

4. Tutorium JavaDoc, Methoden, Arrays

Tutorium 14

Péter Bohner | 23.11.2022



Tutorium 14

22-11-23



23.11.2022



Programmieren Tutorium

Alle 24 von euch haben Abgegeben, super!

Péter Bohner: Tutorium 14

Übungsblatt 1 Wiederholung JavaDoc Methoden Arrays Aufgabe 00000 Ende 000 00000 00000000000 0000000000 0

Tutorium 14 Übungsblatt 1 Übungsblatt 1

Alle 24 von euch haben Abbegeben, auper!



- Alle 24 von euch haben Abgegeben, super!
- WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE!
- → Kompilierfehler, Mandatory Test fails nicht ignorieren



Tutorium 14 Übungsblatt 1 Übungsblatt 1

 Alle 24 von euch haben Abgegeben, auger! WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE! -- Kompilierfehler, Mandatory Test fails nicht ignorieren

Péter Bohner: Tutorium 14 Programmieren Tutorium

23.11.2022

23.11.2022



Alle 24 von euch haben Abgegeben, super!

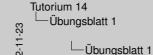
Péter Bohner: Tutorium 14

- WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE!

 → Kompilierfehler, Mandatory Test fails nicht ignorieren
- Dateien: Groß- und Kleinschreibung beachten! Dateiname muss genau der Klassenname sein (.java nicht vergessen) Dateien nur im src/ Ordner ablegen.



Programmieren Tutorium



WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE!
 Kompilerfelbler, Mandatory Teat falls nicht ignorieren

Alle 24 von euch haben Abgegeben, super!

 Dateier: Groß- und Kleinschreibung beschlen! Dateiname muss genau der Klassenname sein (Java nicht vergessen) Dateien nur im src! Orcher ablegen.

23.11.2022



Programmieren Tutorium

Alle 24 von euch haben Abgegeben, super!

Péter Bohner: Tutorium 14

- WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE! → Kompilierfehler, Mandatory Test fails nicht ignorieren
- Dateien: Groß- und Kleinschreibung beachten! Dateiname muss genau der Klassenname sein (.java nicht vergessen) Dateien nur im src/ Ordner ablegen.
- Aufgabenstellungen genau lesen: Wer lesen kann ist klar im Vorteil!



Tutorium 14 └─Übungsblatt 1

 Alle 24 von euch haben Abgegeben, super! WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE;

→ Kompilierfehler, Mandatory Test fails nicht ignorieren Dateien: Groß- und Kleinschreibung beschten! Dateiname muss genau der Klassenname sein Ligwa nich

Aufgebenstellungen genau lesen: Wer lesen kann ist klar im Vorteilt.

└─Übungsblatt 1

23.11.2022



Programmieren Tutorium

Alle 24 von euch haben Abgegeben, super!

Péter Bohner: Tutorium 14

- WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE! → Kompilierfehler, Mandatory Test fails nicht ignorieren
- Dateien: Groß- und Kleinschreibung beachten! Dateiname muss genau der Klassenname sein (.java nicht vergessen) Dateien nur im src/ Ordner ablegen.
- Aufgabenstellungen genau lesen: Wer lesen kann ist klar im Vorteil!
- Beachtet bitte Namenskonventionen und richtige Einrückung (4 Spaces)



Tutorium 14 Übungsblatt 1 └─Übungsblatt 1

 Alle 24 von euch haben Abgegeben, super! WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE; - Kompilierfehler, Mandatory Test falls nicht ignorieren

Dateien: Groß- und Kleinschreibung beschten! Dateiname muss genau der Klassenname sein Ligwa nich

Aufgebenstellungen genau lesen: Wer lesen kann ist klar im Vorteilt.

Beachtet bitte Namerskonventionen und richtige Einrückung (4 Spaces)



Programmieren Tutorium

Alle 24 von euch haben Abgegeben, super!

Péter Bohner: Tutorium 14

- WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE!

 → Kompilierfehler, Mandatory Test fails nicht ignorieren
- Dateien: Groß- und Kleinschreibung beachten! Dateiname muss genau der Klassenname sein (.java nicht vergessen) Dateien nur im src/ Ordner ablegen.
- Aufgabenstellungen genau lesen: Wer lesen kann ist klar im Vorteil!
- Beachtet bitte Namenskonventionen und richtige Einrückung (4 Spaces)
- Aufgabe A: bei allen gut

23.11.2022



Tutorium 14

Übungsblatt 1

Übungsblatt 1

- Alle 24 von euch haben Abgegeben, auper!
 WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH O PUNKTE!

 Komplierfelrier, Mandetory Teat labs nicht isnonieren
- Dateien: Groß- und Kleinschneibung besichten! Dateiname muss genau der Klassenname sein (java nich
- vergessen) Dateien nur im src/ Ordner ablegen.
 - Aufgebenstellungen genau lesen: Wer lesen kann ist klar im Vorteil!
 - Beachtet bitte Namenskonventionen und richtige Einrückung (4 Spaces)
 - Aufgabe A: bei allen gut



- Alle 24 von euch haben Abgegeben, super!
- WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE! → Kompilierfehler, Mandatory Test fails nicht ignorieren
- Dateien: Groß- und Kleinschreibung beachten! Dateiname muss genau der Klassenname sein (.java nicht vergessen) Dateien nur im src/ Ordner ablegen.
- Aufgabenstellungen genau lesen: Wer lesen kann ist klar im Vorteil!
- Beachtet bitte Namenskonventionen und richtige Einrückung (4 Spaces)
- Aufgabe A: bei allen gut
- Aufgabe B: isEmpty() ignoriert kein Whitespace, ist also falsch. isBlank() ist richtig

Übungsblatt Wiederholung JavaDoc Methoden Aufgabe Ende 00000 0

Tutorium 14 Übungsblatt 1

Übungsblatt 1

 Alle 24 von euch haben Abgegeben, super! WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE;

→ Kompilierfehler, Mandatory Test faits nicht ignorierer Dateien: Groß- und Kleinschreibung beschten! Dateiname muss genau der Klassenname sein Ligwa nich

verpessen) Dateien nur im arc/ Ordner ablegen.

Aufgabenstellungen genau lesen: Wer lesen kann ist klar im Vorteil!

Beachtet bitte Namerskonventionen und richtige Einrückung (4 Spaces)

Aufgabe A: bei allen out

Aufgabe B: isEmpty() ignorient kein Whitespace, ist also falsch. isBlank() ist richtig



Programmieren Tutorium

- Alle 24 von euch haben Abgegeben, super!
- WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE! → Kompilierfehler, Mandatory Test fails nicht ignorieren
- Dateien: Groß- und Kleinschreibung beachten! Dateiname muss genau der Klassenname sein (.java nicht vergessen) Dateien nur im src/ Ordner ablegen.
- Aufgabenstellungen genau lesen: Wer lesen kann ist klar im Vorteil!
- Beachtet bitte Namenskonventionen und richtige Einrückung (4 Spaces)
- Aufgabe A: bei allen gut

23.11.2022

- Aufgabe B: isEmpty() ignoriert kein Whitespace, ist also falsch. isBlank() ist richtig
- Aufgabe C: Nicht vergessen den Scanner zu schließen.

Péter Bohner: Tutorium 14



Tutorium 14 Übungsblatt 1

 Aufgebenstellungen genau lesen: Wer lesen kann ist klar im Vorteil? —Übungsblatt 1 Aufgabe A: bei allen out

→ Kompilierfehler, Mandatory Test faits nicht ignorierer verpessen) Dateien nur im arc/ Ordner ablegen. Beachtet bitte Namerskonventionen und richtige Einrückung (4 Spaces)

Alle 24 von euch haben Abgegeben, super!

 Aufgabe B: isEmpty() ignoriert kein Whitespace, ist also falsch. isBlank() ist richtig Aufgabe C: Nicht vergessen den Scanner zu schließen.

WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE;

Dateien: Groß- und Kleinschreibung beschten! Dateiname muss genau der Klassenname sein Ligwa nich



Programmieren Tutorium

- Alle 24 von euch haben Abgegeben, super!
- WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE! → Kompilierfehler, Mandatory Test fails nicht ignorieren
- Dateien: Groß- und Kleinschreibung beachten! Dateiname muss genau der Klassenname sein (.java nicht vergessen) Dateien nur im src/ Ordner ablegen.
- Aufgabenstellungen genau lesen: Wer lesen kann ist klar im Vorteil!
- Beachtet bitte Namenskonventionen und richtige Einrückung (4 Spaces)
- Aufgabe A: bei allen gut

23.11.2022

- Aufgabe B: isEmpty() ignoriert kein Whitespace, ist also falsch. isBlank() ist richtig
- Aufgabe C: Nicht vergessen den Scanner zu schließen.

Péter Bohner: Tutorium 14

 Aufgabe D: Keine Initialisierungen nur Deklarationen, keine main-Methode, Adresse nicht als String, Jede Klasse in eigene Datei.



Tutorium 14 Übungsblatt 1 Übungsblatt 1

 Alle 24 von euch haben Abgegeben, super! WENN SICH ARTEMIS BEI IRGENDWAS BESCHWERT, DANN GIBT ES AUTOMATISCH 0 PUNKTE; → Kompilierfehler, Mandatory Test faits nicht ignorierer

Dateien: Groß- und Kleinschreibung beschten! Dateiname muss genau der Klassenname sein Ligwa nich

verpessen) Dateien nur im arc/ Ordner ablegen. Aufgebenstellungen genau lesen: Wer lesen kann ist klar im Vorteil?

Beachtet bitte Namerskonventionen und richtige Einrückung (4 Spaces)

Aufgabe A: bei allen out

 Aufgabe B: isEmpty() ignoriert kein Whitespace, ist also falsch. isBlank() ist richtig Aufgebe C: Nicht vergessen den Scanner zu schließen.

 Aufgabe D: Keine Initialisierungen nur Deklarationen, keine main-Methode, Adresse nicht als String, Jedi Klasse in eigene Datei.



Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht?

Wiederholung JavaDoc Methoden Ende Arrays •00 00000 00000000000

Aufgabe 0000000000 00000 0

Tutorium 14 ─Wiederholung

─Wiederholung

Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht?

Programmieren Tutorium

Übungsblatt 1

23.11.2022

Péter Bohner: Tutorium 14



Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht? final

JavaDoc

00000

Methoden

00000000000

Arrays

0000000000

Aufgabe

00000

Ende

0

Übungsblatt 1

23.11.2022

•00

Péter Bohner: Tutorium 14

Wiederholung

Programmieren Tutorium

Tutorium 14 ─Wiederholung

─Wiederholung

Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht?



Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht? final

Füllt die gegebene Tabelle aus (x = z = true).

Ausdruck	y = ?	Ergebnis
x && y == y && x		true
(z ^ z) && y		
y == false y == true		
!((z z) & !y)	false	



Tutorium 14
—Wiederholung
—Wiederholung

final. $\label{eq:final_problem}$ Füllt die gegebene Tabelle aus (x=z=trus).

Füllt die gegebene Tabelle aus (x = 2 =

Ausdruck y = 7 Ergebnis x id, y = y id, x true $(2 \wedge 2) \text{ id}, y$ y = false || y = true|

Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht?



Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht?

Füllt die gegebene Tabelle aus (x = z = true).

Ausdruck	y = ?	Ergebnis
x && y == y && x	true	true
(z ^ z) && y		
y == false y == true		
!((z z) & !y)	false	

Tutorium 14
—Wiederholung

Wiederholung

Mit welchem Schfüsselwort werden Variablen unveränderbar gemacht? final. Füllt die gegebene Tabelle aus (x = z = true).

Füllt die gegebene Tabelle aus (x = 2 :

Ausdruck	y = ?	Ergebnis
x 55 y = y 55 x	true	true
(z ^ z) 55 y		
y false y true		
!((z z) 6 ly)	false	



Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht? final

Füllt die gegebene Tabelle aus (x = z = true).

Ausdruck	y = ?	Ergebnis
x && y == y && x	true	true
(z ^ z) && y	false/true	false
y == false y == true		
!((z z) & !y)	false	

9 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14
—Wiederholung
—Wiederholung

Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht? finel.

Stillt die gegebene Yehelle ever (x - x - took)

Füllt die gegebene Tabelle aus (x = 2 =

Ausdruck y = ? Ergebnis $x \le x = y \le x$ true true $(z \land z) \le y = false$ false $y = false \mid |y = true$ $(|z \mid z) \le y | false$



Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht? final

Füllt die gegebene Tabelle aus (x = z = true).

Ausdruck	y = ?	Ergebnis
x && y == y && x	true	true
(z ^ z) && y	false/true	false
y == false y == true	false/true	true
!((z z) & !y)	false	

Übungsblatt 1 Wiederholung JavaDoc Methoden Aufgabe Ende •00 00000 00000000000 0000000000 00000 0

Tutorium 14 ─Wiederholung ─Wiederholung

Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht?



Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht? final

Füllt die gegebene Tabelle aus (x = z = true).

Ausdruck	y = ?	Ergebnis
x && y == y && x	true	true
(z ^ z) && y	false/true	false
y == false y == true	false/true	true
!((z z) & !y)	false	false

 Tutorium 14

Wiederholung

Wiederholung

Fullt die gegebene Tabelle aus (x = 2 =

Ausdruck y = 2 Ergebnis x id y = y id x true true $(z^2 - z) \text{ id} y \text{ false/true}$ false y = false || y = true false/true true ((z + z) id) false false

false

false

Mit welchem Schlüsselwort werden Variablen unveränderbar gemacht?



Welche Ausgabe erzeugt der folgende Code:

```
String s1 = "Hallo ";
String s2 = s1;
s2 = s2 + "Welt";
System.out.println("s1: " + s1 + ", s2: " + s2);
```

JavaDoc

Methoden Aufgabe Ende Arrays 00000000000 0000000000 00000 0

Übungsblatt 1 Wiederholung 000 00000 Tutorium 14 ─Wiederholung ─Wiederholung

Welche Ausgabe erzeugt der folgende Code

Strings sind immutable (unveränderbar), deswegen wird bei der Konkatenation ein neues Objekt erstellt



Welche Ausgabe erzeugt der folgende Code:

```
String s1 = "Hallo ";
String s2 = s1;
s2 = s2 + "Welt";
System.out.println("s1: " + s1 + ", s2: " + s2);
```

Ausgabe: s1: Hallo , s2: Hallo Welt da Strings immutable sind

Übungsblatt 1 Wiederholung JavaDoc Aufgabe Ende Methoden Arrays 00000 0

Tutorium 14 ─Wiederholung

─Wiederholung

Welche Ausgabe erzeugt der folgende Code

Strings sind immutable (unveränderbar), deswegen wird bei der Konkatenation ein neues Objekt erstellt

000 00000 000000000000 0000000000 23.11.2022 Péter Bohner: Tutorium 14

Programmieren Tutorium



Welche Ausgabe erzeugt der folgende Code:

```
String s1 = "Hallo ";
String s2 = s1;
s2 = s2 + "Welt";
System.out.println("s1: " + s1 + ", s2: " + s2);
```

Ausgabe: s1: Hallo , s2: Hallo Welt da Strings immutable sind Womit vergleicht man Strings?



9 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14

Wiederholung

Wiederholung

Wiederholung

Wiederholung

Strings sind immutable (unveränderbar), deswegen wird bei der Konkatenation ein neues Objekt erstellt



Welche Ausgabe erzeugt der folgende Code:

```
String s1 = "Hallo ";
String s2 = s1;
s2 = s2 + "Welt";
System.out.println("s1: " + s1 + ", s2: " + s2);
```

```
Ausgabe: s1: Hallo , s2: Hallo Welt da Strings immutable sind
```

Womit vergleicht man Strings?

```
string1.equals(string2)
```

Übungsblatt 1 Wiederholung Ende JavaDoc Methoden Arrays Aufgabe 0

000 00000 000000000000 0000000000 00000

Tutorium 14 ─Wiederholung

Welche Ausgabe erzeugt der folgende Code

─Wiederholung

Strings sind immutable (unveränderbar), deswegen wird bei der Konkatenation ein neues Objekt erstellt



Welche Ausgabe erzeugt der folgende Code:

```
String s1 = "Hallo ";
String s2 = s1;
s2 = s2 + "Welt";
System.out.println("s1: " + s1 + ", s2: " + s2);
```

Ausgabe: s1: Hallo , s2: Hallo Welt da Strings immutable sind Womit vergleicht man Strings?

```
string1.equals(string2)
```

Sei int i = 1. Welchen Wert hat i nach der Ausführung folgendes Ausdrucks? (1 == 1) || (i++ == 5)

```
Übungsblatt 1
                          Wiederholung
                                                     JavaDoc
                                                                           Methoden
                                                                                                          Arrays
                                                                                                                                     Aufgabe
                                                                                                                                                           Ende
```

000 00000 0000000000 00000 0 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium Tutorium 14 ─Wiederholung ─Wiederholung



Strings sind immutable (unveränderbar), deswegen wird bei der Konkatenation ein neues Objekt erstellt



Welche Ausgabe erzeugt der folgende Code:

```
String s1 = "Hallo ";
String s2 = s1;
s2 = s2 + "Welt";
System.out.println("s1: " + s1 + ", s2: " + s2);
```

Ausgabe: s1: Hallo , s2: Hallo Welt da Strings immutable sind

```
Womit vergleicht man Strings?
```

string1.equals(string2)

Sei int i = 1. Welchen Wert hat i nach der Ausführung folgendes Ausdrucks? (1 == 1) || (i++ == 5) 1, wegen der Kurzschlussauswertung wird die zweite Bedinung nie ausgeführt

```
Übungsblatt 1
                           Wiederholung
                                                      JavaDoc
                                                                            Methoden
                                                                                                                                       Aufgabe
                                                                                                                                                             Ende
```

0.00 00000 Tutorium 14 -Wiederholung ─Wiederholung

Sei int i = 1. Welchen Wert hat i nach der Ausführung folgendes Ausdrucks? (1 == 1) || (i++ == 5) 1. wegen der Kurzschlussauswertung wird die zweite Bedinung nie ausgeführt

Strings sind immutable (unveränderbar), deswegen wird bei der Konkatenation ein neues Objekt erstellt



Welche Ausgabe erzeugt der folgende Code:

```
String s1 = "Hallo ";
String s2 = s1;
s2 = s2 + "Welt";
System.out.println("s1: " + s1 + ", s2: " + s2);
```

Ausgabe: s1: Hallo , s2: Hallo Welt da Strings immutable sind

```
Womit vergleicht man Strings?
```

string1.equals(string2) Sei int i = 1. Welchen Wert hat i nach der Ausführung folgendes Ausdrucks? (1 == 1) || (i++ == 5)

1, wegen der Kurzschlussauswertung wird die zweite Bedinung nie ausgeführt Mit welchem Schlüsselwort verlässt man einen case in einem switch-statement?

```
Übungsblatt 1
                         Wiederholung
                                                   JavaDoc
                                                                         Methoden
                                                                                                                                 Aufgabe
                                                                                                                                                      Ende
                         0.00
                                                                                                                                00000
```

Tutorium 14 -Wiederholung ─Wiederholung



Strings sind immutable (unveränderbar), deswegen wird bei der Konkatenation ein neues Objekt erstellt



Welche Ausgabe erzeugt der folgende Code:

```
String s1 = "Hallo ";
String s2 = s1;
s2 = s2 + "Welt";
System.out.println("s1: " + s1 + ", s2: " + s2);
```

Ausgabe: s1: Hallo , s2: Hallo Welt da Strings immutable sind

```
Womit vergleicht man Strings?
```

string1.equals(string2)

Sei int i = 1. Welchen Wert hat i nach der Ausführung folgendes Ausdrucks? (1 == 1) || (i++ == 5)

1, wegen der Kurzschlussauswertung wird die zweite Bedinung nie ausgeführt

Mit welchem Schlüsselwort verlässt man einen case in einem switch-statement? break

Übungsblatt 1	Wiederholung	JavaDoc	Methoden	Arrays	Aufgabe	Ende	
0	000	00000	00000000000	000000000	00000	0	

Tutorium 14 -Wiederholung ─Wiederholung

1, wegen der Kurzschlussauswertung wird die zweite Bedinung nie ausgeführt Mit welchem Schlüsselwort verlässt man einen case in einem switch-statement!

Strings sind immutable (unveränderbar), deswegen wird bei der Konkatenation ein neues Objekt erstellt

23.11.2022



Programmieren Tutorium

Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Péter Bohner: Tutorium 14



Tutorium 14 ─Wiederholung

─Wiederholung

Wie oft wird ein do-while-Schleife mindestens auspeführt?



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Übungsblatt 1 Wiederholung JavaDoc Methoden Aufgabe Ende Arrays 000 00000 00000000000 0000000000 00000

Tutorium 14 ─Wiederholung ─Wiederholung

Wie oft wird ein do-while-Schleife mindestens auspeführt? Einmal, bevor die Schleifenbedingung zum ersten Mal gegrüft wird.

Programmieren Tutorium



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird Was trifft idealerweise auf eine for-Schleife zu?

Übungsblatt 1 Wiederholung JavaDoc Aufgabe Ende Methoden Arrays 000 00000 00000000000 0000000000 00000

Tutorium 14 ─Wiederholung

Einmal, bevor die Schleiferbedingung zum ersten Mal geprüft wird Was trifft idealerweise auf eine for-Schleife zu?

---Wiederholung



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden



6/39 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14
___Wiederholung

Wie oft wird ein do-white-Schleife mindestens ausgeführt? Einmal, bavor die Schleiferbedingung zum ersten Mal geprüft wird Was trifft idealerweise auf eine for-Schleife zu?

Anzahl der Iterationen bekannt

Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
 for-Schleifen können verschachteit werden

─Wiederholung

23.11.2022



Programmieren Tutorium

Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden

Péter Bohner: Tutorium 14

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen?

Übungsblatt Wiederholung JavaDoc Methoden Arrays Aufgabe Ende 000 00000 0000000000 00000 0

Tutorium 14 -Wiederholung

-Wiederholung

Wie oft wird ein do-while-Schleife mindestens auspeführt? Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird Was trifft idealerweise auf eine for-Schleife zu? Anzahl der Iterationen bekannt Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführ

for-Schleifen k\u00f6nnen verschachteit werden

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen?



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen? continue

Übungsblatt Wiederholung JavaDoc Methoden Arrays Aufgabe Ende 000 00000 0000000000 00000 0

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium Tutorium 14 -Wiederholung

-Wiederholung

Wie oft wird ein do-white-Schleife mindestens auspeführt? Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird Was trifft idealerweise auf eine for-Schleife zu? Anzahl der Iterationen bekannt

- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführ
- for-Schleifen k\u00f6nnen verschachteit werden

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen?



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen? continue

Wie ist die Syntax von einem Konstruktor einer Klasse?

Übungsblatt Wiederholung JavaDoc Methoden Arrays Aufgabe Ende 000 00000 00000 0

Tutorium 14 -Wiederholung

-Wiederholung

Wie oft wird ein do-while-Schleife mindestens auspeführt? Einmal, bevor die Schleifenbedingung zum ersten Mal gegrüft wird Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgefüh
- for-Schleifen k\u00f6nnen verschachteit werden
- Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen?

Wie ist die Syntax von einem Konstruktor einer Klasse?



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen? continue

Wie ist die Syntax von einem Konstruktor einer Klasse?

Klassenname(Parameterliste)

Übungsblatt Wiederholung JavaDoc Methoden Arrays Aufgabe Ende 000 00000 00000 0

Tutorium 14 -Wiederholung

-Wiederholung

Wie oft wird ein do-while-Schleife mindestens auspeführt? Einmal, bevor die Schleifenbedingung zum ersten Mal gegrüft wird Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgefüh
- for-Schleifen k\u00f6nnen verschachteit werden
- Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen?

Wie ist die Syntax von einem Konstruktor einer Klasse?



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen? continue

Wie ist die Syntax von einem Konstruktor einer Klasse?

Klassenname(Parameterliste)

Eine Klasse kann nur einen Konstruktor haben.

Übungsblatt Wiederholung JavaDoc Methoden Aufgabe Ende Arrays 000 00000 00000

Tutorium 14 -Wiederholung -Wiederholung

Einmal, bevor die Schleifenbedingung zum ersten Mal gegrüft wird Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgefüh for-Schleifen k\u00f6nnen verschachteit werden
- Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen

Wie ist die Syntax von einem Konstruktor einer Klasse?

Klassenname(Parameterliste) Fine Klasse kann nur einen Konstruktor haben



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen? continue

Wie ist die Syntax von einem Konstruktor einer Klasse?

Klassenname(Parameterliste)

Eine Klasse kann nur einen Konstruktor haben.

Falsch, man kann die Parameterliste überladen

Übungsblatt Wiederholung JavaDoc Methoden Aufgabe Ende 000 00000

Tutorium 14 -Wiederholung Wiederholung

Einmal, bevor die Schleiferbedingung zum ersten Mal gegrüft wird Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgefüh for-Schleifen k\u00f6nnen verschachteit werden
- Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen
- Wie ist die Syntax von einem Konstruktor einer Klasse'
- Fine Klasse kann nur einen Konstruktor haben Folsch man kann die Parameterliste überlade

Wiederholung



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen? continue

Wie ist die Syntax von einem Konstruktor einer Klasse?

Klassenname(Parameterliste)

Eine Klasse kann nur einen Konstruktor haben.

Falsch, man kann die Parameterliste überladen

Mit welchem Schlüsselwort unterscheidet man zwischen Attributen und anderen Variablen gleichen Namens?

namens?						
ungsblatt 1	Wiederholung	JavaDoc	Methoden	Arrays	Aufgabe	Ende
	○○●	00000	00000000000	000000000	00000	o

Tutorium 14 -Wiederholung Wiederholung

Einmal, bevor die Schleiferbedingung zum ersten Mal gegrüft wird Was trifft idealerwaise out eine fre Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgefü for-Schleifen k\u00f6nnen verschachteit werden
- Mit welchem Schlüsselwort wird eine Schleifeniteration übersprunger
- Wie ist die Syntax von einem Konstruktor einer Klasse'
- Fine Klasse kann nur einen Konstruktor haben
- Folach man kann die Parameterliste überladen
- Mit welchem Schlüsselwort unterscheidet man zwischen Attributen und anderen Variablen gleichen

Wiederholung



Wie oft wird ein do-while-Schleife mindestens ausgeführt?

Einmal, bevor die Schleifenbedingung zum ersten Mal geprüft wird

Was trifft idealerweise auf eine for-Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgeführt
- for-Schleifen können verschachtelt werden.

Mit welchem Schlüsselwort wird eine Schleifeniteration übersprungen? continue

Wie ist die Syntax von einem Konstruktor einer Klasse?

Klassenname(Parameterliste)

Eine Klasse kann nur einen Konstruktor haben.

Falsch, man kann die Parameterliste überladen

Mit welchem Schlüsselwort unterscheidet man zwischen Attributen und anderen Variablen gleichen Namens? this

Namono: this										
bungsblatt 1	Wiederholung ○○●	JavaDoc 00000	Methoden	Arrays	Aufgabe	Ende				

Tutorium 14 -Wiederholung Wiederholung

Einmal, bevor die Schleiferbedingung zum ersten Mal gegrüft wird Was trifft idealerwaise out eine fre Schleife zu?

- Anzahl der Iterationen bekannt
- Die Schrittanweisung wird nach jeder Rumpfausführung ausgefü for-Schleifen k\u00f6nnen verschachteit werden
- Mit welchem Schlüsselwort wird eine Schleifeniteration übersprunger
- Wie ist die Syntax von einem Konstruktor einer Klasse
- Fine Klasse kann nur einen Konstruktor haben
- Folach man kann die Parameterliste überladen
- Mit welchem Schlüsselwort unterscheidet man zwischen Attributen und anderen Variablen gleichen



Tutorium 14

JavaDoc

JavaDoc - Was ist das?

 Übungsblatt 1
 Wiederholung 0
 JavaDoc 0000
 Methoden 000000000
 Arrays 00000000
 Aufgabe 00000
 Ende 00000

 7/39
 23.11.2022
 Péter Bohner: Tutorium 14
 Programmieren Tutorium



Javadoc...

23.11.2022

- ... beschreibt Klassen, Methoden, Konstuktoren und Felder
- ... wird direkt in den Code geschreiben

Péter Bohner: Tutorium 14

... erleichtert das Verständnis

Übungsblatt 1 Wiederholung JavaDoc Methoden Aufgabe Ende Arrays 000 •0000 00000000000 0000000000 00000 0

Programmieren Tutorium

Tutorium 14 -JavaDoc └─JavaDoc - Was ist das?

... beschreibt Klassen, Methoden, Konstuktoren und Felder ... wird direkt in den Code geschreiben ... erleichtert das Verständnis



Javadoc...

- ... beschreibt Klassen, Methoden, Konstuktoren und Felder
- ... wird direkt in den Code geschreiben
- ... erleichtert das Verständnis

Regeln und Ziele

- einheitlich auf Deutsch oder auf Englisch
- vollständige und aussagekräftige Beschreibung

 Übungsblatt 1
 Wiederholung
 JavaDoc
 Methoden
 Arrays
 Aufgabe
 Ende

 0
 000
 ●000
 0000000000
 000000000
 000000000
 0
 0

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14

JavaDoc

JavaDoc - Was ist das?

Brooker

Listenberg France Methods, Kreskkon und Felder

Listenberg France Methods, Kreskkon und Felder

Listenberg France Methods

Listenberg France Methods

Register Zeite

Listenberg France Methods

Listenbe



Tutorium 14 -JavaDoc ☐JavaDoc - Was ist das?

... beschreibt Klassen, Methoden, Konstuktoren und Felder ... wird direkt in den Code geschreiben ... erleichtert das Verständnis einheitlich auf Deutsch oder auf Englisch vollståndige und aussagekräftige Beschreibung

Javadoc...

- ... beschreibt Klassen, Methoden, Konstuktoren und Felder
- ... wird direkt in den Code geschreiben
- ... erleichtert das Verständnis

Regeln und Ziele

- einheitlich auf Deutsch oder auf Englisch
- vollständige und aussagekräftige Beschreibung

Javadoc zur Java API

Übungsblatt 1 Wiederholung Ende JavaDoc Methoden Arrays Aufgabe 000 •0000 000000000000 0000000000 00000

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium



Tutorium 14

—JavaDoc

—JavaDoc für Klassen

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

23.11.2022

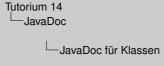
Péter Bohner: Tutorium 14



beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist



Programmieren Tutorium



beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist

23.11.2022



Programmieren Tutorium

- beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist
- mögliche Verwendungszwecke können erwähnt werden

Péter Bohner: Tutorium 14



Tutorium 14

—JavaDoc

—JavaDoc für Klassen

beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist
 mögliche Verwendungszwecke können erwähnt werden



- beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist
- mögliche Verwendungszwecke können erwähnt werden

Péter Bohner: Tutorium 14

der Autor wird angegeben

23.11.2022



Programmieren Tutorium



- beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist
 - der Autor wird angegeben



- beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist
- mögliche Verwendungszwecke können erwähnt werden
- der Autor wird angegeben

23.11.2022

die Versionsnummer wird angegeben

Péter Bohner: Tutorium 14





Tutorium 14 -JavaDoc

Programmieren Tutorium

 beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist

der Autor wird angegeben

die Versionsnummer wird angegeben

JavaDoc für Klassen



- beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionalität sie aufweist
- mögliche Verwendungszwecke können erwähnt werden
- der Autor wird angegeben
- die Versionsnummer wird angegeben

```
/**
  * Kurze aber aussagekräftige Beschreibung der Klasse.
  * @author Autor
  * @version versionsNummer
  */
class ClassName {
```

Tutorium 14

JavaDoc

JavaDoc für Klassen

beschreibt, welchen Zweck die Klasse erfüllt und welche Funktionsfält sie aufweist
 mögliche Verwendungszwecke können erwähnt werden
 der Autor wird angegeben

der Autor wird angegeben
 die Versionsnummer wird angegeber

intax

/**

* Kurze aber sussagekräftige Beschreibung der Klasse

* Gauthor Autor

* Gevesion versionsNummer

er continue (

....

39 23.11.2022

Péter Bohner: Tutorium 14 Programmieren Tutorium

JavaDoc für Klassen - Beispiel

Wiederholung

Péter Bohner: Tutorium 14

000

Übungsblatt 1

23.11.2022



```
/**
 * Modelliert einen zweidimensionalen Vektor.
 * Der Vektor besteht aus einer x- und y-Komponente.
 * @author Peter Bohner
 * @author ugknc
 * @version 1.0
class Vector2D {
    double x;
    double y;
```

JavaDoc

00000

Methoden

00000000000



Ende 0

Programmieren Tutorium

Aufgabe

00000

Arrays

0000000000

Tutorium 14 -JavaDoc * Modelliert einen zweidimensionalen Vektor. . Bauthor upkno JavaDoc für Klassen - Beispiel class Vector2D (double x; double y;



Tutorium 14 -JavaDoc ☐ JavaDoc für Konstruktoren

Übungsblatt 1 Wiederholung JavaDoc Methoden Arrays Aufgabe 00000 Ende 000 00000 00000000000 0000000000 0

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium



beschreibt Zweck der Konstruktors



Tutorium 14 -JavaDoc

beschreibt Zweck der Konstruktors

☐ JavaDoc für Konstruktoren



- beschreibt Zweck der Konstruktors
- beschreibt, in welchem Kontext der Kontruktor verwendet werden kann



23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14

JavaDoc

JavaDoc für Konstruktoren

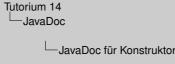
beschreibt Zweck der Konstruktors
 beschreibt, in welchem Kontext der Kontruktor verwendet werden kann

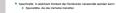


- beschreibt Zweck der Konstruktors
- beschreibt, in welchem Kontext der Kontruktor verwendet werden kann
 - Spezialfälle, die das Verhalten betreffen



Programmieren Tutorium





beschreibt Zweck der Konstruktors



☐ JavaDoc für Konstruktoren



beschreibt Zweck der Konstruktors

23.11.2022

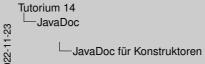
- beschreibt, in welchem Kontext der Kontruktor verwendet werden kann
 - Spezialfälle, die das Verhalten betreffen

Péter Bohner: Tutorium 14

alle Parameter werden beschrieben (Zweck und eventuell Wertebereich)



Programmieren Tutorium



 beschreibt Zweck der Konstruktors beschreibt, in welchem Kontext der Kontruktor verwendet werden kann.

Spezialtille, die das Verhalten betreffen

alle Parameter werden beschrieben (Zweck und eventuell Wertebereich



Aufgabe

Ende

beschreibt Zweck der Konstruktors

Wiederholung

000

Übungsblatt 1

beschreibt, in welchem Kontext der Kontruktor verwendet werden kann

JavaDoc

- Spezialfälle, die das Verhalten betreffen
- alle Parameter werden beschrieben (Zweck und eventuell Wertebereich)

```
Syntax
  * Kurze aber aussagekräftige Beschreibung von Zweck und Kontext des Konstruktors.
 * Sonderfälle werden beschrieben.
  * @param parameter Beschreibung des Parameters
KlassenName(Datentyp parameter, ...) {
```

Methoden 00000 0000000000 00000 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Arrays

Tutorium 14 -JavaDoc ☐ JavaDoc für Konstruktoren

 beschreibt Zweck der Konstruktors beschreibt, in welchem Kontext der Kontruktor verwendet werden kann Spezialtälle, die das Verhalten betreffen

KlassenName(Datentyp parameter, ...) {

JavaDoc für Konstuktoren - Beispiel

Péter Bohner: Tutorium 14

23.11.2022



```
/**
 * Konstruiert einen zwei dimensionalen Vektor mit zwei übergebenen Komponenten.
 * @param x die x-Komponente des Vektors
 * @param y die y-Komponente des Vektors
Vector2D(double x, double y) {
   this.x = x;
   this.y = y;
```



Programmieren Tutorium

Tutorium 14 -JavaDoc JavaDoc für Konstuktoren - Beispiel

- gparae x die x-Komponente des Vektors Vector20(double x, double y) (



Methoden realisieren das dynamische Verhalten von Objekten und führen Berechnungen durch.

Methoden - Einführung

Tutorium 14 └─ Methoden

1. paramter ist der aussagekräftiger Bezeichner des übergebens Arguments

Methoden realisieren das dynamische Verhalten von Obiekten und führen Berechnungen durch.

Übungsblatt 1

Wiederholung

Methoden •00000000000

Arrays 0000000000 Aufgabe

00000

Ende

Programmieren Tutorium

23.11.2022

000

Péter Bohner: Tutorium 14

JavaDoc

00000



Methoden realisieren das dynamische Verhalten von Objekten und führen Berechnungen durch.

Schema:

```
Rückgabetyp methodenName(Parameterliste) {
   // Methodenrumpf
```



Tutorium 14 └─ Methoden

Methoden - Einführung

Methoden realisieren das dynamische Verhalten von Obiekten und führen Berechnungen durch.

1. paramter ist der aussagekräftiger Bezeichner des übergebens Arguments

Methoden





Ende

Programmieren Tutorium



Methoden realisieren das dynamische Verhalten von Objekten und führen Berechnungen durch.

```
Schema:
```

```
Rückgabetyp methodenName(Parameterliste) {
   // Methodenrumpf
```

Der Rückgabetyp **void** steht dabei für keine Rückgabe!

Tutorium 14

Methoden realisieren das dynamische Verhalten von Obiekten und führen Berechnungen durch. Rückpabetyo methodenName(ParameterListe) { Methoden - Einführung

Der Rückgebetyp volld steht dabei für keine Rückgebel

1. paramter ist der aussagekräftiger Bezeichner des übergebens Arguments

Übungsblatt 1

23.11.2022

Péter Bohner: Tutorium 14

Wiederholung

000

•00000000000

Methoden

JavaDoc

00000

0000000000

Aufgabe 00000

Ende

Programmieren Tutorium



Methoden realisieren das dynamische Verhalten von Objekten und führen Berechnungen durch.

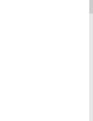
```
Schema:
```

Übungsblatt 1

```
Rückgabetyp methodenName(Parameterliste) {
   // Methodenrumpf
```

Der Rückgabetyp **void** steht dabei für keine Rückgabe!

Parameterliste: Datentyp₁ parameter₁, Datentyp₂ parameter₂, ..., Datentyp_n parameter_n







1. paramter ist der aussagekräftiger Bezeichner des übergebens Arguments

- Wiederholung JavaDoc Aufgabe Ende 00000 0000000000 00000
- •0000000000

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.



Tutorium 14 └─ Methoden

Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.

Methoden - Aufbau

23.11.2022

Péter Bohner: Tutorium 14





Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.

JavaDoc

00000

Die Methodensignatur besteht aus:

Wiederholung

000

Übungsblatt 1



00000000000 23.11.2022 Programmieren Tutorium Péter Bohner: Tutorium 14

Methoden

Tutorium 14 └─ Methoden Methoden - Aufbau

Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur. Die Methodensignatur besteht aus:



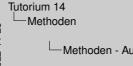
Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.

Die Methodensignatur besteht aus:

dem Namen der Methode



Programmieren Tutorium



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur. Die Methodensignatur besteht aus:

dem Namen der Methode

Methoden - Aufbau



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.

JavaDoc

00000

Methoden

00000000000

Die Methodensignatur besteht aus:

Wiederholung

000

- dem Namen der Methode
- der Anzahl der Parameter

Übungsblatt 1

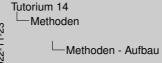
23.11.2022



Péter Bohner: Tutorium 14 Programmieren Tutorium

Arrays

0000000000



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur. Die Methodensignatur besteht aus:

 dem Namen der Methode der Anzahl der Parameter



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.

Die Methodensignatur besteht aus:

- dem Namen der Methode
- der Anzahl der Parameter

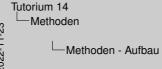
23.11.2022

der Reihenfolge der Parameter

Péter Bohner: Tutorium 14

Übungsblatt 1 Wiederholung JavaDoc Methoden Aufgabe Ende Arrays 000 00000 00000000000 0000000000 00000 0

Programmieren Tutorium



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur. Die Methodensignatur besteht aus:

dem Namen der Methode

der Anzahl der Parameter

der Reihenfolge der Parameter



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.

Die Methodensignatur besteht aus:

- dem Namen der Methode
- der Anzahl der Parameter
- der Reihenfolge der Parameter

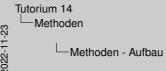
Péter Bohner: Tutorium 14

der Typen der Parameter

23.11.2022



Programmieren Tutorium



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.

Die Methodensignatur besteht aus:

dem Namen der Methode

der Anzahl der Parameter

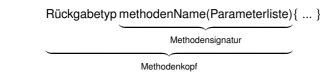
der Reihenfolge der Parameter
 der Typen der Parameter



Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur.

Die Methodensignatur besteht aus:

- dem Namen der Methode
- der Anzahl der Parameter
- der Reihenfolge der Parameter
- der Typen der Parameter



Übungsblatt 1 Wiederholung Ende JavaDoc Methoden Arrays Aufgabe 000 00000 00000000000 0000000000 00000

Programmieren Tutorium

Tutorium 14

Der Methodenkopf besteht aus dem Rückgabetyp und der Methodensignatur

Rückgabetyp methodenName(Parameterliste) (Methodensionstar

Die Methodensignatur besteht aus: dem Namen der Methode der Anzahl der Parameter

der Reihenfolge der Parameter

der Typen der Parameter

23.11.2022

Péter Bohner: Tutorium 14

Methoden - Zugriff auf Attribute



In Methoden kann auch auf Attribute des Objekts zugeriffen werden:



Tutorium 14 └─ Methoden

Methoden - Zugriff auf Attribute

In Methoden kann auch auf Attribute des Objekts zugerillen werden:

000

00000

0000000000

0000000000

00000

23.11.2022

Péter Bohner: Tutorium 14

Programmieren Tutorium

Methoden - Zugriff auf Attribute



In Methoden kann auch auf Attribute des Objekts zugeriffen werden:

```
class Person {
    String name = "Sven";

void greet(String name) {
        System.out.println("Hallo " + name + ", ich heiße " + this.name + ".");
    }
}
```



Tutorium 14

Methoden

Methoden - Zugriff auf Attribute

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Methoden - Zugriff auf Attribute



In Methoden kann auch auf Attribute des Objekts zugeriffen werden:

```
class Person {
    String name = "Sven";

    void greet(String name) {
        System.out.println("Hallo " + name + ", ich heiße " + this.name + ".");
    }
}
```

this erlaubt Unterscheidung zwischen Parameter und Attribut



Tutorium 14

Methoden

In Methoden - Zugriff auf Attribute

Methoden - Zugriff auf Attribute

In Methoden - Zugriff auf Attribute

In Methoden - Zugriff auf Attribute

In Methoden - Zugriff auf Attribute

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Methoden - Aufruf



Methoden können nur auf Objekten aufgerufen werden!

Methoden - Aufruf

Tutorium 14 └─ Methoden

1. Bei Aufruf in selber Klasse nicht nötig: implizit this genutzt

Methoden k\u00fcnnen nur auf Objekten aufgerufen werden!

Wiederholung

JavaDoc

00000

00000000000

Methoden

Arrays

0000000000

Ende

0

Aufgabe 00000

Programmieren Tutorium

Übungsblatt 1

23.11.2022

000 Péter Bohner: Tutorium 14

Methoden - Aufruf

- Methoden können nur auf Objekten aufgerufen werden!
- Dazu wird die Syntax objekt.methode() verwendet.

Methoden - Aufruf

Tutorium 14 └─ Methoden

 Methoden k\u00f6nnen nur auf Objekten aufgerufen werden! Dazu wird die Syntax objekt .eethode() verwendet.

1. Bei Aufruf in selber Klasse nicht nötig: implizit this genutzt

000

Methoden

00000000000

0000000000

















Methoden - Aufruf



- Methoden können nur auf Objekten aufgerufen werden!
 - Dazu wird die Syntax objekt.methode() verwendet.

```
Beispiel
Person person = new Person();
person.greet("Peter");
```

Übungsblatt 1 Wiederholung JavaDoc Aufgabe Ende Arrays 000 00000 00000000000 0000000000 00000

Tutorium 14 └─ Methoden Methoden - Aufruf

 Methoden können nur auf Objekten aufgerufen werden! Dazu wird die Syntax objekt .eethode() verwendet. person.greet("Peter"):

1. Bei Aufruf in selber Klasse nicht nötig: implizit this genutzt

23.11.2022

Péter Bohner: Tutorium 14

Methoden - Aufruf



- Methoden können nur auf Objekten aufgerufen werden!
 - Dazu wird die Syntax objekt.methode() verwendet.

JavaDoc

00000

```
Beispiel

Person person = new Person();
person.greet("Peter");
Ausgabe:
Hallo Peter, ich heiße Sven.
```

Methoden

00000000000

└─Methoden - Aufruf

1. Bei Aufruf in selber Klas

Tutorium 14

Methoden

Bei Aufruf in selber Klasse nicht nötig: implizit this genutzt

Übungsblatt 1

Wiederholung

000

Ende

Aufgabe

00000

Arrays

Methoden - Rückgabe



Wird ein anderer Rückgabetyp als void gewählt, muss die Methode einen Wert vom Rückgabetyp mit dem Schlüsselwort return zurückgeben.



Tutorium 14 └─ Methoden └─Methoden - Rückgabe

Schlüsselwort return zurückgeben.

Wird ein anderer Rückgabetyp als vosid gewählt, muss die Methode einen Wert vom Rückgabetyp mit dem

Übungsblatt 1

23.11.2022

Wiederholung

Péter Bohner: Tutorium 14

000

JavaDoc 00000

Methoden 00000000000

Arrays 0000000000

Aufgabe 00000

Ende 0

Methoden - Rückgabe



Wird ein anderer Rückgabetyp als void gewählt, muss die Methode einen Wert vom Rückgabetyp mit dem Schlüsselwort return zurückgeben.

```
Beispiel
int sum(int a, int b) {
    return a + b;
```









23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Methoden - return bei void



return kann aber auch bei Methoden mit Rückgabetyp void benutzt werden: z.B. um eine Methode gezielt zu verlassen



Methoden - return bei void

return kann aber auch bei Methoden mit Rückgabetvo vold benutzt werden: z.B. um eine Methode gezielt zu verlassen

1. Geschweifte Klammern beim if setzten. Nur für auf Folien oke

Tutorium 14 └─ Methoden

Methoden

000000000000

JavaDoc

00000

Arrays 0000000000 Aufgabe 00000

Ende

Übungsblatt 1

23.11.2022

Péter Bohner: Tutorium 14

Wiederholung

000

Methoden - return bei void



return kann aber auch bei Methoden mit Rückgabetyp void benutzt werden: z.B. um eine Methode gezielt zu verlassen

```
Beispiel
class Job {
    boolean started = false;
    void startJob() {
        if (started)
            return;
        started = true;
        // Do the Job
    }
}
```

Tutorium 14

Methoden

Methoden - return bei void

Methoden - return bei void

Tutorium 14

Tutorium 14

Tutorium 14

Tutorium 15

Tuto

1. Geschweifte Klammern beim if setzten. Nur für auf Folien oke

7/39

9 23.11.2022 Péter Bohner: Tutorium 14



Methoden die mit dem Modifier static beginnen nennt man Klassenmethoden.



Tutorium 14 └─ Methoden

- - 1. Mit static kann man auch Attribute als Klassenvariable definieren 2. Zugriff auf diese Analog zum Methode: Klasse.variable

Methoden die mit dem Modifier status beginnen nernt man Klassenmethoden.

Übungsblatt 1

Wiederholung

Methoden

000000000000

JavaDoc

00000

Arrays 0000000000

Aufgabe

Ende

23.11.2022

000

Péter Bohner: Tutorium 14



Methoden die mit dem Modifier static beginnen nennt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Objekt.

JavaDoc

00000

Methoden

000000000000



00000

Tutorium 14

Methoden die mit dem Modifier status beginnen nennt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Obiekt.

- 1. Mit static kann man auch Attribute als Klassenvariable definieren
- 2. Zugriff auf diese Analog zum Methode: Klasse.variable

Übungsblatt 1

Wiederholung

000



Methoden die mit dem Modifier static beginnen nennt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Objekt.

JavaDoc

00000

Methoden

```
class Math {
   static int sum(int a, int b) {
        return a + b;
```





Ende

0000000000 00000 000000000000

Arrays

Tutorium 14

Methoden die mit dem Modifier status beginnen nernt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Obiekt. static int sum(int a, int b) {

- 1. Mit static kann man auch Attribute als Klassenvariable definieren
- 2. Zugriff auf diese Analog zum Methode: Klasse.variable

Übungsblatt 1

Wiederholung

000

Aufgabe

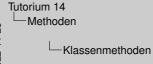


Methoden die mit dem Modifier static beginnen nennt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Objekt.

```
class Math {
    static int sum(int a, int b) {
        return a + b;
    }
}
int c = Math.sum(5, 8); // Nach Ausfühung: c = 13
```



9-11-23





- 1. Mit static kann man auch Attribute als Klassenvariable definieren
- 2. Zugriff auf diese Analog zum Methode: Klasse.variable



Methoden die mit dem Modifier static beginnen nennt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Objekt.

JavaDoc

00000

Methoden

000000000000

Arrays

0000000000

```
class Math {
    static int sum(int a, int b) {
        return a + b;
int c = Math.sum(5, 8); // Nach Ausfühung: c = 13
Es wird kein Objekt benötigt, um die Methode auszuführen.
```



Ende

Aufgabe

00000



-Klassenmethoden

Methoden die mit dem Modifier status beginnen nernt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Objekt. static int sum(int a, int b) { int c = Math.sum(5, 0); // Nach Auxfühung: c = 13 Es wird kein Objekt benötigt, um die Methode auszuführen.

- 1. Mit static kann man auch Attribute als Klassenvariable definieren
- 2. Zugriff auf diese Analog zum Methode: Klasse.variable

Übungsblatt 1

000

Wiederholung



Methoden die mit dem Modifier static beginnen nennt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Objekt.

JavaDoc

00000

Methoden

000000000000

Arrays

```
class Math {
    static int sum(int a, int b) {
        return a + b;
int c = Math.sum(5, 8); // Nach Ausfühung: c = 13
Es wird kein Objekt benötigt, um die Methode auszuführen.
Schema: Klasse.methodenName();
```



Ende

Aufgabe





Methoden die mit dem Modifier status beginnen nernt man Klassenmethoden.

- 1. Mit static kann man auch Attribute als Klassenvariable definieren
- 2. Zugriff auf diese Analog zum Methode: Klasse.variable

Übungsblatt

Wiederholung



Methoden die mit dem Modifier static beginnen nennt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Objekt.

```
class Math {
    static int sum(int a, int b) {
        return a + b;
int c = Math.sum(5, 8); // Nach Ausfühung: c = 13
Es wird kein Objekt benötigt, um die Methode auszuführen.
Schema: Klasse.methodenName();
```

this kann in Klassenmethoden nicht genutzt werden



Tutorium 14 └─ Methoden

-Klassenmethoden

Methoden die mit dem Modifier status beginnen nernt man Klassenmethoden. Diese gehören zu der Klasse statt zu einem Obiekt. static int sum(int a, int b) { Es wird kein Objekt benötigt, um die Methode auszuführen. Schema: Klasse methodenName();

this kann in Klassenmethoden nicht genutzt werder

- 1. Mit static kann man auch Attribute als Klassenvariable definieren
- 2. Zugriff auf diese Analog zum Methode: Klasse.variable

Hilfsmethoden

23.11.2022



Was sind Hilfsmethoden?

- Methoden, die häufig verwendete Funktionalität ausführen
- Können statisch oder nicht statisch sein

Übungsblatt 1 Wiederholung JavaDoc Methoden Aufgabe Ende Arrays 000 00000 000000000000 0000000000 00000 0

> Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14 └─ Methoden Hilfsmethoden

Was sind Hilfsmethoden? Methoden, die häufig verwendete Funktionalität ausführen

Können statisch oder nicht statisch sein



Hilfsmethoden



Was sind Hilfsmethoden?

- Methoden, die häufig verwendete Funktionalität ausführen
- Können statisch oder nicht statisch sein

statisch

- Klassenmethoden
- hängt nicht von Attributen einer Klasse ab
- static int sum(int a, int b)

Übungsblatt 1 Wiederholung JavaDoc Methoden Aufgabe Ende Arrays 000 00000 000000000000 0000000000 00000 0

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium Tutorium 14 └─ Methoden Hilfsmethoden

Was sind Hilfsmethoden? Methoden, die häufig verwendete Funktionalität ausführen Können statisch oder nicht statisch sein

 hängt nicht von Attributen einer Klasse ab * static int sum(int a. int b)

Hilfsmethoden



Was sind Hilfsmethoden?

- Methoden, die häufig verwendete Funktionalität ausführen
- Können statisch oder nicht statisch sein

statisch

23.11.2022

- Klassenmethoden
- hängt nicht von Attributen einer Klasse ab
- static int sum(int a, int b)

nicht statisch

- gliedern innerhalb einer Klasse Funktionalität aus
- hängt von Attributen ab
- int sum() (Summe zweier Attribute)

Übungsblatt 1 Ende Wiederholung JavaDoc Methoden Arrays Aufgabe 000 00000 000000000000 0000000000 00000

Tutorium 14 ☐ Hilfsmethoden

Was sind Hilfsmethoden? Methoden, die häufig verwendete Funktionalität ausführen

Können statisch oder nicht statisch sein

gliedern innerhalb einer Klasse Funktionalität * static int sum(int a. int b)

hängt von Attributen ab

• int sun() (Summe zweier Attribute)

Überladen



Es können mehrere Methoden mit dem gleichen Namen existieren, wenn sich diese in den Datentypen, Reihenfolge und/oder der Anzahl ihrer Parameter(d.h. in ihrer Signatur) unterscheiden.

```
Ok

int sum (int a, int b) { ... }

double sum (double a, double b) { ... }

int sum (int a, int b, int c) { ... }
```

```
Nicht Ok

int sum (int a, int b) { ... }

long sum (int a, int b) { ... }

int sum (int a, int c) { ... }
```

Methoden gleichen Namens, die sich nur im Rückgabetyp unterscheiden, sind nicht möglich.

Tutorium 14

Methoden

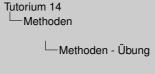
Überladen



Aufgabe

Schreibe eine Methode die eine Ganzzahl an nimmt und zurück gibt ob diese gerade ist.

Übungsblatt 1 Wiederholung JavaDoc Methoden Aufgabe Ende Arrays 000 00000 000000000000 0000000000 00000 0







Schreibe eine Methode die eine Ganzzahl an nimmt und zurück gibt ob diese gerade ist.

Tipp

Mit dem Modulo Operator % erhaltet ihr den Rest der Ganzzahldivision.

 Übungsblatt 1
 Wiederholung
 JavaDoc ooo 0000000 €00
 Methoden ooo 0000000 €00
 Arrays ooo 000000 €00
 Aufgabe ooo 0000 000
 Ende ooo 0000000 €00

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14

Methoden

Methoden - Übung





Aufgabe

Schreibe eine Methode die eine Ganzzahl an nimmt und zurück gibt ob diese gerade ist.

Tipp

- Mit dem Modulo Operator % erhaltet ihr den Rest der Ganzzahldivision.
- Der Rückgabetyp euer Methode muss boolean sein.

Programmieren Tutorium



└─Methoden - Übung





Aufgabe

Schreibe eine Methode die eine Ganzzahl an nimmt und zurück gibt ob diese gerade ist.

Tipp

- Mit dem Modulo Operator % erhaltet ihr den Rest der Ganzzahldivision.
- Der Rückgabetyp euer Methode muss boolean sein.

```
Lösung
```

```
static boolean isEven(int number) {
   return (number % 2 == 0);
```

Übungsblatt 1 Wiederholung JavaDoc Arrays Aufgabe Ende 000 00000 0000000000 00000 000000000000

-Methoden Methoden - Übung

Tutorium 14

Mit dem Modulo Operator % erhaltet ihr den Rest der Ganzzahldivision Der Rückgabetyp euer Methode muss booless sein. static boolean isEven(int number) {
 return (number % 2 == 0);



Tutorium 14

beschreibt Zweck der Methode

└─ Methoden beschreibt Zweck der Methode Methoden - JavaDoc

1. Beispiel Spezialfälle: Wurzel ziehen, Methode verarbeitet keine negativen werte: muss angegeben werden

Übungsblatt 1

JavaDoc 00000 Methoden 00000000000

Arrays 0000000000

Aufgabe 00000

Ende

0

23.11.2022

Wiederholung

000

Péter Bohner: Tutorium 14

- beschreibt Zweck der Methode
- beschreibt in welchem Kontext die Methode verwendet werden kann

JavaDoc

00000

Methoden

00000000000



Ende

0

Aufgabe

00000

Tutorium 14 └─ Methoden

Methoden - JavaDoc

 beschreibt Zweck der Methode beschreibt in welchem Kontext die Methode verwendet werden kann

1. Beispiel Spezialfälle: Wurzel ziehen, Methode verarbeitet keine negativen werte: muss angegeben werden

Übungsblatt 1

Wiederholung

000

Arrays

- beschreibt Zweck der Methode
- beschreibt in welchem Kontext die Methode verwendet werden kann
 - Vor- und Nachbedingungen der Ausführung



Tutorium 14 Methoden - JavaDoc

 beschreibt Zweck der Methode beschreibt in welchem Kontext die Methode verwendet werden kann Vor- und Nachbedingungen der Ausführung

1. Beispiel Spezialfälle: Wurzel ziehen, Methode verarbeitet keine negativen werte: muss angegeben werden

Methoden

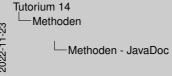
00000000000

Wiederholung



- beschreibt Zweck der Methode
- beschreibt in welchem Kontext die Methode verwendet werden kann
 - Vor- und Nachbedingungen der Ausführung
 - Spezialfälle, die das Verhalten betreffen





 beschreibt Zweck der Methode beschreibt in welchem Kontext die Methode verwendet werden kann Vor- und Nachbedingungen der Ausführung Spezialtälle, die das Verhalten betreffen

1. Beispiel Spezialfälle: Wurzel ziehen, Methode verarbeitet keine negativen werte: muss angegeben werden



- beschreibt Zweck der Methode
- beschreibt in welchem Kontext die Methode verwendet werden kann

JavaDoc

00000

- Vor- und Nachbedingungen der Ausführung
- Spezialfälle, die das Verhalten betreffen
- alle Parameter werden beschrieben (Zweck und Wertebereich)



Ende 0

Aufgabe

00000



Methoden

00000000000

Arrays

0000000000



 beschreibt in welchem Kontext die Methode verwendet werden kann Vor- und Nachbedingungen der Ausführung Spezialtille, die das Verhalten betreffen alle Parameter werden beschrieben (Zweck und Wertebereich)

beschreibt Zweck der Methode

1. Beispiel Spezialfälle: Wurzel ziehen, Methode verarbeitet keine negativen werte: muss angegeben werden

Übungsblatt 1

Wiederholung



- beschreibt Zweck der Methode
- beschreibt in welchem Kontext die Methode verwendet werden kann
 - Vor- und Nachbedingungen der Ausführung
- Spezialfälle, die das Verhalten betreffen
- alle Parameter werden beschrieben (Zweck und Wertebereich)
- der Rückgabetyp wird beschrieben (Inhalt)



23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium



1. Beispiel Spezialfälle: Wurzel ziehen, Methode verarbeitet keine negativen werte: muss angegeben werden

23.11.2022



Programmieren Tutorium

- beschreibt Zweck der Methode
- beschreibt in welchem Kontext die Methode verwendet werden kann
 - Vor- und Nachbedingungen der Ausführung
- Spezialfälle, die das Verhalten betreffen

Péter Bohner: Tutorium 14

- alle Parameter werden beschrieben (Zweck und Wertebereich)
- der Rückgabetyp wird beschrieben (Inhalt)

```
Syntax
  * Kurze aber aussagekräftige Beschreibung von Zweck und Kontext der Methode.
  * @param parameter Beschreibung des Parameters
  * @return Beschreibung Rückgabewert
 RückgabeDatentyp methodenName(DatenTyp parameter) {
Übungsblatt 1
                 Wiederholung
                                   JavaDoc
                                                  Methoden
                                                                      Arrays
                                                                                         Aufgabe
                                                                                                        Ende
                 000
                                   00000
                                                  00000000000
                                                                                         00000
```

Tutorium 14

Methoden

- Methoden - JavaDoc

Methoden - JavaDoc

- Methoden - JavaDoc

- Methoden -

1. Beispiel Spezialfälle: Wurzel ziehen, Methode verarbeitet keine negativen werte: muss angegeben werden

23.11.2022

Péter Bohner: Tutorium 14



Programmieren Tutorium

```
/**
  * Returns the absolute value of an int value.
  * If the argument is not negative, the argument is returned.
  * If the argument is negative, the negation of the argument is returned.
  * Note that if the argument is equal to the value of Integer.MIN_VALUE,
  * the most negative representable int value, the result is that same value, which is negative.
  * @param a the argument whose absolute value is to be determined
  * @return the absolute value of the argument.
 static int abs(int a) {
     return (a < 0) ? -a : a;
Übungsblatt
                Wiederholung
                                 JavaDoc
                                                                                    Aufgabe
                                                                                                  Ende
                                                                   Arrays
                000
                                               00000000000
                                                                                    00000
                                                                                                   0
```

Arrays - Einführung



- sind Folgen von *n* Elementen **desselben** Datentyps
- Zugriff auf einzelne Elemente erfolgt über Index
 - Index des ersten Elements: 0
 - Index des letzten Elements: n 1
- Arrays werden intern wie Objekte behandelt
 - Referenzierung des Arrays
- einfachste Datenstruktur



23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium Tutorium 14 —Arrays Arrays - Einführung

 sind Folgen von n Elementen desselben Datentyps Zugriff auf einzelne Elemente erfolgt über Index Index des ersten Elements: 0

Index des ersten Elements: 0
Index des letzten Elements: n -

 Arrays werden intern wie Obiekte behandelt Referenzierung des Arrays einfachste Datenstruktur



Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":

Übungsblatt 1 Wiederholung JavaDoc Methoden Arrays Aufgabe Ende 000 00000 00000000000 00000

Tutorium 14 —Arrays

0

000000000 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium Arrays - Deklaration

Deklariere Array vom Datentvo "Tvp" mit dem Bezeichner "name":



Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":

JavaDoc

00000

Typ[] name;

Wiederholung

000

Übungsblatt 1

 Methoden
 Arrays
 Aufgabe
 Ende

 0000000000
 0●0000000
 00000
 0

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14
EATRAYS

Arrays - Deklaration

Deklariere Array vom Daterriyp "Typ" mit dem Bezeichner "name":
 Typ]] nase;



Aufgabe

00000

Arrays

000000000

Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":

JavaDoc

00000

Typ[] name;

Wiederholung

Péter Bohner: Tutorium 14

000

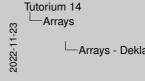
Übungsblatt 1

23.11.2022

Reserviere im gleichen Schritt Speicher für N Elemente und initialisiere diese mit ihren Default Werten:

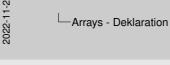
Methoden

00000000000



Reserviere im gleichen Schritt Speicher für N Elemente und initialisiere diese mit ihren Default Werten:

Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":





Ende

0



Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":

JavaDoc

00000

Typ[] name;

Übungsblatt 1

- Reserviere im gleichen Schritt Speicher für *N* Elemente und initialisiere diese mit ihren Default Werten:
 - Typ[] name = new Typ[N];

Wiederholung

000

Ende

Aufgabe

00000

0

000000000000 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Arrays

000000000

Methoden

Tutorium 14 —Arrays

Arrays - Deklaration

 Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name": Reserviere im gleichen Schrift Speicher für N Elemente und initialisiere diese mit ihren Default Werten:



- Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":
 - Typ[] name;

Übungsblatt 1

- Reserviere im gleichen Schritt Speicher für *N* Elemente und initialisiere diese mit ihren Default Werten:
- Typ[] name = new Typ[N];

Wiederholung

000

Deklariere ein Array und fülle es direkt mit spezifizierten Werten auf:

JavaDoc

00000



 Arrays
 Aufgabe
 End

 ○●○○○○○○○
 ○

39 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Methoden

000000000000

Tutorium 14
Arrays

Arrays - Deklaration

Dablairiere Array vom Datentyp, Typ' mit dem Bazeichner, name!
 17(p): name;
 Reservices im gleichen Schritt Speicher f
ür N Bernente und eritätisiere dese mit ihren Detault Werten:
 17(p): name - new Typ(N):
 Dablairiere ein Array und f
ülle ein deind mit soezifisierten Werten auf:



- Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":
 - Typ[] name;
- Reserviere im gleichen Schritt Speicher für *N* Elemente und initialisiere diese mit ihren Default Werten:
- Typ[] name = new Typ[N];
- Deklariere ein Array und fülle es direkt mit spezifizierten Werten auf:
 - Typ[] name = { wert₁, wert₂, ..., wert_n};



Arrays - Dek

Tutorium 14

Deldurier Array vom Datentyp, Typ' mit dem Bezeichner "name":
 17(9) | name;
 Reserviers im gleichen Schrift Speicher für N Elemente und initialisiene diese mit ihren Detault Werten:
 Reserviers im geleichen Schrift Speicher für N Elemente und initialisiene diese mit ihren Detault Werten:
 Reserviers im Re

Deklariere ein Array und f
ülle es direkt mit spezifizierten Werten auf:
 Typ:] nane = { wert₂, wert₂, ..., wert₃};

Arrays - Deklaration

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Arrays - Deklaration



- Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":
 - Typ[] name;

23.11.2022

- Reserviere im gleichen Schritt Speicher für *N* Elemente und initialisiere diese mit ihren Default Werten:
- Typ[] name = new Typ[N];
- Deklariere ein Array und fülle es direkt mit spezifizierten Werten auf:
 - Typ[] name = { wert₁, wert₂, ..., wert_n};

Péter Bohner: Tutorium 14

Array hat damit automatisch die Größe n.



Ende

Programmieren Tutorium





 Tvall name = new TvalNl; Deklariere ein Array und f
ülle es direkt mit spezifizierten Werten auf:

Typ[] name = { wert₁, wert₂, ..., wert₄};

Arrays - Deklaration



- Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":
 - Typ[] name;
- Reserviere im gleichen Schritt Speicher für *N* Elemente und initialisiere diese mit ihren Default Werten:
 - Typ[] name = new Typ[N];
- Deklariere ein Array und fülle es direkt mit spezifizierten Werten auf:
 - Typ[] name = { wert₁, wert₂, ..., wert_n};
 - Array hat damit automatisch die Größe n.

Achtung!

Die Größe eines Arrays ist nach anfordern des Speicherplatzes fest!

 Übungsblatt 1
 Wiederholung o
 JavaDoc o
 Methoden o
 Arrays o
 Aufgabe o
 Ende o

39 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14
—Arrays

—Arrays - Deklaration

- Deklariere Array vom Datentyp "Typ" mit dem Bezeichner "name":
 17(p1) asse;
 Reserviere in olicichen Schrift Soeicher für N Elemente und initialisiere diese mit ihren Detault Werten:
- Reserviere im gleichen Schrift Speicher für N Elemente und initialisiere diese mit ihnen Default \
 1791 nase new Typ(N)
 Deblariere ein Army und fülle es direkt mit spezifizierten Werten auf:
- Typ | name = { worts, worts, ..., worts};

 Array hat damit automatisch die Gottle m.

Achtung!



double[] a = new double[8];

Übungsblatt 1 Wiederholung JavaDoc Methoden Arrays Ende 000 00000

Aufgabe 00000 00000000000 000000000 0

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium Tutorium 14 —Arrays Arrays - Beispiele

double[] a = new double[f];



double[] a = new double[8];

0.0 0.0 0.0 0.0



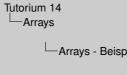
Arrays

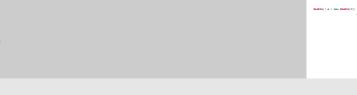
000000000

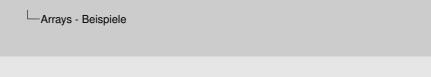






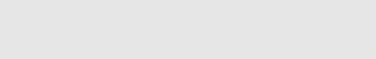












A- 00 00 00 00 00 00 00 00

- Ende 0
- Programmieren Tutorium

Aufgabe 00000

Übungsblatt 1



```
double[] a = new double[8];
```

Übungsblatt 1

23.11.2022

String[] b = new String[5];

Wiederholung

Péter Bohner: Tutorium 14

000

JavaDoc

00000

0.0 0.0 0.0 0.0 0.0

Methoden

00000000000

Arrays

000000000

Aufgabe

00000





Ende 0

Programmieren Tutorium









- —Arrays Arrays - Beispiele

Tutorium 14



A- 00 00 00 00 00 00 00 00

double[] a = new double[f];



Tutorium 14 —Arrays

Arrays - Beispiele

double[] a = new double[f];

String[] b = new String[5];

A- 00 00 00 00 00 00 00 00

the said and said said said

JavaDoc

00000

Methoden

00000000000

Arrays

000000000



00000

0

Ende

23.11.2022 Péter Bohner: Tutorium 14

Wiederholung

000

Übungsblatt 1



```
double[] a = new double[8];
                                  0.0 0.0 0.0 0.0 0.0
String[] b = new String[5];
                                    null null null null null
                                             2 3 4
int[] c = {3, -5, 9, 11};
```



Arrays - Beispiele

Tutorium 14 —Arrays



23.11.2022

Péter Bohner: Tutorium 14







Übungsblatt 1

23.11.2022

Wiederholung

Péter Bohner: Tutorium 14

000

JavaDoc

00000



Aufgabe

00000

Ende

0

Programmieren Tutorium

Tutorium 14

```
double[] a = new double[8];
                                      0.0 0.0 0.0 0.0
String[] b = new String[5];
                                    null null null null null
                                             2 3 4
int[] c = {3, -5, 9, 11};
```

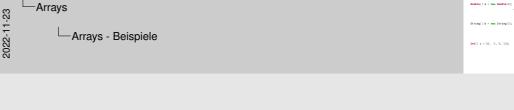
Methoden

00000000000

2 3

Arrays

000000000



A- 00 00 00 00 00 00 00 00

the said and said said said



Zugriff auf ein Element des initialisierten Arrays a erfolgt mit:

—Arrays Arrays - Zugriff

Tutorium 14

Zuoriff auf ein Element des initialisierten Arrays a erfolot mit:

1. haben wir schon bei main-methode genutzt um auf Konsolenargumente zu zugreifen

JavaDoc

00000000000

Methoden

Arrays

Aufgabe

Ende

Übungsblatt 1

Wiederholung 000

Péter Bohner: Tutorium 14

00000

000000000 00000

0

23.11.2022

dsruher Institut für Technologie

Programmieren Tutorium

Zugriff auf ein Element des initialisierten Arrays a erfolgt mit:

Péter Bohner: Tutorium 14

a[index]

isierten Arrays a erfolgt mit:





Tutorium 14

ays - Zugriff

Zugriff auf ein Element des initialisierten Arrays a erfolot mit:

1. haben wir schon bei main-methode genutzt um auf Konsolenargumente zu zugreifen



- Zugriff auf ein Element des initialisierten Arrays a erfolgt mit:
 - a[index]
 - Für Index gilt: 0 ≤ index < a.length



└─Arrays └─Arrays - Zugriff

Tutorium 14

riff

Zugriff auf ein Element des initialisierten Arrays a erfolgt mit:
 a [Index]
 Pfor Index git: 0 ≤ Index < a. Length

1. haben wir schon bei main-methode genutzt um auf Konsolenargumente zu zugreifen

Ende



- Zugriff auf ein Element des initialisierten Arrays a erfolgt mit:
 - a[index]
 - Für Index gilt: 0 ≤ index < a.length</p>
 - Sonst: ArrayIndexOutOfBoundException





Tutorium 14

 Zugriff auf ein Element des initialisierten Arrays a erfolgt mit: s | index |
 Für Index gilt: 0 ≤ Index < a. Length
 Sonst: ArsylindexOutDiffoundException

1. haben wir schon bei main-methode genutzt um auf Konsolenargumente zu zugreifen



- Zugriff auf ein Element des initialisierten Arrays a erfolgt mit:
 - a[index]
 - Für Index gilt: 0 ≤ index < a.length
 - Sonst: ArrayIndexOutOfBoundException

```
Beispiel
int[] a = {3, 7, 1, -5, 9};
System.out.println(a[3]); // Ausgabe: -5
a[3] = 11;
System.out.println(a[3]); // Ausgabe: 11
a[5] = 7; // ArrayIndexOutOfBoundException
```

```
        Übungsblatt 1
        Wiederholung
        JavaDoc
        Methoden
        Arrays
        Aufgabe
        Ende

        0
        000
        0000
        00000000000
        000●00000
        00000
        0
```

Tutorium 14
—Arrays
—Arrays - Zugriff

(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(1000)
(

a|s| = 1; System.out.println(a|3|); // Ausgabe: 21 a|5| = 7; // ArrayIndexOutOfBoundException

Zugriff auf ein Element des initialisierten Arrays a erfolgt mit:

1. haben wir schon bei main-methode genutzt um auf Konsolenargumente zu zugreifen

27/39

23.11.2022

Péter Bohner: Tutorium 14



Arrays werden in Java wie Objekte behandelt:

Übungsblatt 1 Wiederholung JavaDoc Methoden Arrays Aufgabe 00000 Ende 000 00000 00000000000 0000000000 0

Tutorium 14 —Arrays Arrays - Aufbau im Speicher

Arraya werden in Java wie Objekte behandelt:



Arrays werden in Java wie Objekte behandelt:

int[] a = {3, 8, 6};

Übungsblatt 1 Wiederholung JavaDoc Methoden Aufgabe Ende Arrays 000 00000 00000000000 0000000000 00000 0

Tutorium 14 —Arrays Arrays - Aufbau im Speicher

Arraya werden in Java wie Objekte behandelt: * int[] a = {1, 0, 0};

23.11.2022

Péter Bohner: Tutorium 14



Arrays werden in Java wie Objekte behandelt:

- int[] a = {3, 8, 6};
 - erstellt ein Array und weist der Variable a die Referenz auf das Array zu



Programmieren Tutorium

Tutorium 14 —Arrays

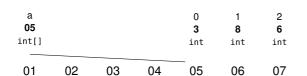
* int[] a = {1, 0, 0}; enstellt ein Array und weist der Variable a die Referenz auf das Array zu

Arraya werden in Java wie Objekte behandelt: Arrays - Aufbau im Speicher



Arrays werden in Java wie Objekte behandelt:

- int[] a = {3, 8, 6};
 - erstellt ein Array und weist der Variable a die Referenz auf das Array zu





19 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14
—Arrays
—Arrays - Aufbau im Speicher

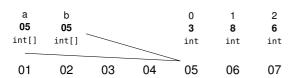






Arrays werden in Java wie Objekte behandelt:

- int[] a = {3, 8, 6};
 - erstellt ein Array und weist der Variable a die Referenz auf das Array zu
 - int[] b = a;





9 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14
—Arrays
—Arrays - Aufbau im Speicher

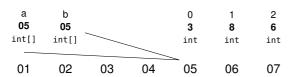






Arrays werden in Java wie Objekte behandelt:

- int[] a = {3, 8, 6};
 - erstellt ein Array und weist der Variable a die Referenz auf das Array zu
 - int[] b = a;
 - a und b referenzieren dasselbe Array





23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium Tutorium 14 —Arrays Arrays - Aufbau im Speicher



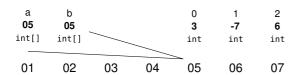






Arrays werden in Java wie Objekte behandelt:

- int[] a = {3, 8, 6};
 - erstellt ein Array und weist der Variable a die Referenz auf das Array zu
 - int[] b = a;
 - a und b referenzieren dasselbe Array
 - b[1] = -7;





9 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

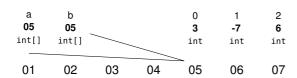






Arrays werden in Java wie Objekte behandelt:

- int[] a = {3, 8, 6};
 - erstellt ein Array und weist der Variable a die Referenz auf das Array zu
 - int[] b = a;
 - a und b referenzieren dasselbe Array
 - b[1] = -7;
 - Der Ausdruck a[1] wertet zu -7 aus.





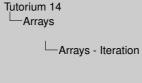
23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium Tutorium 14 —Arrays Arrays - Aufbau im Speicher

Arraya werden in Java wie Objekte behandelt: * int[] a = {1, 0, 0}; erstellt ein Array und weist der Variable a die Referenz auf das Array zu



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.





- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen

Iteriere über Array sint (1 a = 13 , ii , iii); um Summe der Elemente zu bestimmen.

- 4. for each bevorzugt benutzen
 - 5. element kann näturlich auch anders bennant werden s. Bsp.



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

JavaDoc

00000

Möglichkeit 1



000000000000

Arrays

0000000000

Methoden

Arrays - Iteration

Tutorium 14

—Arrays

Iteriere über Array sint (1 a = 13 , ii , iii); um Summe der Elemente zu bestimmen.

- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- - 4. for each bevorzugt benutzen 5. element kann näturlich auch anders bennant werden s. Bsp.

Übungsblatt 1

Wiederholung

000

Ende

Aufgabe



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

Möglichkeit 1

Iteriere mit for-Schleife über mögliche Indizes



Tutorium 14 —Arrays

Iteriere über Array sint (1 a = 13 , ii , iii); um Summe der Elemente zu bestimmen. Iteriere mit for-Schleife über mögliche Indizes

Arrays - Iteration

- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- 4. for each bevorzugt benutzen
- 5. element kann näturlich auch anders bennant werden s. Bsp.



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

JavaDoc

00000

Methoden

000000000000

Arrays

0000000000

Möglichkeit 1

- Iteriere mit for-Schleife über mögliche Indizes
- greife im Schleifenrumpf über Index auf Elemente zu



Aufgabe 00000

Ende 0

Programmieren Tutorium

Tutorium 14

—Arrays

Arrays - Iteration

Iteriere über Array sint (1 a = 13 , ii , iii); um Summe der Elemente zu bestimmen. Iteriere mit for-Schleife über mögliche Indizes

greife im Schleifenrumpf über Index auf Elemente

- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- 4. for each bevorzugt benutzen
- 5. element kann näturlich auch anders bennant werden s. Bsp.

Übungsblatt 1

Wiederholung



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

Methoden

Möglichkeit 1

- Iteriere mit for-Schleife über mögliche Indizes
- greife im Schleifenrumpf über Index auf Elemente zu

```
Ende
O
```

Tutorium 14 Arrays Arrays - Iteration Arrays - Iteration

- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- 4.
 - 4. for each bevorzugt benutzen5. element kann n\u00e4turlich auch anders bennant werden s. Bsp.

Aufgabe

00000



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

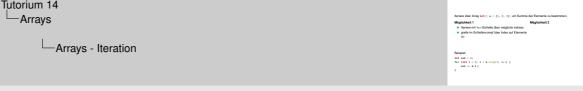
Möglichkeit 1

Möglichkeit 2

- Iteriere mit for-Schleife über mögliche Indizes
- greife im Schleifenrumpf über Index auf Elemente zu

```
Beispiel:
 int sum = 0;
 for (int i = 0; i < a.length; i++) {</pre>
      sum += a[i];
Übungsblatt 1
                    Wiederholung
                                        JavaDoc
                                                         Methoden
                    000
                                        00000
```

```
Tutorium 14
—Arrays
```



- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- 4. for each bevorzugt benutzen
 - 5. element kann näturlich auch anders bennant werden s. Bsp.

23.11.2022

Ende

Aufgabe

00000



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

Möglichkeit 1

- Iteriere mit for-Schleife über mögliche Indizes
- greife im Schleifenrumpf über Index auf Elemente zu

Möglichkeit 2

Methoden

Nutze for-each Schleife

```
Beispiel:
 int sum = 0;
 for (int i = 0; i < a.length; i++) {</pre>
      sum += a[i];
Übungsblatt 1
                   Wiederholung
                                       JavaDoc
                   000
                                       00000
```

Aufgabe

00000

Arrays

0000000000

Tutorium 14 —Arrays Arrays - Iteration



- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- - 4. for each bevorzugt benutzen 5. element kann näturlich auch anders bennant werden s. Bsp.

23.11.2022

Ende



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

Möglichkeit 1

- Iteriere mit for-Schleife über mögliche Indizes
- greife im Schleifenrumpf über Index auf Elemente zu

Möglichkeit 2

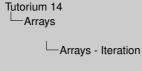
- Nutze for-each Schleife
- Führe Anweisung für jedes Element des Arrays

```
Beispiel:
 int sum = 0;
 for (int i = 0; i < a.length; i++) {</pre>
      sum += a[i];
Übungsblatt 1
                    Wiederholung
                                        JavaDoc
                   000
                                       00000
```

Methoden

- aus

0000000000





- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- - 4. for each bevorzugt benutzen 5. element kann näturlich auch anders bennant werden s. Bsp.

Ende

0

Aufgabe



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

JavaDoc

Möglichkeit 1

- Iteriere mit for-Schleife über mögliche Indizes
- greife im Schleifenrumpf über Index auf Elemente zu

Möglichkeit 2

- Nutze for-each Schleife
- Führe Anweisung für jedes Element des Arrays aus
- Schema:

Methoden

■ **for** (Typ element : array) { }

```
Beispiel:
int sum = 0;
for (int i = 0; i < a.length; i++) {</pre>
    sum += a[i];
```

Wiederholung

000

```
Aufgabe
                                        Ende
Arrays
0000000000
                      00000
                                        0
```

Tutorium 14 —Arrays Schema: Arrays - Iteration int sum = 0: for (int i = 0: i < a.length: i++) {

- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- - 4. for each bevorzugt benutzen 5. element kann näturlich auch anders bennant werden s. Bsp.

Übungsblatt 1



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

Möglichkeit 1

- Iteriere mit for-Schleife über mögliche Indizes
- greife im Schleifenrumpf über Index auf Elemente zu

Péter Bohner: Tutorium 14

Möglichkeit 2

- Nutze for-each Schleife
- Führe Anweisung für jedes Element des Arrays aus
- Schema:
 - **for** (Typ element : array) { }
 - "Typ" ist Datentyp der Elemente des Arrays

```
Beispiel:
```

23.11.2022

```
int sum = 0;
for (int i = 0; i < a.length; i++) {</pre>
    sum += a[i];
```

Übungsblatt Wiederholung 000

JavaDoc

Methoden

Arrays 0000000000

Aufgabe 00000

Ende 0



- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- 4. for each bevorzugt benutzen
 - 5. element kann näturlich auch anders bennant werden s. Bsp.



Iteriere über Array int[] a = {3, 8, 6}; um Summe der Elemente zu bestimmen.

Möglichkeit 1

Übungsblatt

Iteriere mit for-Schleife über mögliche Indizes

Wiederholung

000

greife im Schleifenrumpf über Index auf Elemente zu

Möglichkeit 2

- Nutze for-each Schleife
- Führe Anweisung für jedes Element des Arrays aus
- Schema:
 - **for** (Typ element : array) { }
 - "Typ" ist Datentyp der Elemente des Arrays

Aufgabe

00000

Ende

0

```
Beispiel:
                                                   Beispiel:
int sum = 0;
                                                   int sum = 0;
for (int i = 0; i < a.length; i++) {</pre>
                                                   for (int summand : a) {
    sum += a[i];
                                                        sum += element;
```

JavaDoc

```
Arrays
0000000000
```

Tutorium 14 —Arravs Arrays - Iteration



- 1. for each Schleifen auch für andere Datentypen
- 2. Reihenfolge hängt von Datenstruktur (Iterator ab)
- 3. for each kann nur benutzt werden wenn Indizes egal sind und alle Elemente gleich behandelt werden sollen
- - 4. for each bevorzugt benutzen 5. element kann näturlich auch anders bennant werden s. Bsp.

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Methoden



sind "Arrays von Arrays"



Ende

Übungsblatt 1 Wiederholung 000

JavaDoc 00000

Methoden 00000000000

Arrays 0000000000

Aufgabe 00000

0

Programmieren Tutorium

—Arrays

Tutorium 14

Mehrdimensionale Arrays

2. letzter Punkt: so kann man mehrdim. Arrays mit verschieden großen Arrays erstellen

1. Abbild: Annahme: Werte wurden gesetzt

sind "Arrays yon Arrays"

- sind "Arrays von Arrays"
- häufiger Anwendungsfall: Matrizen



Aufgabe Ende



Tutorium 14 —Arrays

Mehrdimensionale Arrays

- 1. Abbild: Annahme: Werte wurden gesetzt 2. letzter Punkt: so kann man mehrdim. Arrays mit verschieden großen Arrays erstellen

häufiger Anwendungsfall: Matrizer

Übungsblatt 1

Wiederholung

000

JavaDoc

00000

Methoden

00000000000

Arrays

0000000000



- sind "Arrays von Arrays"
- häufiger Anwendungsfall: Matrizen
- Beispiel:
- int[][] m = new int [2][3];



Aufgabe Ende 00000 0



Mehrdimensionale Arrays

- 1. Abbild: Annahme: Werte wurden gesetzt 2. letzter Punkt: so kann man mehrdim. Arrays mit verschieden großen Arrays erstellen

 sind "Arrays yon Arrays" häufiger Anwendungsfall: Matrizer Beispiel:
 Sat[][] n = new Sat [2][2];

Übungsblatt 1

Wiederholung

000

JavaDoc

00000

Methoden

00000000000

Arrays



- sind "Arrays von Arrays"
- häufiger Anwendungsfall: Matrizen
- Beispiel:
 - int[][] m = new int [2][3];
 - erzeugt 2-dimensionales Array



Ende 0



Aufgabe

Arrays

0000000000

Methoden 23.11.2022 Péter Bohner: Tutorium 14

00000000000

Tutorium 14 —Arrays sind "Arrays yon Arrays" häufiger Anwendungsfall: Matrizer * Sat[][] m = new Sat [2][2]; Mehrdimensionale Arrays

- 1. Abbild: Annahme: Werte wurden gesetzt
- 2. letzter Punkt: so kann man mehrdim. Arrays mit verschieden großen Arrays erstellen

Übungsblatt 1

JavaDoc

00000

Wiederholung

Mehrdimensionale Arrays



- sind "Arrays von Arrays"
- häufiger Anwendungsfall: Matrizen
- Beispiel:
 - int[][] m = new int [2][3];
 - erzeugt 2-dimensionales Array
 - m ist ein Array der Größe 2 von Arrays der Größe 3 vom Datentyp int

JavaDoc

00000



00000

Ende

0

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

000000000000

Methoden

Tutorium 14 —Arrays sind "Arrays yon Arrays" häufiger Anwendungsfall: Matrizen Beispiel: * Sat[][] n = new Sat [2][2]; eczaugt 2-dimensionales Array
 mist ein Array der Größe 2 von Arrays der Größe 3 vom Datentyp in Mehrdimensionale Arrays

- 1. Abbild: Annahme: Werte wurden gesetzt
- 2. letzter Punkt: so kann man mehrdim. Arrays mit verschieden großen Arrays erstellen

Übungsblatt 1

000

Wiederholung

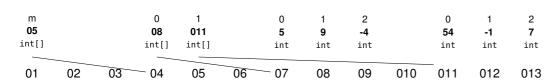
Arrays

0000000000

Mehrdimensionale Arrays



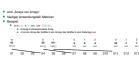
- sind "Arrays von Arrays"
- häufiger Anwendungsfall: Matrizen
- Beispiel:
 - int[][] m = new int [2][3];
 - erzeugt 2-dimensionales Array
 - m ist ein Array der Größe 2 von Arrays der Größe 3 vom Datentyp int





Tutorium 14 —Arrays

Mehrdimensionale Arrays



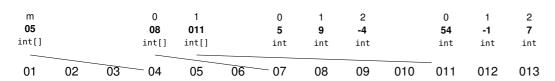
- 1. Abbild: Annahme: Werte wurden gesetzt
- 2. letzter Punkt: so kann man mehrdim. Arrays mit verschieden großen Arrays erstellen

23.11.2022

Mehrdimensionale Arrays

- sind "Arrays von Arrays"
- häufiger Anwendungsfall: Matrizen
- Beispiel:
- int[][] m = new int [2][3];

 - erzeugt 2-dimensionales Array
 - m ist ein Array der Größe 2 von Arrays der Größe 3 vom Datentyp int



int[][] a = new int [3][]; erzeugt Array mit 3 Referenzen des Typs int[]



Tutorium 14 —Arrays

Mehrdimensionale Arrays



- 1. Abbild: Annahme: Werte wurden gesetzt
- 2. letzter Punkt: so kann man mehrdim. Arrays mit verschieden großen Arrays erstellen

23.11.2022

Péter Bohner: Tutorium 14

Programmieren Tutorium

Mehrdimensionale Arrays - Iteration



Iterieren über Mehrdimensionale Arrays meist mit for-Schleifen:

 Übungsblatt 1
 Wiederholung
 JavaDoc
 Methoden
 Arrays
 Aufgabe
 Ende

 0
 000
 0000
 0000000000
 000000000
 00000
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0

23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14

RATION Arrays

Arrays

Mehrdimensionale Arrays - Iteration

berieve über Mahrdmanisionale Armays maint mit for Schheller: for ($\pm k$ ± \otimes \pm < $\pm k$.Lengthy $\pm m$) { for ($\pm k$ ± \otimes \pm < $\pm k$.Lengthy $\pm m$) { //// Nor whose soil Element $\pi(2)(2)$ } }

-Mehrdimensionale Arrays - Iteration

Arrays - Aufgabe



Aufgabe

Vervollständige den unten stehenden Code, um das Produkt von zwei Elementen mit gleichem Index aus den untenstehenden **gleich langen** Arrays zu berechnen.

Speichere das Ergebnis in einem weiteren Array und gebe das Ergebnis auf der Konsole aus.

00000

```
Code
class ArrayMultiplicator {
    public static void main(String[] args) {
        int[] a = {-1, 5, 3, 5, 8};
        int[] b = {4, 3, 8, 1, -1};
        // Ergänze deinen Code hier
    }
}
Übungsblatt 1 Wiederholung JavaDoc Methoden Arrays Aufgabe Ende
```

000000000

Tutorium 14
__Arrays
__Arrays - Aufgabe

Tellifonia

Terrelationing des services solvened Code, so dels Problet son and Themeries and glockine holes aus des

personal-codes glock linguige Anaya as benefities.

Specimen del Egyptime in more services from ord place due Egyptime and der Armerie aus.

COCT

000

00000

Arrays - Lösung

23.11.2022

Péter Bohner: Tutorium 14



Programmieren Tutorium

```
class ArrayMultiplicator {
     public static void main(String[] args) {
          int[] a = {-1, 5, 3, 5, 8};
          int[] b = {4, 3, 8, 1, -1};
          int[] result = new int[5];
          for (int i = 0; i < a.length; i++) {</pre>
              int product = a[i] * b[i];
              result[i] = product;
              System.out.println(a[i] + " * " + b[i] + " = " + product);
Übungsblatt 1
                                                                                                       Ende
                 Wiederholung
                                   JavaDoc
                                                  Methoden
                                                                      Arrays
                                                                                        Aufgabe
                 000
                                   00000
                                                  000000000000
                                                                      000000000
                                                                                        00000
                                                                                                       0
```

Tutorium 14

Arrays

Arrays - Lösung

Arrays - Lösung

Aufgabe



Tutorium 14 -Aufgabe

—Aufgabe

Buch

Modelliert eine Klasse Book, die die Attribute Titel, Autor, Erscheinungsjahr und Id hat. Die Id soll unveränderbar sein. Schreibt einen Konstruktor, der alle Attribute übergeben bekommt und diese setzt. Außerdem soll ein Konstruktor definiert werden, der immer das Jahr 2020 als Erscheinungsdatum setzt.

Für Schnelle

Schreibt ein ausführbares Programm, mit dem pro Konstruktor jeweils ein Objekt erstellt wird und Titel und Erscheinungsjahre der Bücher jeweils in getrennten Zeilen ausgegeben werden.

Übungsblatt Wiederholung JavaDoc Aufgabe Ende 000 00000 •0000

23.11.2022 Péter Bohner: Tutorium 14

Programmieren Tutorium



```
/**
 * Modelliert ein Buch zur Vereinfachung der Digitalisierung von Büchern.
 * Ein Buch besteht aus einer eindeutigen Identifikationsnummer,
 * dem Erscheinungsjahr, einem Titel und dem Namen des Autors.
 * @author Leon Wittemund
 * @version 1.0
class Book {
    final int id;
    int year;
    String title;
    String author;
```



Übungsblatt 1 Wiederholung 000

JavaDoc 00000

Methoden 00000000000 Arrays 0000000000 Aufgabe 00000 Ende

0

23.11.2022

Péter Bohner: Tutorium 14

Programmieren Tutorium

23.11.2022

Péter Bohner: Tutorium 14



Programmieren Tutorium

```
/**
      * Erstellt Instanzen der Klasse Buch
      * @param id eindeutige Identifikationsnummer
      * @param year Erscheinungsjahr des Buches
      * @param title Buchtitel
      * @param author Buchautor
     Book(int id, int year, String title, String author) {
          this.id = id;
          this.year = year;
          this.title = title;
          this.author = author;
Übungsblatt 1
                                                                                      Aufgabe
                                                                                                     Ende
                 Wiederholung
                                  JavaDoc
                                                 Methoden
                                                                    Arrays
                 000
                                  00000
                                                 00000000000
                                                                    0000000000
                                                                                      00000
                                                                                                     0
```

Tutorium 14

Aufgabe

- contil between or black bus
- green of standing best forticement or black bus
- green of standing best forticement
- green or between green or black busines
- green or blac

Übungsblatt 1

23.11.2022

Wiederholung

Péter Bohner: Tutorium 14

000



```
/**
* Erstellt Instanzen der Klasse Buch,
* wobei das Erscheinungsjahr auf das Jahr 2020 gesetzt wird.
 * @param id eindeutige Identifikationsnummer
* @param title Buchtitel
* @param author Buchautor
 */
Book(int id, String title, String author) {
    this(id, 2020, title, author);
```

JavaDoc

00000

Methoden

00000000000

```
Ende
```

Aufgabe

00000

Arrays

0000000000

0

Programmieren Tutorium

Tutorium 14 —Aufgabe

Lösung

. Erstellt Instanzen der Klasse Such, * wobei des Erscheinungsjahr auf des Jahr 2020 gesetzt wird. . (param id eindeutige Identifikationsnummer . @param title Suchtitel . Operan author Suchautor Book(ist id, String title, String author) {



```
/**
 * Die Klasse BookApp ermöglicht es,
 * die Funktionen der Klasse Book benutzen zu können.
 * @author Leon Wittemund
 * @version 1.0
class BookApp
   /**
     * Methode, mit der die Funktionen der Klasse Book getestet werden.
     * @param args übergebene Argumente
    public static void main(String[] args) {
        Book bookA = new Book(1, "Java ist auch eine Insel", "Christian Ullenboom");
        Book bookB = new Book(20, 2009, "Clean Code", "Robert Cecil Martin");
        System.out.println(bookA.title + ": " + bookA.year);
        System.out.println(bookB.title + ": " + bookB.year);
```

 Übungsblatt 1
 Wiederholung
 JavaDoc
 Methoden
 Arrays
 Aufgabe
 Ende

 0
 000
 0000
 0000000000
 000000000
 0000
 0

/39 23.11.2022 Péter Bohner: Tutorium 14 Programmieren Tutorium

Tutorium 14

E Aufgabe

Lösung

The professional and the profe

