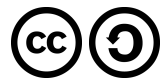


*This work is licensed under the Creative Commons  
Attribution-ShareAlike 3.0 License.*



# Arrays und Schleifen

Andreas & Tim

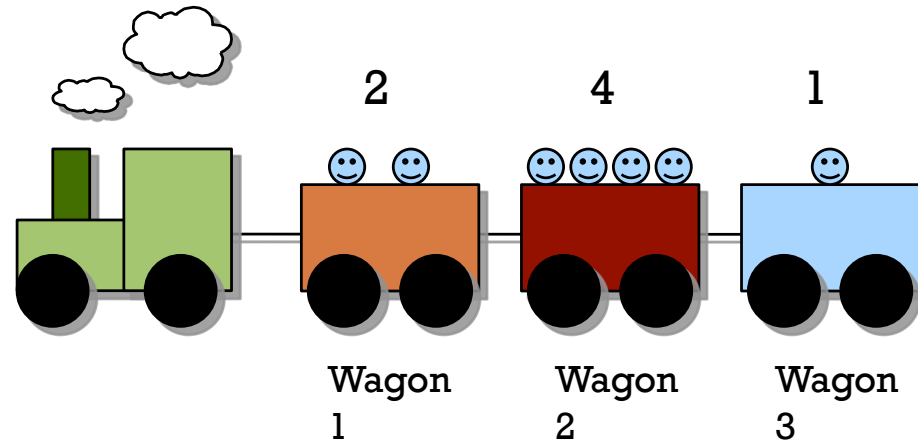


# Inhaltsverzeichnis

- Was ist ein Array?
- Array
  - Bauanleitung
  - Beispiel: Matrix
  - Fehler #1
  - Fehler #2
- Schleifen
  - Der Anfang einer Schleife
  - Die while Schleife
  - Fehler #1
  - Fehler #2
  - Der letzte Ausweg
  - Beispiel: Collatz-Folge
  - Die for Schleife
  - Beispiel: String Array
  - Beispiel: Matrix

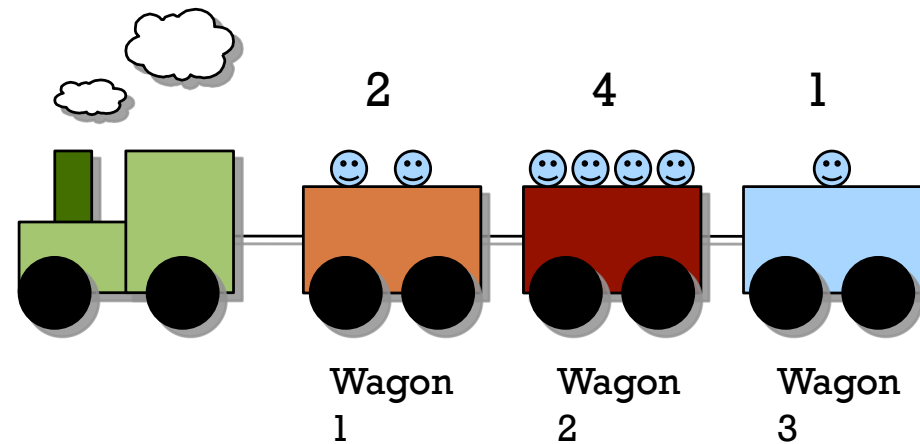
# + Was ist ein Array?

- Ein Array kann man sich wie einen Zug vorstellen

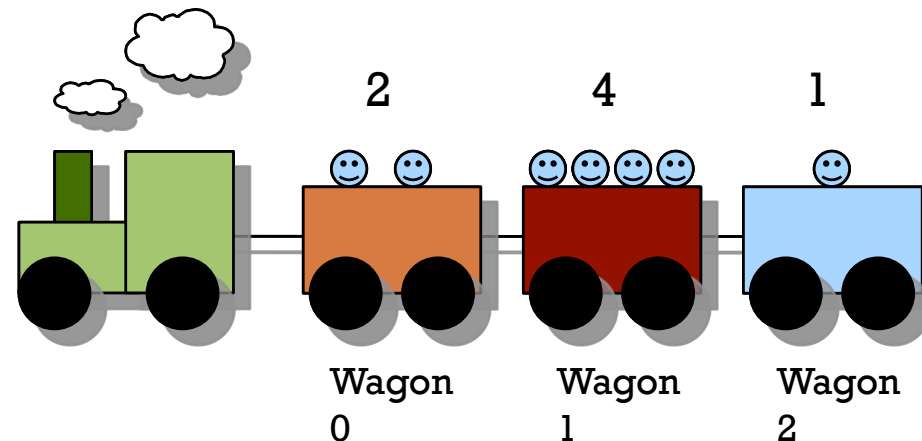


# + Was ist ein Array?

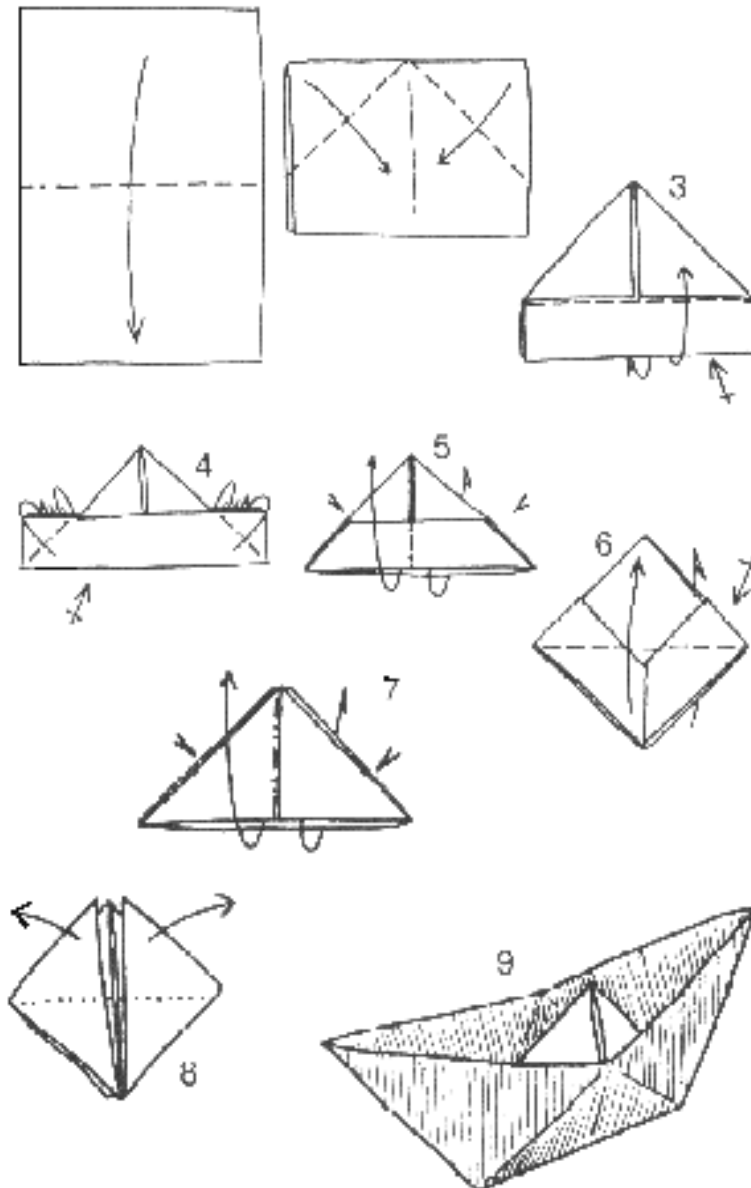
- Ein Array kann man sich wie einen Zug vorstellen



- Java beginnt die Nummerierung mit 0 also:



# + Array - Bauanleitung

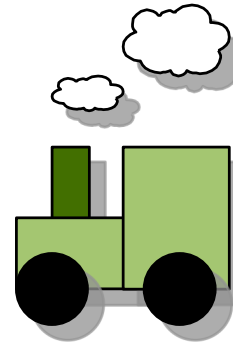


# + Array - Bauanleitung

6

■ Wie sieht das in Java aus?

① wir haben einen Zug mit Zahlen

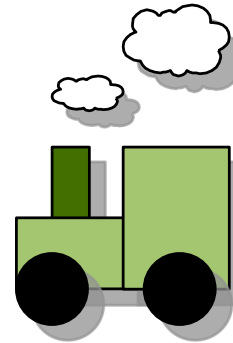


```
int[] zug;
```

# + Array - Bauanleitung

## ■ Wie sieht das in Java aus?

- ① wir haben einen Zug mit Zahlen



```
int[] zug;
```

Datentyp

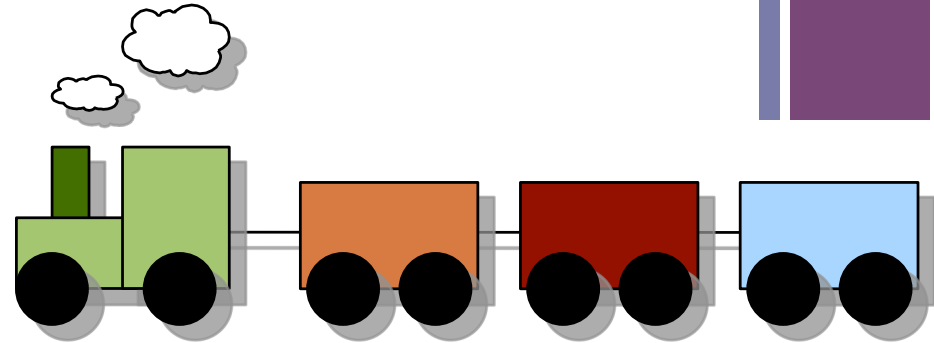
Variablenname

[] Arraysymbol

# + Array - Bauanleitung

## ■ Wie sieht das in Java aus?

- ① wir haben einen Zug mit Zahlen



```
int[] zug;
```

- ② wir müssen Java sagen das wir einen neuen Zug mit 3 Wagons haben wollen

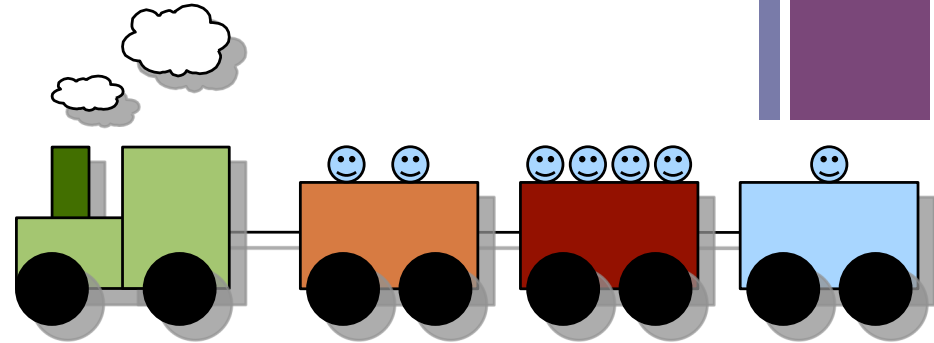
```
zug = new int[3];
```



# + Array - Bauanleitung

## ■ Wie sieht das in Java aus?

- ① wir haben einen Zug mit Zahlen



```
int[] zug;
```

- ② wir müssen Java sagen das wir einen neuen Zug mit 3 Wagons haben wollen

```
zug = new int[3];
```

- ③ erst jetzt kann man den Wagons Werte (Personen) zuweisen (bzw. einsteigen lassen)

```
zug[0] = 2;  
zug[1] = 4;  
zug[2] = 1;
```

# + Array – Bauanleitung

10

## ■ Bisherige Deklaration

```
String name;
```

# + Array – Bauanleitung

## ■ Bisherige Deklaration

```
String name;
```

## ■ Bisherige Initialisierung

```
name = "Tim";
```

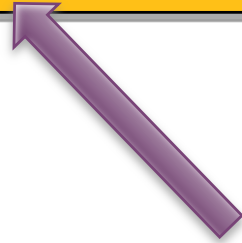
name

Tim

# + Array – Bauanleitung

## ■ Deklaration von Arrays

```
String[] namen;
```



# + Array – Bauanleitung

## ■ Deklaration von Arrays

```
String[] namen;
```

```
String namen[];
```

# + Array – Bauanleitung

## ■ Deklaration von Arrays

```
String[] namen;
```

```
String namen[];
```

```
String []namen;
```

# + Array – Bauanleitung

## ■ Deklaration von Arrays

```
String[] namen;
```

```
String namen[];
```

```
String []namen;
```

## ■ Wobei der Datentyp **String** durch jeden Beliebigen ersetzt werden kann:

- Bsp: int, byte, long, double, boolean, usw.....

# + Array – Bauanleitung

## ■ Deklaration von Arrays

```
String[] namen;
```

```
String namen[];
```

```
String []namen;
```

## ■ Initialisierung von Arrays

```
namen = {"Tim","Mario","Sebastian"};
```



# + Array – Bauanleitung

## ■ Deklaration von Arrays

```
String[] namen;
```

```
String namen[];
```

```
String []namen;
```

## ■ Initialisierung von Arrays

```
namen = {"Tim","Mario","Sebastian"};
```

namen



# + Array – Bauanleitung

## ■ Initialisierung von Arrays

```
String[] namen = new String[3];
```

namen



# + Array – Bauanleitung

## ■ Initialisierung von Arrays

```
String[] namen = new String[3];
```

namen



# + Array – Bauanleitung

## ■ Initialisierung von Arrays

```
String[] namen = new String[3];
```

```
namen[0] = "Tim";
```

namen



# + Array – Bauanleitung

## ■ Initialisierung von Arrays

```
String[] namen = new String[3];
```

```
name[0] = "Tim";  
name[1] = "Mario";
```

namen



# + Array – Bauanleitung

## ■ Initialisierung von Arrays

```
String[] namen = new String[3];
```

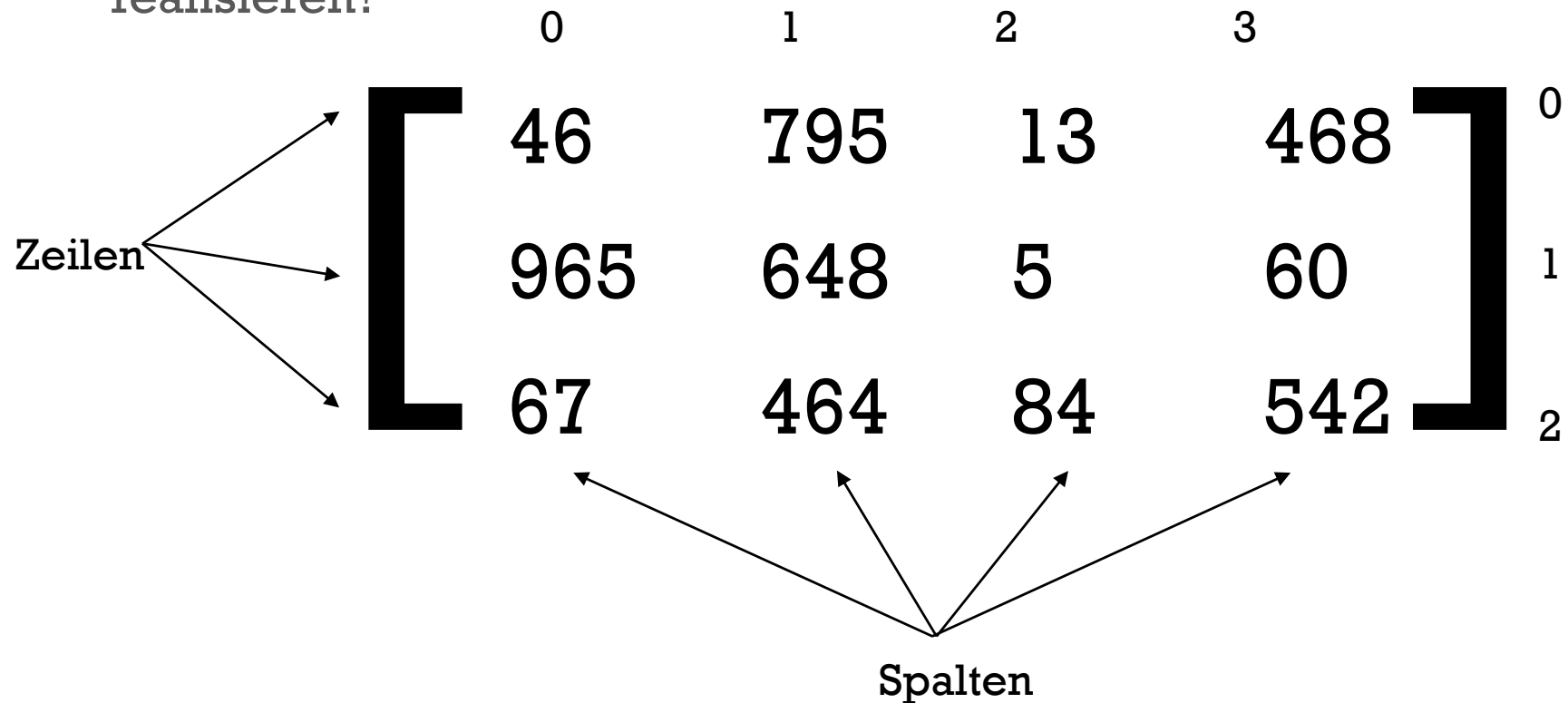
```
name[0] = "Tim";  
name[1] = "Mario";  
name[2] = "Sebastian";
```

namen



## + Array – Beispiel: Matrix

- Wie würde man eine Matrix in Java mittels Array's realisieren?



## + Array – Beispiel: Matrix

	0	1	2	3	
0	46	795	13	468	0
1	965	648	5	60	1
2	67	464	84	542	2

```
int[][] matrix = new int[3][4];
```

```
matrix[0][0] = 46;
```

```
matrix[0][1] = 795;
```

```
matrix[0][2] = 13;
```

```
matrix[0][3] = 468;
```

```
matrix[1][0] = 965;
```

```
matrix[1][1] = 648;
```

```
matrix[1][2] = 5;
```

```
Matrix[1][3] = 60;
```

```
matrix[2][0] = 67;
```

```
matrix[2][1] = 464;
```

```
matrix[2][2] = 84;
```

```
matrix[2][3] = 542;
```



## + Array – Beispiel: Matrix

	0	1	2	3	
0	46	795	13	468	0
1	965	648	5	60	1
2	67	464	84	542	2

```
int[][] matrix = new int[3][4];
```

```
matrix[0][0] = 46;  
matrix[0][1] = 795;  
matrix[0][2] = 13;  
matrix[0][3] = 468;
```

```
matrix[1][0] = 965;  
matrix[1][1] = 648;  
matrix[1][2] = 5;  
matrix[1][3] = 60;
```

```
matrix[2][0] = 67;  
matrix[2][1] = 464;  
matrix[2][2] = 84;  
matrix[2][3] = 542;
```

Alternativ

```
int[][] matrix = new int[3][4];
```

```
int[] zeile0 = {46,795,13,468};
```

```
int[] zeile1 = {965,648,5,60};
```

```
int[] zeile2 = {67,464,84,542};
```

```
matrix[0] = zeile0;
```

```
matrix[1] = zeile1;
```

```
matrix[2] = zeile2;
```

wie so etwas eleganter zu  
lösen ist, erklärt euch Tim

# + Array – Fehler #1

```
int[] zug;  
zug[0] = 2;  
zug[1] = 4;  
zug[2] = 1;
```

- wo liegt der Fehler?

## + Array – Fehler #1

```
int[] zug;  
zug[0] = 2;  
zug[1] = 4;  
zug[2] = 1;
```

Fehler der beim Compilieren erkannt wird

- wo liegt der Fehler?
- Fehlermeldung von Java:

```
ZugFehler1.java:5: variable zug might not have been initialized  
    zug[0] = 2;  
    ^  
1 error
```

## + Array – Fehler #1

```
int[] zug = new int[3];  
zug[0] = 2;  
zug[1] = 4;  
zug[2] = 1;
```

- wo liegt der Fehler?
  - Fehlermeldung von Java:

*COMPILIERT OHNE FEHLERMELDUNG*

## + Array – Fehler #2

```
int[] zug = new int[3];  
zug[3] = 5;
```

- wo liegt der Fehler?

## + Array – Fehler #2

```
int[] zug = new int[3];  
zug[3] = 5;
```

Fehler der erst zur Laufzeit erkannt wird

- wo liegt der Fehler?
- Fehlermeldung von Java:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 3  
    at ZugFehler2.main(ZugFehler2.java:5)
```

## + Array – Fehler #2

```
int[] zug = new int[4];  
zug[3] = 5;
```

- wo liegt der Fehler?
  - Fehlermeldung von Java:

*AUSFÜHRUNG OHNE FEHLER*

# + Array

32

- Vermeidung eines **ArrayIndexOutOfBoundsException**
  - mit Hilfe einer if-Bedingung:

```
int[] zug = new int[3];  
int i = 3;  
if ( zug.length < i ) {  
    zug[i] = 5;  
} else {  
    //Fehlerbehandlung  
}
```



# + Array

33

- Man erhält die Länge (Größe) eines Arrays

```
variablenname.length
```

```
int[] zug = {2,4,1};  
int size = zug.length;
```

- Welche Zahl steht in size?

# + Array

34

- Man erhält die Länge (Größe) eines Arrays

```
variablenname.length
```

```
int[] zug = {2,4,1};  
int size = zug.length;
```

- Welche Zahl steht in size?

3

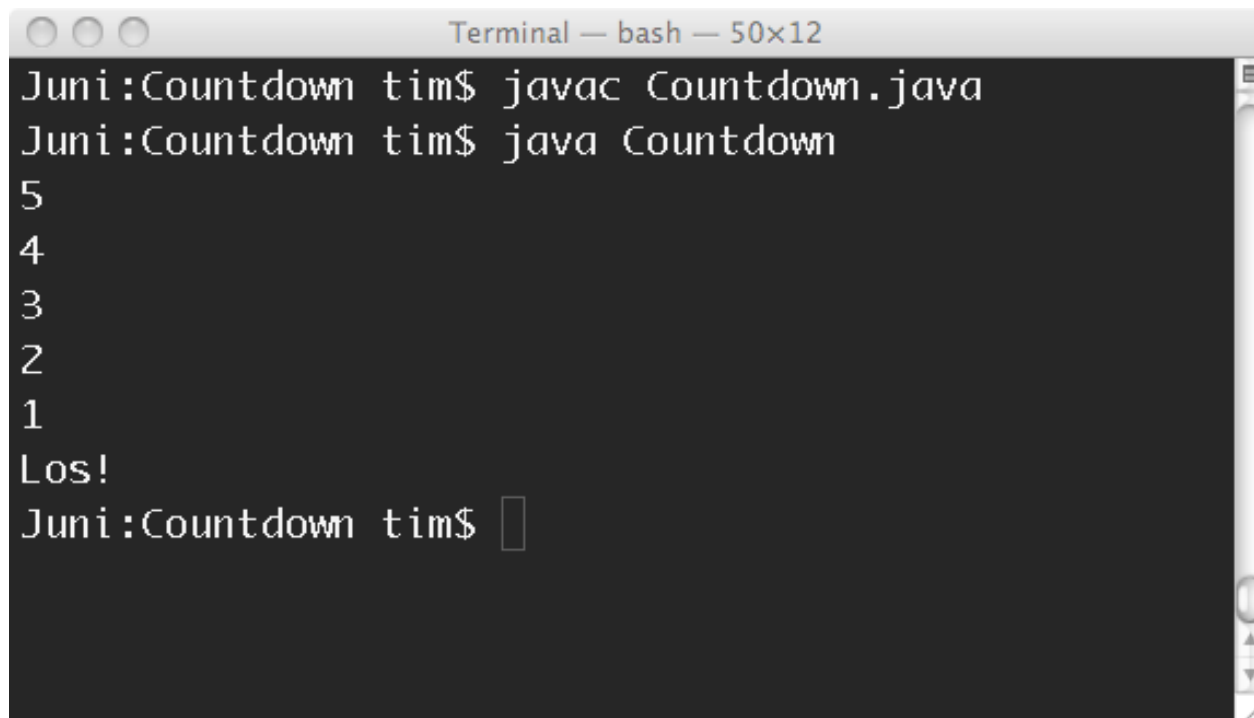


# Inhaltsverzeichnis

- Was ist ein Array?
- Array
  - Bauanleitung
  - Beispiel: Matrix
  - Fehler #1
  - Fehler #2
- Schleifen
  - Der Anfang einer Schleife
  - Die while Schleife
  - Fehler #1
  - Fehler #2
  - Der letzte Ausweg
  - Beispiel: Collatz-Folge
  - Die for Schleife
  - Beispiel: String Array
  - Beispiel: Matrix

## + Der Anfang einer Schleife

- Aufgabe: Schreibe ein Java Programm welches einen Countdown von 5 abwärts auf der Konsole ausgibt:



```
Terminal — bash — 50x12
Juni:Countdown tim$ javac Countdown.java
Juni:Countdown tim$ java Countdown
5
4
3
2
1
Los!
Juni:Countdown tim$
```

# + Der Anfang einer Schleife

## ■ Folgender Lösungsansatz:

```
1      System.out.println("5");  
2      System.out.println("4");  
3      System.out.println("3");  
4      System.out.println("2");  
5      System.out.println("1");  
6      System.out.println("Los!");
```

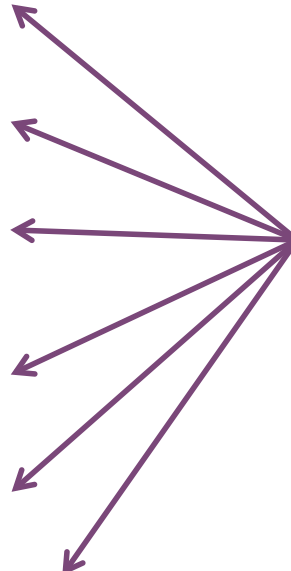
# + Der Anfang einer Schleife

■ Folgender Lösungsansatz:

```
1      System.out.println("5");  
2      System.out.println("4");  
3      System.out.println("3");  
4      System.out.println("2");  
5      System.out.println("1");  
6      System.out.println("Los!");
```

**Unsauber!**

Warum?



# + Der Anfang einer Schleife

- Aufgabe: Schreibe ein Java Programm welches einen Countdown von 1000 abwärts auf der Konsole ausgibt!

```
System.out.println("1000");  
System.out.println("999");  
System.out.println("998");  
System.out.println("997");  
System.out.println("996");  
System.out.println("995");  
System.out.println("994");  
System.out.println("993");  
System.out.println("992");  
System.out.println("991");  
System.out.println("990");  
System.out.println("989");  
System.out.println("988");  
System.out.println("987");  
...
```

Ist das elegant?

# + Der Anfang einer Schleife

- Aufgabe: Schreibe ein Java Programm welches einen Countdown von 1000 abwärts auf der Konsole ausgibt!

```
System.out.println("1000");  
System.out.println("999");  
System.out.println("998");  
System.out.println("997");  
System.out.println("996");  
System.out.println("995");  
System.out.println("994");  
System.out.println("993");  
System.out.println("992");  
System.out.println("991");  
System.out.println("990");  
System.out.println("989");  
System.out.println("988");  
System.out.println("987");  
...
```

**NEIN!**

Mehrfacher Code

Unnötige Schreibarbeit

Copy & Paste Fehler

Speicherverbrauch



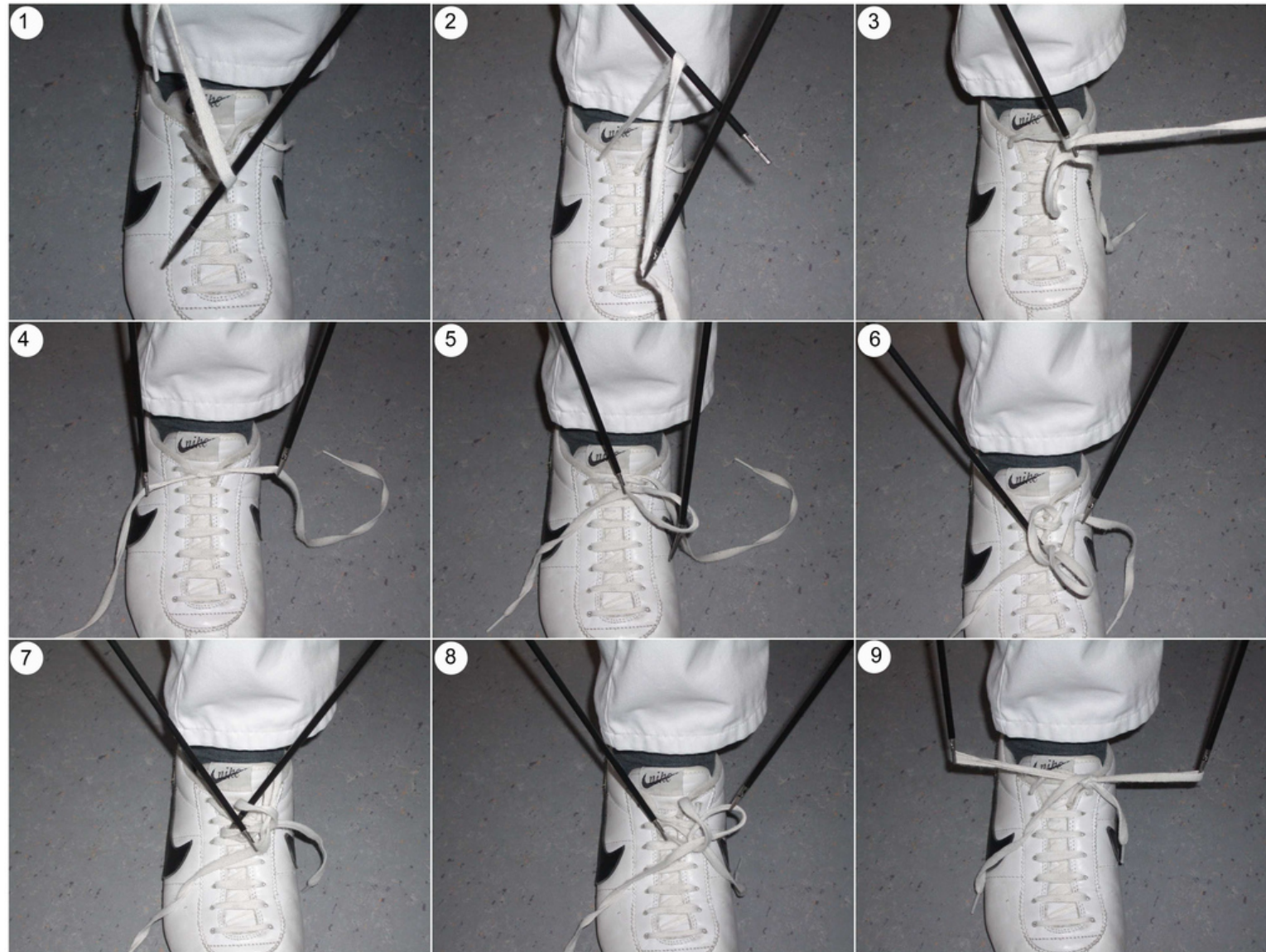
# + Schleifen

41



# + Wie bindet man eine Schleife?

42



# + Die while Schleife

- Aufgabe: Countdown von 1000 abwärts

```
1  int counter = 1000;  
2  
3  while(counter > 0){  
4      System.out.println(counter);  
5      counter = counter - 1;  
6  }  
7  System.out.println("Los!");
```

# + Die while Schleife

- Aufgabe: Countdown von 1000 abwärts

```
1  int counter = 1000;  
2  
3  while(counter > 0){  
4      System.out.println(counter);  
5      counter = counter - 1;  
6  }  
7  System.out.println("Los!");
```

Initialisierung

Bedingung

Dekrement



# + mögliche Fehlerquellen

45





# Fehler #1

46

- Aufgabe: Gib alle ungeraden positiven Zahlen kleiner 10 aus!

```
1  int grenze = 10;
2  int zahl = 1;
3
4  while(zahl < grenze){
5      // zahl ist ungerade
6      if(zahl % 2 == 1){
7          System.out.println(zahl);
8          zahl = zahl + 1;
9      }
10 }
```

Ist das korrekt?

## + Fehler #1

- Aufgabe: Gib alle ungeraden positiven Zahlen kleiner 10 aus!

```
1  int grenze = 10;
2  int zahl = 1;
3
4  while(zahl < grenze){
5      // zahl ist ungerade
6      if(zahl % 2 == 1){
7          System.out.println(zahl);
8          zahl = zahl + 1;
9      }
10 }
```

Ist das korrekt?

**NEIN!**

Endlosschleife



# Fehler #1

48

- Aufgabe: Gib alle ungeraden positiven Zahlen kleiner 10 aus!

```
1  int grenze = 10;
2  int zahl = 1;
3
4  while(zahl < grenze){
5      // zahl ist ungerade
6      if(zahl % 2 == 1){
7          System.out.println(zahl);
8      }
9      zahl = zahl + 1;
10 }
```

Korrekt!



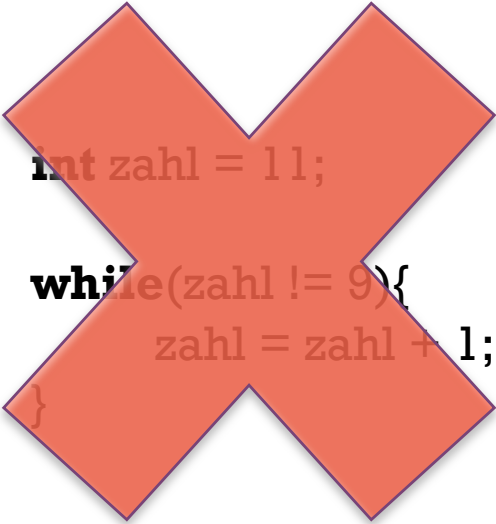
## + Fehler #2

49

```
1  int zahl = 11;  
2  
3  while(zahl != 9){  
4      zahl = zahl + 1;  
5  }
```

## + Fehler #2

50



```
1 int zahl = 11;  
2  
3 while(zahl != 9){  
4     zahl = zahl + 1;  
5 }
```

```
1 int zahl = 11;  
2  
3 while(zahl < 9){  
4     zahl = zahl + 1;  
5 }
```

toter Code

# + Der letzte Ausweg

Der Befehl **break**

```
while(zahl < 9){  
    if(zahl == -1){  
        break;  
    }  
    zahl = zahl + 1;  
}
```

Bitte nur im Ausnahmefall benutzen.

## + Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

$$c_{n+1} := \begin{cases} 3c_n + 1, & \text{falls } c_n \text{ ungerade und } \neq 1, \\ \frac{c_n}{2}, & \text{falls } c_n \text{ gerade,} \\ 1, & \text{falls } c_n = 1. \end{cases}$$

**Handsimulation:**

# + Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

```
1 int counter = 0;  
2 int collatz = 11;  
3
```

$$c_{n+1} := \begin{cases} 3c_n + 1, & \text{falls } c_n \text{ ungerade und } \neq 1, \\ \frac{c_n}{2}, & \text{falls } c_n \text{ gerade,} \\ 1, & \text{falls } c_n = 1. \end{cases}$$

# + Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

```
1  int counter = 0;
2  int collatz = 11;
3
4  while(collatz > 1){
5
6
7
8
9
10
11      counter = counter + 1;
12 }
```

$$c_{n+1} := \begin{cases} 3c_n + 1, & \text{falls } c_n \text{ ungerade und } \neq 1, \\ \frac{c_n}{2}, & \text{falls } c_n \text{ gerade,} \\ 1, & \text{falls } c_n = 1. \end{cases}$$

# + Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

```
1  int counter = 0;
2  int collatz = 11;
3
4  while(collatz > 1){
5      if(collatz % 2 == 1 && collatz != 1){
6          collatz = 3 * collatz + 1;
7      }
8      else if(collatz % 2 == 0){
9          collatz = collatz / 2;
10     }
11     counter = counter + 1;
12 }
13 System.out.println(counter);
```

$$c_{n+1} := \begin{cases} 3c_n + 1, & \text{falls } c_n \text{ ungerade und } \neq 1, \\ \frac{c_n}{2}, & \text{falls } c_n \text{ gerade,} \\ 1, & \text{falls } c_n = 1. \end{cases}$$

## + Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

```
1  int counter = 0;
2  int collatz = 11;
3
4  while(collatz > 1){
5      if(collatz % 2 == 1 && collatz != 1){
6          collatz = 3 * collatz + 1;
7      }
8      else if(collatz % 2 == 0){
9          collatz = collatz / 2;
10     }
11     counter = counter + 1;
12 }
13 System.out.println(counter);
```

$$c_{n+1} := \begin{cases} 3c_n + 1, & \text{falls } c_n \text{ ungerade und } \neq 1, \\ \frac{c_n}{2}, & \text{falls } c_n \text{ gerade,} \\ 1, & \text{falls } c_n = 1. \end{cases}$$

Ausgabe:

14



# + Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

```
1  int counter = 0;
2  int collatz = 11;
3
4  while(collatz > 1){
5      if(collatz % 2 == 1 && collatz != 1){
6          collatz = 3 * collatz + 1;
7      }
8      else if(collatz % 2 == 0){
9          collatz = collatz / 2;
10     }
11     counter = counter + 1;
12 }
13 System.out.println(counter);
```

Initialisierung

Bedingung

Inkrement

## + Die for Schleife

- Die Initialisierung, die Bedingung und das Inkrement/Dekrement lassen sich leicht in einer for Schleife zusammenfassen.

```
for(int i = 1000; i > 0; i = i - 1){  
    System.out.println(i);  
}  
System.out.println("Los!");
```

## + Die for Schleife

- Die Initialisierung, die Bedingung und das Inkrement/Dekrement lassen sich leicht in einer for Schleife zusammenfassen.

Initialisierung

Bedingung

Dekrement

```
for(int i = 1000; i > 0; i = i - 1){  
    System.out.println(i);  
}  
System.out.println("Los!");
```



## Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

```
1  int counter = 0;
2  int collatz = 11;
3
4  while(collatz > 1){
5      if(collatz % 2 == 1 && collatz != 1){
6          collatz = 3 * collatz + 1;
7      }
8      else if(collatz % 2 == 0){
9          collatz = collatz / 2;
10     }
11     counter = counter + 1;
12 }
13 System.out.println(counter);
```

## + Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

```
1  int counter = 0;
2  int collatz = 11;
3
4  while(collatz > 1){
5      if(collatz % 2 == 1 && collatz != 1){
6          collatz = 3 * collatz + 1;
7      }
8      else if(collatz % 2 == 0){
9          collatz = collatz / 2;
10     }
11     counter = counter + 1;
12 }
13 System.out.println(counter);
```



## Beispiel: Collatz-Folge

- Aufgabe: Nach wie vielen Berechnungen ist die Collatz-Folge von 11 gleich 1?

```
1  int collatz = 11;
2  int counter;
3
4  for(counter = 0; collatz > 1; counter = counter + 1){
5      if(collatz % 2 == 1 && collatz != 1){
6          collatz = 3 * collatz + 1;
7      }
8      else if(collatz % 2 == 0){
9          collatz = collatz / 2;
10     }
11 }
12 System.out.println(counter);
```

## + Beispiel: String-Array

- Gib alle Elemente des Arrays `gedicht` auf der Konsole aus!

```
1 String[] gedicht = new String[4];  
2 gedicht[0] = "Ein Mops kam in die Küche,";  
3 gedicht[1] = "und stahl dem Koch ein Ei.";  
4 gedicht[2] = "Da nahm der Koch den Löffel";  
5 gedicht[3] = "und schlug den Mops zu Brei.";
```



## + Beispiel: String-Array

- Gib alle Elemente des Arrays `gedicht` auf der Konsole aus!

```
1  String[] gedicht = new String[4];
2  gedicht[0] = "Ein Mops kam in die Küche,";
3  gedicht[1] = "und stahl dem Koch ein Ei."
4  gedicht[2] = "Da nahm der Koch den Löffel"
5  gedicht[3] = "und schlug den Mops zu Brei."
6
7  for(int i = 0;
8
9
```



## + Beispiel: String-Array

- Gib alle Elemente des Arrays `gedicht` auf der Konsole aus!

```
1  String[] gedicht = new String[4];
2  gedicht[0] = "Ein Mops kam in die Küche,";
3  gedicht[1] = "und stahl dem Koch ein Ei."
4  gedicht[2] = "Da nahm der Koch den Löffel"
5  gedicht[3] = "und schlug den Mops zu Brei."
6
7  for(int i = 0; i < gedicht.length;
8
9
```

## + Beispiel: String-Array

- Gib alle Elemente des Arrays `gedicht` auf der Konsole aus!

```
1  String[] gedicht = new String[4];
2  gedicht[0] = "Ein Mops kam in die Küche,";
3  gedicht[1] = "und stahl dem Koch ein Ei."
4  gedicht[2] = "Da nahm der Koch den Löffel"
5  gedicht[3] = "und schlug den Mops zu Brei."
6
7  for(int i = 0; i < gedicht.length; i = i + 1)
8
9
```

## + Beispiel: String-Array

- Gib alle Elemente des Arrays `gedicht` auf der Konsole aus!

```
1  String[] gedicht = new String[4];
2  gedicht[0] = "Ein Mops kam in die Küche,";
3  gedicht[1] = "und stahl dem Koch ein Ei."
4  gedicht[2] = "Da nahm der Koch den Löffel"
5  gedicht[3] = "und schlug den Mops zu Brei."
6
7  for(int i = 0; i < gedicht.length; i = i + 1){
8      System.out.println(gedicht[i]);
9  }
```



## Beispiel Matrix

68

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542

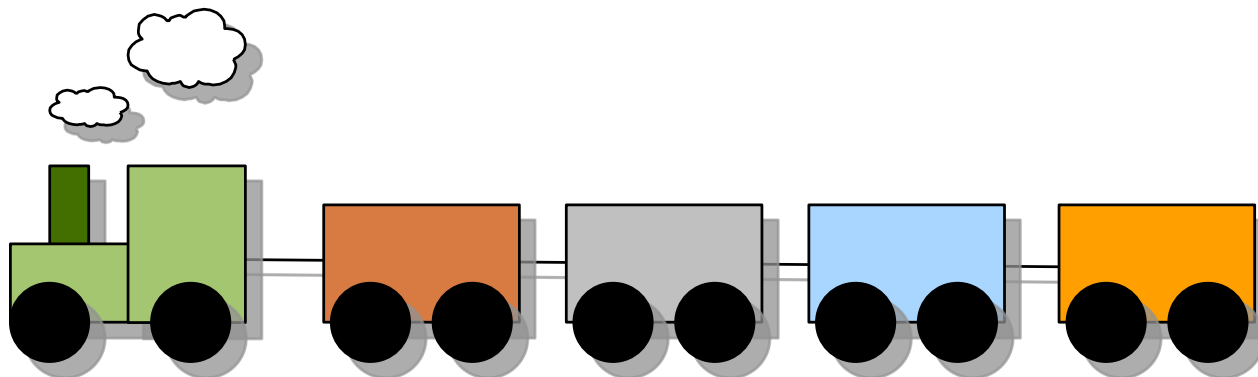


# Beispiel Matrix

69

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542

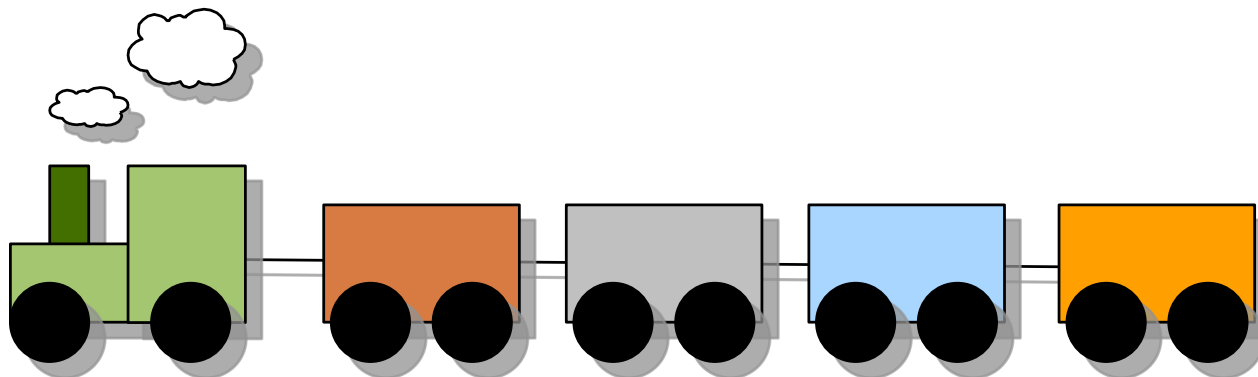


# + Beispiel Matrix

70

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

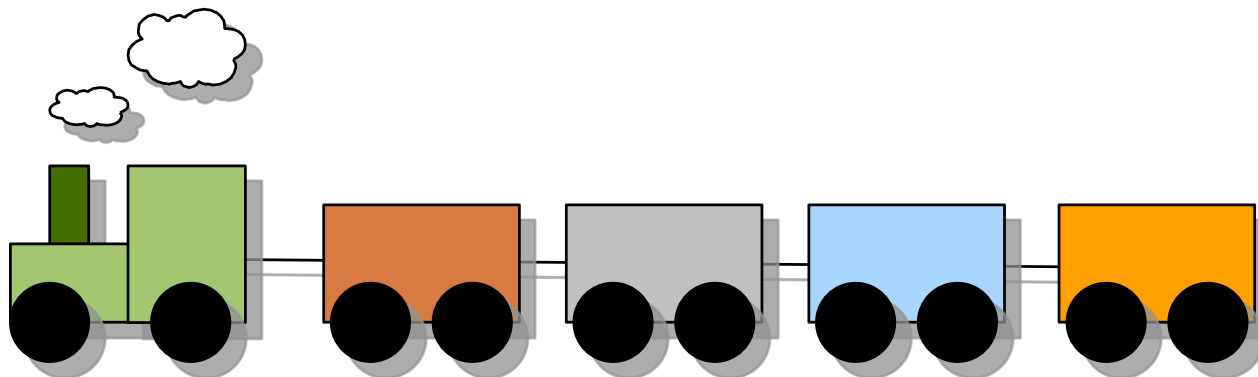
0

## + Beispiel Matrix

71

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

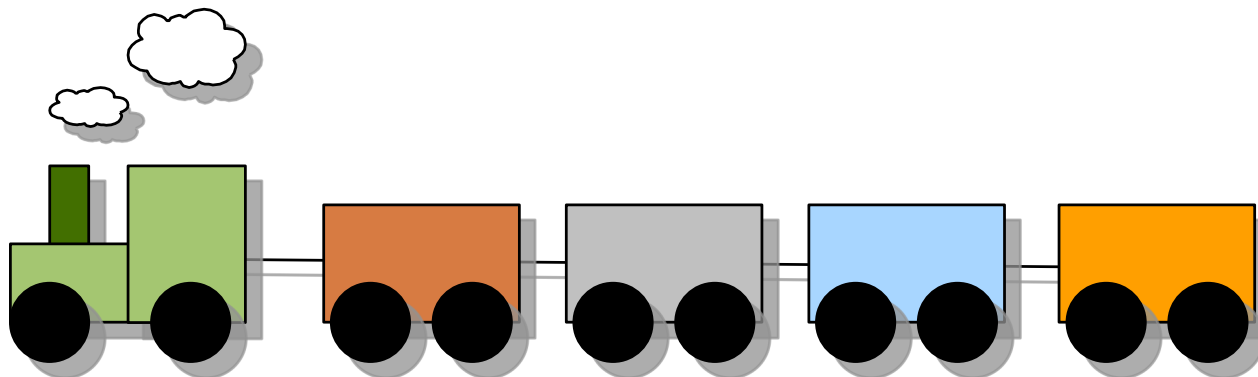
46

## + Beispiel Matrix

72

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

46

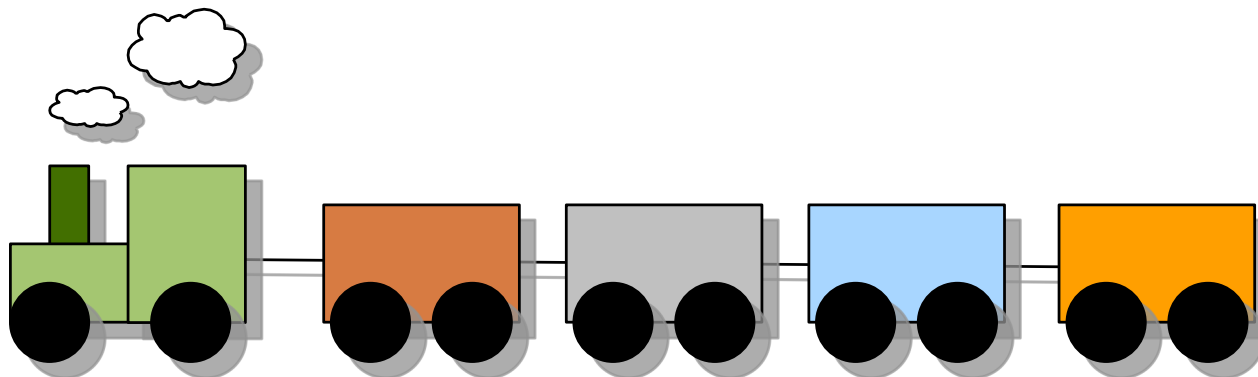


## + Beispiel Matrix

73

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

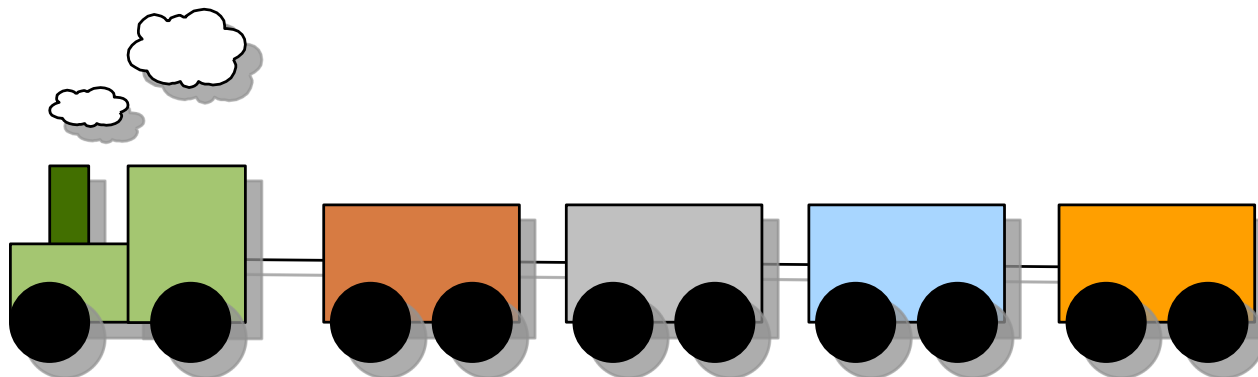
965

# + Beispiel Matrix

74

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

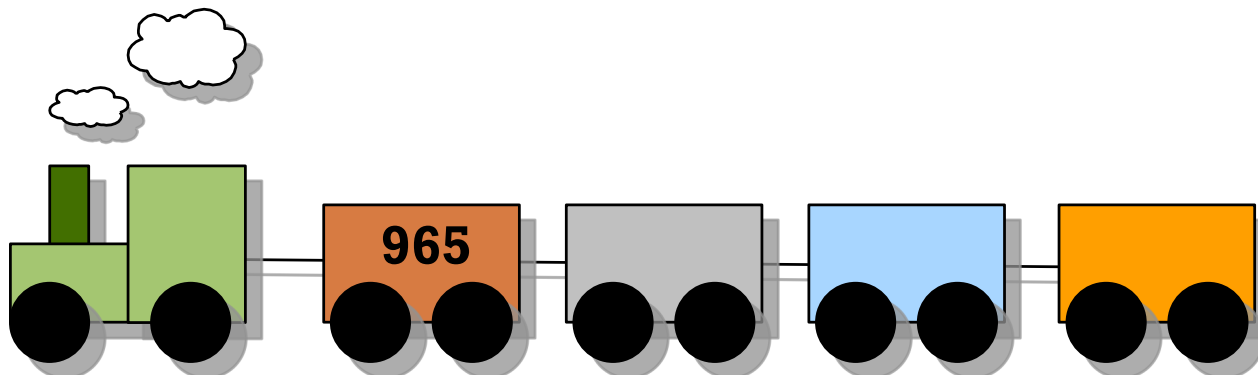
965

# + Beispiel Matrix

75

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

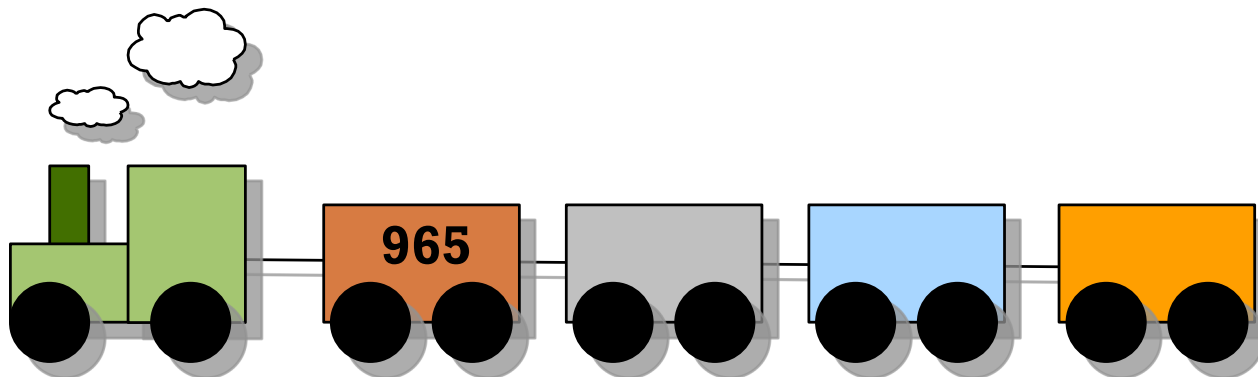
965

# + Beispiel Matrix

76

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

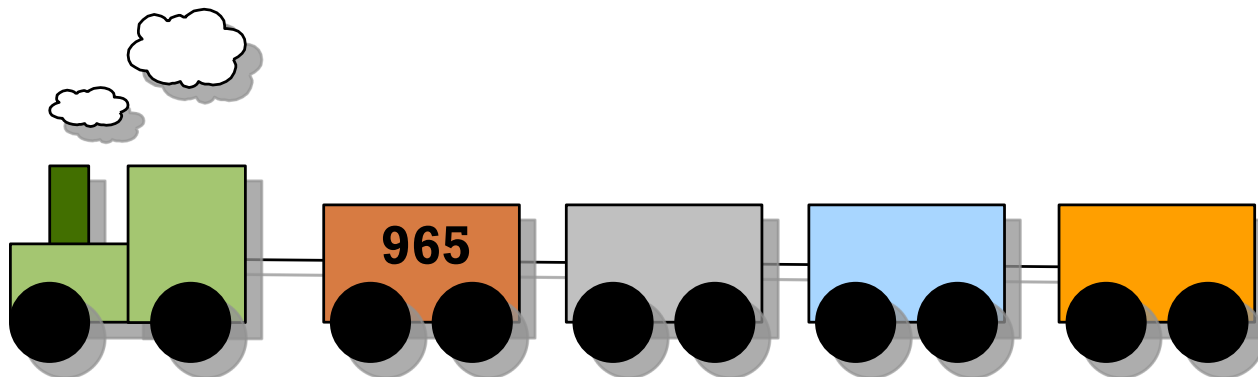
0

## + Beispiel Matrix

77

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

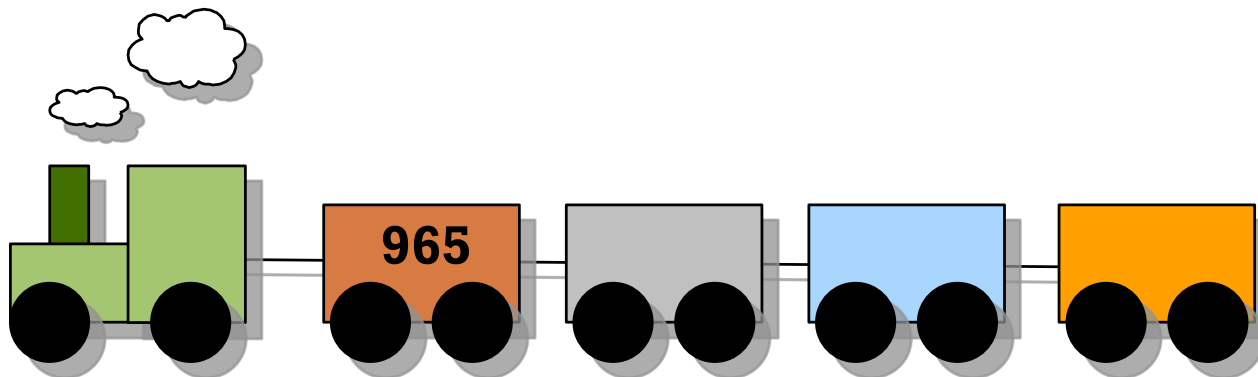
795

# + Beispiel Matrix

78

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

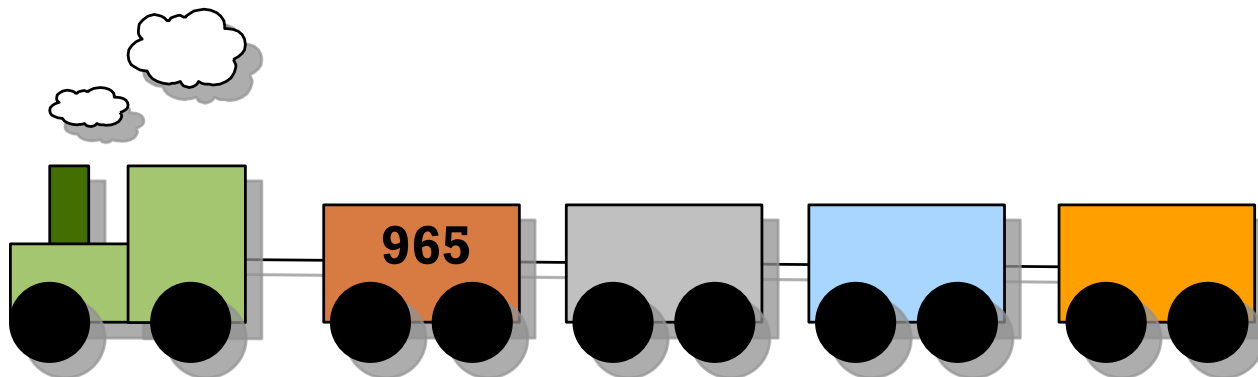
795

# + Beispiel Matrix

79

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

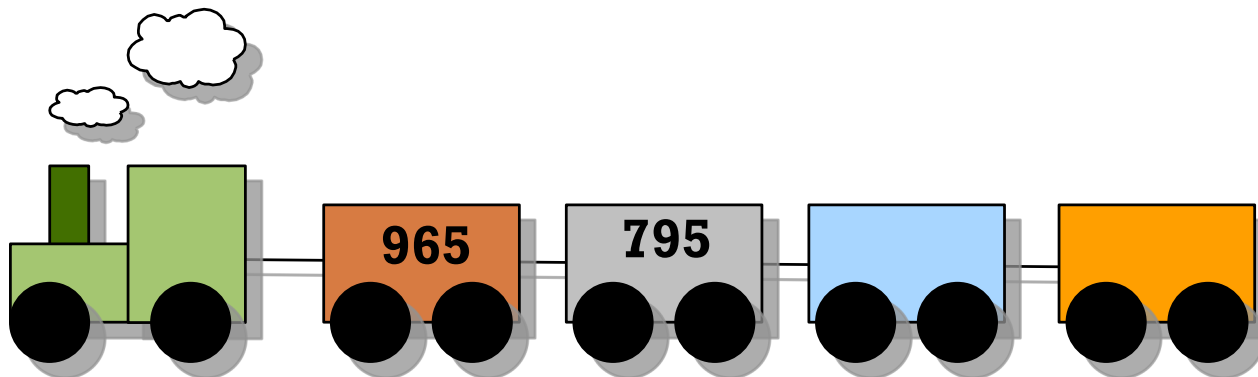
795

## + Beispiel Matrix

80

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

795

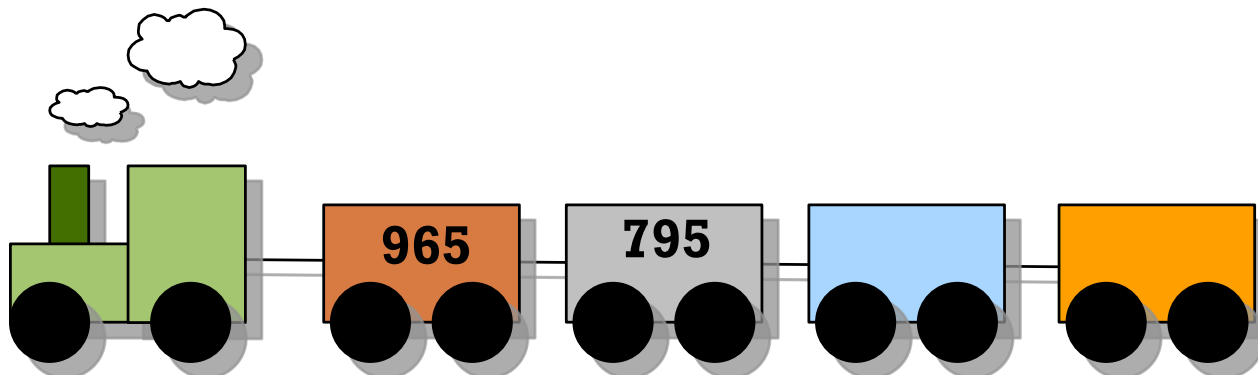


## + Beispiel Matrix

81

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

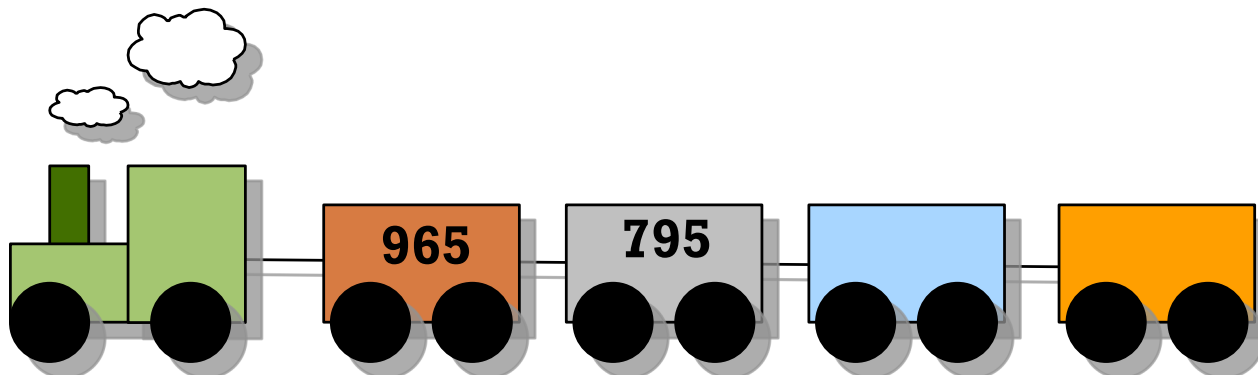
0

# + Beispiel Matrix

82

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

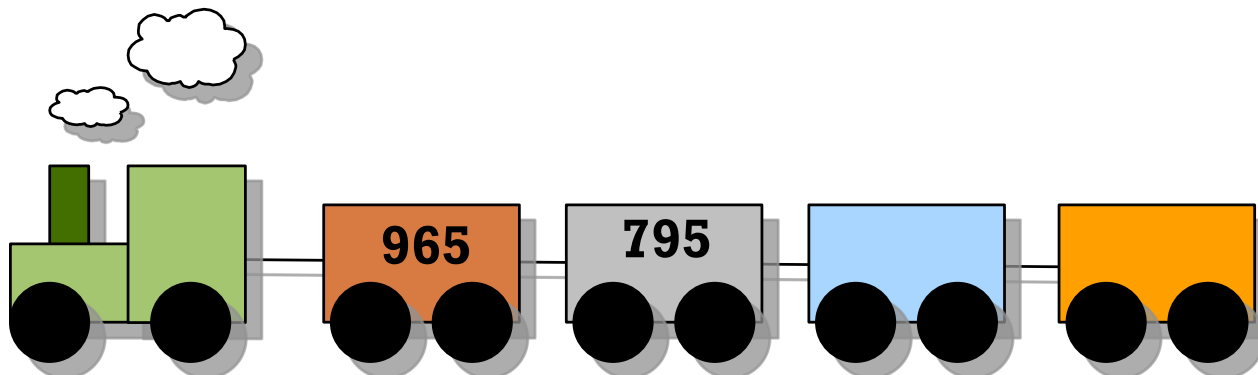
13

## + Beispiel Matrix

83

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

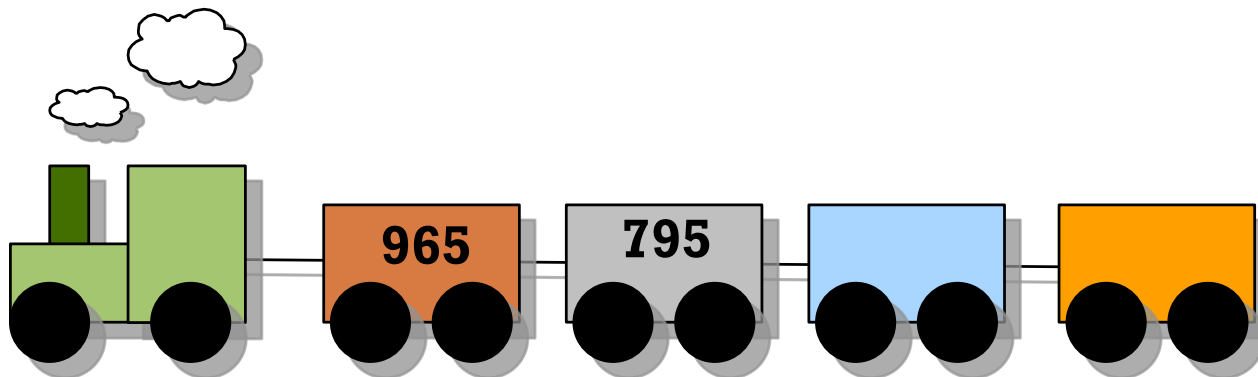
13

## + Beispiel Matrix

84

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

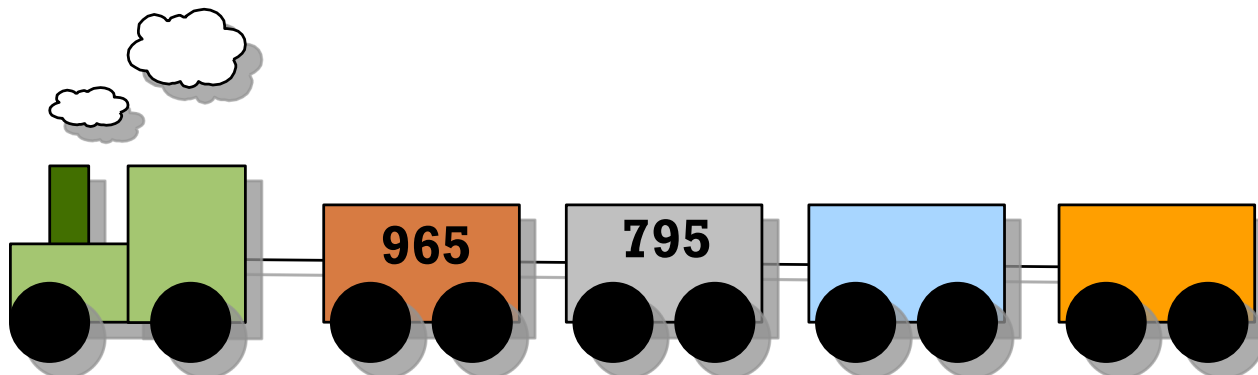
13

## + Beispiel Matrix

85

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

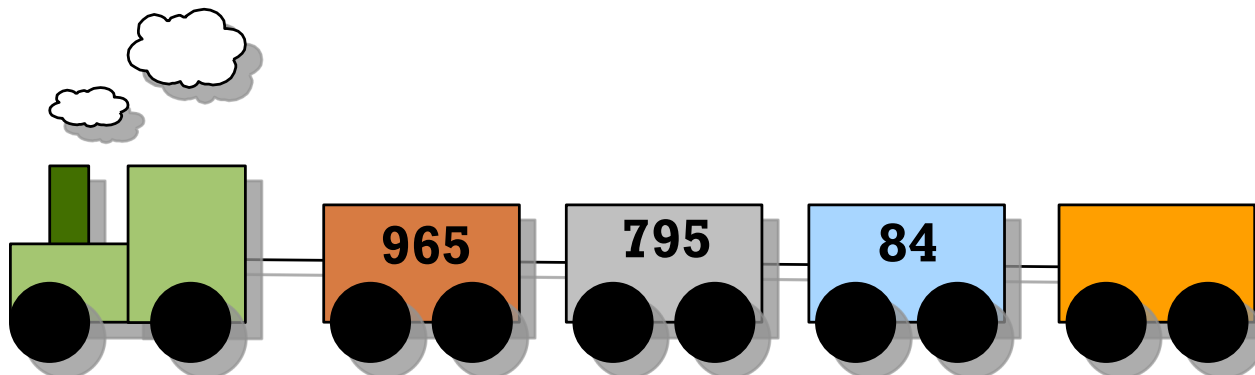
84

## + Beispiel Matrix

86

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

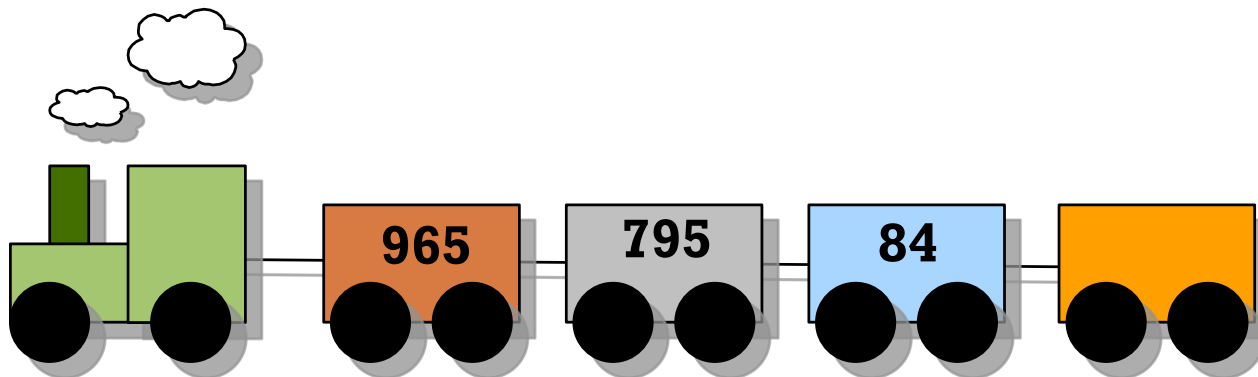
84

## + Beispiel Matrix

87

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



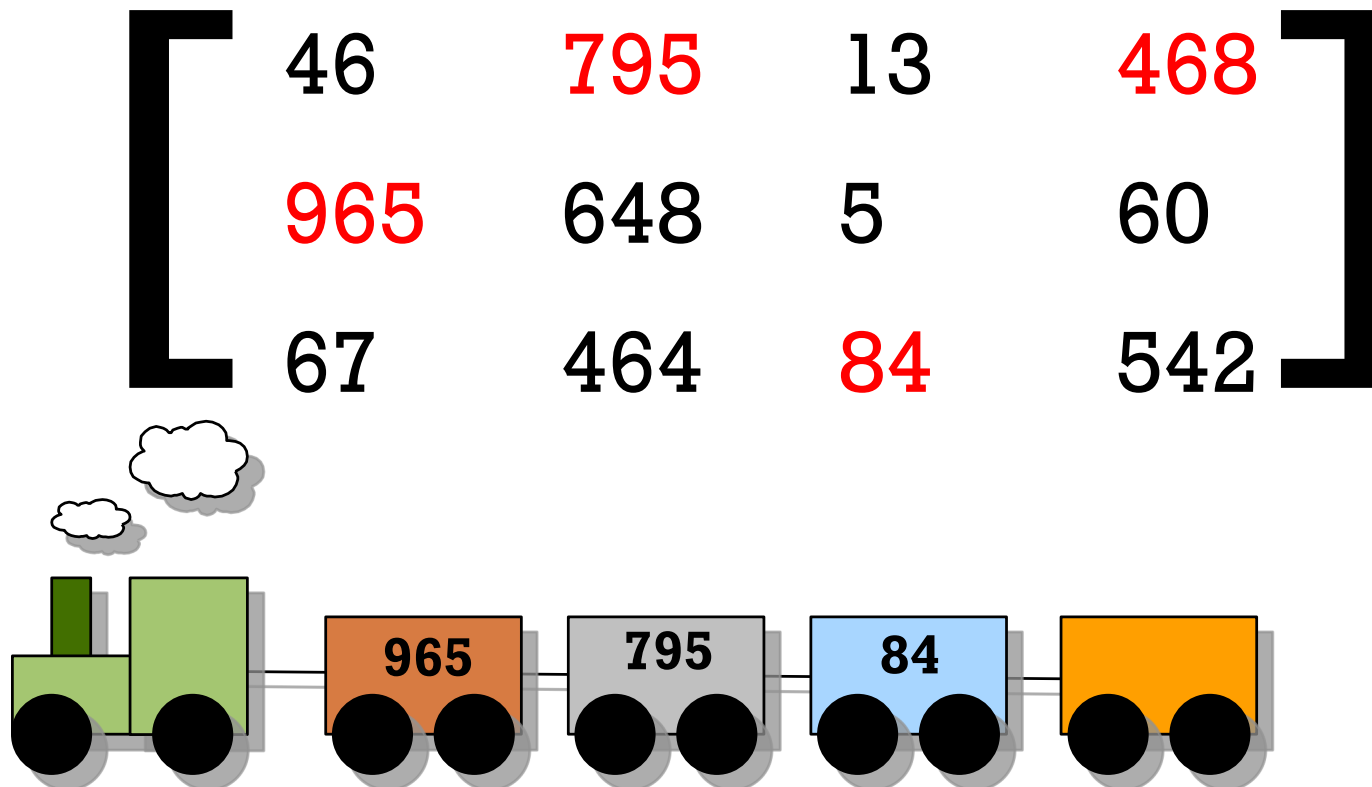
Aktuelles  
Maximum:

0

## + Beispiel Matrix

88

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!



Aktuelles  
Maximum:

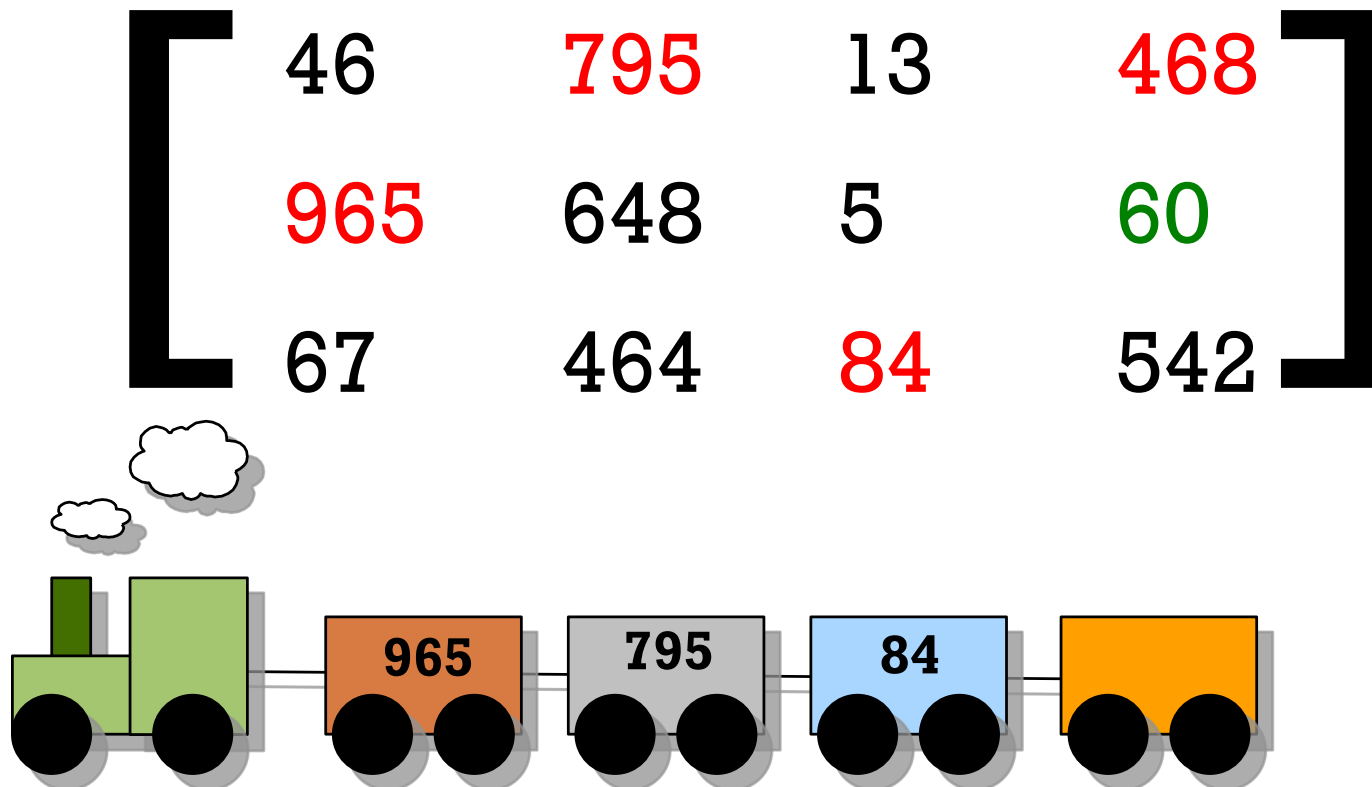
468



# + Beispiel Matrix

89

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!



Aktuelles  
Maximum:

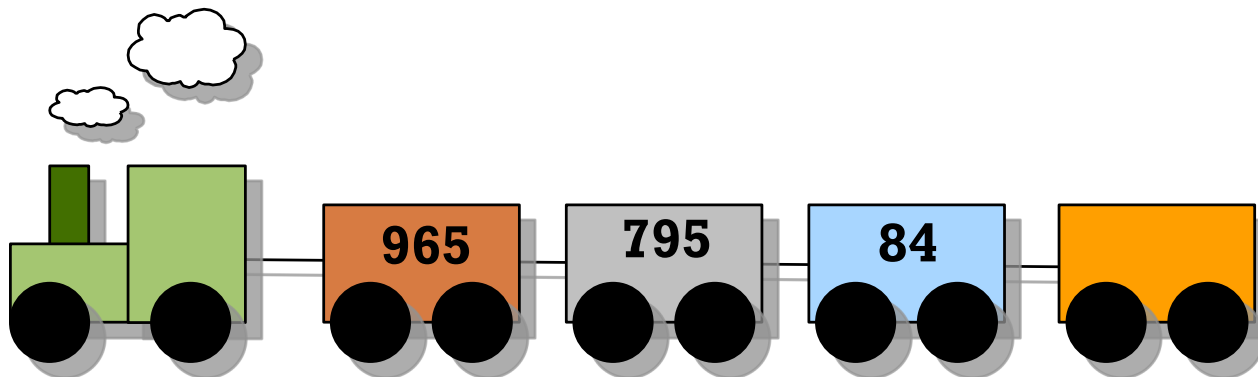
468

# + Beispiel Matrix

90

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



Aktuelles  
Maximum:

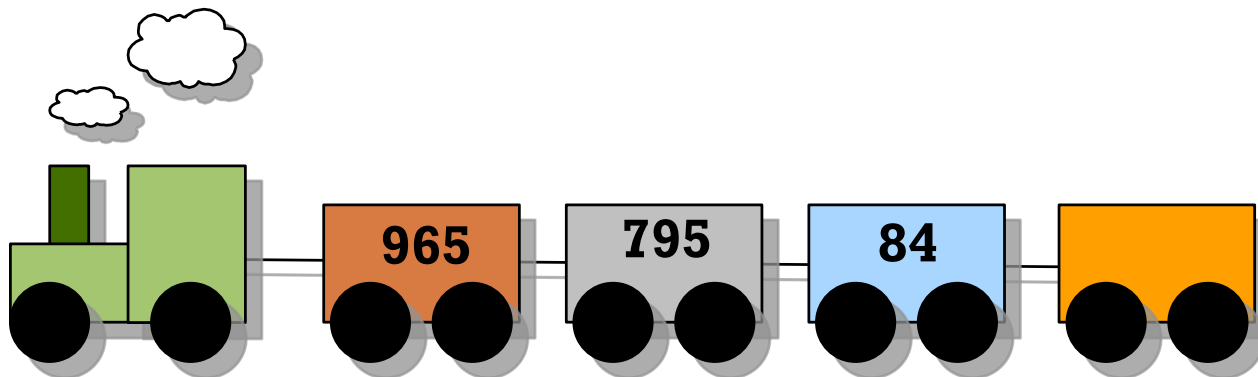
468

## + Beispiel Matrix

91

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!

46	795	13	468
965	648	5	60
67	464	84	542



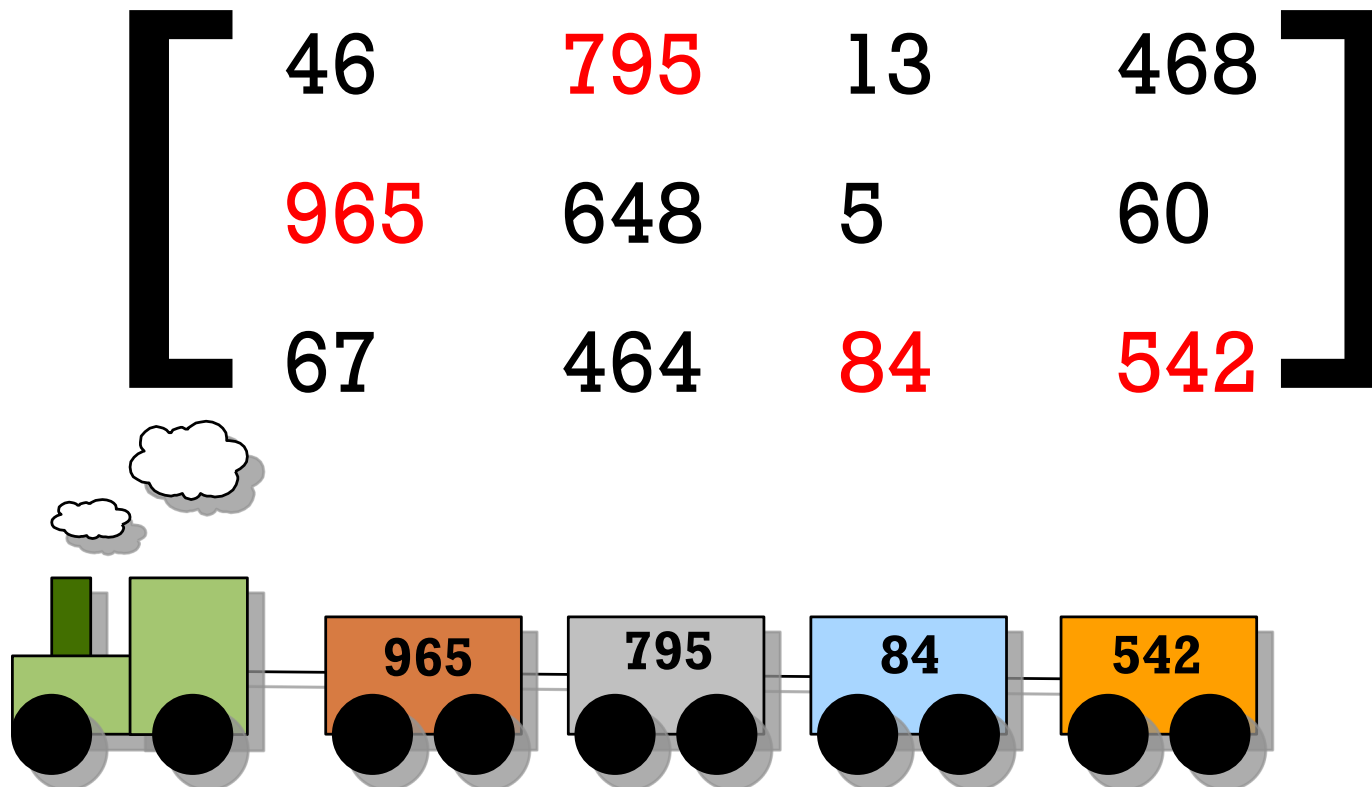
Aktuelles  
Maximum:

542

# + Beispiel Matrix

92

- Aufgabe: Finde das Maximum aus jeder Spalte und speichere das Ergebnis in einem neuen Array!



Aktuelles  
Maximum:

542

## + Beispiel Matrix

```
int[][] matrix =
```

46	795	13	468
965	648	5	60
67	464	84	542

## + Beispiel Matrix

94

```
int[][] matrix =
```

46	795	13	468
965	648	5	60
67	464	84	542

```
int[] ergebnis = new int[4];
```

## + Beispiel Matrix

95

```
int[][] matrix =
```

46	795	13	468
965	648	5	60
67	464	84	542

```
int[] ergebnis = new int[4];
```

```
for(int j = 0; j < 4; j++){
```

```
    for(int i = 0; i < 3; i++){
```

```
    }
```

```
}
```

## + Beispiel Matrix

96

```
int[][] matrix =
```

46	795	13	468
965	648	5	60
67	464	84	542

```
int[] ergebnis = new int[4];
```

```
for(int j = 0; j < 4; j++){  
    int maximum = 0;  
    for(int i = 0; i < 3; i++){
```

```
    }
```

```
}
```



## + Beispiel Matrix

```
int[][] matrix =
```

46	795	13	468
965	648	5	60
67	464	84	542

```
int[] ergebnis = new int[4];
```

```
for(int j = 0; j < 4; j++){  
    int maximum = 0;  
    for(int i = 0; i < 3; i++){  
        if(maximum < matrix[i][j]){  
            maximum = matrix[i][j];  
        }  
    }  
}
```

## + Beispiel Matrix

98

`int[][] matrix =`  $\begin{bmatrix} 46 & 795 & 13 & 468 \\ 965 & 648 & 5 & 60 \\ 67 & 464 & 84 & 542 \end{bmatrix}$

```
int[] ergebnis = new int[4];
```

```
for(int j = 0; j < 4; j++){  
    int maximum = 0;  
    for(int i = 0; i < 3; i++){  
        if(maximum < matrix[i][j]){  
            maximum = matrix[i][j];  
        }  
    }  
    ergebnis[j] = maximum;  
}
```

Fertig

+  
Ende

99

# Vielen Dank für die Aufmerksamkeit

Bitte Feedbackzettel abgeben

Jetzt: Übungsaufgaben im TEL

Für Fragen stehen wir jederzeit bereit

Quellen der Bilder chronologisch:

[1] User: mightymikey73, URL: <http://www.flickr.com/photos/mightymikey73/2459063122/sizes/o/>

[2] User: mirgo, URL: [www.flickr.com/photos/migro/218662105/sizes/l/](http://www.flickr.com/photos/migro/218662105/sizes/l/)

[3] User: oliklee, URL: [http://farm4.static.flickr.com/3276/2945259290\\_df52829f09\\_o.jpg](http://farm4.static.flickr.com/3276/2945259290_df52829f09_o.jpg)

[4] User: tottix, URL: <http://www.flickr.com/photos/tottix/4279051374/sizes/l/>