FACHSCHAFT INFORMATIK
HS Karlsruhe

Programmiervorkurs

Einführung in Java

Tag 4

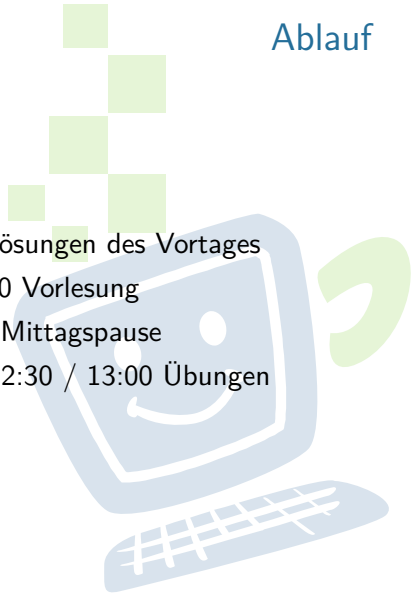
Jonas Westenhoff

Sommersemester 2020

Inhaltsübersicht Vorkurs

- Tag 1: Zustände, Variablen, Datentypen, Konvertierungen, Arithmetik, Eclipse Livedemo
- Tag 2: Kommentare, Boolesche Ausdrücke, If-Abfragen, Switch-Case
- Tag 3: Arrays, (Do-)While-Schleife, For-Schleifen, Weiterführung Debugging
- Tag 4: (statische) Methoden, Klassenvariablen

Ablauf



- 09:30 Lösungen des Vortages
- ab 10:00 Vorlesung
- 90 min Mittagspause
- gegen 12:30 / 13:00 Übungen

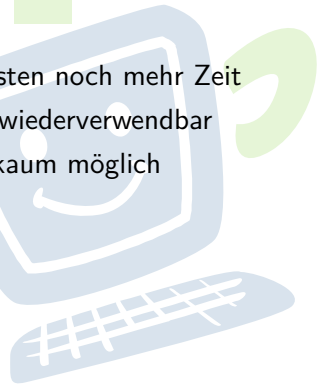
Beispiel ohne Methoden

```java
public class Main {
    public static void main() {
        for (int i = 1; i <= 10; i++) {
            System.out.println(i);
        }

        // ...

        for (int i = 1; i <= 10; i++) {
            System.out.println(i);
        }
    }
}
```

Warum ist dieser Code problematisch?

Probleme:

- Zeitaufwändig
- (zu) viel Code
- Unübersichtlich
- Änderungen kosten noch mehr Zeit
- Code oft nicht wiederverwendbar
- Arbeitsteilung kaum möglich



Warum ist dieser Code problematisch?

Probleme:

- Zeitaufwändig
- (zu) viel Code
- Unübersichtlich
- Änderungen kosten noch mehr Zeit
- Code oft nicht wiederverwendbar
- Arbeitsteilung kaum möglich

Lösungen:

- Ähnlichen Code auslagern
- Wiederverwendbaren Code schreiben
- **Methoden!**

Beispiel mit Methoden

```
public class Main {
    public static void main () {
        zaehlBisZehn();

        // ...

        zaehlBisZehn();
    }

    public static void zaehlBisZehn() {
        for (int i = 1; i <= 10; i++) {
            System.out.println(i);
        }
    }
}
```

Methodenaufbau

```
public static Rückgabetyp Name(Parameter){
    // Methodenrumpf
    return ... ;
}
```

Beispiel

```
public static void zaehlBisZehn () {
    // ...
}
```

Anfang: public static
Rückgabetyp: void
Methodenname: zaehlBisZehn
Parameter: keine

Aufruf einer Methode

```
public static void main() {
    zaehlBisZehn();
    int x = gibMir42();
    int y = verdoppelWert(x);
```

- Methodenname
- (Parameter)
- ;

Beispiel ohne Methoden

```
public class Main () {
    public static void main() {
        for (int i = 1; i <= 9; i++) {
            System.out.println(i);
        }

        // ...

        for (int i = 1; i <= 10; i++) {
            System.out.println(i);
        }
    }
}
```

Beispiel mit Methoden

```
public class Main() {
    public static void main() {
        zaehleBis(9);

        // ...

        zaehleBis(10);
    }

    public static void zaehleBis(int zahl) {
        for (int i = 1; i <= zahl; i++) {
            System.out.println(i);
        }
    }
}
```

Parameter

```
public static void zaehleBis(int zahl) {
    // ...
}
```

- In die runden Klammern kommen die Parameter
- Parameter werden mit Komma getrennt:

  ```
  (int a, boolean b, double c)
  ```

- Ein Parameter besteht aus Datentyp und Bezeichner

Was passiert?

```
zaehleBis(9);
```

```
public static void zaehleBis(int zahl) {
    // "zahl" wird der Wert 9 zugewiesen
}
```

```
zaehleBis(10);
```

```
public static void zaehleBis(int zahl) {
    // "zahl" wird der Wert 10 zugewiesen
}
```

Beispiel mit 2 Parametern

```java
public class Main {
    public static void main() {
        zaehleVonBis(1, 9);
        // ...
        zaehleVonBis(5, 10);
    }

    public static void
        zaehleVonBis(int von, int bis) {
        for (int i = von; i <= bis; i++) {
            System.out.println(i);
        }
    }
}
```

Beispiel mit Rückgabewert

```java
public class Main {
    public static void main() {
        int x = gibMir42();
    }

    public static int gibMir42() {
        return 42;
    }
}
```

Rückgabewert

```
public static int gibMir42() {
    ...
    return 42;
}
```

- Möchte man einen Wert zurück geben, so wird der entsprechende Datentyp angegeben (z.B. int, double, String)
- Wenn es keinen Rückgabewert gibt wird das Schlüsselwort void als Datentyp angegeben
- Wenn es einen Rückgabewert gibt wird dieser in der Methode mit dem Befehl return zurückgegeben

Was passiert?

```
int x = gibMir42();
```

- Rechte Seite von "=" wird zuerst ausgewertet
  - gibMir42();

```
public static int gibMir42() {
    return 40 + 2;
}
```

- return 40 + 2
  - return 42
    - 42 wird zurückgegeben
- x wird der Wert 42 zugewiesen

Lesbarkeit durch Methoden

- Berechnung der Fakultät: $n! = 1 \times 2 \times \cdots \times n$

```
public static long fak(int n){
    long faku = 1;

    // Iterative Berechnung

    for(int i = 1; i<=n; i++) {
      faku *= i;
    }

    return faku;
}
```

Binomialkoeffizient berechnen

$$\binom{n}{k} = \frac{n!}{(n-k)! \times k!}$$

- Für die Berechnung braucht man 3 Mal eine Fakultät
- Somit 3 Schleifen in der Berechnung
- Code unlesbar

Binomialkoeffizient berechnen

$$\binom{n}{k} = \frac{n!}{(n-k)! \times k!}$$

- Für die Berechnung braucht man 3 Mal eine Fakultät
- Somit 3 Schleifen in der Berechnung
- Code unlesbar

Lösung

- Methode `long fak(int n)` benutzen

Binomialkoeffizient berechnen

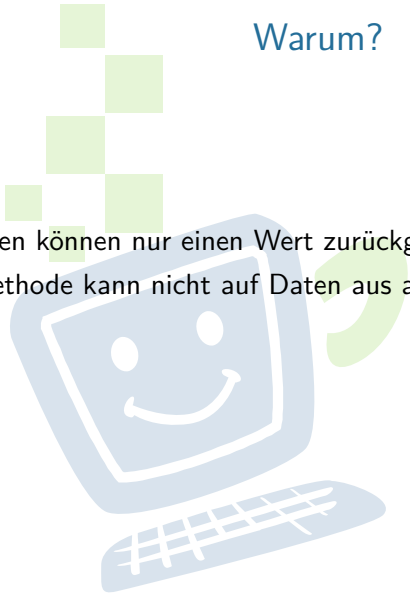
$$\binom{n}{k} = \frac{n!}{(n-k)! \times k!}$$

```
public static double binomial(int n, int k) {
    double bin = fak(n) / (fak(n - k) * fak(k));

    return bin;
}
```

Beispiel ohne Klassenvariablen

```java
public class Vorkurs {
    public static void main() {
        int x = zaehleVonBis(1, 9);
        System.out.println(wurdeAusgegeben);
        // Das funktioniert so leider nicht
    }

    public static int zaehleVonBis(int von, int bis) {
        for (int i = von; i <= bis; i++) {
            System.out.println(i);
        }

        boolean wurdeAusgegeben = bis >= von;
        return bis - von + 1;
    }
}
```

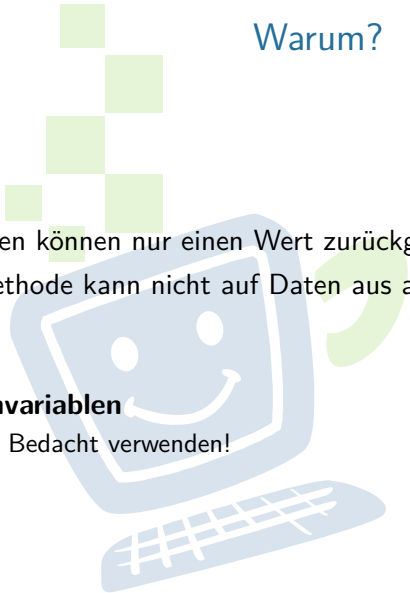
Warum?



Probleme

- Methoden können nur einen Wert zurückgeben
- Eine Methode kann nicht auf Daten aus anderen Methoden zugreifen

Warum?



Probleme

- Methoden können nur einen Wert zurückgeben
- Eine Methode kann nicht auf Daten aus anderen Methoden zugreifen

Lösung

- **Klassenvariablen**
  - mit Bedacht verwenden!

Beispiel mit Klassenvariablen

```java
public class Vorkurs {
    public static boolean wurdeAusgegeben;

    public static void main() {
        int x = zaehleVonBis(1, 9);
        System.out.println(wurdeAusgegeben);
    }

    public static int zaehleVonBis(int von, int bis) {
        for (int i = von; i <= bis; i++) {
            System.out.println(i);
        }

        wurdeAusgegeben = bis >= von;
        return bis - von + 1;
    }
}
```

Deklaration von Klassenvariablen

- Deklaration direkt nach Klassendeklaration
- **public static** Datentyp Variablenname;
- Sichtbar in der ganzen Klasse
- Ohne manuelle Zuweisung wird der Defaultwert zugewiesen
- Beispiel:

```
public static boolean wurdeAusgegeben;
```

Vorstellung der Aufgabe

Vorstellung der Aufgabe

addNewTicket

```
public static String[] addNewTicket(String[] oldTickets,
    String newTicket,boolean bahncard)
```

Beschreibung:
Soll den Parameter ***oldTickets*** um den Parameter ***newTicket*** erweitern und für den Fall dass der Parameter ***bahncard*** true ist um „(Bahncard)“ erweitern und anschließend zurück geben.

getSum

```
public static double getSum()
```

Beschreibung:
Gibt den aktuellen Gesamtpreis als double in Euro zurück

calculateNewSum

```
public static void calculateNewSum(int distance, boolean
    bahncard)
```

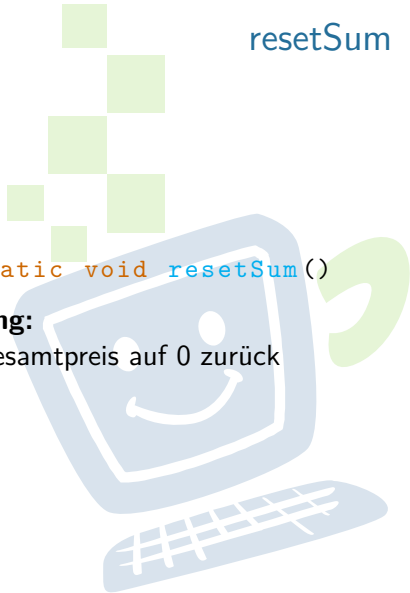
Beschreibung:

Berechnet den Gesamtpreis aller bisher sowie dem aktuell ausgewählten Ticket.

Dabei gilt:

- Bis 200km: 10 €+ 0.20 €pro km
- Ab 200km: 5 €+ 0.15 €pro km
- Mit einer Bahncard erhält man immer 25% Rabatt

resetSum



```
public static void resetSum()
```

Beschreibung:
Setzt den Gesamtpreis auf 0 zurück

insertMoney


```
public static void insertMoney(int amount)
```

Beschreibung:
Wenn der Kunde einen Geldschein einwirft wird diese Methode aufgerufen. Der noch zu bezahlende Betrag muss dementsprechend angepasst werden.

getAmountLeft


```
public static double getAmountLeft()
```

Beschreibung:
Als Rückgabewert soll hier der noch zu zahlende Betrag als double in Euro zurück gegeben werden.

getChangeAmount

```
public static double getChangeAmount()
```

Beschreibung:
Wird aufgerufen, wenn der Kunde ausreichend Geld eingeworfen hat und gibt den Betrag des Wechselgeldes zurück (in Euro). Dieser muss positiv sein.

getChangeCoins

```
public static int[] getChangeCoins()
```

Beschreibung:

Berechnet, wie viele Münzen von jeder Sorte der Kunde zurück bekommt. Gibt ein Array zurück, das die Anzahl der entsprechenden Münzen enthält

- Rückgabe[0]: enthält die Anzahl der 2 €-Münzen
- Rückgabe[1]: enthält die Anzahl der 1 €-Münzen
- ...
- Rückgabe[7]: enthält die Anzahl der 1 ct-Münzen

beginPayment (OPTIONAL)

```
public static void beginPayment()
```

- Diese Methode ist optional
- Wird nicht immer benötigt
- Kann für spezielle Aktionen zu Beginn des Bezahlvorgangs verwendet werden

Quellen und Lizenz

FACHSCHAFT INFORMATIK
HS Karlsruhe

- Original von Samuel Zeitvogel
- Überarbeitet 2012 von Daniel Hoff
- Überarbeitet 2013 von Tristan Wagner
- Überarbeitet 2015 von Tobias Kerst
- Überarbeitet 2016 von Christian Wernet
- Überarbeitet 2018 von Jan Oliver Zerulla
- Überarbeitet 2018 von Johannes Beierle
- Überarbeitet 2020 von Jonas Westenhoff