technische universität
dortmund

Fakultät für Informatik
Lehrstuhl 14
Lars Hildebrand, Marcel Preuß,
Iman Kamehkhosh, Marc Bury, Diana Howey

Praktikum zu

Einführung in die Informatik für LogWIIngs und WiMas

Wintersemester 2015/16

Übungsblatt 6

Besprechung:
07.–11.12.2015
(KW 50)

Vorbereitende Aufgaben

Auf diesem Übungsblatt sollen Sie sich genauer mit den Eigenarten der Programmiersprache Java befassen.

Aufgabe 6.1: Lokale Variablen

In dieser Aufgabe sollen Sie sich mit der Sichtbarkeit von Variablen in Blöcken beschäftigen.

Sie haben Blöcke ({...}) bereits verwendet, um in bedingten Anweisungen und Schleifen die bedingte Ausführung oder Wiederholung von mehr als einer Operation durchzuführen. Blöcke können jedoch auch als eine allein stehende Kontrollstruktur verwendet werden, um Code zu strukturieren.

Geben Sie an, ob die folgenden Codefragmente vom Java Compiler akzeptiert werden würden. Falls nicht, geben Sie an, wo sich der Fehler verbirgt:

a)
```
public static void main(String[] args) {
    int m;
    if(2 > 17) {
        m = 4;
    }
    else {
        m = 9;
    }
    System.out.println("m: " + m);
}
```

b)
```
public static void main(String[] args) {
    int m;
    if(2 > 17) {
        int n = 7;
        m = 4;
    }
    else {
        m = 9;
    }
    System.out.println("m: " + m + " n: " + n);
}
```

---

c)
```
public static int doSomething(int y) {
    int z = 32;
    {
        int z = 2;
        y = y - z;
    }
    return y;
}
```

---

d)
```
public static void doSomethingElse(double x) {
    {
        double e = 18.0;
        x *= e + 4.3;
    }
    e = x / 2.0;
    System.out.println("e: " + e + " x: " + x);
}
```

---

e)
```
public static void doSomethingElseAgain(double x) {
    {
        double e = 18.0;
        x *= e + 4.3;
    }
    double e = x / 2.0;
    System.out.println("e: " + e + " x: " + x);
}
```

Präsenzaufgaben

Aufgabe 6.2: Umwandlung von Schleifen

In dieser Aufgabe sollen Sie Schleifen von einem Typ in einen anderen umwandeln.

a) Wandeln Sie die folgende Zählschleife in eine **while**-Schleife (kopfgesteuert) um.

```
int negative = 0;
for(int counter = -47; counter < 0; counter++) {
    negative++;
}
System.out.println("negative: " + negative);
```

b) Wandeln Sie die folgende **do-while**-Schleife (fußgesteuert) in eine **while**-Schleife um.

```
int n = 7;
int m = 256;
do {
    m = m - 20;
    n--;
} while(n > 0);
System.out.println("m :" + m);
```

c) Wandeln Sie die folgende **while**-Schleife in eine **for**-Schleife (Zählschleife) um.

```
int p = 0;
int q = 7;
int e = 1;
while(p < 15 && q < 9) {
    e = e + p * q - 2;
    p = p + 2;
}
System.out.println("e: " + e);
```

Aufgabe 6.3: Unterfunktionen und Subroutinen – Geschwindigkeit

In dieser Aufgabe sollen Sie sich mit der Deklaration und dem Aufruf statischer Methoden vertraut machen.

Bei der Konstruktion eines Geschwindigkeitsmessers für Automobile soll ein Programm die derzeitige Geschwindigkeit in Kilometern pro Stunde (km/h) ausgeben. Die Sensoren geben allerdings die zurückgelegte Strecke in Metern und die dabei vergangene Zeit in Sekunden an.

a) Da das Umrechnen von Geschwindigkeiten eine sehr allgemeine Aufgabe ist, die an vielen Stellen nützlich sein kann, wollen wir eine Funktion schreiben, die diese Umrechnung für uns übernimmt.

- Überlegen Sie sich, wie man eine Geschwindigkeit von m/s in km/h umrechnet.

- Legen Sie eine neue Klasse **SpeedSensor** im Paket **blatt06** an.
- Fügen Sie die **main**-Methode ein, in der Sie zwei Variablen vom Typ double anlegen: **currentMeters** und **currentSeconds**. Diese sollen die Sensorwerte repräsentieren. Denken Sie sich beliebige Werte dafür aus.
- Legen Sie innerhalb der SpeedSensor-Klasse eine statische Methode an:
  `public static double velocity`
  Diese soll zwei Parameter vom Typ **double** haben: **meters** und **seconds**.
- Bestimmen Sie anschließend die **Geschwindigkeit** in km/h und geben Sie diese mit Hilfe des **return**-Statements zurück.
- Geben Sie in der **main**-Methode mit folgendem Code die derzeitige Geschwindigkeit aus:

```
System.out.println("Current speed is: " +
            velocity(currentMeters, currentSeconds) + " km/h");
```

b) Jetzt wollen wir dem Fahrer auch noch den Bremsweg anzeigen. Sei v die aktuelle Geschwindigkeit in km/h, dann ist eine Näherung für den Bremsweg s in Metern:

$$s = \frac{1}{2}\left(\frac{v}{10}\right)^2$$

- Schreiben Sie eine Funktion **brakingDistance** vom Typ **double**, die die Sensorwerte **meters** und **seconds** als Parameter nimmt und den Bremsweg in Metern zurückgibt.
- Geben Sie den Bremsweg in der main-Funktion aus mit

```
System.out.println("Current braking distance is: " +
            brakingDistance(currentMeters, currentSeconds) + " m");
```

Aufgabe 6.4: Aufrufverhalten

In dieser Aufgabe sollen Sie sich mit dem Aufruf von statischen Methoden mit Parametern primitiver Datentypen vertraut machen.

Betrachten Sie folgendes Programm:

```
1  package blatt06;
2
3  public class Program {
4      public static void add5ToInt(int x) {
5          x = x + 5;
6          System.out.println("x in function add5ToInt: " + x);
7      }
8
9      public static void main(String[] args) {
10         int x = 8;
11
12         System.out.println("x before function call: " + x);
13         add5ToInt(x);
14         System.out.println("x after function call: " + x);
15     }
16 }
```

- Was erwarten Sie als Ausgabe des Programms?

- Was ist die Ausgabe des Programms?

- Was fällt Ihnen auf? Was ist der Grund für dieses Programmverhalten?

Ergänzende Aufgaben

Aufgabe 6.5: Programmstrukturierung

In dieser Aufgabe sollen Sie sich mit der Möglichkeit befassen, ein Programm durch Umstrukturierung in Subroutinen verständlicher zu machen.

Unser Aquarium ist kaputt gegangen und wir wollen ein neues kaufen. Dieses soll möglichst viel Wasser fassen. Zur Auswahl stehen drei Aquarien, deren Maße in **Höhe**, **Breite** und **Tiefe** angegeben sind.

- Markieren Sie zuerst, in welchem Teil des Programmes sich komplexe, sich wiederholende Strukturen befinden.
- Lagern Sie diese Strukturen in Subroutinen aus, indem Sie das Programm umschreiben. Legen Sie dazu eine neue Klasse mit dem Namen **BiggestAquarium** im Paket **blatt06** an.

```
1  package blatt06;
2
3  public class BiggestAquarium {
4      public static void main(String[] args) {
5          double firstHeight = 16.5, firstWidth = 27.5, firstDepth = 38.0;
6          double secondHeight = 20.0, secondWidth = 20.0, secondDepth = 20.0;
7          double thirdHeight = 40.2, thirdWidth = 22.5, thirdDepth = 18.5;
8
9          if((firstHeight * firstWidth * firstDepth >
10                 secondHeight * secondWidth * secondDepth) &&
11                 (firstHeight * firstWidth * firstDepth >
12                 thirdHeight * thirdWidth * thirdDepth)) {
13             System.out.println("Aquarium number 1 is the biggest");
14         }
15         else if((secondHeight * secondWidth * secondDepth >
16                 firstHeight * firstWidth * firstDepth) &&
17                 (secondHeight * secondWidth * secondDepth >
18                 thirdHeight * thirdWidth * thirdDepth)) {
19             System.out.println("Aquarium number 2 is the biggest");
20         }
21         else {
22             System.out.println("Aquarium number 3 is the biggest");
23         }
24     }
25 }
```