



FACHSCHAFT INFORMATIK
HS Karlsruhe

Programmiervorkurs

Tag 2

Boris Bopp

Ablauf

- 09:30 Uhr Vorstellung der Lösungen des Vortages
- 10:00 Uhr Vorlesung
- 11:30 Uhr Mittagspause
 - 60 Minuten
- gegen 12:30 Uhr Übungen im LI 136 u. E 203

Inhaltsübersicht Vorkurs

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

Inhalt Tag 2

- **Kommentare**
 - Zeilenkommentar
 - Blockkommentar
 - JavaDoc
- **Boolesche Ausdrücke**
 - Wahrheitswerte
 - Verknüpfungen
- **Fallunterscheidungen**
 - If-Abfragen
 - Switch-Case

Kommentare

- Sind euch gestern schonmal begegnet:

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
}
```

- z.B. für Body Mass Index

```
// The body mass index is defined as the body mass in kg  
// divided by the square of the body height in meters.  
double bodyMass = 80, bodyHeight = 1.9;  
double bmi = bodyMass / (bodyHeight * bodyHeight);
```

Kommentare

- Haben **keinen** Einfluss auf den Programmablauf
- Erleichtern das Verständnis des Quelltextes
 - Für andere Entwickler
 - Für sich selbst: „Was habe ich da vor 4 Monaten nochmal gemacht?“
- Quellcode ist primär verständlich für Computer
Kommentare beschreiben die dahinterliegende Lösungsidee der Entwickelnden.

Kommentare

- **Nicht** das offensichtliche beschreiben:

```
// Die Summe besteht aus der Addition von input und 8.  
int sum = input + 8;
```

- Können sinnvolle Namensgebung **nicht** ersetzen:

```
// x ist der bruttopreis  
// h ist der nettopreis und p die Mehrwertsteuer.  
x = h + p;
```

- Besser ohne Kommentar:

```
grossPrice = netPrice + taxes;
```

Zeilenkommentar

Kommentare



- Für Anmerkungen & Gedächtnisstützen
- Beginnen mit zwei Schrägstrichen „//“
- Enden mit einem Zeilenumbruch
- Beziehen sich gewöhnlich auf die nächsten 1-3 Zeilen Quelltext
- z.B als Gedächtnisstütze:

```
// TODO: Stürzt ab, wenn personsCount 0 ist.  
int cookiesPerPerson = cookiesCount / personsCount;
```


Blockkommentar

Kommentare



- Zum kurzzeitigen Auskommentieren von Code, wenn man verschiedene Lösungsansätze ausprobiert, da das Programm sonst nicht kompilieren würde.
- Beginnen mit `/*` und enden mit `*/`
- z.B.

```
/*  
int result = 23;  
result += 5  
*/  
int result = 42;
```

JavaDoc

Kommentare



- Habt ihr vielleicht schon gesehen:

```
System.out.println(result);
```

● **void java.io.PrintStream.println(int x)**

println

```
public void println(int x)
```

Prints an integer and then terminate the line. This method behaves as though it invokes [print\(int\)](#) and then [println\(\)](#).

Parameters:

Press 'F2' for focus

JavaDoc

Kommentare



- Aus diesen Kommentaren kann eine Webseite (HTML) generiert werden, die den Code beschreibt
- Beginnen mit `/**` und enden mit `*/`
- Sogenannte Tags, die mit `@` beginnen, zeichnen den Inhalt aus.

- z.B. Am Anfang einer Java-Datei

```
/**  
 * Prints a String and then terminate the line. This method behaves as  
 * though it invokes {@link #print(String)} and then  
 * {@link #println()}.  
 *  
 * @param x The String to be printed.  
 */
```

- mehr dazu an Tag 4

Wahrheitswerte

Boolesche Ausdrücke



- Werden zur Entscheidungsfindung verwendet
- Können zwei Werte annehmen
 - **true**
 - **false**
- Oft das Ergebnis eines Vergleiches
- z.B Glückspielprogramm

```
bool gameAllowed = age > 17;
```

Arithmetische Operatoren

Boolesche Ausdrücke

- > größer
- < kleiner

- >= größer oder gleich
- <= kleiner oder gleich

- == gleich
- != ungleich

Häufiger Anfängerfehler:

```
boolean isAllowed;  
/* isAllowed wird gesetzt  
 * [...]  
 */  
isAllowed == true    =>    isAllowed  
isAllowed == false  =>    !isAllowed
```

Arithmetische Operatoren

Boolesche Ausdrücke



- > größer
- < kleiner

- >= größer oder gleich
- <= kleiner oder gleich

- == gleich
- != ungleich

Häufiger Flüchtigkeitsfehler:

```
int number = 1;  
System.out.println( number == 2 );  
System.out.println( number = 2 );
```

Verknüpfungen

Boolesche Ausdrücke



- Oft muss man Wahrheitswerte verknüpfen
- $a \ \& \ b$ „A und B“
- $!a$ „Nicht a“
- $a \ \wedge \ b$ „Entweder A oder B“ (aber nicht beides)
 - z.B. `boolean multiwaySwitchingLight = switch1 ^ switch2;`
- $a \ | \ b$ „A oder B“ (oder beides)
 - z.B. `boolean hasAccess = isStudent | isEmployee;`

Verknüpfungen

Boolesche Ausdrücke

A	B	&
0	0	0
0	1	0
1	0	0
1	1	1

A	B	^
0	0	0
0	1	1
1	0	1
1	1	0

A	B	
0	0	0
0	1	1
1	0	1
1	1	1

A	!
0	1
1	0

Boolesche Ausdrücke



- Den Satz „Die Exmatrikulation wird durchgeführt bei mehr als 10 Semestern oder nicht bezahltem Semesterbeitrag“ kann der Computer so nicht auswerten.
- Den booleschen Ausdruck jedoch schon:

```
bool exmatriculation = (semester > 10) | (!semesterFeePaid);
```

Exkurs: Gleitkommazahlen



- Gleitkommazahlen in Java sind anfällig für Rundungsfehler
- Diese dürfen niemals auf Gleichheit (mit ==) geprüft werden, denn z.B. ist $0.1 + 0.1 \neq 0.2$
- Stattdessen prüfen ob die Abweichung minimal ist:
double a = 0.1 + 0.1;
double b = 0.2;
bool isEqual = Math.abs(a-b) < 1.0E-10;

If-Abfrage

Fallunterscheidungen



- Anweisung wird nur dann ausgeführt, wenn eine bestimmte Bedingung erfüllt ist
- Diese Bedingungen haben wir bereits als boolesche Ausdrücke kennen gelernt.
- „else if“ und „else“ sind optional

```
if ( /* Bedingung */ ) {  
    // mach was  
} else if ( /* andere Bedingung */ ) {  
    // mach was anderes  
} else {  
    // mach was ganz anderes  
}
```

If-Abfrage

Fallunterscheidungen



```
if(weHaveMate) {  
    System.out.println(  
        "Nimm 'ne Mate und geh zur Vorlesung");  
} else if (weHaveKaffee) {  
    System.out.println(  
        "Nimm 'nen Kaffee und geh zur Vorlesung");  
} else {  
    System.out.println("Viel Glueck da draussen!"  
        + " Du bist auf dich alleine gestellt.");  
}
```

Switch-Case

Fallunterscheidungen



- Wird leicht unübersichtlich

```
char operation;  
int result, a, b;  
if(operation == '+') {  
    result = a+b;  
} else if (operation == '-') {  
    result = a-b;  
} else if (operation == '*') {  
    result = a*b;  
} else if (operation == '/') {  
    result = a/b;  
} else {  
    result = 0;  
}
```

Switch-Case

Fallunterscheidungen

```
char operation;  
int result, a, b;  
  
switch(operation) {  
    case '+':  
        result = a+b;  
        break;  
    case '-':  
        result = a-b;  
        break;  
    case '*':  
        result = a*b;  
        break;  
    case '/':  
        result = a/b;  
        break;  
    default:  
        result = 0;  
        break;  
}
```

- Datentypen:
 - Ganzzahlen
 - char
 - sogar String!

Switch-Case

Fallunterscheidungen

```
char input;  
boolean isVowel;  
switch(input) {  
    case 'a':  
    case 'e':  
    case 'i':  
    case 'o':  
    case 'u':  
        isVowel = true;  
        break;  
    default:  
        isVowel = false;  
}
```

- Datentypen:
 - Ganzzahlen
 - char
 - sogar String!
- Lauft durch bis
 - break
 - zum Ende des Switch

Gültigkeitsbereiche

Fallunterscheidungen



```
char input;  
boolean isVowel;  
switch(input) {  
    case 'a':  
    case 'e':  
    case 'i':  
    case 'o':  
    case 'u':  
        isVowel = true;  
        break;  
    default:  
        isVowel = false;  
}
```

Warum so?

Gültigkeitsbereiche

Fallunterscheidungen



Und nicht so?

```
char input;  
switch(input) {  
    case 'a':  
    case 'e':  
    case 'i':  
    case 'o':  
    case 'u':  
        boolean isVowel = true;  
        break;  
    default:  
        boolean isVowel = false;  
}
```

Ablauf

- 09:30 Uhr Vorstellung der Lösungen des Vortages
- 10:00 Uhr Vorlesung
- 11:30 Uhr Mittagspause
 - 60 Minuten
- gegen 12:30 Uhr Übungen im LI 136 u. E203