Lehrstuhl für Praktische Informatik IV
Holger Füßler
A5, 6, Raum B 219
68131 Mannheim
Telefon: (0621) 181–2605
Email: fuessler@informatik.uni-mannheim.de

Robert Schiele
B6, 29, Raum C0.04
68131 Mannheim
Telefon: (0621) 181–2214
Email: rschiele@uni-mannheim.de

UNIVERSITÄT MANNHEIM

Praktische Informatik I — 4. Übungsblatt
Wintersemester 2005/2006 — Abgabe: 23. November 2005

Allgemeine Bearbeitungshinweise

Verwenden Sie für die Programmieraufgaben die in den Aufgaben angegebenen Klassennamen und geben Sie den Klassen-Quellcode sowohl auf Papier als auch per e-mail bei Ihrem Tutor ab. Verwenden Sie javadoc-Kommentare zur Dokumentation. Bitte schreiben Sie für jede Aufgabe auf, wie lange Sie für die Bearbeitung der Aufgabe gebraucht haben. Seien Sie dabei möglichst ehrlich.

Aufgabe 1 — 10 Punkte

Die Methode `Math.random` liefert bei jedem Aufruf eine Zufallszahl zwischen 0 und 1 als `double`. Übernehmen Sie die Methode `wurf` aus den Vorlesungsfolien, welche mit Hilfe von `Math.random` eine gleichverteilte ganzzahlige Zufallszahl von 1 bis 6 zurückliefert, also einen Würfel simuliert.
Schreiben Sie ein Programm, welches diese Methode `anzahl` mal ausführt und zurückgibt, wie oft welche Zahl aufgetreten ist.
Schreiben Sie weiter eine Methode `histogram`, welche vom Programm zur Ausgabe der Ergebnisse in einem Histogramm aufgerufen wird.
Kam also zum Beispiel die 1 5-mal vor, die 2 7-mal, die 3 4-mal, die 4 7-mal, die 5 6-mal und die 6 5-mal, so sollte die Ausgabe etwa so aussehen:

```
1: =====
2: =======
3: ====
4: =======
5: ======
6: =====
```

Das Programm soll der Methode die Ergebnisse als Parameter übergeben.
Testen Sie Ihr Programm mindestens mit `anzahl = 10`, `anzahl = 100` und `anzahl = 400` und geben Sie das dabei entstandene Histogramm jeweils mit ab.
Nennen Sie die Klasse `WUERFELSPIEL`.

(keine Programmieraufgabe!) Betrachten Sie die folgende Klasse.

```
1  class alpha {
       static int beta;                              // 1
       static int gamma;                             // 2
       int delta;                                    // 3
       int epsilon;                                  // 4
6      static void zeta(int beta,                    // 5
                        int delta,                   // 6
                        int eta) {                   // 7
           System.out.println(beta);          // a
           System.out.println(gamma);         // b
11         System.out.println(delta);         // c
           System.out.println(epsilon);       // d
           System.out.println(eta);           // e
       }
       void iota(int beta,                           // 8
16             int delta) {                          // 9
           int eta = 42;                             // 10
           System.out.println(this.beta);     // f
           System.out.println(gamma);         // g
           System.out.println(delta);         // h
21         System.out.println(epsilon);       // i
           System.out.println(eta);           // j
       }
   }
```

Die Klasse enthält u.a. Variablendeklarationen, die durch Kommentare mit den Nummer 1-10 und Ausgaben, die durch Kommentare mit den Buchstaben Buchstaben a-j gekennzeichnet sind

Geben Sie für jede der Ausgabezeilen an, welchen Variablen-Inhalt sie ausgibt, d.h. in welcher Zeile die Variable deklariert ist, die hier ausgegeben wird. Entdecken Sie eine ungültige Zeile, so geben Sie an, welche Zeile ungültig ist.

Wenn Sie zum Beispiel der Überzeugung wären, dass die Zeile `a` („`System.out.println(beta);`“) den Inhalt der Variablen `beta` in der mit `1` markierten Zeile ausgibt, dann würden Sie zum Beispiel angeben: `a --> 1`

Begründen Sie ihre Zuordnungen kurz.

Aufgabe 3 — $\Sigma = 5$ Punkte

(keine Programmieraufgabe!) In den folgenden Code-Schnipseln sind jeweils ein oder mehrere Fehler versteckt. Markieren Sie diese(n) (oder geben Sie die Zeilennummer(n) an) und **erklären** Sie kurz, was der Fehler ist.

Aufgabe 3 a) — 3 Punkte

Listing i:

```
1  public class X {
     static final int i = 1;
     private int j = 0;

     static {
6      int p = 9;
       i = p;
       p = 3;
     }

11   public X(int p) {
       j = 0;
     }
   }
```

Listing ii:

```
1  public class X {
     static int i = 1;
     protected int j = 0;

     static {
6      int p = 9;
       i = p;
       j = p;
     }

11   public X(int p) {
       j = 0;
     }
   }
```

Listing iii:

```
1  public class X {
     static final int i;
     static int k;
     private int j = 0;

6    static {
       int p = 9;
       p = Math.PI;
     }

11   public X(int j) {
       this.j = 0;
       X.i = j;
       X.k = X.i;
     }
16 }
```

Aufgabe 3 b) 2 Punkte

Gegeben sei die Klasse Y mit dem u.a. Quellcode. Welche Methode (Zeilen 2 - 5) wird in den Zeilen 12 - 15 aufgerufen und warum? Welche Typkonversionen werden dabei durchgeführt?

```
public class Y {
  static void test(double i) { }
  static void test(long l) { }
4 static void test(char c) { }
  static void test(short s) { }

  public static void main(String[] args) {
    int i = 1;
9   short s = 2;
    float f = 3.0;
    char c = 'u';
    test(i);
    test(s);
14  test(f);
    test(c);
  }
}
```

Aufgabe 4 — $\Sigma = 10$ Punkte

Die Angabe von Positionen auf der Erdoberfläche erfolgt meistens durch zwei Erdmittelpunktswinkel, den Längengrad (oft longitude λ) und den Breitengrad (oft latitude ϕ). Für die computerbezogene Handhabbarkeit der Koordinaten kann es sinnvoll sein, diese Koordinaten in ein kartesisches Korrdinatensystem (
`http://de.wikipedia.org/wiki/Geographische_Koordinaten`,
`http://de.wikipedia.org/wiki/Kugel`) umzurechnen. Hierfür bietet sich ein objektorientierter Ansatz an.

Aufgabe 4 a) — 3 Punkte

Implementieren Sie zunächst die Klasse `Angle` (Winkel), die einen Winkel intern als `double` im Bogenmaß speichert. Der Winkel soll über die Methoden `double getAsRad()` und `setAsRad(double)` im Bogenmaß und über die Methoden `double getAsGrad()` bzw. `setAsGrad(double)` in Grad-Skala abgefragt bzw. gesetzt werden können. Überprüfen Sie jeweils die Gültigkeit des Wertebereichs. Zur Erinnerung:

$$\alpha_{\text{grad}} = \alpha_{\text{rad}} \cdot \frac{180}{\pi} \qquad \alpha_{\text{rad}} \in [0; 2\pi)$$

Aufgabe 4 b) — 5 Punkte

Schreiben Sie jetzt unter Verwendung der `Angle`-Klasse die Klasse `GeoCoordinate`, deren Konstruktor aber Längen- und Breitengrad als `Angle`-Objekte übernimmt und diese dann als Instanzvariablen speichert. Implementieren Sie auch die Methode `double[] getCartesianCoordinates()`, die die Position im kartesischen Koordinatensystem als double-Feld der Länge 3 zurückgibt.
Hinweis: Unter der Annahme, dass die Erde eine Kugel ist, kann man die kartesische Koordinaten mit den folgenden Formeln berechnen.

$$\begin{aligned} x &= r_{\text{earth}} \cdot \cos(\lambda) \cdot \sin\left(\frac{\pi}{2} - \varphi\right) \\ y &= r_{\text{earth}} \cdot \sin(\lambda) \cdot \sin\left(\frac{\pi}{2} - \varphi\right) \\ z &= r_{\text{earth}} \cdot \cos\left(\frac{\pi}{2} - \varphi\right) \end{aligned}$$

wobei $r_{\text{earth}} = 6371000.0$ der mittlere Erdradius ist und $0 \leq \lambda < \pi$ bzw. $-\pi \leq \varphi \leq \pi$.

Aufgabe 4 c) 2 Punkte

Schreiben Sie eine Main-Funktion, die jede implementierte Methode mindestens einmal aufruft und Argumente und Ergebnisse ausgibt. Testen Sie die Koordinantenumrechnung für die geographischen Pole ($\varphi = \pm 90$) und den Schnittpunkt des Äquators mit der 0ten Längengrad ($\varphi = 0, \lambda = 0$) und der Datums-/Zeitgrenze ($\varphi = 0, \lambda = 180$).

Benutzen Sie für π die Konstante aus dem `Math`-package. Im selben package befinden sich auch die benötigten trigonometrischen Funktionen `Math.sin()` bzw. `Math.cos()`.