Folge 4.1
Einfache Arrays

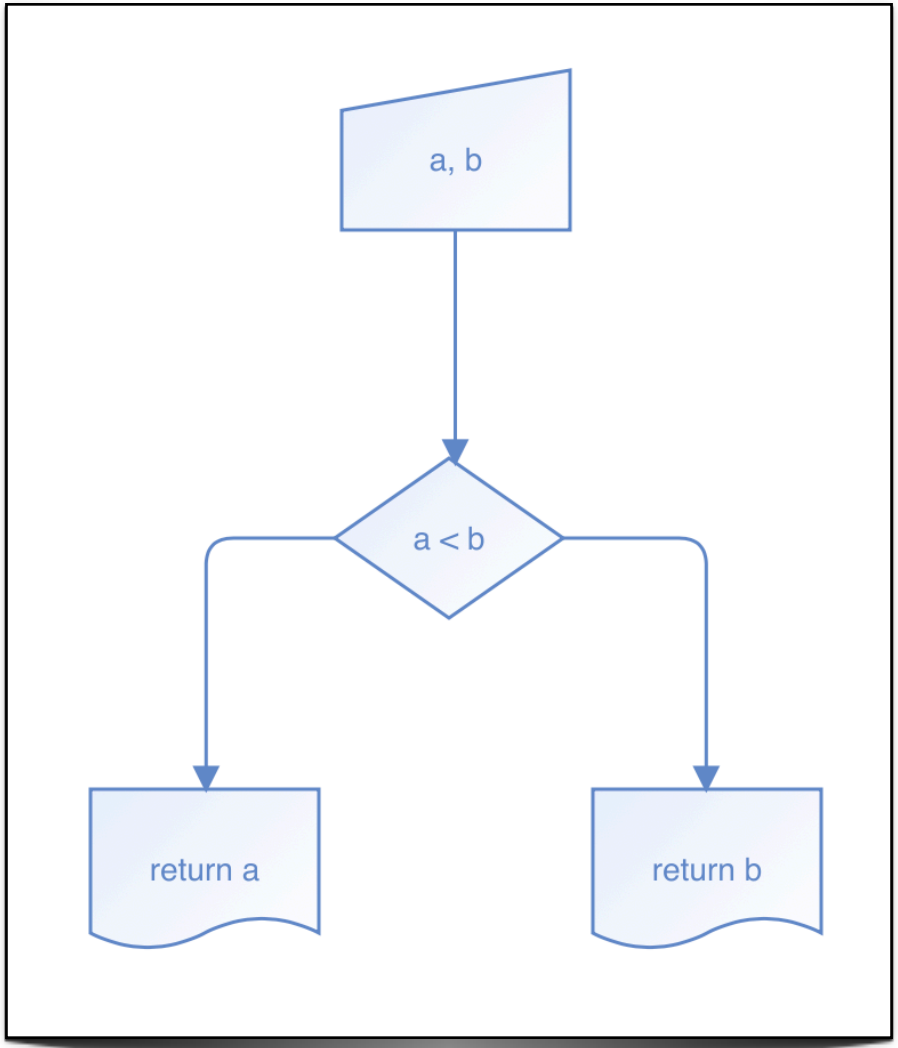
4.1 Einfache Arrays

1. Einführendes Beispiel
2. Gründliche Analyse eines Programms
3. Theorie: Was ist überhaupt ein Array?
4. Aufgaben

4.1.1 Einführendes Beispiel

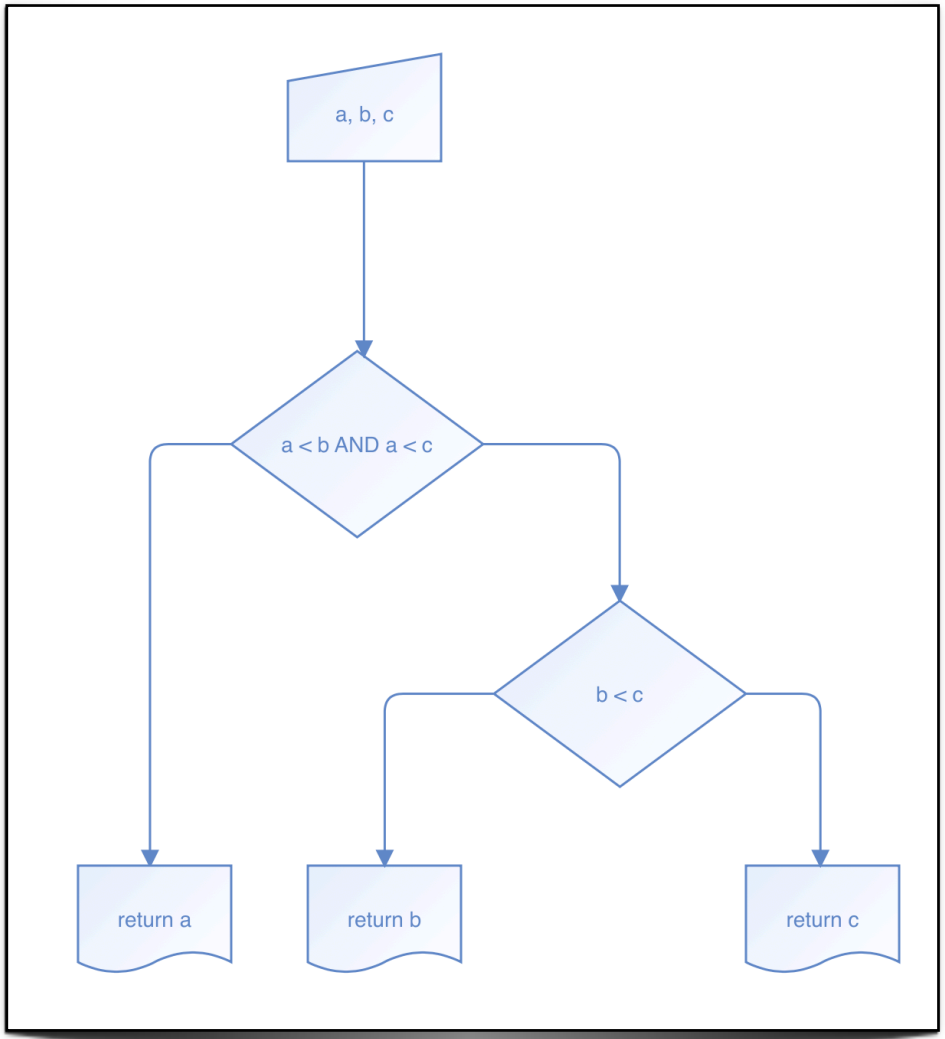
```java
public int mini(int a, int b)
{
    if (a < b)
        return a;
    else
        return b;
}
```

```java
public int miniTernaer(int a, int b)
{
    return (a < b) ? a : b;
}
```



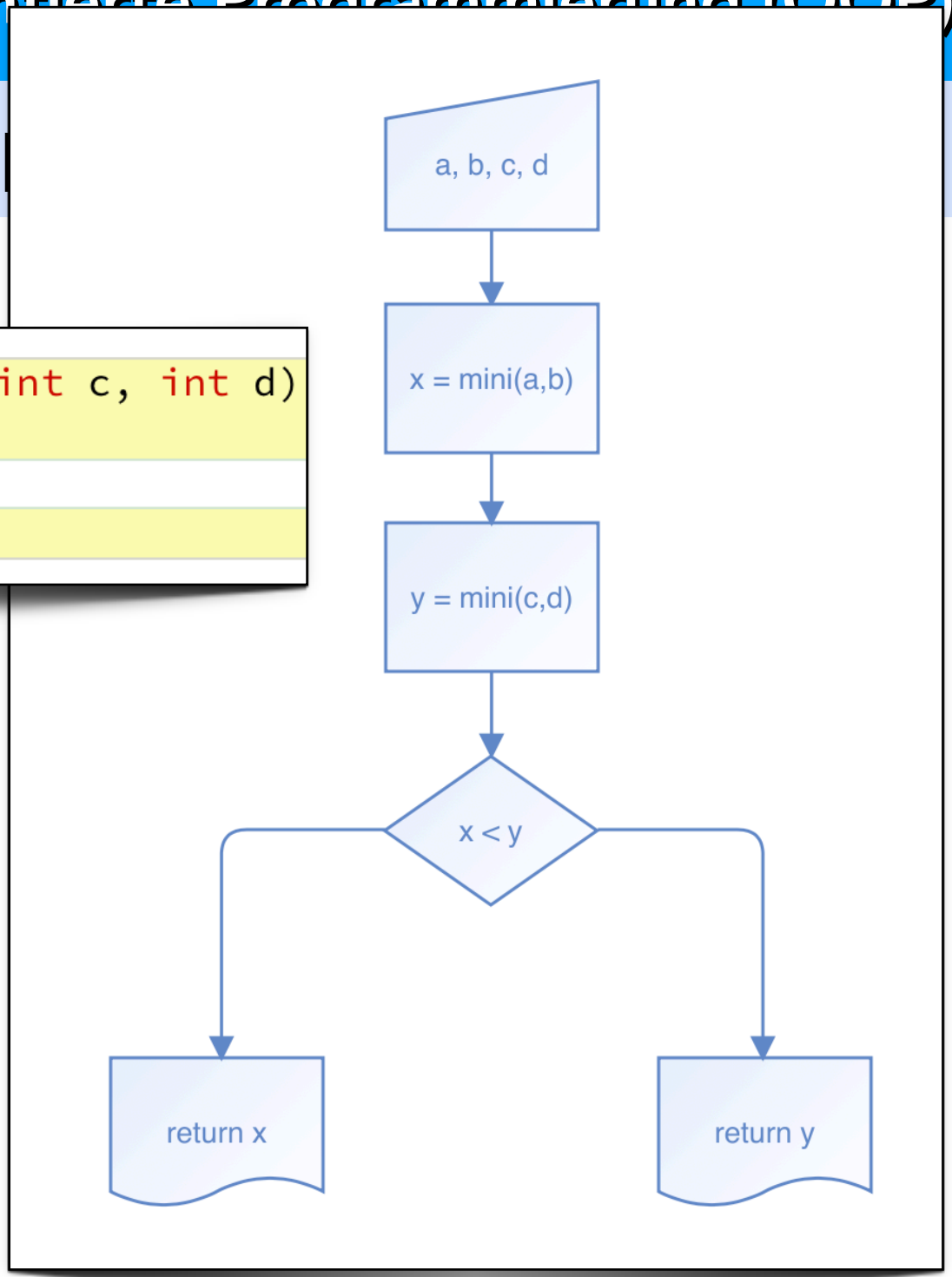
4.1.1 Einführendes Beispiel

```
public int mini(int a, int b, int c)
{
    if ((a < b) && (a < c))
        return a;
    else if (b < c)
        return b;
    else
        return c;
}
```



4.1.1 Einführendes Beis

```
public int miniVonVier(int a, int b, int c, int d)
{
    return mini(mini(a,b),mini(c,d));
}
```



4.1.1 Einführendes Beispiel

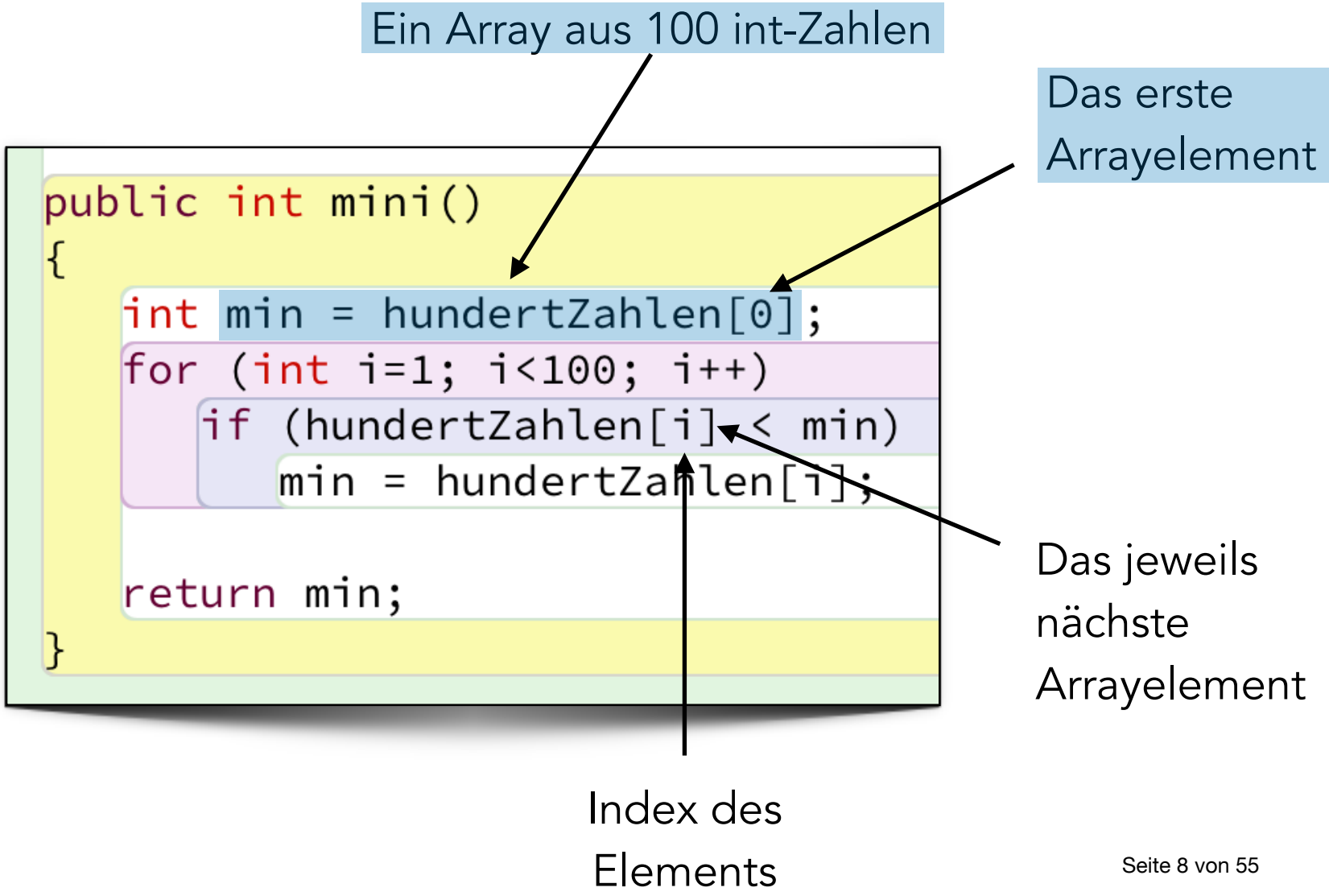
```java
public int miniVonVierV1(int a, int b, int c, int d)
{
    return mini(mini(a,b),mini(c,d));
}
```

```java
public int miniVonVierV2(int a, int b, int c, int d)
{
    int mini = a;
    if (b < mini) mini = b;
    if (c < mini) mini = c;
    if (d < mini) mini = d;

    return mini;
}
```

4.1.1 Einführendes Beispiel

```
public int mini()
{
    int min = hundertZahlen[0];
    for (int i=1; i<100; i++)
        if (hundertZahlen[i] < min)
            min = hundertZahlen[i];

    return min;
}
```

4.1.1 Einführendes Beispiel



4.1.

```java
public class ArrayTest
{
    int[] hundertZahlen;

    public ArrayTest()
    {
        hundertZahlen = new int[100];
    }

    public void erzeugeArray()
    {
        for (int i=0; i<100; i++)
            hundertZahlen[i] = (int) (Math.random()*1000) + 1;
    }

    public int mini()
    {
        int min = hundertZahlen[0];
        for (int i=1; i<100; i++)
            if (hundertZahlen[i] < min)
                min = hundertZahlen[i];

        return min;
    }
}
```

4.1.

```
1  public class ArrayTest
2  {
3      int[] hundertZahlen;
4
5      public ArrayTest()
6      {
7          hundertZahlen = new int[100];
8      }
9
10     public void erzeugeArray()
11     {
12         for (int i=0; i<100; i++)
13             hundertZahlen[i] = (int) (Math.random()*1000) + 1;
14     }
15
16     public int mini()
17     {
18         int min = hundertZahlen[0];
19         for (int i=1; i<100; i++)
20             if (hundertZahlen[i] < min)
21                 min = hundertZahlen[i];
22
23         return min;
24     }
25 }
```

Deklaration des Arrays

Initialisierung des Arrays

```
1  public class ArrayTest
2  {
3      int[] hundertZahlen;
4
5      public ArrayTest()
6      {
7          hundertZahlen = new int[100];
8      }
9
10     public void erzeugeArray()
11     {
12         for (int i=0; i<100; i++)
13             hundertZahlen[i] = (int) (Math.random()*1000) + 1;
14     }
15
16     public int mini()
17     {
18         int min = hundertZahlen[0];
19         for (int i=1; i<100; i++)
20             if (hundertZahlen[i] < min)
21                 min = hundertZahlen[i];
22
23         return min;
24     }
25 }
```

Deklaration des Arrays

Initialisierung des Arrays

Füllen des Arrays mit 100 Zufallszahlen

4.1.

```
1  public class ArrayTest
2  {
3      int[] hundertZahlen;
4
5      public ArrayTest()
6      {
7          hundertZahlen = new int[100];
8      }
9
10     public void erzeugeArray()
11     {
12         for (int i=0; i<100; i++)
13             hundertZahlen[i] = (int) (Math.random()*1000) + 1;
14     }
15
16     public int mini()
17     {
18         int min = hundertZahlen[0];
19         for (int i=1; i<100; i++)
20             if (hundertZahlen[i] < min)
21                 min = hundertZahlen[i];
22
23         return min;
24     }
25 }
```

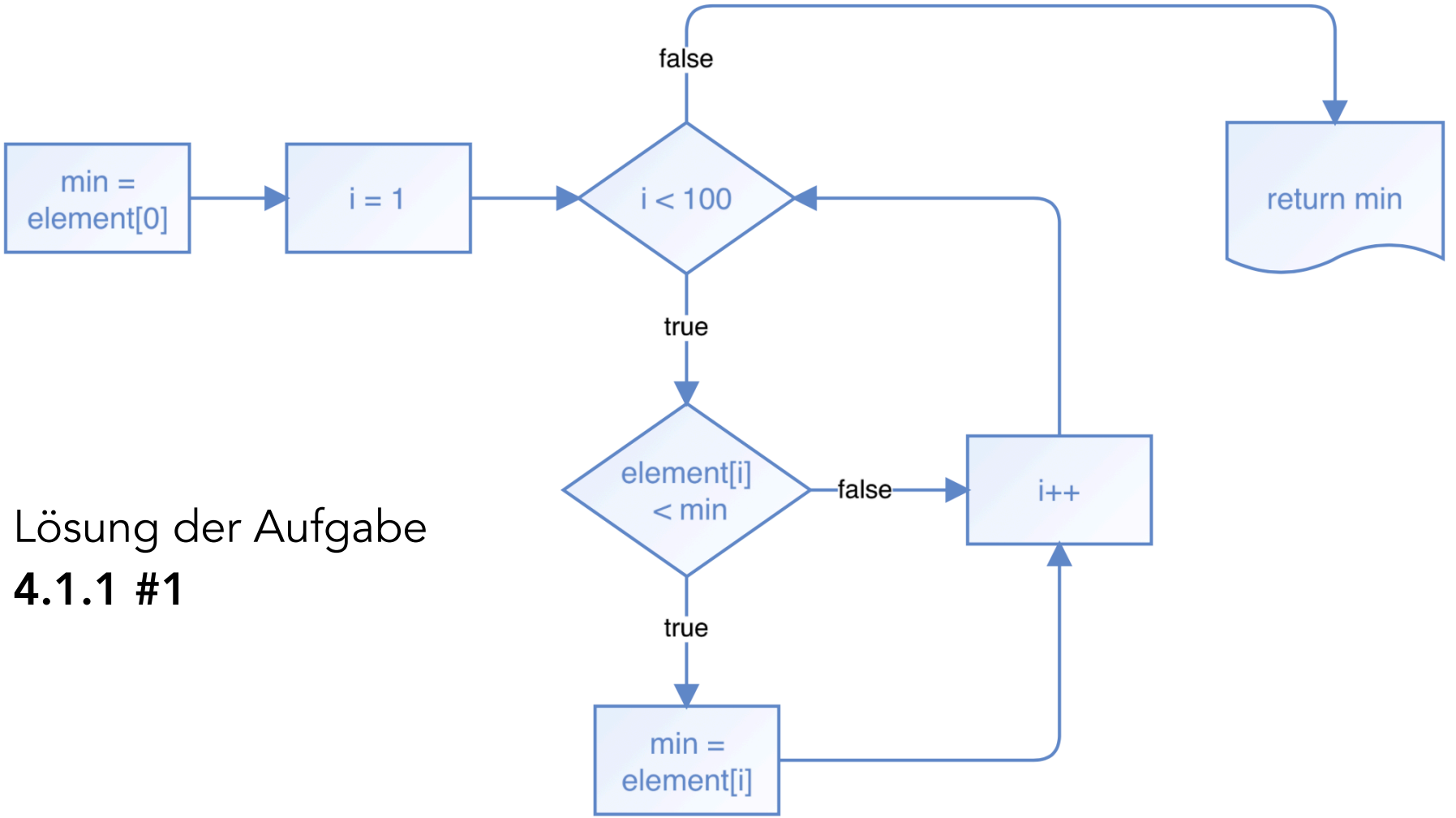
Deklaration des Arrays

Initialisierung des Arrays

Füllen des Arrays mit 100 Zufallszahlen

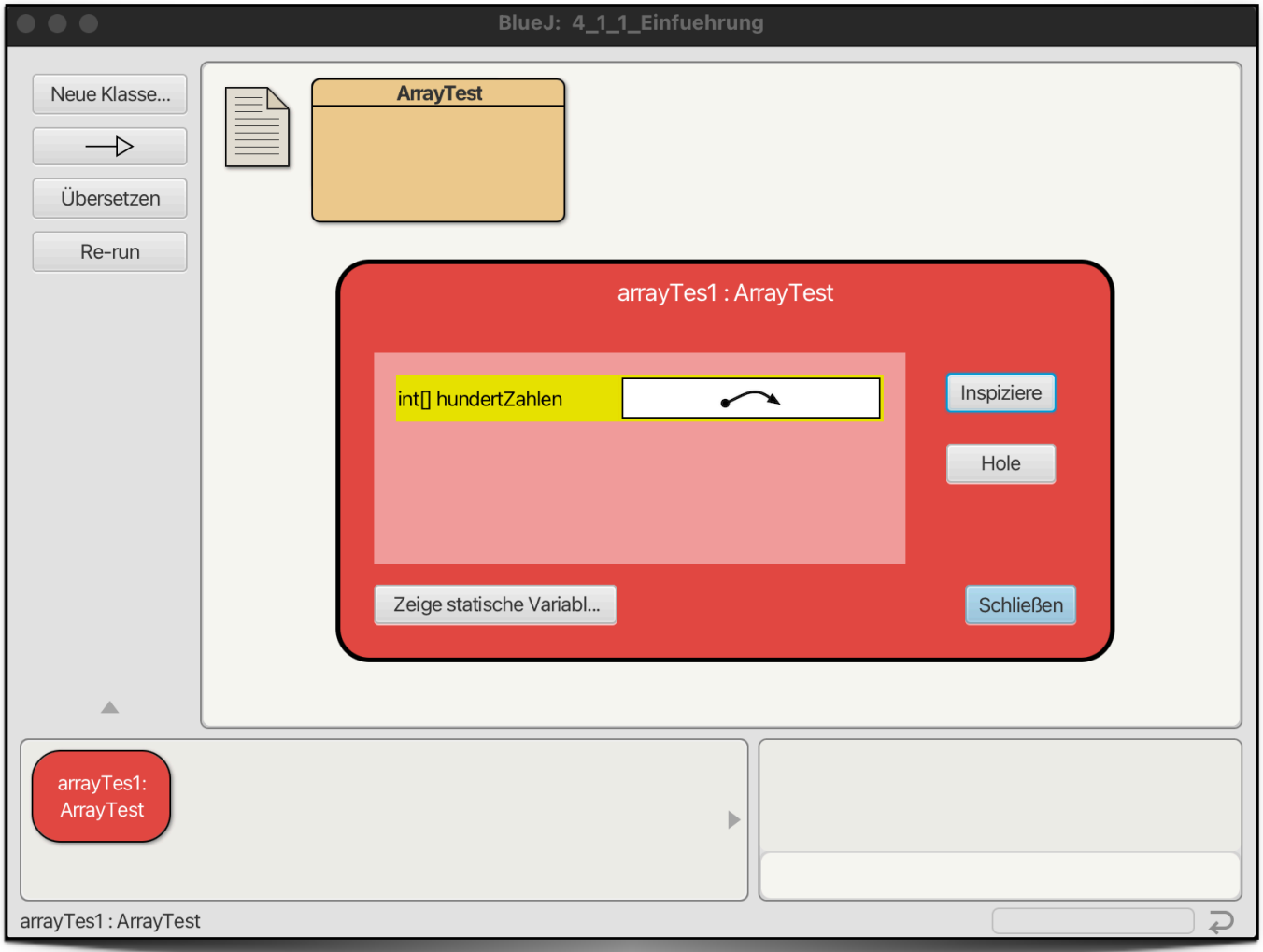
Suchen des kleinsten Elementes im Array

4.1.1 Einführendes Beispiel



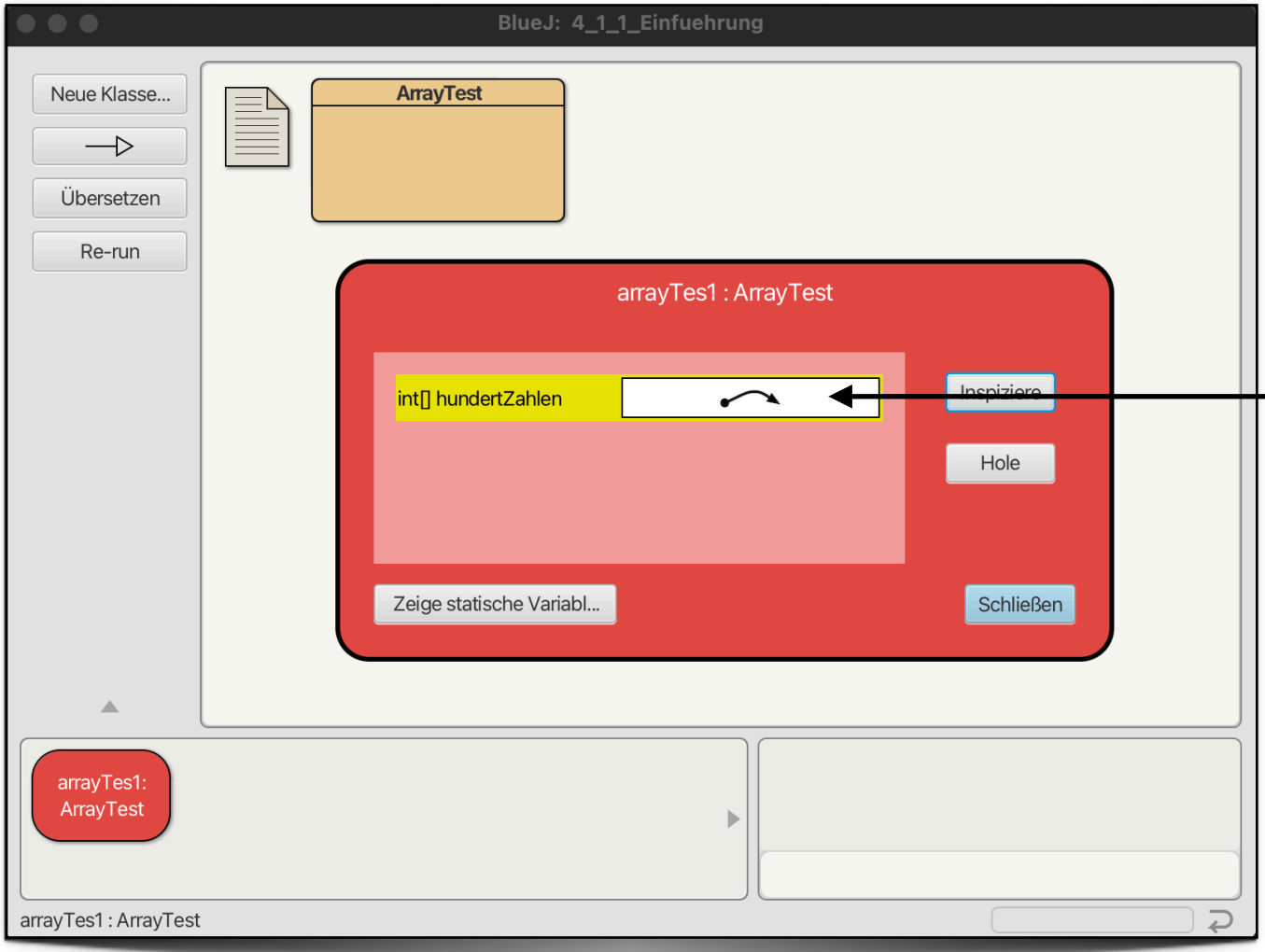
Lösung der Aufgabe
4.1.1 #1

4.1.1 Einführendes Beispiel



Untersuchung eines Objekts der Klasse **ArrayTest**

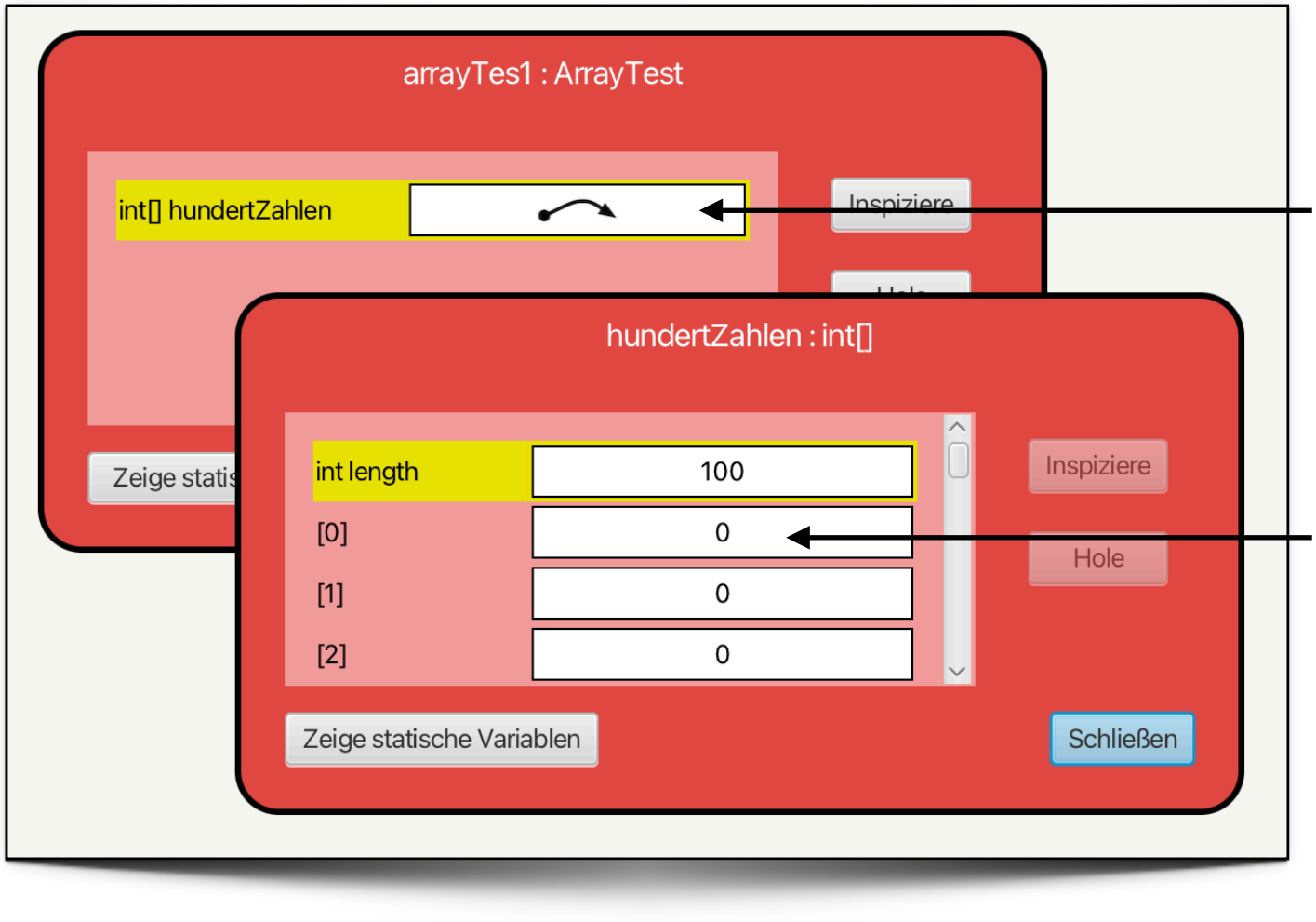
4.1.1 Einführendes Beispiel



Untersuchung eines Objekts der Klasse **ArrayTest**

Das Array **int[] hundertZahlen**

4.1.1 Einführendes Beispiel



Doppelklick auf den Pfeil

Die ersten drei Elemente des Arrays

4.1.1 Einführendes Beispiel

Ein Student, der unbedingt alle 100 Element im Objektinspektor sehen möchte.

4.1.2 Gründliche Analyse des Programms

```
3  public class ArrayTest
4  {
5      int[] hundertZahlen;
6
7      public ArrayTest()
8      {
9          hundertZahlen = new int[100];
10     }
```

Deklaration des Arrays

4.1.2 Gründliche Analyse des Programms

```
3  public class ArrayTest
4  {
5      int[] hundertZahlen;
6
7      public ArrayTest()
8      {
9          hundertZahlen = new int[100];
10     }
```

Deklaration des Arrays

`hundertZahlen` ⟶ `null`

4.1.2 Gründliche Analyse des Programms

```
3  public class ArrayTest
4  {
5      int[] hundertZahlen;
6
7      public ArrayTest()
8      {
9          hundertZahlen = new int[100];
10     }
```

← ***Initialisierung*** des Arrays

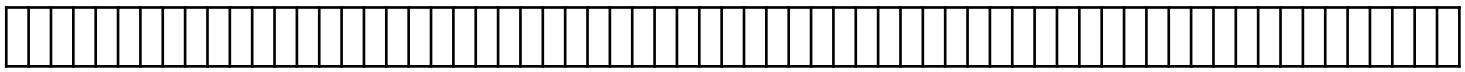
`hundertZahlen` ⟶ `null` nach Deklaration, vor Initialisierung

4.1.2 Gründliche Analyse des Programms

```
3  public class ArrayTest
4  {
5      int[] hundertZahlen;
6
7      public ArrayTest()
8      {
9          hundertZahlen = new int[100];
10     }
```

← ***Initialisierung*** des Arrays

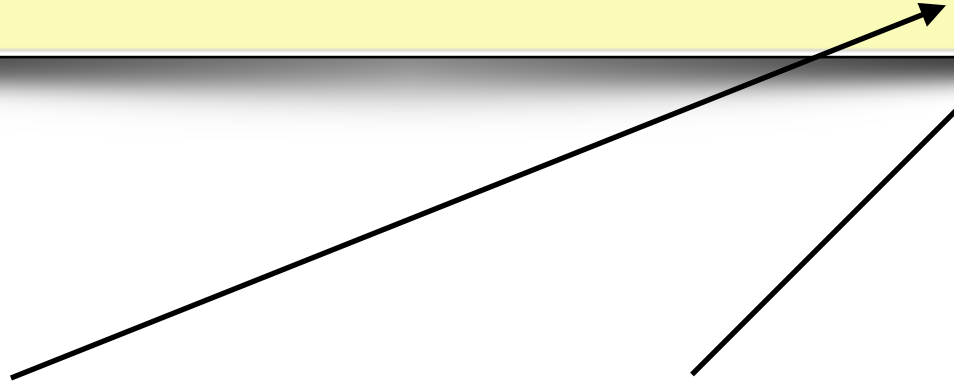
`hundertZahlen` → `null` nach Deklaration, vor Initialisierung

`hundertZahlen` → 

nach Initialisierung

4.1.2 Gründliche Analyse des Programms

```
10    public void erzeugeArray()
11    {
12        for (int i=0; i<100; i++)
13            hundertZahlen[i] = (int) (Math.random()*1000) + 1;
14    }
```



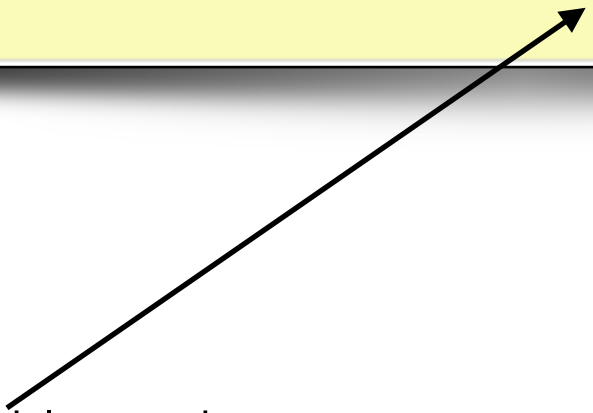
Zufallszahl zwischen
0 (inklusive)
und 1.0 (exklusive)

Zufallszahl zwischen
0 (inklusive)
und 1000.0 (exklusive)

Zufallszahl zwischen
1.0 (inklusive)
und 1001.0 (exklusive)

4.1.2 Gründliche Analyse des Programms

```
10      public void erzeugeArray()
11      {
12          for (int i=0; i<100; i++)
13              hundertZahlen[i] = (int) (Math.random()*1000) + 1;
14      }
```



int-Zufallszahl zwischen
1 (inklusive)
und 1000 (exklusive)

4.1.2 Gründliche Analyse des Programms

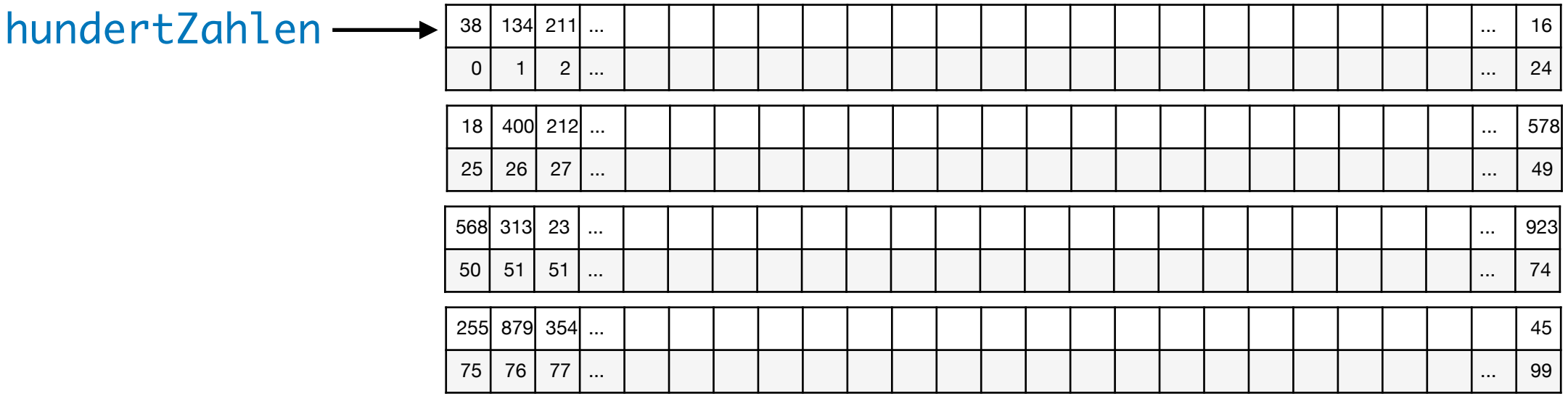
Die ersten 14 Elemente des Arrays nach Auffüllen mit Zufallszahlen

4.1.2 Gründliche Analyse des Programms

```
public int mini()
{
    int min = hundertZahlen[0];
    for (int i=1; i<100; i++)
        if (hundertZahlen[i] < min)
            min = hundertZahlen[i];

    return min;
}
```

Die Suche nach der kleinsten Zahl im Array.

4.1.3 Was ist überhaupt ein Array?



4.1.3 Was ist überhaupt ein Array?

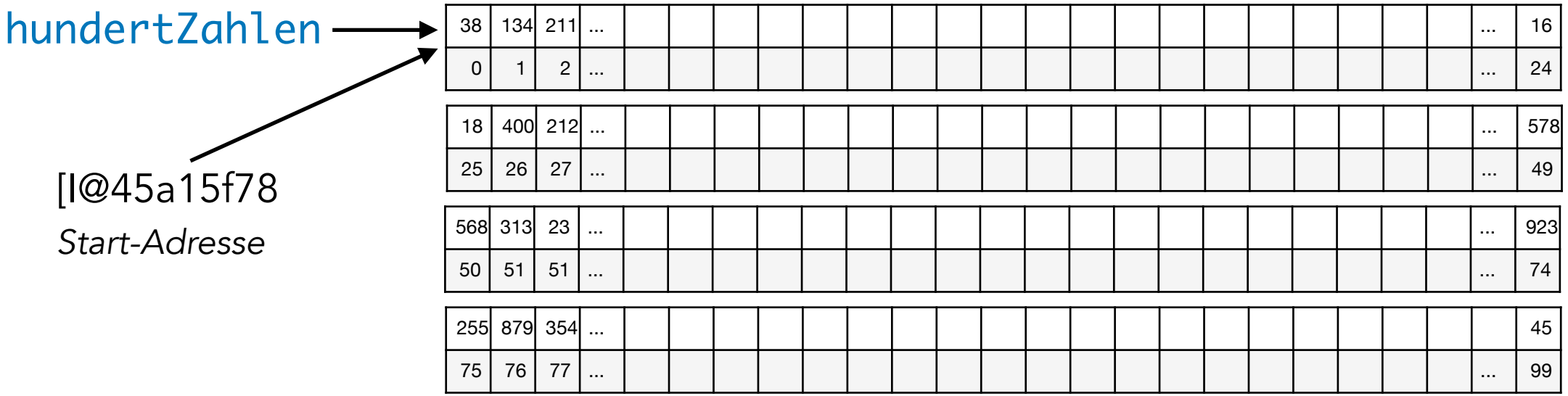
hundertZahlen →

38	134	211	...																		...	16
0	1	2	...																		...	24
18	400	212	...																		...	578
25	26	27	...																		...	49
568	313	23	...																		...	923
50	51	51	...																		...	74
255	879	354	...																			45
75	76	77	...																		...	99

```
public ArrayTest()
{
    hundertZahlen = new int[100];
    System.out.println(hundertZahlen);
}
```

→ [I@45a15f78

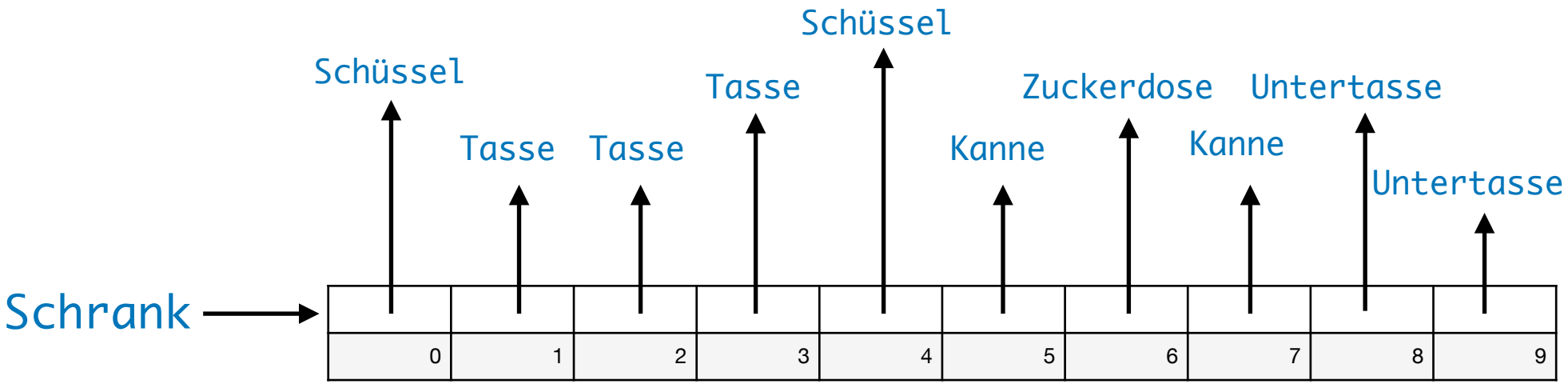
4.1.3 Was ist überhaupt ein Array?




```
public ArrayTest()
{
    hundertZahlen = new int[100];
    System.out.println(hundertZahlen);
}
```

→ [I@45a15f78

4.1.3 Was ist überhaupt ein Array?



Ein Array aus zehn Objekten der Klasse Geschirr.

Die Array-Elemente sind selbst Referenzen auf Objekte.

4.1.4 Aufgaben

Übung 4.1.4 #1

Schreiben Sie für die Klasse **`NZahlen`** eine sondierende Methode

```
public int gibSumme()
```

welche alle 100 Zufallszahlen des Arrays addiert und das Ergebnis dann zurückliefert.

4.1.4 Aufgaben

Übung 4.1.4 #1

Schreiben Sie für die Klasse `NZahlen` eine sondierende Methode

```
public int gibSumme()
```

welche alle 100 Zufallszahlen d

```
public int gibSumme()
// Übung 4.1.4 #1
{
    int sum = 0;
    for (int i=0; i < zahlen.length; i++)
        sum += zahlen[i];
    return sum;
}
```

4.1.4 Aufgaben

Übung 4.1.4 #2

Schreiben Sie für die Klasse `NZahlen` eine sondierende Methode

```
public int gibAnzahlVon(int z)
```

welche als Ergebnis zurückliefert, wie oft die Zahl `z` in dem Array vorkommt.

4.1.4 Aufgaben

Übung 4.1.4 #2

Schreiben Sie für die Klasse **`NZahlen`** eine sondierende Methode

```
public int gibAnzahlVon(int z)
```

welche als Ergebnis zurückliefert, wie oft

```
public int gibAnzahlVon(int z)
// Übung 4.1.4 #2
{
    int sum = 0;
    for (int i=0; i < zahlen.length; i++)
        if (zahlen[i] == z)
            sum++;
    return sum;
}
```

4.1.4 Aufgaben

Übung 4.1.4 #3

Schreiben Sie für die Klasse **`NZahlen`** eine sondierende Methode

```
public int gibAnzahlGerade(int z)
```

welche als Ergebnis zurückliefert, wie oft gerade Zahlen (2, 4, 6, ...) in dem Array vorkommen.

4.1.4 Aufgaben

Übung 4.1.4 #3

Schreiben Sie für die Klasse **`NZahlen`** eine sondierende Methode

```
public int gibAnzahlGerade(int z)
```

welche als Ergebnis zurückliefert, wie oft

```
public int gibAnzahlGerade(int z)
// Übung 4.1.4 #3
{
    int sum = 0;
    for (int i=0; i < zahlen.length; i++)
        if (zahlen[i] % 2 == 0)
            sum++;
    return sum;
}
```

4.1.4 Aufgaben

Übung 4.1.4 #4

Schreiben Sie für die Klasse **`NZahlen`** eine manipulierende Methode

```
public void tausch(int indexA, int indexB)
```

welche die beiden Arrayelemente an den Indices **`indexA`** und **`indexB`** miteinander vertauscht.

4.1.4 Aufgaben

Übung 4.1.4 #4

Schreiben Sie für die Klasse `NZahlen` eine manipulierende Methode

```
public void tausch(int indexA, int indexB)
```

welche die beiden Arrayele

```
public void tausch(int indexA, int indexB)
{
    int temp = zahlen[indexA];
    zahlen[indexA] = zahlen[indexB];
    zahlen[indexB] = temp;
}
```

4.1.4 Aufgaben

Übung 4.1.4 #5

Schreiben Sie für die Klasse **`NZahlen`** eine manipulierende Methode

```
public void minNachVorne()
```

welche die kleinste Zahl des Array mit der ersten Zahl (Index 0) des Arrays vertauscht. Verwenden Sie in dieser Methode die Methode **`tausch()`** aus #4 sowie die Methode **`mini()`** aus dem vorherigen Abschnitt.

4.1.4 Aufgaben

Übung 4.1.4 #5

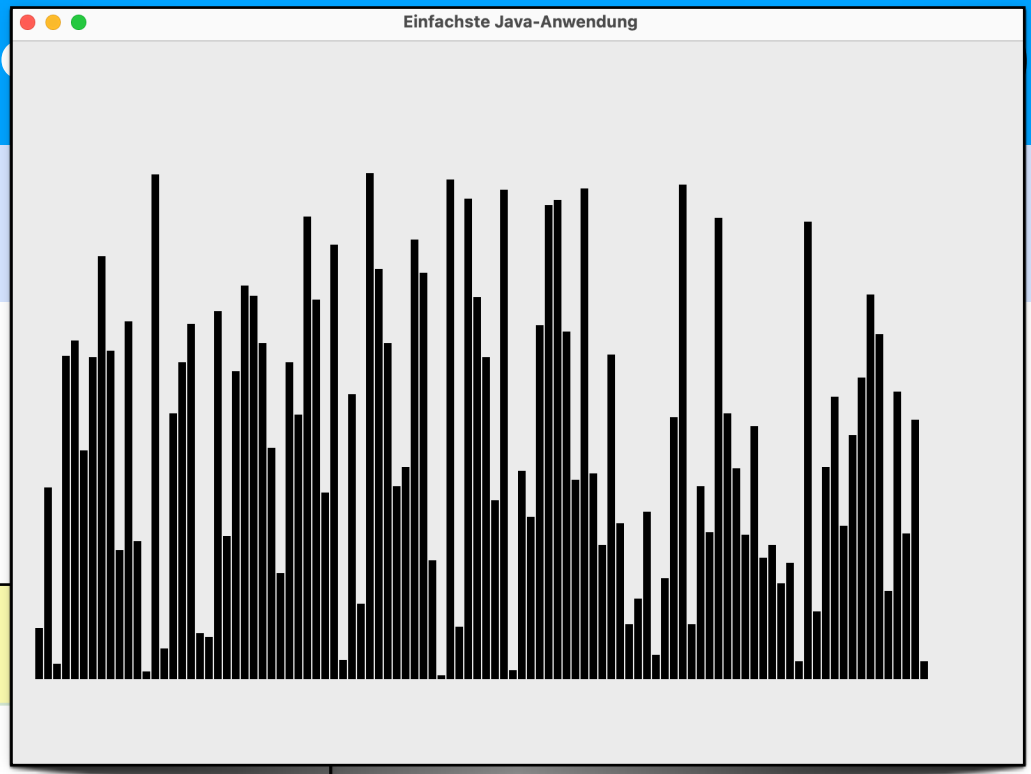
```
public int gibMiniPos()
{
    int mini = zahlen[0];
    int miniPos = 0;

    for (int i=1; i < zahlen.length; i++)
        if (zahlen[i] < mini)
        {
            mini = zahlen[i];
            miniPos = i;
        }

    return miniPos;
}
```

```
public void nachVorne()
{
    int mp = gibMiniPos();
    tausch(0,mp);
}
```

0) des Arrays vertauscht. Verwenden Sie in
hode **mini()** aus dem vorherigen Abschnitt.

4.1.4 Aufgaben

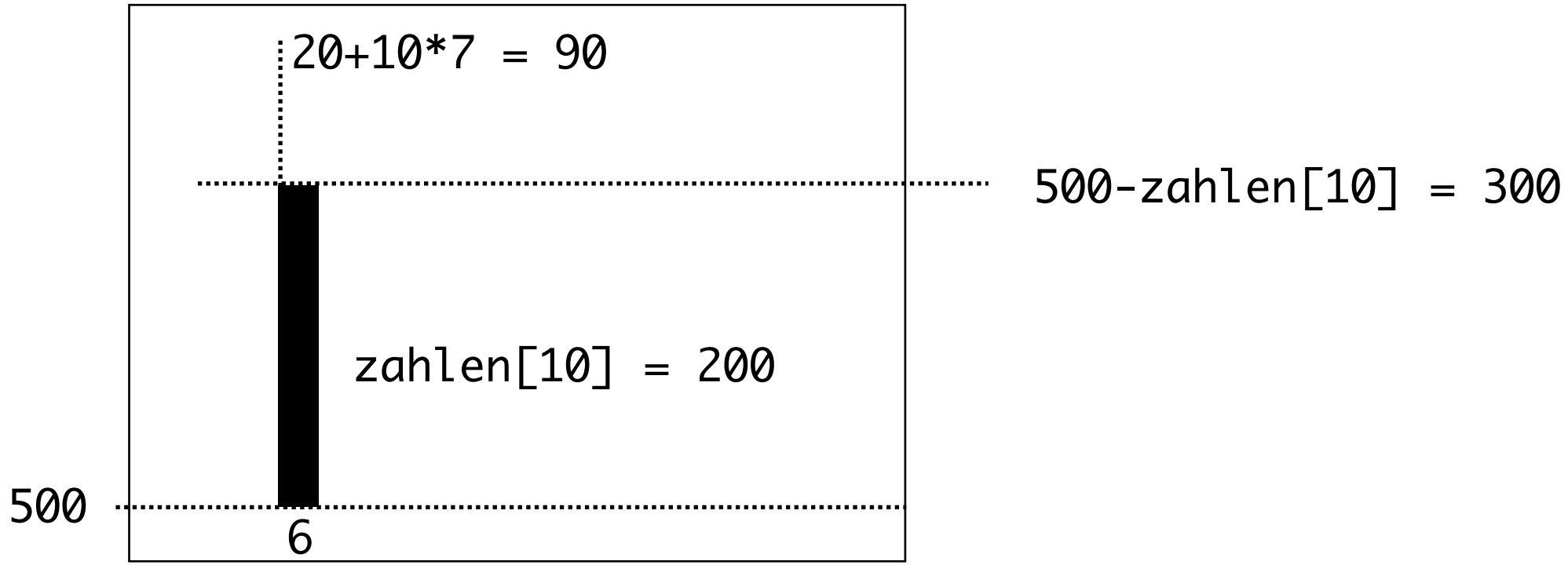
Übung 4.1.4 #6




```
protected void paintComponent(Graphics g)
{
    super.paintComponent(g);

    int[] zahlen = new int[100];
    for (int i=0; i<100; i++)
       zahlen[i] = (int) (Math.random()*400) + 1;

    for (int i=0; i<100; i++)
       g.fillRect(20+i*7,500-zahlen[i],6,zahlen[i]);
}
```

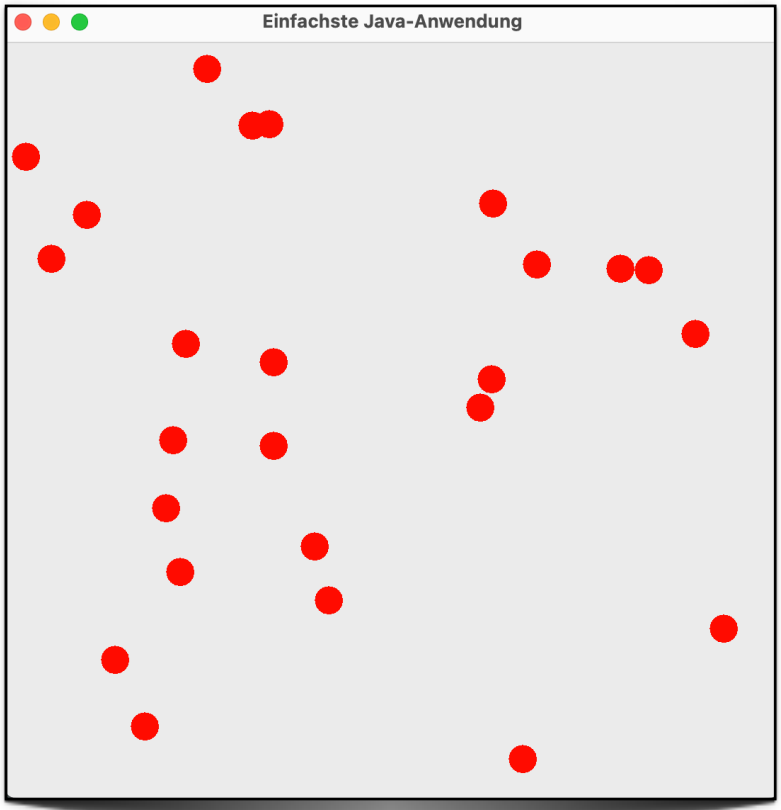
4.1.4 Aufgaben

```
for (int i=0; i<100; i++)
   g.fillRect(20+i*7,500-zahlen[i],6,zahlen[i]);
```



4.1.4 Aufgaben

```
protected void paintComponent(Graphics g)
{
    super.paintComponent(g);

    int[] zahlen = new int[100];
    for (int i=0; i<100; i++)
       zahlen[i] = (int) (Math.random()*500) + 1;

    g.setColor(Color.RED);
    for (int i=0; i<50; i+=2)
       g.fillOval(zahlen[i],zahlen[i+1],20,20);
}
```

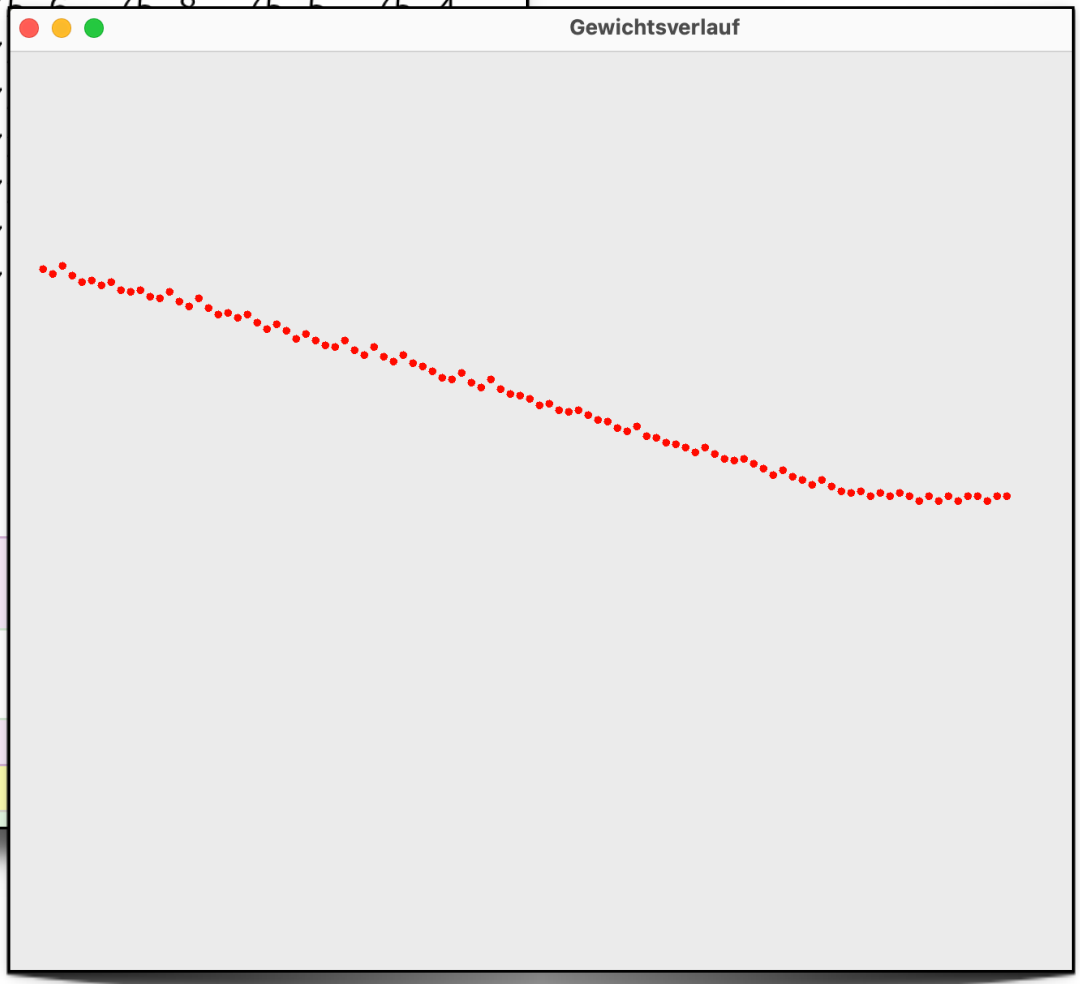


Übung 4.1.4 #7

Einführung in die Objektorientierte Programmierung (OOP)

Übung 4.1.4 #8

```
protected void paintComponent(Graphics g)
{
    super.paintComponent(g);

    double[] gewicht = {
            78.4, 78.3, 78.5, 78.2, 78.0, 78.1, 77.9, 78.0, 77.8, 77.7,
            77.8, 77.6, 77.5, 77.7, 77.4, 77.3, 77.5, 77.2, 77.0, 77.1,
            76.9, 77.0, 76.8, 76.6, 76.7, 76.5, 76.3, 76.4, 76.2, 76.1,
            76.0, 76.2, 75.9, 75.8, 76.0, 75.7, 7
            75.3, 75.1, 75.0, 75.2, 74.9, 74.8, 7
            74.4, 74.2, 74.3, 74.1, 74.0, 74.1, 7
            73.4, 73.6, 73.3, 73.2, 73.1, 73.0, 7
            72.6, 72.5, 72.6, 72.4, 72.3, 72.1, 7
            71.9, 71.7, 71.6, 71.5, 71.6, 71.4, 7
            71.3, 71.4, 71.3, 71.4, 71.3, 71.4, 7
        };

    int gew;
    g.setColor(Color.RED);

    for (int i=0; i<gewicht.length; i++)
    {
        gew = (int) ((gewicht[i]-60)*20);
        g.fillOval(20+i*6,500-gew,5,5);
    }
}
```



4.1.4 Aufgaben

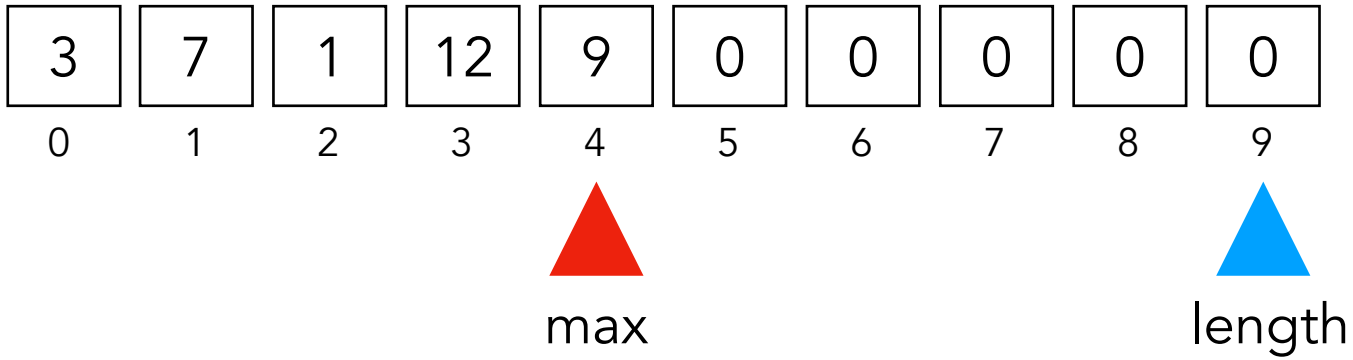
Übung 4.1.4 #9

```
public void append(int zahl)
// Übung 4.1.4 #9
{
    if (max < 100)
    {
        zahlen[max] = zahl;
        max++;
    }
}
```

4.1.4 Aufgaben

Übung 4.1.4 #9

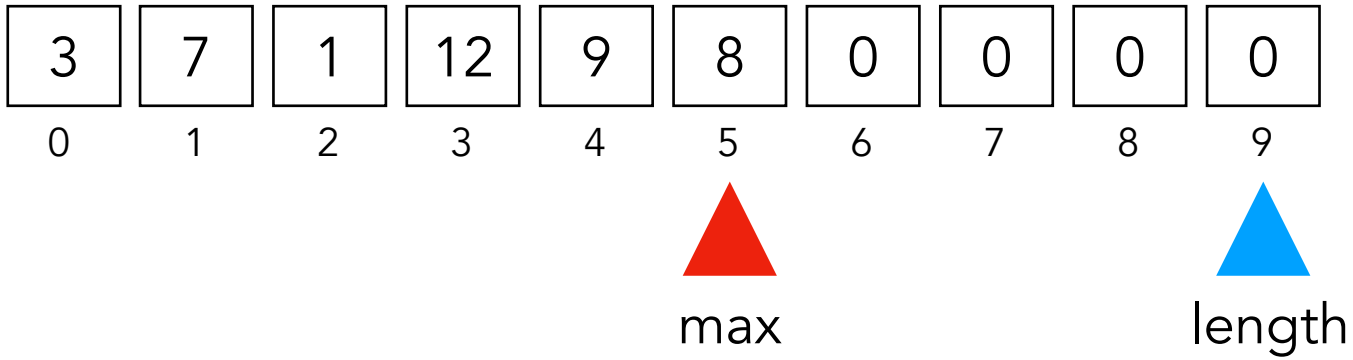
```
public void append(int zahl)
// Übung 4.1.4 #9
{
    if (max < 100)
    {
        zahlen[max] = zahl;
        max++;
    }
}
```



4.1.4 Aufgaben

Übung 4.1.4 #9

```
public void append(int zahl)
// Übung 4.1.4 #9
{
    if (max < 100)
    {
        zahlen[max] = zahl;
        max++;
    }
}
```



4.1.4 Aufgaben

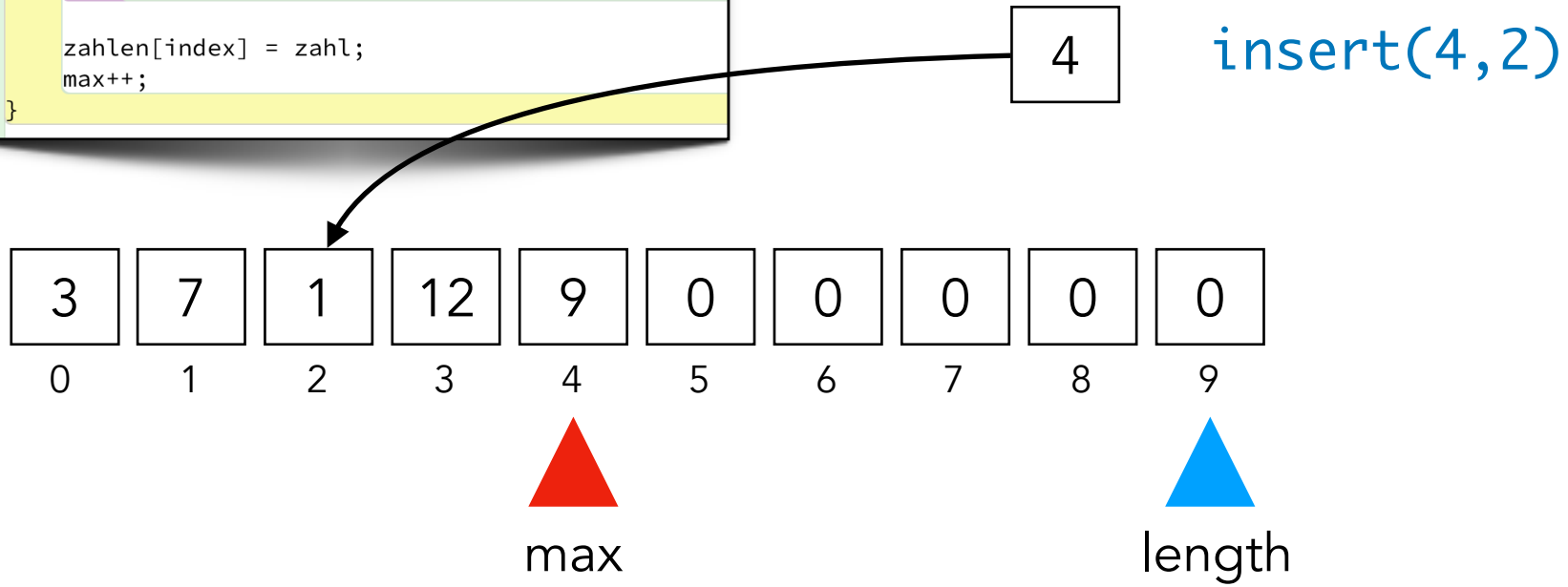
Übung 4.1.4 #10/11

```
public void insert(int zahl, int index)
{
    if (max >= 100 || index < 0 || index > max)
        return;

    for (int i = max; i > index; i--)
        zahlen[i] = zahlen[i - 1];

    zahlen[index] = zahl;
    max++;
}
```

4.1.4 Aufgaben

Übung 4.1.4 #10/11

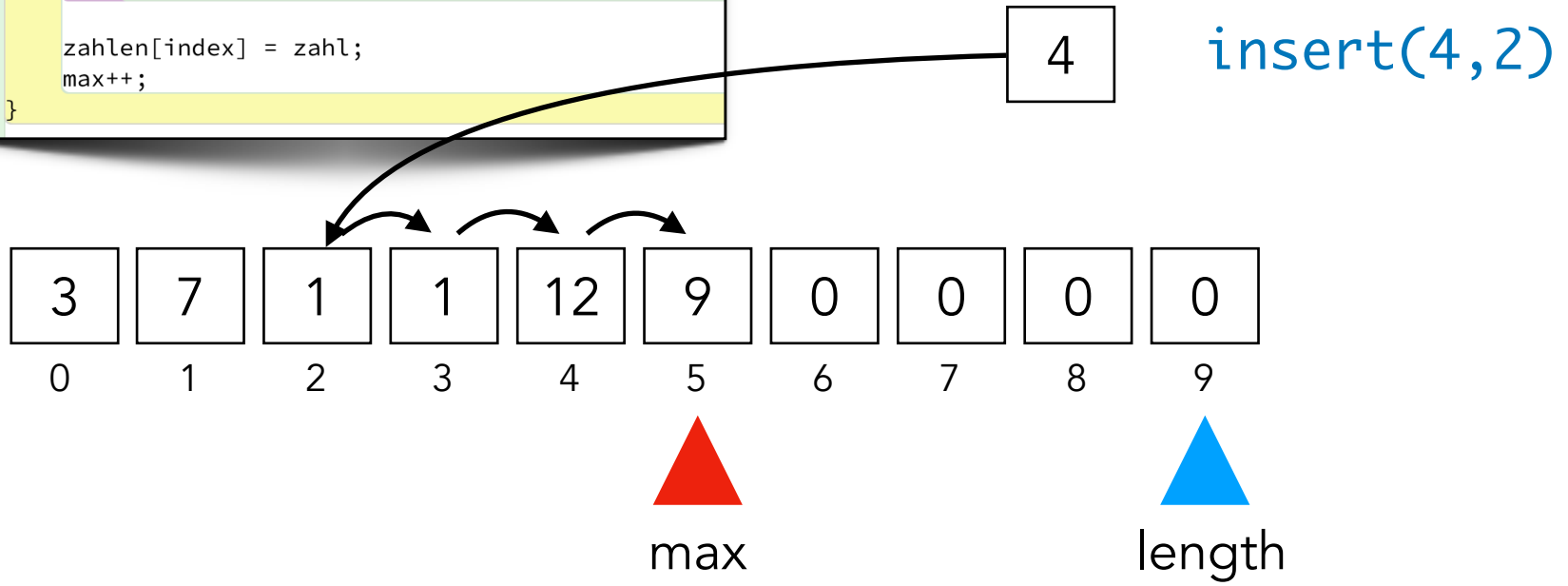
```
public void insert(int zahl, int index)
{
    if (max >= 100 || index < 0 || index > max)
        return;

    for (int i = max; i > index; i--)
        zahlen[i] = zahlen[i - 1];

    zahlen[index] = zahl;
    max++;
}
```



4.1.4 Aufgaben

Übung 4.1.4 #10/11

```
public void insert(int zahl, int index)
{
    if (max >= 100 || index < 0 || index > max)
        return;

    for (int i = max; i > index; i--)
        zahlen[i] = zahlen[i - 1];

    zahlen[index] = zahl;
    max++;
}
```

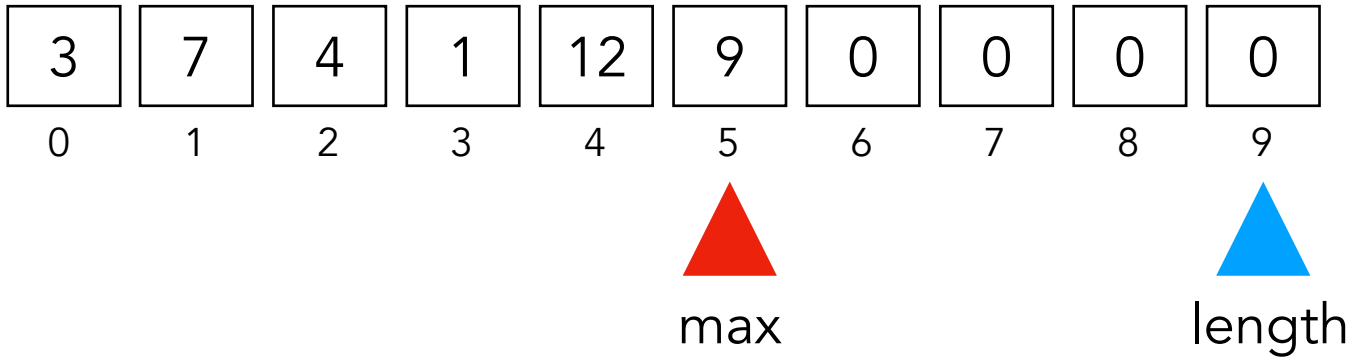


4.1.4 Aufgaben

Übung 4.1.4 #10/11

```
public void insert(int zahl, int index)
{
    if (max >= 100 || index < 0 || index > max)
        return;

    for (int i = max; i > index; i--)
        zahlen[i] = zahlen[i - 1];

    zahlen[index] = zahl;
    max++;
}
```

insert(4,2)



4.1.4 Aufgaben

Übung 4.1.4 #10/11

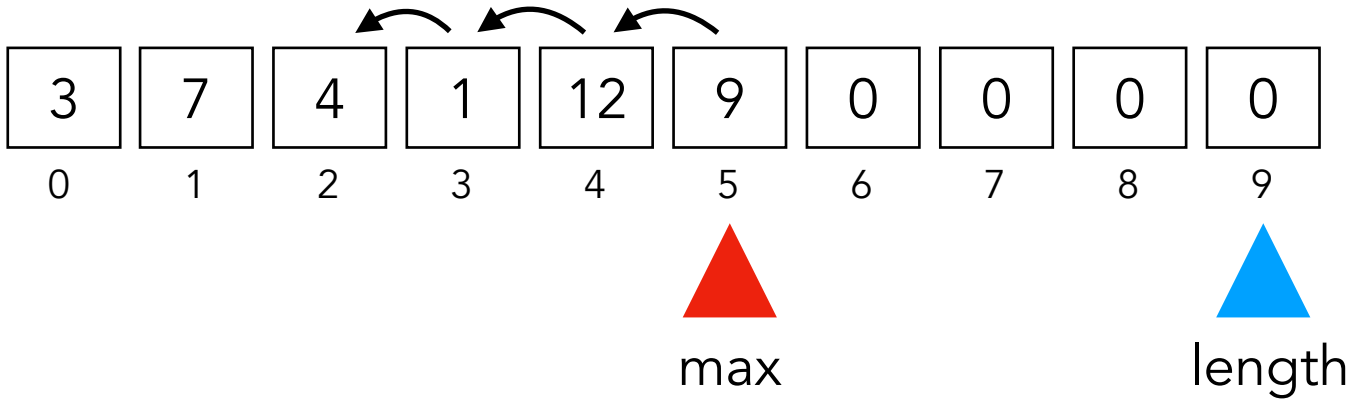
```
public void delete(int index)
{
    if (index < 0 || index >= max)
        return;

    for (int i = index; i < max - 1; i++)
        zahlen[i] = zahlen[i + 1];

    zahlen[max - 1] = 0;
    max--;
}
```

4.1.4 Aufgaben

Übung 4.1.4 #10/11

```
public void delete(int index)
{
    if (index < 0 || index >= max)
        return;

    for (int i = index; i < max - 1; i++)
        zahlen[i] = zahlen[i + 1];

    zahlen[max - 1] = 0;
    max--;
}
```

delete(2)

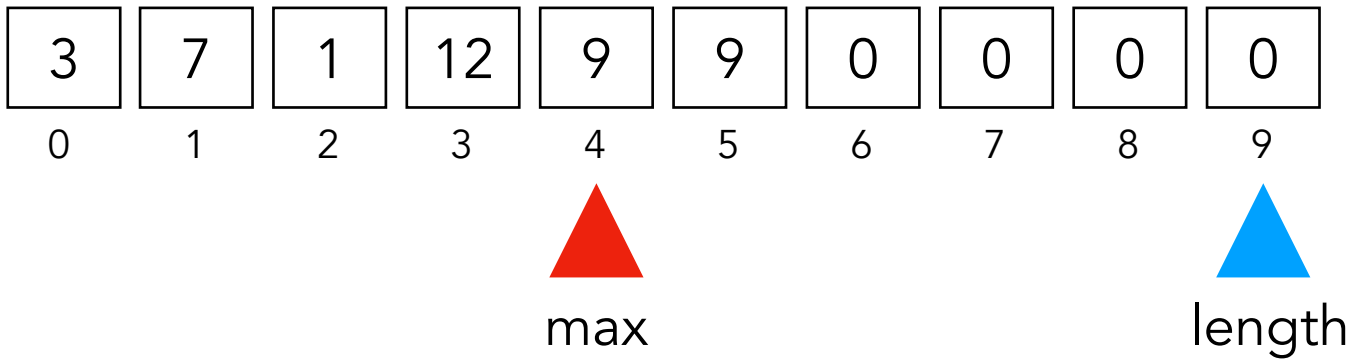


4.1.4 Aufgaben

Übung 4.1.4 #10/11

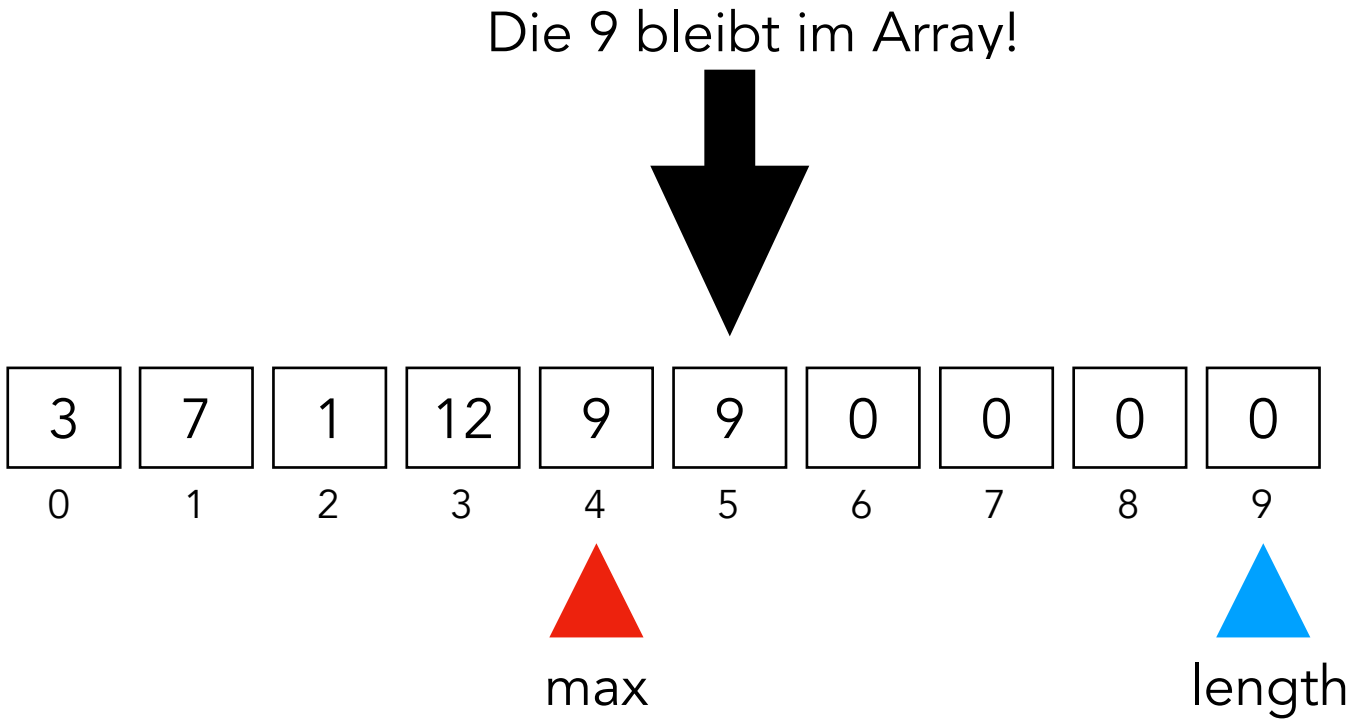
```
public void delete(int index)
{
    if (index < 0 || index >= max)
        return;

    for (int i = index; i < max - 1; i++)
        zahlen[i] = zahlen[i + 1];

    zahlen[max - 1] = 0;
    max--;
}
```

delete(2)



4.1.4 Aufgaben

Übung 4.1.4 #10/11



4.1.4 Aufgaben

```
public void ausgeben()
{
    for (int i=0; i < max; i++)
        System.out.println("Zahl " + i + " = " + zahlen[i]);
}
```

Die 9 bleibt im Array!