



Einfache Arrays

Annabelle Klarl

Zentralübung zur Vorlesung

„Einführung in die Informatik: Programmierung und Softwareentwicklung“

<http://www.pst.ifi.lmu.de/Lehre/wise-13-14/infoeinf>



Action required now



1. Smartphone: installiere die App "socrative student" **oder**
Laptop: öffne im Browser m.socrative.com
2. Betrete den Raum **16485 16485**.
3. Beantworte die erste Frage sofort!



Arrays: Wiederholung

Ein Array ist ein Tupel von Elementen gleichen Typs

$$\mathbf{a} = [w_1, w_2, w_3, \dots, w_n]$$

- Elemente müssen den gleichen Typ haben
 - Grunddatentyp z.B. `int`-Array `[1, 2, 3, 4]`, aber nicht `[1, 1.0, 2]`
 - Klassentyp (später)
 - Arraytyp z.B. doppeltes `int`-Array
`[[1, 2], [3, 4]]`, aber nicht `[[1, 2], [1.0, 2.0]]`
 - Reihenfolge relevant: $[w_1, w_2] \neq [w_2, w_1]$
 - Zugriff auf ein bestimmtes Element möglich z.B. `a[0] = w1`
- Achtung! Array: $[w_1, w_2, w_3, \dots, w_n]$
- Position: $0 \quad 1 \quad 2 \quad \dots \quad n-1$



Arrays: Elemente gleichen Typs

Ein Array ist ein Tupel von Elementen gleichen Typs

$$a = [w_1, w_2, w_3, \dots, w_n]$$



Welche Anweisung wird vom Compiler **nicht** akzeptiert?

- a) `int [] a = {1, 2};`
- b) `int [] a = {1, 2.0};`
- c) `double [] a = {1, 2};`
- d) `double [] a = {1, 2.0};`

Wiederholung von ZÜ3: **automatische Typkonversion** zum größeren Typ

`byte < short < int < long < float < double`



Aufgabe 1: Arithmetisches Mittel

Schreibe eine Methode, die das arithmetische Mittel aller Zahlen in einem `int`-Array berechnet.

Algorithmusidee:

- Summiere alle Elemente des Arrays auf
- Teile die Summe durch die Anzahl der Elemente



Aufgabe 1: Arithmetisches Mittel (Lösungsidee 1)

```
public static double durchschnitt(int[] daten) {  
    int sum = 0;  
    for (int i = 0; i < daten.length; i++)  
        sum = sum + daten[i];  
    return sum / daten.length;  
}
```



Raum: 16485 16485

Welches Problem tritt
in dieser Methode auf?

- a) Die Methode darf nicht **static** sein.
- b) Es werden nicht alle Elemente aufsummiert.
- c) Die Division liefert nicht das gewünschte Ergebnis.
- d) Es kann eine Division durch 0 auftreten.
- e) Es tritt kein Problem auf.



Aufgabe 1: Arithmetisches Mittel (Lösungsidee 2)

```
public static double durchschnitt(int[] daten) {  
    int sum = 0;  
    for (int i = 0; i < daten.length; i++)  
        sum = sum + daten[i];  
    return (double)sum / daten.length;  
}
```

Array daten muss mind.
ein Element enthalten,
sonst Division durch 0!



Aufgabe 2: Sortierung (I)

Schreibe eine Methode, die überprüft, ob ein `double`-Array aufsteigend sortiert ist.

Algorithmusidee: Durchlaufe das Eingabe-Array elementweise

- Falls **ein** Element größer als sein rechter Nachbar ist, gib `false` zurück
- Falls **alle** Elemente kleiner als ihr rechter Nachbar sind, gib `true` zurück

```
public static boolean istSortiert(double[] arr) {  
    for (int i = 0; i < arr.length-1; i++) {  
        if (arr[i] > arr[i+1])  
            return false;  
    }  
    return true;  
}
```

sofortige Beendigung der Methode



Aufgabe 2: Sortierung (II)

Schreibe eine Methode, die überprüft, ob ein `double`-Array aufsteigend sortiert ist.

Algorithmusidee: Durchlaufe das Eingabe-Array elementweise

- Falls **ein** Element größer als sein rechter Nachbar ist, gib `false` zurück
- Falls **alle** Elemente kleiner als ihr rechter Nachbar sind, gib `true` zurück

```
public static boolean istSortiert(double[] arr) {  
    for (int i = 1; i < arr.length; i++) {  
        if (arr[i - 1] > arr[i])  
            return false;  
    }  
    return true;  
}
```

Achtung: Beginne mit dem Index 1, sonst ist `arr[i-1]` nicht definiert.



Aufgabe 2: Sortierung (III)

```
public static boolean istSortiert(double[] arr) {  
    for (int i = 1; i < arr.length; i++) {  
        if (arr[i - 1] > arr[i])  
            return false;  
    }  
    return true;  
}
```



Raum: 16485 16485

Wann tritt in dieser
Methode ein Problem auf?

- a) ... falls `arr` leer ist.
- b) ... falls `arr` nur ein Element hat.
- c) ... falls `arr` zwei Elemente hat.
- d) ... nie.



Aufgabe 3a: Kleines Einmaleins mit Arrays

Gib das kleine Einmaleins auf der Konsole mit Hilfe von Arrays aus.

Algorithmusidee: (siehe ZÜ5)

```
x * y = produkt
```

- Erzeuge ein leeres doppeltes Array (=> Matrix)
- Für x : gehe die Zahlen von 1 bis 10 durch
- Für y : gehe nochmal die Zahlen 1 bis 10 durch für jede Möglichkeit der Zahl x
- Berechne das Produkt aus den Zahlen x und y und speichere den Wert in der entsprechenden Zelle des doppelten Arrays.



Aufgabe 3a: Kleines Einmaleins mit Arrays

```
public class EinmaleinsArrays {  
    public static void main(String[] args) {  
  
        int[][] elements = new int[10][10];  
  
        for (int i = 1; i <= 10; i++) {  
  
            for (int j = 1; j <= 10; j++) {  
  
                elements[i-1][j-1] = i*j;  
  
            }  
  
        }  
  
    }  
}
```

Erzeuge ein doppeltes
leeres Array

Für x : gehe die Zahlen von
1 bis 10 durch

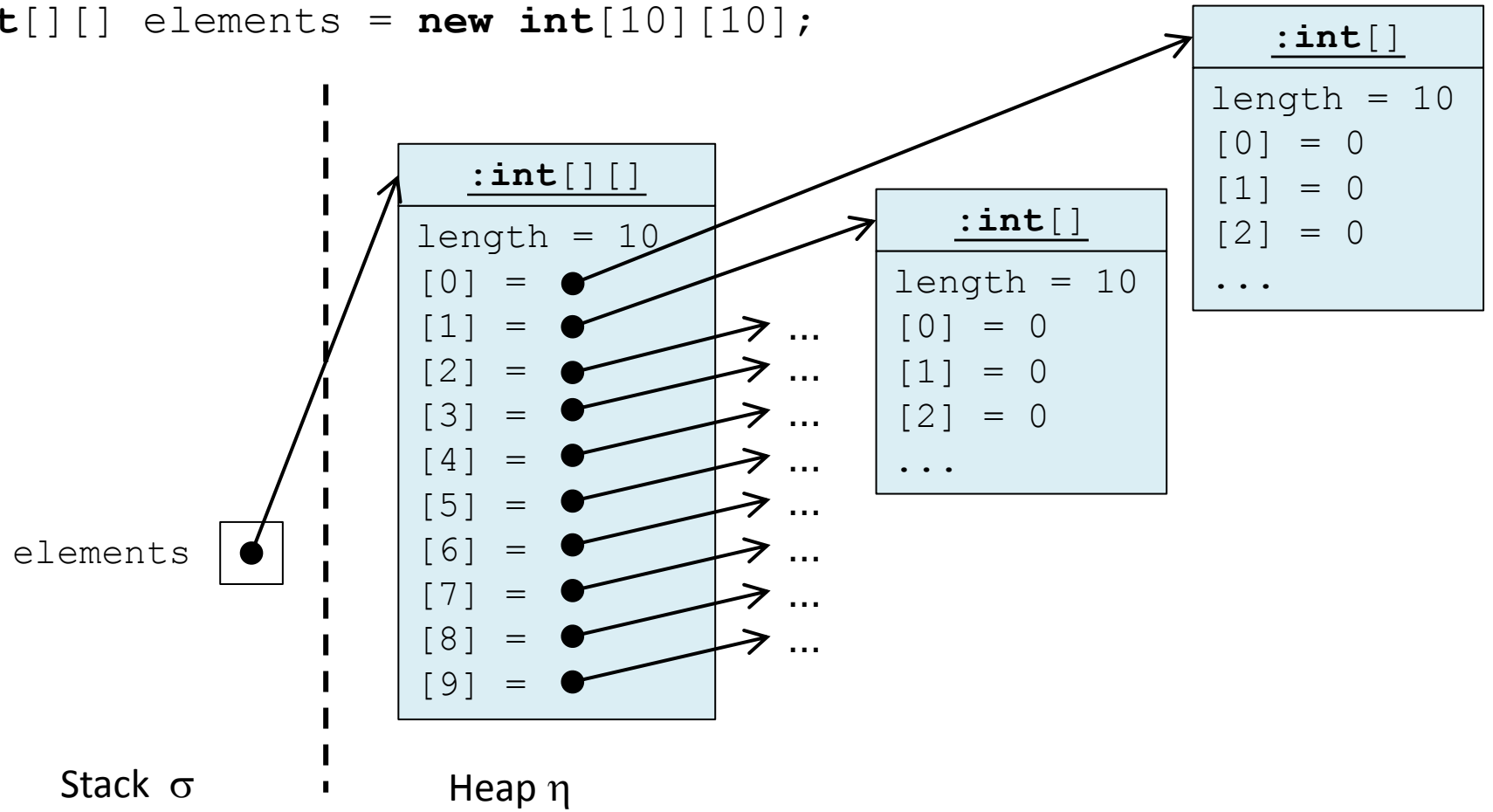
Für y : gehe nochmal die
Zahlen 1 bis 10 durch für
jede Möglichkeit der Zahl x

Berechne das Produkt aus
den Zahlen x und y und
speichere den Wert.



Aufgabe 3b: Speicherdarstellung

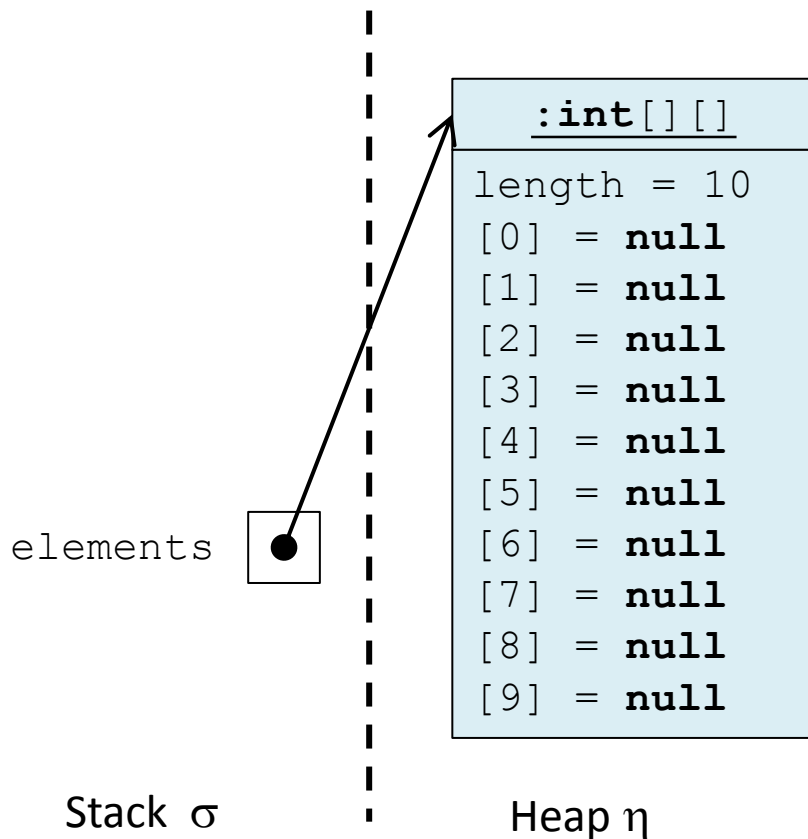
```
int[][] elements = new int[10][10];
```





Aufgabe 3b: Speicherdarstellung (partiell initialisiert)

```
int[][] elements = new int[10][];
```



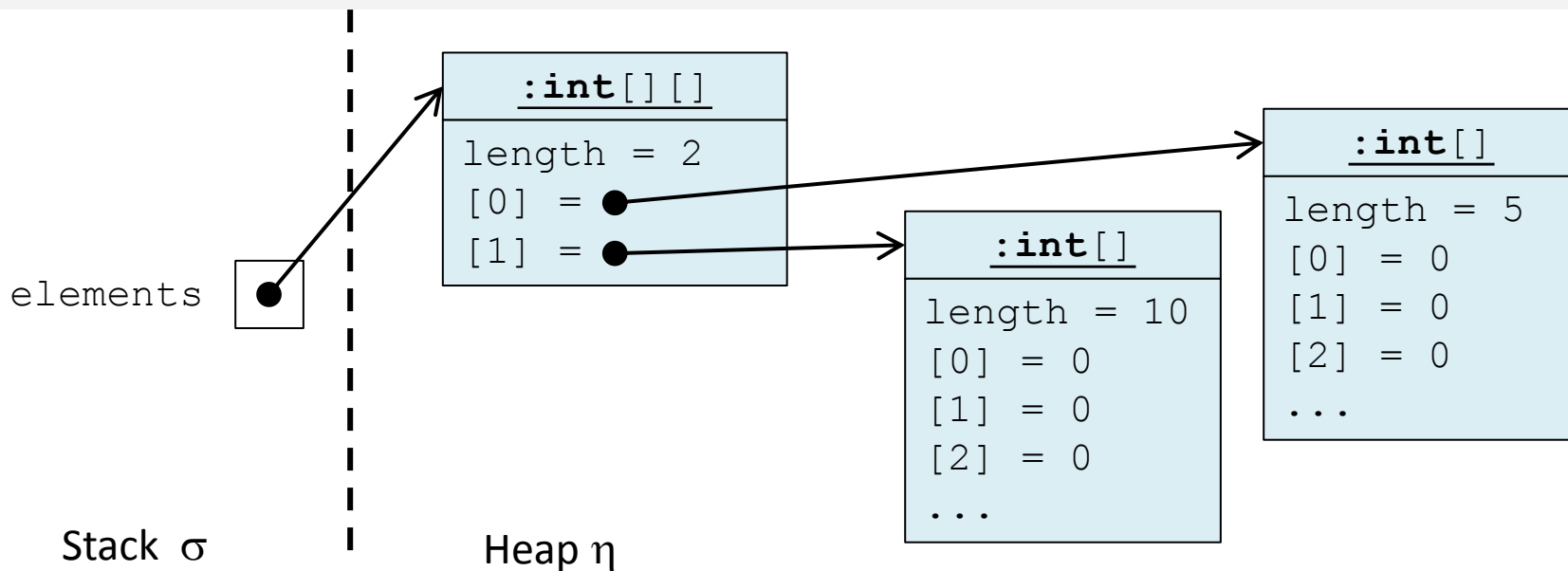


Einschub: **socratic**
Raum: 16485 16485

Kann ein doppeltes Array unterschiedlich lange "Zeilen" haben?

Codeausschnitt:

```
int[][] elements = new int[2][];
elements[0] = new int[5];
elements[1] = new int[10];
```





Aufgabe 3c: Ausgeben einer Matrix

```
public class EinmaleinsArrays {  
    public static void main(String[] args) {  
        int[][] elements = new int[10][10];  
        for (int i = 1; i <= 10; i++) {  
            for (int j = 1; j <= 10; j++) {  
                elements[i-1][j-1] = i*j;  
            }  
        }  
  
        for (int i = 0; i < elements.length; i++) {  
            for (int j = 0; j < elements[i].length; j++) {  
                System.out.print(elements[i][j] + " ");  
            }  
            System.out.println();  
        }  
    }  
}
```




Aufgabe 3c: Ausgeben einer Matrix

The screenshot shows a Java IDE console window with the following output:

```
<terminated> Matrix [Java Application] C:\Program Files\Java\jdk1.7.0_07\bin\javaw.exe (10.12.2012 10:02:54)
1 2 3 4 5 6 7 8 9 10
2 4 6 8 10 12 14 16 18 20
3 6 9 12 15 18 21 24 27 30
4 8 12 16 20 24 28 32 36 40
5 10 15 20 25 30 35 40 45 50
6 12 18 24 30 36 42 48 54 60
7 14 21 28 35 42 49 56 63 70
8 16 24 32 40 48 56 64 72 80
9 18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100
```