Prinzip auch für das Verschieben der kleineren Türme zum Tragen kommt.

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. Überprüfen Sie Ihre Antwort indem Sie das Programm ausführen.

```
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 Rückgabetyp macht bei einer Beobachterfunktion keinen Sinn?

(d) Welche Access Modifier gibt es und was bedeuten sie?