

2. Tutorium

Datentypen, Operatoren, Referenzen, Strings

Tutorium 14

Péter Bohner | 09.11.2022



Inhaltsverzeichnis

1. Wiederholung
2. Artemis
3. Klassen und Objekte
 - 3.1 Einführung
 - 3.2 Objekte
 - 3.3 Modellierung
4. Operatoren
5. Datentypen II
6. Referenzen
 - 6.1 Speicherung
7. Scanner

Wiederholung
○○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Welche Befehlsformel innerhalb einer Klasse ermöglicht die Ausführung einer Java-Applikation?

Die *main*-Methode ist der Haupteinstiegspunkt in die Anwendung. Hier startet die Programmausführung.

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

Java Dateien werden **exakt** nach der Klasse benannt: Sonst Compilerfehler.

Wiederholung
●○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Wiederholung

Mit welchem Befehl kann ein Java-Programm kompiliert werden?

Kompilieren: `javac JavaApp.java`

Ausführen: `java JavaApp`

Wie kann eine Ausgabe auf der Konsole erzeugt werden?

`System.out.println();` oder `System.out.print();`

Sind Klassen Datentypen in Java? Ja...

```
class Car {  
    Body body;  
    Engine engine;  
    ...  
}
```

Mit welchem Schlüsselwort kann ein neues Objekt einer Klasse erzeugt werden?

`new Classname();`

Wiederholung primitive Datentypen

Typ	Erklärung	Wertebereich	Beispielwerte
boolean	Wahrheitswerte	true oder false	true, false
char	16-Bit-Unicode	0x0000 ... 0xffff	'A', '\n', '\u05D0'
byte	8-Bit-Integer	$-2^7 \dots 2^7 - 1$	12
short	16-Bit-Integer	$-2^{15} \dots 2^{15} - 1$	12
int	32-Bit-Integer	$-2^{31} \dots 2^{31} - 1$	12
long	64-Bit-Integer	$-2^{63} \dots 2^{63} - 1$	12L, 14L
float	32-Bit-Gleitk.	1,40239846E-45f... 3,40282347E+38f	9.81F, 0.3E-8F, 2f
double	64-Bit-Gleitk.	4,94065645841246544E-324... 1,79769131486231570E+308	9.81, 3e1

Wiederholung
○○●○○

Artemis
○○○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Wiederholung Variablen

„Platzhalter“ für Werte eines Datentyps

Deklaration

- Name und Datentyp der Variable
- `Datentyp Name;`

Zuweisung

- Wert der Variable
- `Name = Wert;`

Initialisierung

- Kombination aus Deklaration und Zuweisung
- `Datentyp Name = Wert;`

Wiederholung
○○●○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Wie greift man auf Attribute von Objekten zu?

- Zugriff mit `Objektname.Variablenname`
- Umgang wie mit „normalen“ Variablen
- Also Initialisierung mit:
⇒ `Objektname.Variablenname = Wert;`

Wiederholung
○○○●

Artemis
oooooooooooooooo

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○

Datentypen II
○○○

Referenzen
○○○

Scanner
○○

Ende
○○

Artemis

- neues Abgabesystem
- nutzt im Hintergrund GitLab und Jenkins
- Online-Editor: Code kann direkt komfortable in Artemis bearbeitet werden
- Oder: Aufgabe mit git clonen, lokal bearbeiten und Änderungen pushen

Registrierung

- Registrierung drücken und mit KIT Account einloggen
- E-Mail mit Link zur Passwort-Setzung für Artemis (GitLab, Jenkins)
- Anmeldung mit Passwort und Kurs beitreten

Wiederholung
○○○○

Artemis
●○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Git Basics

- `git clone url` - Clont das GitLab Repository lokal auf euren Rechner
- `git add file` - „staged“ (markiert) Datei für den nächsten Commit
- `git add -A` - Alle neuen/veränderten Dateien werden „gestaged“
- `git commit -m „Nachricht“` „gestagete“ Änderungen werden in einem Commit zusammengefasst
- `git pull` - Änderungen aus GitLab Repository werden heruntergeladen
- `git push` - Änderungen aus lokalen Repository werden hochgeladen
- mehr zu Git in SWT I oder in der [Git Dokumentation](#)
- ...oder Git über eclipse oder eine der vielen anderen [GUIs](#) benutzen

Nach jedem commit führt Artemis einen build aus und führt automatisiert die public Tests durch

Wiederholung
○○○○○

Artemis
●○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

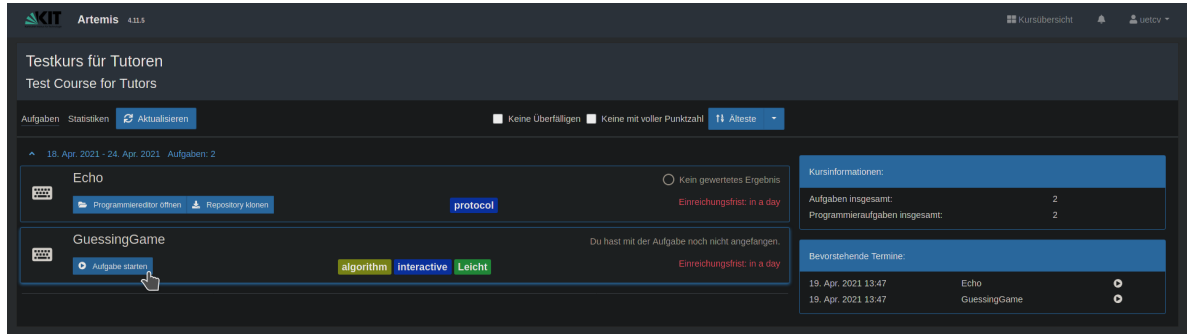
Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Artemis - Aufgabe starten



Testkurs für Tutoren
Test Course for Tutors

Aufgaben Statistiken Aktualisieren

Keine Überfalligen Keine mit voller Punktzahl Alteste

18. Apr. 2021 - 24. Apr. 2021 Aufgaben: 2

Echo ☐ Kein gewertetes Ergebnis
 Programmiereditor öffnen Repository klonen protocol Einreichungsfrist: in a day

GuessingGame Du hast mit der Aufgabe noch nicht angefangen.
 Aufgabe starten algorithm interactive Leicht Einreichungsfrist: in a day

Kursinformationen:

Aufgaben insgesamt:	2
Programmieraufgaben insgesamt:	2

Bevorstehende Termine:

19. Apr. 2021 13:47	Echo	<input type="radio"/>
19. Apr. 2021 13:47	GuessingGame	<input type="radio"/>

1. „Aufgabe starten“ anklicken

Wiederholung
○○○○○

Artemis
○○●○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

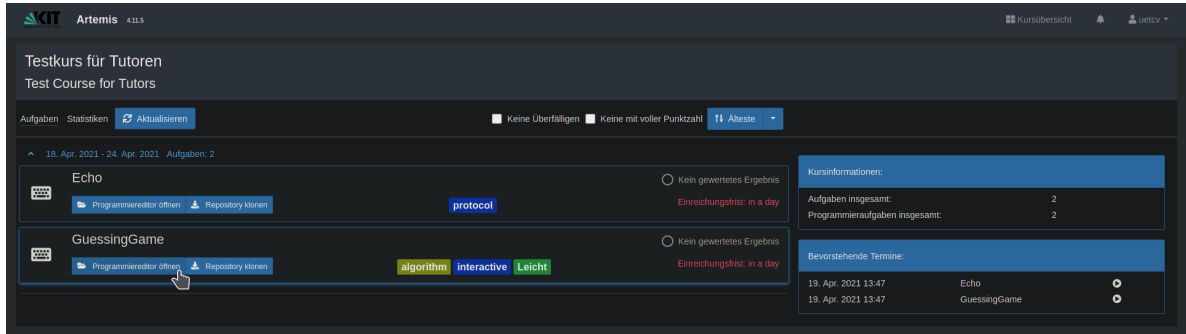
Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Artemis - Webeditor nutzen



Testkurs für Tutoren
Test Course for Tutors

Aufgaben Statistiken Aktualisieren

Keine Überfalligen Keine mit voller Punktzahl Alteste

18. Apr. 2021 - 24. Apr. 2021 Aufgaben: 2

Echo ☐ Kein gewertetes Ergebnis
Programmiereditor öffnen Repository klonen protocol
Einreichungsfrist: in a day

GuessingGame ☐ Kein gewertetes Ergebnis
Programmiereditor öffnen Repository klonen algorithm interactive Leicht
Einreichungsfrist: in a day

Kursinformationen:
Aufgaben insgesamt: 2
Programmieraufgaben insgesamt: 2

Bevorstehende Termine:
19. Apr. 2021 13:47 Echo
19. Apr. 2021 13:47 GuessingGame

2. „Programmiereditor öffnen“ anklicken

Wiederholung
○○○○○

Artemis
○○●○○○○○○○○○

Klassen und Objekte
○
○
○○○○○


Operatoren
○○○○


Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○


Artemis 4.11.5

 Kursübersicht
 
 uetcv

Echo [5 Punkte]

☐ Kein gewertetes Ergebnis
 Aktualisieren
Speichern
Abenden

Dateibrowser

+
 +
 ↗

src

✓ Gespeichert.
 ✓ Abgesendet.

Wähle eine Datei aus, um zu beginnen!

Aufgabenstellung

Echo

Write a program, that reads lines from System.in and outputs the lines again. Remember to send a newline character "ln" after each line or use System.out.println.

Use only java.io, java.lang

The program quits when reading the special input "quit".

- Try to run in all sorts of Timeouts
- Try to use restricted packages

Please report back (eg. Tutor-Forum in Ilias):

- How many tests were you able to see?
- How useful are the error-messages? Please be specific in what you did and what you expect
- Did the tests run as expected?
- Did the system crash? How?

Wiederholung
○○○○○

Artemis
○○○○●○○○○○○○

Klassen und Objekte
○
○
○○○○○

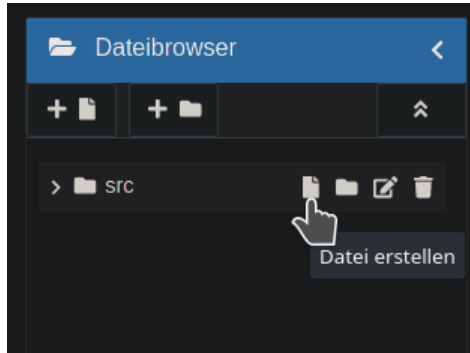
Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○○

Ende
○○



3. Über den Dateibrowser können neue Dateien angelegt werden
4. Code schreiben

Wiederholung
○○○○○

Artemis
○○○○○●○○○○○○○

Klassen und Objekte
○
○
○○○○○

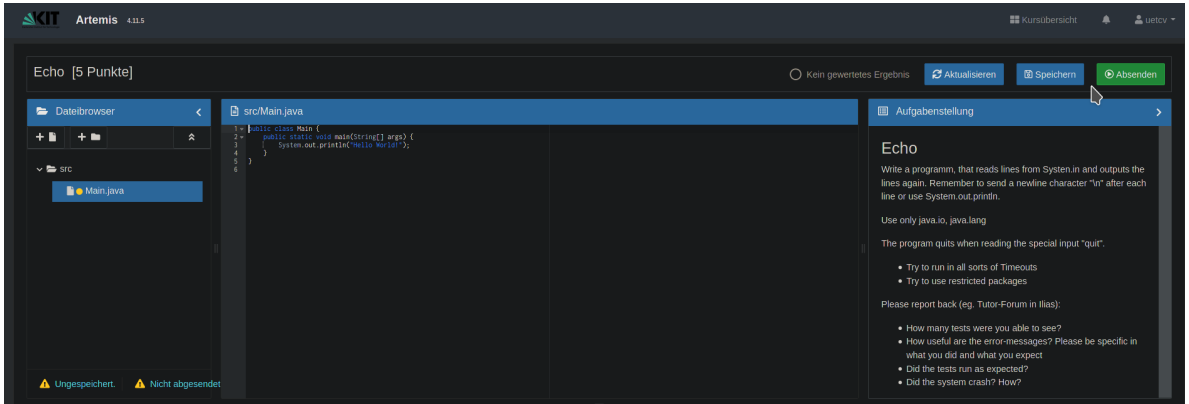
Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○



5. „Speichern“ um Änderung zu sichern

6. „Absenden“ um Projekt zu builden, testen und abzugeben

Wiederholung
○○○○○

Artemis
○○○○○○●○○○○○

Klassen und Objekte
○
○
○○○○○

Operatoren
○○○○

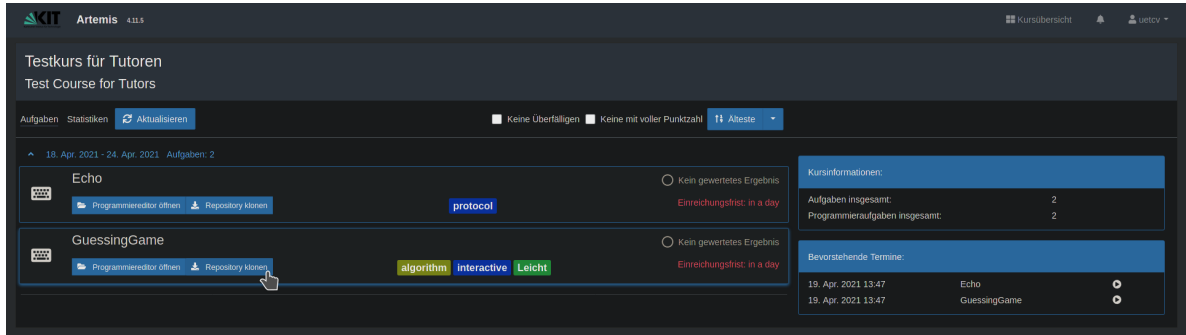
Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Artemis - Lokal arbeiten



Artemis 4.11.5

Kursübersicht

uetcv

Testkurs für Tutoren
Test Course for Tutors

Aufgaben Statistiken Aktualisieren

Keine Überfalligen Keine mit voller Punktzahl Alteste

18. Apr. 2021 - 24. Apr. 2021 Aufgaben: 2

Echo

Programmieditor öffnen Repository klonen protocol

Kein gewertetes Ergebnis

Einreichungsfrist: in a day

GuessingGame

Programmieditor öffnen Repository klonen algorithm interactive Leicht

Kein gewertetes Ergebnis

Einreichungsfrist: in a day

Kursinformationen:

Aufgaben insgesamt:	2
Programmieraufgaben insgesamt:	2

Bevorstehende Termine:

19. Apr. 2021 13:47	Echo	○
19. Apr. 2021 13:47	GuessingGame	○

2. „Repository klonen“ anklicken

Wiederholung
○○○○○

Artemis
○○○○○○●○○○○○

Klassen und Objekte
○
○
○
○○○○

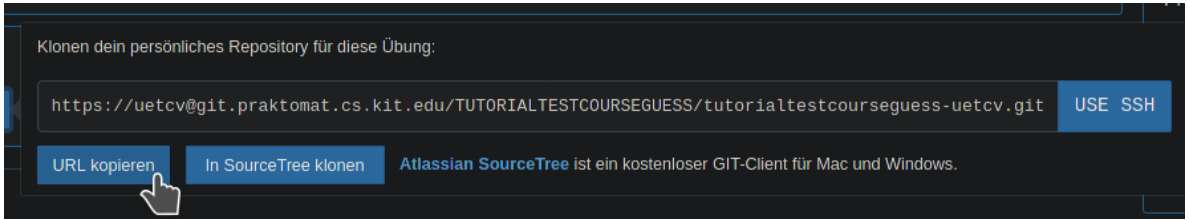
Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○



3. „URL kopieren“ anklicken
4. eclipse öffnen
5. In eclipse: „File“ → „Import“
6. „Git“ → „Projects from Git“
7. „Next“ klicken

Wiederholung
○○○○○

Artemis
○○○○○○○●○○○○○

Klassen und Objekte
○
○
○○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Source Git Repository

Enter the location of the source repository.

Location

URI:

Host:

Repository path:

Connection

Protocol:

Port:

Authentication

User:

Password:

☐ Store in Secure Store

8. Unter „URI“ kopierte URL einfügen
9. Unter „Authentication“ Artemis Passwort eintragen
10. 3 x „Next“ klicken
11. „Finish“ klicken

Wiederholung
○○○○○

Artemis
○○○○○○○○○●○○○

Klassen und Objekte
○
○
○
○○○○○

Operatoren
○○○○

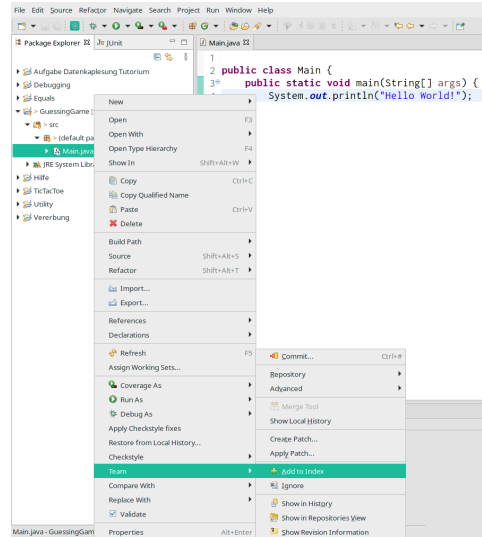
Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

12. Repository ist nun lokal verfügbar
13. Nun kann normal lokal in eclipse programmiert werden
14. geänderte/erstellte Dateien müssen nun „gestaged“ und „committed“ werden
15. „Rechtsklick“ auf geänderte Datei → „Team“ → „Add to Index“
16. wurden alle geänderten Dateien „gestaged“:
„Rechtsklick“ auf Projekt → „Team“ → „Commit“



Wiederholung
○○○○○

Artemis
○○○○○○○○○○●○○

Klassen und Objekte
○
○
○○○○

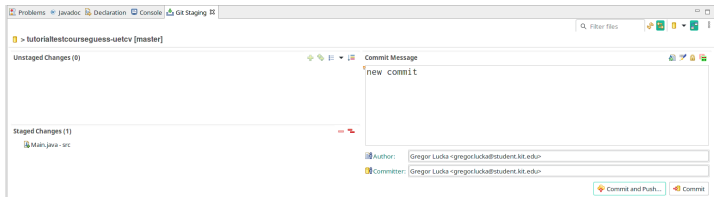
Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○



17. beliebige „Commit Message“ wählen
18. „Commit and Push...“ klicken
19. „Preview“ klicken
20. „Push“ klicken
21. „Close“ klicken
22. Änderungen sind nun in Artemis eingebucht und die Tests werden ausgeführt
23. Ergebnisse der Tests sind in Artemis verfügbar

Wiederholung
○○○○○

Artemis
○○○○○○○○○○○●○○

Klassen und Objekte
○
○
○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Fragen?

Wiederholung
○○○○○

Artemis
○○○○○○○○○○○○●

Klassen und Objekte
○
○
○
○○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Klassen und Objekte

Allgemein

- Modellierung der Realität
- Objekte mit Eigenschaften und Verhalten

Was ist eine Klasse?

- Objekte, die ähnliche Eigenschaften und dasselbe Verhalten haben, werden zusammengefasst
- Klasse stellt Bauplan für Objekte dar
- Attribute und Methoden werden festgelegt

Wiederholung
○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
●
○
○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Was ist ein Objekt?

Instanzen, die aus dem Bauplan entstehen können

Charakterisierung eines Objekts

- **Identität:** Bleibt immer gleich, egal ob sich der Zustand ändert
- **Zustand:** Menge aller Attributwerte
- **Verhalten:** Methoden

Wiederholung
○○○○

Artemis
oooooooooooooooo

Klassen und Objekte
○
●
○○○○

Operatoren
○○○

Datentypen II
○○○

Referenzen
○○○

Scanner
○○

Ende
○○

Klasse

Wie bauen die Klasse **Laptop**:

Eigenschaften

- Arbeitsspeicher
- Tastatur
- Display
- Ist es ein Convertible?

Verhalten

- Hochfahren
- Aufklappen
- Programm ausführen

Wiederholung
○○○○

Artemis
oooooooooooooooo

Klassen und Objekte
○
○
●○○○

Operatoren
○○○

Datentypen II
○○○

Referenzen
○○○

Scanner
○○

Ende
○○

Attribute

```
class Laptop {  
    int ram;  
    Keyboard keyboard;  
    Display display;  
    boolean isConvertible;  
  
    void powerOn() { ... }  
    void open(){ ... }  
    void runProgram(String name) { ... }  
}
```

Wiederholung
○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○●○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Wie erzeugt man ein Objekt?

Der Ausdruck **new** `Classname()`; erzeugt ein neues Objekt.

Mit Laptop `laptop = new Laptop();` wird das Objekt `laptop` vom Typ `Laptop` erstellt.

`laptop.ram = 16;` ändert den Wert des Arbeitsspeichers.

Mehr zu Konstruktoren kommt bald.

Wiederholung
○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○○●○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Aufgaben

Teil A

Modelliert eine Klasse Motor. Ein Motor besitzt eine gewisse Anzahl an PS und Zylindern. Er hat außerdem ein Gewicht und Hubraum (in l). Wählt geeignete Datentypen.

Teil B

Modelliert eine Klasse Karosserie. Sie hat ein Gewicht, Anzahl an Sitzplätzen und eine bestimmte Form (Name der Form).

Teil C

Überlegt euch, wie man aus den beiden gegebenen Klassen in Java-Code eine Klasse Auto entwerfen könnte. Zur Info: Ein Auto hat eine Karosserie und einen Motor.

Wiederholung
○○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○○●○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Teil A und B

```
class Engine {  
    int horsepower;  
    int numberOfCylinders;  
    int weight;  
    double engineDisplacement;  
}
```

```
class Body {  
    int weight;  
    int numberOfSeats;  
    String type;  
}
```

Teil C

```
class Car {  
    Body body;  
    Engine engine;  
    ...  
}
```

- Klassen können als eigener Datentyp genutzt werden
- Attribute können also als Typ eine bestimmte Klasse haben
- Sie nutzen deren Eigenschaften und Verhalten

Wiederholung
○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○○○○●

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Arithmetische Operationen

		Präzedenz
unäres -/+		1
+	Addition	3
-	Subtraktion	3
*	Multiplikation	2
/	Division	2
%	Modulo (Rest bei Ganzzahldivision)	2

Präzedenz

Der Wert gibt an wie stark der Operator bindet.

Je **kleiner** der Wert, desto **stärker** bindet der Operator.

Beispiel aus der Mathematik: $5 \cdot 3 + 2 = 17 \neq 25$ da \cdot stärker als $+$ bindet

Wiederholung
○○○○

Artemis
oooooooooooooooo

Klassen und Objekte
○
○
○○○○

Operatoren
●○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

In-/Dekrement-Operator

++ erhöht Variable um eins
-- verringert Variable um eins

Notationen:

Präfix	$y = ++x$	$x = x + 1;$	$y = x;$
Postfix	$y = x++$	$y = x;$	$x = x + 1;$

Analog für --

Wiederholung
○○○○

Artemis
oooooooooooooooo

Klassen und Objekte
○
○
○
○○○○

Operatoren
●○○

Datentypen II
○○○

Referenzen
○○○

Scanner
○○

Ende
○○

Vergleichs Operationen

Liefern Wahrheitswert (boolean):

	Präzedenz
< kleiner	5
<= kleiner-gleich (entspricht \leq)	5
> größer	5
>= größer-gleich (entspricht \geq)	5
== Gleichheit	6
!= Ungleichheit	6

Wiederholung
○○○○

Artemis
oooooooooooooooo

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○●○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Operationen auf boolean-Werten

		Präzedenz
!	Negation	1
&&	logisches Und	10
	logisches Oder	11

Wiederholung
○○○○○

Artemis
○○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○○

Operatoren
○○○●

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Ungenauigkeit double

```
double a = 2.0;
double b = 1.9;
double c = a - b;
```

Wert von c: 0.100000000000000009

Zahlen sind nur **endlich** genau \Rightarrow *Rundungsfehler*

Achtung bei Vergleichen

Statt Vergleich mit `actual == expected`, geeignetes Delta verwenden:

`Math.abs(expected - actual) < delta`

Hier: Statt `c == 0.1`, verwende `Math.abs(c - 0.1) < 0.0001`

Wiederholung
○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

Datentypen II
●○○○

Referenzen
○○○○

Scanner
○○

Ende
○○

Was ist die Ausgabe des folgenden Codes?

```
byte a = 127;  
System.out.println(a++ + ", " + a);
```

127, -128

Warum ist das so?

Der Wertebereich von byte geht von -2^7 (-128) bis $2^7 - 1$ (127).

Inkrementieren vom größten Wert

⇒ kleinster Wert

Dekrementieren vom kleinsten Wert

⇒ größter Wert

Wie ist bei Gleitkommazahlen?

Overflow:

```
System.out.println(Double.MAX_VALUE);  
System.out.println(Double.MAX_VALUE + 1.0);  
System.out.println(Double.MAX_VALUE * 2);
```

```
1.7976931348623157E308  
1.7976931348623157E308  
INFINITY
```

Underflow:

```
System.out.println(Math.pow(2, -1074));  
System.out.println(Math.pow(2, -1075));
```

```
4.9E-324  
0.0
```

Benennungs-Koventionen

Variablen, Attribute und Methoden
Klassen, Enums und Interfaces
Klassenkonstanten, Einträge in Enums
Pakete

lowerCamelCase
UpperCamelCase
GROSS_GESCHRIEBEN
kleinbuchstaben (umgedrehte Domain)

userCounter
StringBuilder
GRAVITATION_EARTH
com.java.util

Wiederholung
○○○○

Artemis
oooooooooooooooo

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

Datentypen II
○○○●

Referenzen
○○○○

Scanner
○○

Ende
○○

Speicherung

Abstrakter Speicheraufbau:

x	y	z	a	b	c
0	2	-2	1.6f	-2.5f	50
int	int	int	float	float	int
01	02	03	04	05	06

- Variablen werden an aufeinanderfolgenden Adressen im Speicher gespeichert
- Pro Adresse werden immer nur 8 Bits gespeichert
- int braucht eigentlich 4 Adressen (hier vereinfacht)

Wiederholung
○○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○

Operatoren
○○○○

Datentypen II
○○○○

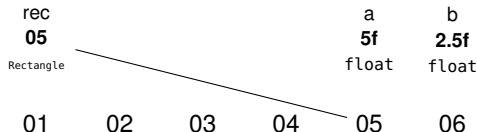
Referenzen
●○○○

Scanner
○○

Ende
○○

Speicherung von Objekten

- Für ein Objekt wird immer nur eine Referenz auf die Speicheradressen der Objektattribute gespeichert
- Die Objektvariable zeigt auf die Objektidentität
- Beispiel: Rectangle rec mit rec.a = 5f und rec.b = 2.5f

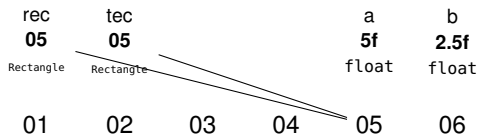


Zuweisung von Objekten

An der Situation von eben (Rectangle rec mit $\text{rec.a} = 5f$ und $\text{rec.b} = 2.5f$) wird folgendes geändert:

Rectangle tec = rec;

Wie sieht der Speicher anschließend aus?



null

- Wichtiges Element!
- `Rectangle rec = null;`
- Referenz auf nichts
- Es wird auf „kein Objekt“ referenziert
- Zugriff auf das Objekt führt zur Exception

```
rec.a = 10;
```

⇒ `java.lang.NullPointerException`

Wiederholung
oooo

Artemis
oooooooooooooooo

Klassen und Objekte
o
o
o
oooo

Operatoren
oooo

Datentypen II
oooo

Referenzen
ooo●

Scanner
oo

Ende
oo

Scanner

Für Eingaben während das Programm läuft, verwende die Klasse Scanner:

Importieren mit `import java.util.Scanner`; vor der Klassendefinition!

Neuen Scanner erstellen mit: `new Scanner(System.in)`

Einlesen mit:

<code>nextLine()</code>	für Strings
<code>nextInt()</code>	für Ganzzahlen
<code>nextDouble()</code>	für Gleitkommazahlen
...	...

Wiederholung
○○○○○

Artemis
○○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
●○

Ende
○○

Scanner - Beispiel

```
import java.util.Scanner;

class ReadTerminal {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Gib eine Zahl ein!");
        int input = scanner.nextInt();
        System.out.println("Deine Zahl war " + input + ".");
    }
}
```

```
$ java ReadTerminal
Gib eine Zahl ein!
> 5
Deine Zahl war 5.
```

Wiederholung
oooo

Artemis
oooooooooooooooo

Klassen und Objekte
o
o
oooo

Operatoren
oooo

Datentypen II
oooo

Referenzen
oooo

Scanner
o●

Ende
oo

Wie weit sind wir?

02.11.2022
 Orga
 Java
 Klassen und Objekte

09.11.2022
 Datentypen
 Operatoren
 Referenzen

16.11.2022
 Schleifen
 Konstruktoren
 JavaDoc
 Kontrollstrukturen

Wiederholung
 ○○○○

Artemis
 ○○○○○○○○○○○○

Klassen und Objekte
 ○
 ○
 ○○○○

Operatoren
 ○○○○

Datentypen II
 ○○○○

Referenzen
 ○○○○

Scanner
 ○○

Ende
 ●○

Bis zum nächsten Tutorium am 16.11.2022!

Wiederholung
○○○○○

Artemis
○○○○○○○○○○○○○○

Klassen und Objekte
○
○
○
○○○○○

Operatoren
○○○○

Datentypen II
○○○○

Referenzen
○○○○

Scanner
○○

Ende
○●