Prof. Dr. Stefan Brass
Dipl.-Inform. Annett Thüring
Dipl.-Inform. Steffen Schiele

Martin-Luther-Universität Halle-Wittenberg
Institut für Informatik
Wintersemester 2013/ 2014

Übungsblatt 6: Objektorientierte Programmierung

Ausgabe: 22.11.2013 Abgabe: 29.11.2013

Aufgabe 1: Duplikatvermeidung (8 Punkte)

Schreiben Sie ein Java-Programm, welches

- solange Werte einliest, bis eine negative Zahl eingegeben wird,
- diese (nicht-negativen) Werte dabei in ein Feld der Größe 20 speichert, falls diese noch nicht vorkommen,
- eine Fehlermeldung ausgibt (das Programm aber nicht beendet!), wenn versucht wird mehr als 20 verschiedene Werte einzugeben und
- am Ende die im Feld gespeicherten Zahlen (und nur diese) kommagetrennt hintereinander ausgibt.

Es dürfen somit in dem Feld keine Duplikate gespeichert werden und es können (müssen aber nicht!) bis zu 20 Werte abgespeichert werden. Gehen Sie davon aus, dass der Benutzer nur ganze Zahlen eingibt (aber möglicherweise auch mehr als 20).

Aufgabe 2: Operatorbäume (4 Punkte)

Zeichnen Sie für die folgenden Wertausdrücke die Operatorbäume:

(a) `i * (j - 2 * k)`

(b) `i - 10 > 0 || i < 0`

(c) `i - j + k == 0 || i != j && k != 0`

(d) `i - i % j * j`

Aufgabe 3: Operatoren (keine Abgabe)

Was gibt das folgende Programm aus? Begründen Sie!

```
1  public class Ausgabe
2  {
3      public static void main(String[] args)
4      {
5          int i = 12;
6          int j = 5;
7          double x = 7.6;
8
9          x = x + i / j;
10         System.out.println("x: " + x);
11         i = j / 2 * 3;
12         System.out.println("i: " + i);
13         j = i = i - j;
14         System.out.println("j: " + j);
15     }
16 }
```

Aufgabe 4: Typfehler (keine Abgabe)

Welche der folgenden Ausdrücke enthalten Typ-Fehler? Erklären Sie diese!
Deklarationen: `int i, j; boolean b; double d;`
(Initialisierungen sind bereits erfolgt.)

- `i = (j==0)`
- `b && i = j`
- `d = i - j`
- `b || i < 5`
- `j = d - 2`
- `j == 0 && d < 2.0`

Aufgabe 5: Memory (keine Abgabe)

Programmieren Sie eine Ein-Spieler-Variante des bekannten Memory-Spiels. Zuerst gibt der Benutzer an, wie viele verschiedenen Buchstaben vorkommen. Anschließend sollen diese in einem Feld so zufällig verteilt werden, dass jeder Buchstabe genau zweimal vorkommt (die Buchstaben entsprechen somit den Memory-Karten).

Nun wird der Spieler aufgefordert zwei Karten aufzudecken (dazu muss er den Index der ersten und den Index der zweiten aufzudeckenden Karte angeben). Stimmen diese überein, so bleiben diese für den gesamten Spielverlauf aufgedeckt (es wird also der entsprechende Buchstabe angezeigt). Nicht aufgedeckte Karten werden durch ein * symbolisiert. Sobald der Spieler alle Paare gefunden hat, wird noch ausgegeben, wie viele Versuche er benötigte. Damit es übersichtlich ist, welche beiden Karten gerade aufgedeckt wurden und welche Karten zu bereits gefundenen Paaren gehören, soll unter jeder gerade aufgedeckten Karten das Symbol ^ ausgegeben werden.

Ein möglicher Ablauf könnte wie folgt aussehen:

```
Wie viele verschiedene Zeichen? 4
Decke folgendes Feld auf: 1
Decke folgendes Feld auf: 2
*DC*****
 ^^
Decke folgendes Feld auf: 0
Decke folgendes Feld auf: 3
A**B****
^  ^
Decke folgendes Feld auf: 4
Decke folgendes Feld auf: 5
****DB**
    ^^
Decke folgendes Feld auf: 3
Decke folgendes Feld auf: 5
***B*B**
   ^ ^
Decke folgendes Feld auf: 6
Decke folgendes Feld auf: 7
***B*BAC
      ^^
```

```
Decke folgendes Feld auf: 0
Decke folgendes Feld auf: 6
A**B*BA*
^     ^
Decke folgendes Feld auf: 2
Decke folgendes Feld auf: 4
A*CBDBA*
  ^ ^
Decke folgendes Feld auf: 2
Decke folgendes Feld auf: 7
A*CB*BAC
  ^    ^
Decke folgendes Feld auf: 1
Decke folgendes Feld auf: 4
ADCBDBAC
 ^  ^
Sie haben 9 Versuche benoetigt!
```