25. Oktober 2010

Übungen zur Vorlesung

# Einführung in die Informatik: Programmierung und Software-Entwicklung

Musterlösung Blatt 2

**Aufgabe 2-1. (4 Punkte)** Folgende Java-Fragmente sind fehlerhaft. Welche Fehler enthalten sie? Können Sie die Fehler korrigieren?
**Hinweis:** Lassen Sie sich vom Java-Compiler helfen, indem Sie die Java-Fragmente in eine `main`-Methode schreiben und das Programm vom Compiler überprüfen lassen.

1.
```
int x = 3;
System.out.println(2 * x); // gibt das Doppelte von x aus
```

2.
```
System.out.println("Zeile 1");
System.out.println(" Zeile 2");
System. out.println("Zeile 3");
```

3.
```
int x = 3;
String s = "x = 12";
System.out.println(s) ;
```

4.
```
int x = 3;
int y = 5;
// vertausche die Werte von x und y
System.out.println("x = " + x + " y = " + y);
int z;
z = x; x = y; y = z;
System.out.println("x = " + x + " y = " + y);
```

5.
```
double x = 3.0;
System.out.println("x = " + x);
System.out.println () ;   // neue Zeile
x = x + 1;
System.out.println("x = " + x); // x soll um 1 erhöht sein
```

6.
```
S tring s = "a";
String t = s + 3;
String u = "3" ;
System.out.println(s + t + u);
```

7.
```
int x1 = 1;
int x2 = 2;
int x3 = 3;
System.out.println(x1 + x2 + x3 );
```

**Aufgabe 2-2. (4 Punkte)** Gegeben seien die folgenden Booleschen Java-Ausdrücke:

1. `!(b1 & b2)`
2. `!(b1 | b2)`
3. `(!b1) & (!b2)`
4. `(!b1) | (!b2)`

Wir nennen zwei solche Booleschen Ausdrücke *äquivalent*, falls sie den gleichen Wert haben, egal was wir für `b1` und `b2` einsetzen.

Welche der obigen vier Booleschen Ausdrücke sind zueinander äquivalent?
**Hinweis:** Eine mögliches Vorgehen ist, die Wertetabellen für die einzelnen Ausdrücke aufzustellen und diese dann zu vergleichen. Sollten Sie sich unsicher über den Wert eines Ausdrucks sein, so können ein Java-Programm schreiben, um die verschiedenen Werte der Ausdrücke zu überprüfen. Für den ersten Ausdruck könnte man folgendermaßen beginnen:

```
System.out.println(!(false & false));
System.out.println(!(true & false));
...
```

**Lösung:** Äquivalent sind 1 und 4, sowie 2 und 3. [1]

**Bewertung:** Jeweils ein Punkt für jede korrekt identifizierte Äquivalenz; sowie jeweils ein Punkt für eine korrekte Begründung (z.B. vollständige Wertetabelle).

**Aufgabe 2-3. (4 Punkte)** Den Durchschnitt zweier Zahlen $x$ und $y$ kann man bekanntlich mit der Formel $(x + y)/2$ berechnen. In Java ist das analog mit dem Ausdruck `(x+y)/2` möglich, wie man mit folgendem Programmfragment nachvollziehen kann:

```
int x = 2;
int y = 7;
int d = (x+y)/2;
System.out.println(d);
```

Der Durchschnitt `d` wird hier ganzzahlig berechnet, das Ergebnis ist gerundet.

Betrachten Sie nun folgendes Programmfragment:

```
int x = 2147483646;
int y = 2;
int d = (x+y)/2;
System.out.println(d);
```

Welches Verhalten beobachten Sie? Wie können Sie dieses Verhalten erklären? Unter welchen Umständen kann man ein korrektes Ergebnis erwarten?

Ändern Sie den Ausdruck `(x+y)/2` so ab, dass der Durchschnitt wenigstens für alle *positiven* `int`-Werte `x` und `y` korrekt berechnet wird!

**Lösung:** Im zweiten Programmstück wird ein negativer Wert ausgegeben, obwohl der Wert eigentlich positiv sein müsste. Das Problem ist, dass $x + y$ zu groß ist, um als `int` gespeichert zu werden. Es kommt zu einem Überlauf.

[1] Siehe `http://de.wikipedia.org/wiki/De_Morgansche_Gesetze`

Richtiges Verhalten kann man erwarten, wenn die Zahl $x + y$ als `int`-Wert kodiert werden kann, d.h. $-2^31 \leq x + y \leq 2^31 - 1$

Eine Möglichkeit ist, den Durchschnitt als `long` zu berechnen:

```
int d = (int)(((long)x + (long)y) / 2L);
```

Es ist aber auch möglich, nur `int`-Berechnungen zu benutzen:

```
int d = x + (y-x)/2;
```

Das liefert für positive Zahlen immer das richtige Ergebnis.

ACHTUNG: Die folgende Variante ist *falsch*:

```
int d = x/2 + y/2;
```

da hier Rundungsfehler auftreten können, z.B. für $x = 1$ und $y = 3$ bekommt man als Ergebnis 1, und nicht 2.

**Bewertung:**

Ein Punkt für die korrekte Beobachtung, dass ein negatives Ergebnis ausgegeben wird; ein weiterer Punkt für die Erklärung, dass der Fehler auf einen Überlauf zurückzuführen ist; sowie bis zu zwei Punkte für die Korrektur des Problems.