Einführung in die Programmierung

by André Karge
Übung - Static & Arrays

letzte Woche

- Klassen & Objekte
- Konstruktor
- Instanziierung von Objekten
- Methodenaufrufe mit Objekten

diese Woche

- Besprechung Übungsblatt 3
- Floating Point Checks
- statische Methoden & Attribute
- Arrays
- Besprechung Übungsblatt 4

Übungsblatt 3

Übungsblatt 3

Aufgabe 1

Code Beispiel

Übungsblatt 3

Aufgabe 1

Code Beispiel

Aufgabe 2

Code Beispiel

Übungsblatt 3

Aufgabe 1

Code Beispiel

Aufgabe 2

Code Beispiel

Aufgabe 3

Code Beispiel

Floating Point Checks

Floating Point Checks

```
float a = 1.0f/3.0f;
float b = 0.33;
// a == b?
```

Wie vergleicht man zwei Floating Point Nummern?

Floating Point Check

Lösung

- wir können float Werte nicht direkt vergleichen
- dazu brauchen wir ein kleinen Floatwert epsilon ϵ
- zum Vergleich berechnen wir die Differenz und vergleichen mit epsilon
- zum Beispiel: $\epsilon = 10^{-14} = \frac{1}{10^{14}}$

Floating Point Check

Lösung

- wir können float Werte nicht direkt vergleichen
- dazu brauchen wir ein kleinen Floatwert epsilon ϵ
- zum Vergleich berechnen wir die Differenz und vergleichen mit epsilon
- zum Beispiel: $\epsilon = 10^{-14} = \frac{1}{10^{14}}$

Übungsaufgabe

1. Schreiben sie eine Methode, die zwei float-werte übergeben bekommt und true oder false zurück gibt, jenachdem, ob die Zahlen gleich oder ungleich sind.

Floating Point Check

Lösung

```
public class Test {
  private static final float epsilon = 10e-14;
  public static void main(String[] args) {
    float a = 1.0f/3.0f;
    float b = 0.33;
    boolean result = floatEqual(a, b);
  }
  public boolean floatEqual(float a, float b) {
    float differenz = Math.abs(a - b);
    if(differenz < Test.epsilon) {
      System.out.println("Werte sind gleich");
      return true;
    }
    else {
      System.out.println("Werte sind nicht gleich");
      return false;
    }
  }
}
```

Static Methoden & Attribute

Static Methoden & Attribute - Insel 5.3

- statische Methoden und Attribute vereinbaren, dass sie den Zustand von Objekten nicht ändern
- Schlüsselwort *static*
- Kapselt feste Klassenzugehörige Eigenschaften und Funktionen innerhalb der Klasse
- Alle Instanzen haben somit Zugriff auf die selbe Version der Methode / Attributes
- Beispielsweise letztes Übungsblatt:

Static Methoden & Attribute - Insel 5.3

- statische Methoden und Attribute vereinbaren, dass sie den Zustand von Objekten nicht ändern
- Schlüsselwort *static*
- Kapselt feste Klassenzugehörige Eigenschaften und Funktionen innerhalb der Klasse
- Alle Instanzen haben somit Zugriff auf die selbe Version der Methode / Attributes
- Beispielsweise letztes Übungsblatt:

```
class Complex {
  // ...
  // Es wird kein Zustand von vorhandenen Objekten geändert - static
  public static final Complex add(Complex z1, Complex z2) {
    return new Complex(z1.real + z2.real, z1.imag + z2.imag);
  }
  // ändert den Zustand des Objektes, auf dem .add(...) aufgerufen wird - nicht static
  public void add(Complex z2) {
    this.real += z2.real;
    this.imag += z2.imag;
  }
```

Static Methoden & Attribute - Insel 5.3

statische Attribute

- haben wir schon auf der vorletzten Folie gesehen:

```
public class Test {
  private static final float epsilon = 10e-14;
  ...
}
```

- der Wert soll für alle Ausprägungen (Instanzen) der Klasse gelten
- wird zusätzlich das Schlüsselwort *final* verwendet, dann darf der Wert nicht geändert werden

Static Methoden & Attribute - Insel 5.3

Übungsaufgabe

1. Erweitern Sie die gegebene Klasse `Triangle` (Webseite Intelligente Softwaresysteme). Verwenden Sie dabei `static` wo notwendig
2. Jedes Dreieck soll wissen, wieviele Dreiecke generiert worden sind. Erstellen Sie zusätzlich eine Methode um diese Zahl auszulesen.
3. Erstellen Sie eine eigene Klasse (Testklasse mit Main-Methode) um Ihre Implementierung zu testen.

Arrays

Arrays - Insel 3.7

- *Array* ist ein spezieller Datentyp
- auch *Feld* oder *Reihung* genannt
- fasst mehrere Werte in einer Einheit zusammen
- deklariert mit: Typ[] bezeichner = element1, element2, ...
- Index = Addresse von Werten im Array
- Index = Ganzzahl vom Typ integer und startet in Java bei 0
- Beispiel:

```
int[] meinArray = {15,12,13,1,14};
// indexing:       0  1  2  3 4
int length = meinArray.length; // = 5
int erstes = meinArray[0]; // = 15
```

Arrays - Insel 3.7

Mehrdimensionale Arrays

```
// mit konkreten Werten:
int[][] = {{1, 2}, {3, 4}, {5, 6}};
// leeres Array mit definierter Größe:
int[][] = new int[4][8]; // 4 zeilen und 8 Spalten
```

Arrays - Insel 3.7

Übungsaufgabe

1. Schreiben Sie eine Methode, die ein festes Float Array der Größe 42 erstellt
2. Iterieren Sie mit einer for-Schleife über dieses Array und schreiben an jede Stelle die aktuelle Schleifenvariable geteilt durch 7
3. geben Sie die Quersumme der Zahlen im Array auf der Konsole aus

Übungsblatt 4

Fragen?