Programmierung 1

Kapitel 1 – Hallo Welt

Prof. Dr. Jörg Kreiker

joerg.kreiker@informatik.hs-fulda.de

Fachbereich Angewandte Informatik
Hochschule Fulda – University of Applied Sciences

19./20. Oktober 2017

Java

- Unsere Programmiersprache: Java
  - Benannt nach dem Kaffee
  - Erfinder: James Gosling von Sun Microystems Anfang der 1990er Jahre
  - Populärste Sprache weltweit
  - Plattformunabhängig
  - Objektorientiert
- Aktuelle Version: Java 9 von Oracle

Agenda

1. Teaser
2. Orgakram
3. Unser erstes Programm
4. Aus- und Eingabe
5. Bedingte Anweisungen

Material

- Moodle Seite der Veranstaltung:
  https://elearning.hs-fulda.de/ai/course/view.php?id=381
- Buch: Nachschlagewerk *Java ist auch eine Insel*
  - Online Version unter
    http://openbook.rheinwerk-verlag.de/javainsel/
- Offizielle Sprachdokumentation von Oracle
  - https://docs.oracle.com/javase/9/
- Forum: http://stackoverflow.com/

Drei Säulen

1. Vorlesung
   - Videomitschnitt freitags
   - Live-Coding
2. Praktikum
   - Ergänzender Inhalt
   - Live Übungen
   - Fragestunde
3. Selbststudium
   - Nacharbeiten/Erarbeiten
   - Hausaufgaben

Wertungen

1. Gruppenwertung: Jede Woche gibt es bis zu 2×11 Plazierungspunkte, je 11 Punkte für die Gruppe mit der
   - höchsten Anwesenheitsquote und
   - dem besten Live Score
2. Individualpunkte:
   - Hausaufgaben werden individuell bewertet
   - Top 5 mit den meisten Punkten am Ende $\rightarrow$ Pizza
   - Obere Hälfte all derjenigen mit mindestens 10 Punkten $\rightarrow$ Duplo zur Klausur
3. Antwort des Tages: Pechkeks

Klausur

- Die Prüfung in Proggen1 findet erstmals als Semi-E-Klausur in den Rechnerräumen statt.
- Programmieraufgaben am Rechner
- Abgabe auf Papier

Unser erstes Programm

Folgendes Programm im Texteditor in eine Datei Hello.java

Hello World!

```
public class Hello {
   public static void main(String[] args) {
      System.out.println("Hello World!");
   }
}
```

Kompilieren mit `javac` und ausführen mit `java` in der Konsole

Kompilieren und Ausführen

```
> javac Hello.java

> java Hello
Hello World!
```

Das erste Programm

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

- Jedes Programm hat einen Namen (hier: Hello).
- Der Name steht hinter dem Schlüsselwort `class` (was eine Klasse, was `public` ist, lernen wir später)
- Der Datei-Name muss zum Namen des Programms “passen”, d.h. in diesem Fall `Hello.java` heißen.

Das erste Programm

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

- Das Java-Programm ist der Rumpf des Hauptprogramms, d.h. der `main()` Methode.
- Die Programm-Ausführung eines Java-Programms startet stets mit einem Aufruf der `main()` Methode.
- Die Methode `System.out.println` ist vordefiniert und schreibt ihr Argument auf die Standardausgabe (Konsole).
- Hier ist das Argument eine Zeichenkette (String), der in Anführungszeichen gesetzt ist.

Variablen

Alter und Delta

```
int alter = 41;
int delta = 10;
```

- Um Daten zu speichern und auf gespeicherte Daten zugreifen zu können, stellt Java Variablen zur Verfügung.
- Variablen müssen erst einmal eingeführt, d.h. deklariert werden.
- Diese beiden Deklarationen führen die beiden Variablen mit den Namen `alter` und `delta` ein.
- Gleichzeitig werden die Variablen mit den Werten 41 bzw. 10 initialisiert

Erklärung

- Das Schlüsselwort `int` besagt, dass diese Variablen ganze Zahlen ("Integers") speichern sollen.
- `int` heißt auch Typ der Variablen `alter` und `delta`.
- Am Ende einer Deklaration steht ein Semikolon ";".
- Syntax (was wir schreiben):

$$typ \quad name \quad = \quad wert \quad ;$$

- Semantik (was das Programm tut): siehe oben

Ausgabe

Schreiben auf Konsole

```
System.out.println("Hello World!");
System.out.println("Ich bin " + alter + " Jahre alt");
```

- `System.out.println` schreibt das Argument (in Klammern) auf die Standardausgabe (Konsole)
- Das Argument kann eine Zeichenkette sein oder
- mehrere mit "+" verbundene Strings
- Auch Integer Variablen können übergeben werden
- Jede Zeile ist eine Anweisung, genauer ein Methodenaufruf

Eingabe

Scanner

```
import java.util.Scanner;
Scanner in = new Scanner(System.in);
alter = in.nextInt();
```

- `import` lädt existierenden Java Code
- Die zweite Zeile erzeugt ein neues `Scanner` Objekt (→ später), welches von der Standardeingabe liest
- `in.nextInt()` liest die nächste ganze Zahl ein (bis zum nächsten Leerzeichen oder Return)
- Jede Zeile ist eine Anweisung

Zuweisung

Zuweisungen

```
alter = 102;
alter = alter + delta;
```

- Anweisungen gestatten, die Werte von Variablen zu modifizieren.
- Eine wichtige Anweisung ist die Zuweisung.
- Die Variable `alter` erhält den Wert 102.
- In der Zuweisung `alter = alter + delta;` greift das `alter` auf der rechten Seite auf den Wert vor der Zuweisung zu.
- D.h. die Werte von `alter` und `delta` werden ermittelt, addiert und der Variablen `alter` zugewiesen.

Zuweisungen

```
alter = 102;
alter = alter + delta;
```

Semantik der Zuweisung

linke Seite	rechte Seite
Variable die sich ändert	der neue Wert berechnet aus den alten

Ausdrücke

- Auf der rechten Seite von Zuweisungen an Integer Variablen können Integer Ausdrücke stehen
- Ein Integer Ausdruck kann sein
  - Eine ganze Zahl als Konstante
  - Zwei Ausdrücke mit einem Operator in der Mitte, z.B.
    - `+`, `-`, `*` wie erwartet
    - `/` ganzzahlige Division (`5/2` ergibt `2`)
    - `%` modulo, Rest bei ganzzahliger Division
  - Ein geklammerter Ausdruck

Sequenz

- Mehrere Anweisungen können hintereinander stehen: Sequenz
- Zu jedem Zeitpunkt wird nur eine Anweisung ausgeführt.
- Jede Anweisung wird genau einmal ausgeführt. Keine wird wiederholt, keine ausgelassen.
- Die Reihenfolge, in der die Anweisungen ausgeführt werden, ist die gleiche, in der sie im Programm stehen (d.h. nacheinander).
- Mit Beendigung der letzten Anweisung endet die Programm-Ausführung.

Bedingte Anweisung

If-else

```
if (alter > 29) {
    System.out.println("Aemon!");
} else {
    System.out.println("Arya!");
}
```

- Zuerst wird die Bedingung ausgewertet: `alter > 29`
- Ist sie erfüllt, wird die nächste Anweisung ausgeführt.
- Ist sie nicht erfüllt, wird die Anweisung nach dem `else` ausgeführt.
- Geschweifte Klammern definieren Blöcke aus einem oder mehreren Anweisungen
- Den `else`-Teil darf man weglassen.
- Innerhalb der Blöcke dürfen erneut bedingte Anweisungen vorkommen: geschachtelte Bedingungen.

Bedingungen

Die Bedingungen innerhalb einer bedingten Anweisung können unter anderem sein:

- Ein Vergleich zwischen zwei Integer Ausdrücken, z.B.
  - `<`, `>`, `<=`, `>=` wie erwartet
  - `==` Test auf Gleichheit
  - `!=` Test auf Ungleichheit
- Eine Konjunktion von Vergleichen, z.B. `x > 1 && x < 100`, `x` liegt zwischen 1 und 100
- Eine Disjunktion von Vergleichen, z.B. z.B. `x < 1 || x > 100`, `x` liegt nicht zwischen 1 und 100