Guten Nachmittag!

Ablauf

- **Puzzle-Gruppenarbeit:**
  Unterrichtsrunde ca. 45 Minuten

- **Weitere Java-Elemente / Repetition**

Puzzle-Gruppenarbeit

- **Gruppenweise zusammensitzen (alle roten, alle grünen, ...)**
- **Ziel: Den anderen der Gruppe die erarbeiteten Themen erklären**
- **Zeit: 40 Minuten (10 Minuten für jeden)**

Wir haben gesehen

- **Das Gerüst ist bei JavaKara fix vorgegeben**

```
import JavaKaraProgram;
public class ____________ extends JavaKaraProgram {
  public void myProgram() {
    // hier kommt das Hauptprogramm hin

    (...)

  }
}
```

- **Wir müssen nur noch „Befehle einfüllen“**

Befehle einfüllen

- **Die Befehle, die man aufruft, nennt man Methoden**
- **Die Befehle werden mit Strichpunkten abgeschlossen**

```
kara.move();
```

Geschweifte Klammern { }

- **Umklammern Programmblöcke**
- **Z.B. eine Methode oder ein Verzweigungs-Block**
- **Können auch verschachtelt angewendet werden -> Blöcke einrücken**

```
import JavaKaraProgram;
public class ____________ extends JavaKaraProgram {
  public void myProgram() {

    (...)

  }
}
```

Einrücken

```
import JavaKaraProgram; public class Slalom
extends JavaKaraProgram { void eat() { if
(kara.onLeaf()) { kara.removeLeaf(); } } void
eat_right() { eat(); kara.move();
kara.turnRight(); } void eat_left() { eat();
kara.move(); kara.turnLeft(); } public void
myProgram() { while (!kara.treeFront()) {
eat_right(); eat_left(); } }
```

Einrücken

```
import JavaKaraProgram;

public class Slalom extends JavaKaraProgram {

  void eat() { if (kara.onLeaf()) { kara.removeLeaf(); } }

  void eat_right() { eat(); kara.move(); kara.turnRight(); }

  void eat_left() { eat(); kara.move(); kara.turnLeft(); }

  public void myProgram() { while (!kara.treeFront()) { eat_right(); eat_left(); } }

}
```

Bildschirm

Einrücken

```
import JavaKaraProgram;

public class Slalom extends
      JavaKaraProgram {

  void eat() {

     if (kara.onLeaf()) {
       kara.removeLeaf();
     }

  }

  void eat_right() {

     eat();
     kara.move();
     kara.turnRight();

  }
```

...

...

```
void eat_left() {

    eat();
    kara.move();
    kara.turnLeft();

  }

  public void myProgram() {

    while (!kara.treeFront()) {
      eat_right();
      eat_left();
    }
  }
}
```

Wir kennen nun schon

- **Boole‘sche Ausdrücke**

- **Verzweigung (if)**

```
if (richtungBasel()) { einspuren(); }
```

- **Schleifen (while, for)**

```
while (lebeNoch()) { herzSchlag(); }
```

- **Methoden**

Herr Boole & Verzweigungen

- **Wichtig, wenn festgestellt werden muss, ob ein bestimmter Fall eingetreten ist.**
- **Mehrere Bedingungen können kombiniert werden**

```
if ( sonne.Scheint() && draussen.Warm() ) {
   marc.nimmLiegestuhHervor();
   marc.liegeAnDieSonne();
}
```

Objekt „marc“

„Solange-Bis“ Schleifen

- **while Schleife, wenn man etwas solange macht, bis etwas passiert**

```
while ( sonne.Scheint() ) {
  marc.bleibAufDemLiegestuhlLiegen();
}
```

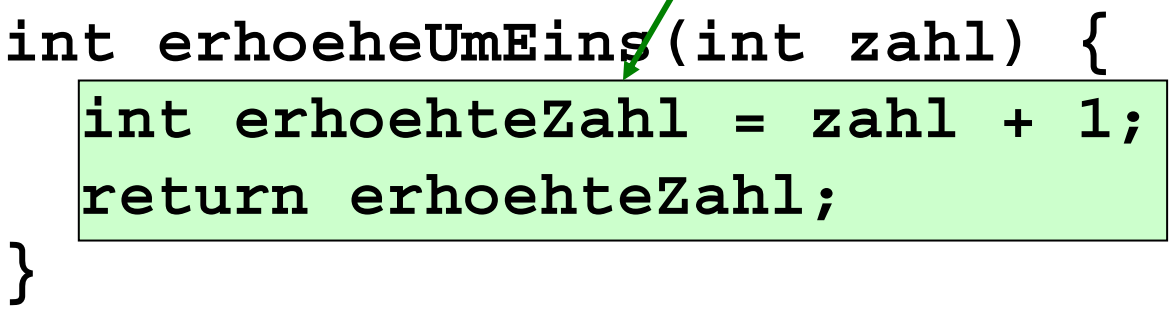
„x-Mal“ Schleifen

- **for Schleife, bei einer Anzahl Schleifendurchgänge**

```
for (int i=1; i<17; i++) {
  marc.leseHarryPotterBand1Seite(i);
}
```

Methoden

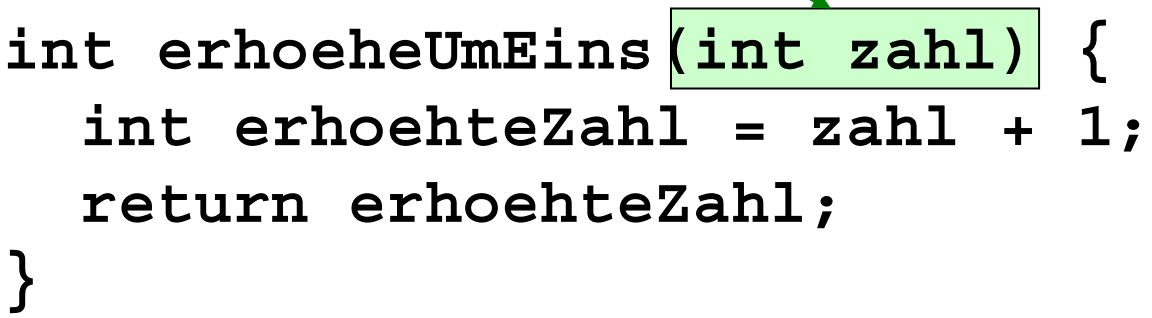
- **Methoden fassen mehrere Befehle zusammen**
- **Parameter erlauben es, Werte zu übergeben.**
- **Methoden können Werte zurückgeben**



```
int erhoeheUmEins(int zahl) {
    int erhoehteZahl = zahl + 1;
    return erhoehteZahl;
}
```

Methoden

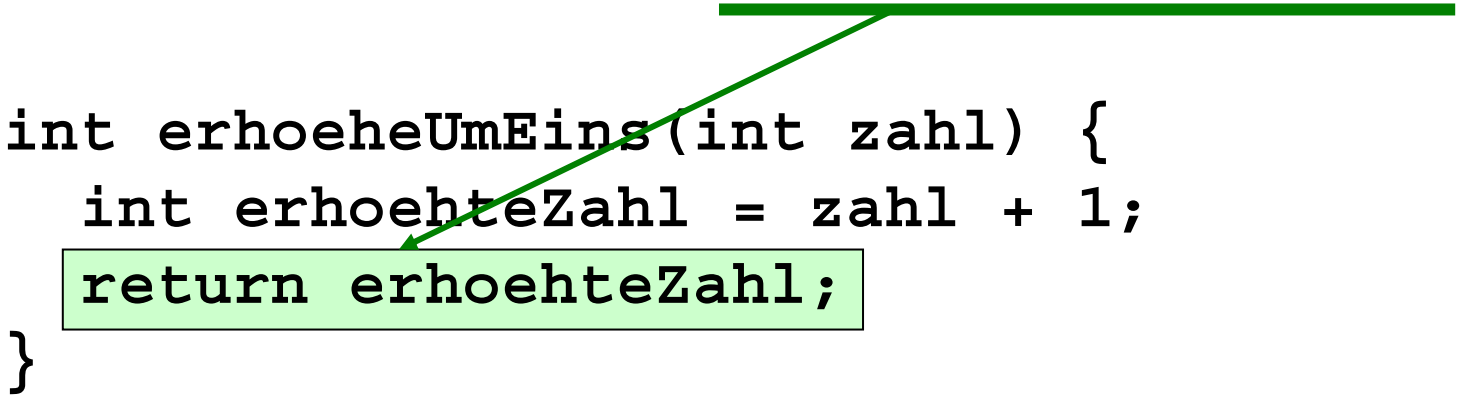
- **Methoden fassen mehrere Befehle zusammen**
- **Parameter erlauben es, Werte zu übergeben.**
- **Methoden können Werte zurückgeben**



```
int erhoeheUmEins(int zahl) {
  int erhoehteZahl = zahl + 1;
  return erhoehteZahl;
}
```

Methoden

- **Methoden fassen mehrere Befehle zusammen**
- **Parameter erlauben es, Werte zu übergeben.**
- **Methoden können Werte zurückgeben**



```
int erhoeheUmEins(int zahl) {
  int erhoehteZahl = zahl + 1;
  return erhoehteZahl;
}
```

Methoden

- **Details: Mehrere Parameter werden durch Kommas getrennt**
- **Mehrere Werte zurückgeben: Geht nicht!**


```
int erhoeheUm(int zahl, int wert) {
  int erhoehteZahl = zahl + wert;
  return erhoehteZahl;
}
```

Methoden

```
int erhoeheUmEins(int zahl) {
  int erhoehteZahl = zahl + 1;
  return erhoehteZahl;
}
```

- **Parameter werden in der Methode wie Variablen angeschaut**
- **Auf die Variablen des Hauptprogramms soll man nicht zugreifen**

Weitere Java-Elemente

- **Kommentare:**
  Die habt Ihr schon gesehen

- **2 Möglichkeiten:**

```
/* Das ist ein Kommentar.
   Der kann auch mehrere Zeilen umfassen. */

// Das ist auch ein Kommentar
// der geht aber nur bis zum Schluss der Zeile
```

Gross- / Kleinschreibung

- **Java unterscheidet zwischen Gross- und Kleinschreibung!**

  `Kara.Move();` **wird nicht akzeptiert**

Variablen

- **Schon kennengelernt bei for-Schleife**
- **Kennt Ihr von der Mathematik:**

$$\begin{aligned} 3x+4 &= 16 \\ 3x &= 12 \\ x &= 4 \end{aligned}$$

- **Platzhalter für einen Wert**

Variablen

- **In Mathematik:**
  x kann ganze Zahl sein, oder reell, oder eine natürliche Zahl
- **In Java:**
  jede Variable hat einen Typ!

- **Variablen müssen definiert werden!**

Variablen definieren

```
int counter = 0;
```


```
int counter;
(...)
counter = 0;
```

Initialisierungswert kann
auch weggelassen werden!

Verschiedene Typen

- **int: ganze Zahlen**
  ```
  int counter = 2;
  ```
- **double: Gleitkommazahl (rationale Zahlen)**
  ```
  double pi = 3.14159;
  ```
- **String: Zeichenketten (Achtung: grosses S)**
  ```
  String bewertung = "Java ist cool!";
  ```
- **boolean: wahr oder nicht wahr**
  ```
  boolean schoenesWetter = true;
  ```

Typen in der freien Wildbahn!

```
int erhoeheUmEins(int zahl) {
  int erhoehteZahl = zahl + 1;
  return erhoehteZahl;
}

void viertelDrehung() {
  kara.move();
  kara.turnRight();
  kara.move();
}

boolean warm() {
  if (draussen.temperatur() > 20) {
    return true;
  }
  return false;
}
```

Konstanten

- **Gibt’s in der Mathematik auch:**
  Zahl π = 3.14159265358...
  Zahl e = 2.71828182845...

- **Wert wird vor dem Start zugewiesen – und nicht mehr verändert**

- **Auch Konstanten müssen definiert werden**

Konstanten definieren

- **Konvention: Für den Konstantennamen GROSSBUCHSTABEN verwenden**
- **Vor dem Typ muss `final` stehen**

```
final int FENSTERBREITE = 800;
```

Ablauf von JavaKara Programmen

- **Start: `myProgram()` wird gesucht**
- **Abarbeiten der Befehle in `myProgram()`**
- **Programm ist fertig, wenn die letzte Anweisung von `myProgram()` abgearbeitet wurde**

```
import JavaKaraProgram;
public class GeheUmBaumHerum

  void viertelDrehung() {
    kara.move();
    kara.turnRight();
    kara.move();
  }

  public void myProgram() {
    kara.turnLeft();
    viertelDrehung();
    viertelDrehung();
    kara.turnLeft();
  }
}
```

public – jetzt wird's kompliziert

- `public` **heisst "von aussen sichtbar"**
- **Brauchts, damit das JavaKara-Programm die Klasse** `GeheUmBaumHerum` **und darin das Program** `myProgram()` **"sieht"**

```
import JavaKaraProgram;
public class GeheUmBaumH

  void viertelDrehung()
    kara.move();
    kara.turnRight();
    kara.move();
  }

  public void myProgram(
    kara.turnLeft();
    viertelDrehung();
    viertelDrehung();
    kara.turnLeft();
  }
}
```

Beispiel Restaurant

- “Public”
  - `entrée(KLEINER_SALAT);`
  - `hauptgang(STEAK, MEDIUM);`
  - `rechnung(KREDITKARTE_VISA);`

- Nicht Public
  - `wascheSalat(GRUENDLICH);`
  - `schäleKartoffeln(FLOTT);`
  - `aufschlag(“10%”);`

public – jetzt wird's kompliziert

- **viertelDrehung() muss nicht public sein**
- **Wird ja auch nicht vom JavaKara Programm direkt aufgerufen**

```
import JavaKaraProgram;
public class GeheUmBaumH

  void viertelDrehung()
    kara.move();
    kara.turnRight();
    kara.move();
  }

  public void myProgram(
    kara.turnLeft();
    viertelDrehung();
    viertelDrehung();
    kara.turnLeft();
  }
}
```