

# Informatik II Übung, Woche 09

Giuseppe Accaputo

3. März, 2016

## Plan für heute

1. Java: kurze Wiederholung
2. Nachbesprechung Übung 1
3. Vorbesprechung Übung 2

## Java Typen Übersicht

Typ	Grösse (Bits)	Min. Wert	Max. Wert
<code>byte</code>	8	-128	127
<code>short</code>	16	-32768	32767
<code>int</code>	32	$-2^{31}$	$2^{31} - 1$
<code>long</code>	64	$-2^{63}$	$2^{63} - 1$
<code>float</code>	32	$-3.4 \cdot 10^{38}$	$3.4 \cdot 10^{38}$
<code>double</code>	64	$-1.80 \cdot 10^{308}$	$1.80 \cdot 10^{308}$
<code>boolean</code>	n.d.	Entweder <code>true</code> oder <code>false</code>	
<code>char</code>	16	0x0000	0xffff

## Programmausführung

Wenn ein Programm gestartet wird, so führt die Java Virtual Machine (JVM) die `main()`-Methode aus:

```
class MyClass{  
    public static void main(String[] args){  
        // Mein Code  
    }  
}
```

## If / Else Abfragen

- Bedingung muss zu einem bool evaluieren.

Beispiel:

```
if(testscore >= 90){
    grade = 'A';
}
else if (testscore >= 80){
    grade = 'B';
}
else{
    grade = 'C';
}
```

## Schleifen: Do-While

- ▶ Schleifenbody wird mindestens einmal ausgeführt.

Beispiel:

```
int count = 1;
do{
    System.out.println("Count is " + count);
    count++;
} while(count < 11);
```

## Schleifen: While

- ▶ Schleifenbody wird evtl. nie ausgeführt.

Beispiel:

```
int count = 1;
while(count < 11){
    System.out.println("Count_ is_" + count);
    count++;
}
```

## Schleifen: For

- ▶ `for`-Schleifen bieten eine Alternative zu `while`-Schleifen

Beispiel:

```
int count = 1;
for(int i=1; i<11; i++){
    System.out.println(count);
}
```

- ▶ Die eben gezeigte `for`-Schleife macht dasselbe wie

```
int count = 1;
while(count < 11){
    System.out.println(count);
    count++;
}
```



## Strings

Strings können mittels + konkateniert werden:

```
String myString = "Foo" + "Bar" + " " + 123;
```

- Die Umwandlung von Datentypen und Objekten zu Strings erfolgt mittels implizitem Aufruf der toString()-Methode.

## Methoden

```
public static int MyMethod(int x){  
    return x*2;  
}
```

Diese Methodendeklaration besteht aus fünf Komponenten:

1. Modifiers (`public static`)
2. Rückgabetyt: Datentyp des Wertes, welches von der Methode zurückgegeben wird (`int`)
3. Methodenname (`MyMethod`)
4. Parameter-Liste in Klammern: eine durch Kommas separierte Liste von Eingabeparametern inkl. Datentypen (`int x`)
5. Methodenbody (`return x*2;`)

## Allgemeines

- ▶ Code Style: Verwendet die Tastenkombination CTRL + SHIFT + F um den Eclipse Code Formatter zu verwenden.
- ▶ Java Doc:  
`http://docs.oracle.com/javase/7/docs/api/`
- ▶ Verwendet Kommentare um zu beschreiben was ihr implementiert.

## Übung 1: Einige Statistiken

- ▶ 66 Judge Benutzer
- ▶ Hello World
  - ▶ Submitted: 203
  - ▶ Total correct: 62
- ▶ Hello Name
  - ▶ Submitted: 163
  - ▶ Total correct: 54

## Aufgabe 2.1: Präzedenz der Operatoren in Java

Kategorie	Operator	Assoziativität
1. Unäre Operatoren	++ -- + - !	Linksassoziativ
2. Arithmetische Operatoren	* / % + -	Linksassoziativ
3. Schiebe-Operatoren	<< >> >>>	Linksassoziativ
4. Vergleichs-Operatoren	< <= > >= == != instanceof	Linksassoziativ
5. Logische Operatoren	& ^   &&    ?:	Linksassoziativ
6. Zuweisungs-Operatoren	=, op=, wobei op: + - / % & ! ^ << >>	Rechtsassoziativ

## Aufgabe 2.1: Assoziativität

Linksassoziativ:

$$\blacktriangleright A \text{ op } B \text{ op } C \iff (A \text{ op } B) \text{ op } C$$

Rechtsassoziativ:

$$\blacktriangleright A \text{ op } B \text{ op } C \iff A \text{ op } (B \text{ op } C)$$

## Aufgabe 2.1: Beispiel zu Präzedenz und Assoziativität

**Frage:** Ist der folgende Code-Ausschnitt syntaktisch korrekt?  
Wenn ja, was hat die Variable `b` für einen Wert nach der Ausführung des Codes?

```
boolean b = 7 >= 5 >= 3;
```

## Aufgabe 2.1: Beispiel zu Präzedenz und Assoziativität

**Frage:** Ist der folgende Code-Ausschnitt syntaktisch korrekt?  
Wenn ja, was hat die Variable `b` für einen Wert nach der Ausführung des Codes?

```
boolean b = 7 >= 5 >= 3;
```

**Antwort:** Der Code-Ausschnitt ist syntaktisch falsch.  
*Grund:* Der `<=` Operator ist Linksassoziativ, d.h.

```
boolean b = (7 >= 5) >= 3;
```

Da `(7 >= 5)` zu `true` evaluiert, wird in einem nächsten Schritt versucht ein `boolean` mit einem `int` zu vergleichen, jedoch verlangt der `<=` Operator zwei Variablen vom gleichen Typ.



## Aufgabe 3: Math Klasse

- ▶ Bietet Implementierungen von verschiedenen Mathematik-Funktionen, unter anderem auch `sin` und `sqrt` (Wurzel ziehen)
- ▶ `double r = Math.sqrt(1000);`
- ▶ Übersicht: <https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html>