

### ÜBERSICHT

Ein **Array** ist eine **Datenstruktur**. In einer Datenstruktur werden auf eine bestimmte Art und Weise Daten gespeichert. Die Variablen, die du bisher kennengelernt hast, sind also auch Datenstrukturen. In einer Variablen wurde ein Wert von einem bestimmten Datentyp gespeichert. In einem Array dagegen werden **mehrere Daten vom gleichen Datentyp** gespeichert.

Man kann sich ein Array wie eine Liste vorstellen. In einem Array steht also eine Information nach der anderen. Deshalb nennt man Arrays auch Reihungen. Es gibt auch komplexere Arrays, die sogenannten zweidimensionalen Arrays. Diese kann man sich am ehesten als Tabellen vorstellen.



### LERNZIEL

In diesem Kapitel lernst du den Umgang mit ein- und zweidimensionalen Arrays. Dazu benutzt du die Kontrollstrukturen und Anweisungen, die du in den vorigen Kapiteln kennengelernt hast.



### THEORIE

Du weißt inzwischen schon, dass man in einem Array mehrere Daten vom gleichen Typ speichern kann. Man kann sich ein Array wie im folgenden **Beispiel** vorstellen:

x ist ein Array für Integerwerte und hat 3 Speicherzellen.

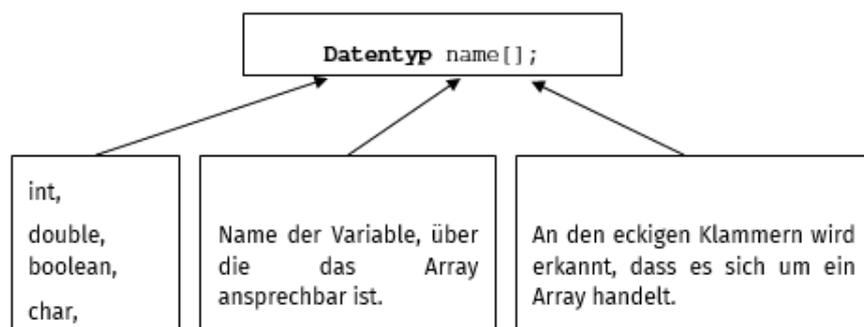
In der Speicherzelle 0 befindet sich eine 12, in der Speicherzelle 1 die Zahl -23 und in der Speicherzelle 2 eine 3.

x	
0	12
1	-23
2	3

Wie realisiert man nun Arrays in einem Programm?

### Deklaration eines Arrays

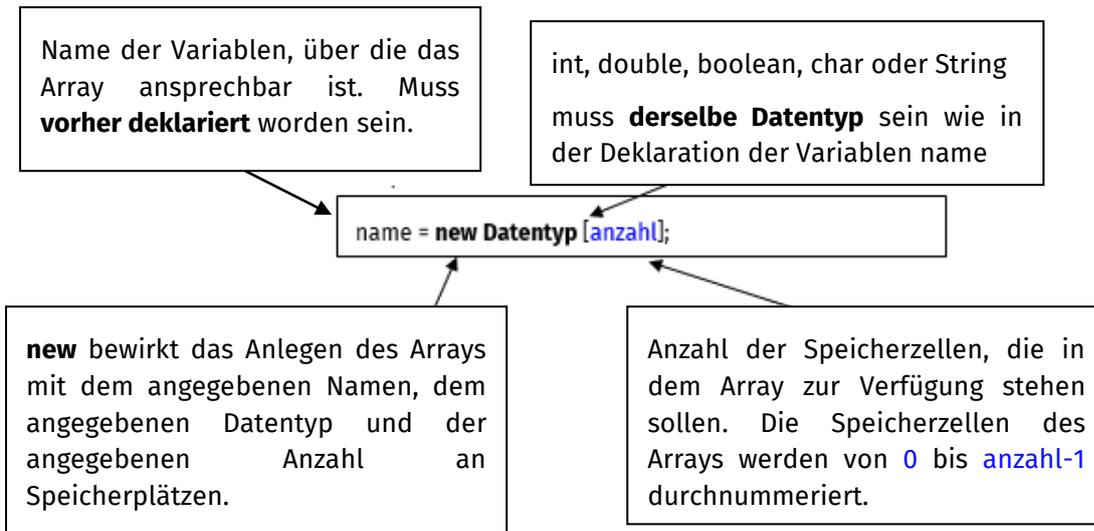
Zunächst muss das Array deklariert werden. In der Deklaration von Arrays wird mitgeteilt, welcher Variablen das Array zugewiesen werden soll und von welchem Datentyp die Werte des Arrays sind.



### Anlegen eines Arrays

Nach der Deklaration muss das Array angelegt werden. Das bedeutet, dass Speicherplatz für das Array reserviert werden muss. Damit das passieren kann, muss bekannt sein, wie viele Daten in

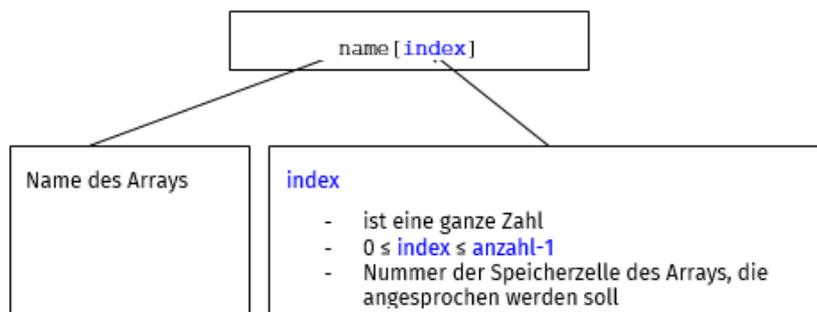
dem Array abgelegt werden sollen. Für jeden Wert, der gespeichert werden soll, wird eine Speicherzelle angelegt. Die Speicherzellen des Arrays werden von 0 an durchnummeriert.



Nachdem das passiert ist, existiert das Array und ist über den Namen, den man ihm zugewiesen hat, ansprechbar. Jede Speicherzelle des Arrays ist bereits mit einem Standardwert belegt worden. Das nennt man **Initialisierung**. Ein Array mit dem Datentyp Integer wird mit dem Wert 0 initialisiert.

### Speichern und Auslesen bei Arrays

Jede einzelne Speicherzelle kann mit dem Namen des Arrays und der Nummer der Speicherzelle (hier index) angesprochen werden. Die Werte für index müssen zwischen 0 und `anzahl-1` liegen. `anzahl` ist die Zahl, die beim Anlegen des Arrays für die Anzahl der Speicherzellen angegeben wurde.



`name[index]` kann jetzt wie eine Variable verwendet werden:

Wert wird im Array name an die Speicherzelle mit der nummer index gespeichert.	<code>name[index] = Wert;</code>
Der Wert, der im Array name an der Speicherzelle mit der Nummer index gespeichert ist, wird mit 2 addiert und in der Variablen x gespeichert.	<code>x = name[index]+2;</code>

## Beispielprogramm zu dem Array:

x	
0	12
1	-23
2	3

Deklaration von x als Array mit integer-Werten

Anlegen des Arrays x als Array mit 3 integer-Werten.

### Programm Reihung

```
public class Reihung {  
    public static void main (String [] arguments){  
  
        int x[];  
  
        x = new int [3];  
  
        x[0] = 12;  
        x[1] = -23;  
        x[2] = 3;  
  
    }  
}
```

Belegung des Arrays mit integer-Werten. Die einzelnen Speicherzellen des Arrays können über die Indizes 0, 1 und 2 angesprochen werden.

## SICHERUNGSPHASE



### Aufgabe 1 – Arrays deklarieren, anlegen und beschreiben

Programmiere die folgenden Arrays!



x	
0	12
1	11
2	-1

y	
0	1.3
1	1.4
2	-12.3
3	2.23

z	
0	'a'
1	'#'

v	
0	"Ar"
1	"ra"
2	"y"

Lasse in deinem Programm folgende Werte am Bildschirm ausgeben:

- Inhalt der Speicherzelle 0 von Array x
- Inhalt der Speicherzelle 3 von Array y
- Inhalt der Speicherzelle 1 von Array z
- Inhalt der Speicherzelle 0 von Array v



### Aufgabe 2 – Arrays deklarieren, anlegen und beschreiben

Schreibe ein Programm, in dem du je ein Array für zwei unterschiedliche Datentypen deklarierst, anlegst und mit Werten füllst.

Die beiden Arrays sollen eine unterschiedliche Anzahl an Speicherzellen haben.

Der Inhalt jedes Arrays soll am Bildschirm ausgegeben werden. Formatiere deine Ausgabe wie in dem folgenden **Beispiel**:

doubleArray ist ein array, das 3 double-Werte enthält.

```
| doubleArray
-----
0 | 3.1
1 | 12.0
2 | -3.4
```



### Aufgabe 3 – Schleifen und Arrays

Schreibe ein Programm, in dem alle Zahlen von 1 bis 100 in ein geeignetes Array geschrieben werden. Verwende eine Schleife.



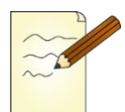
### Aufgabe 4 – Notenverwaltung

Schreibe ein Programm zur Verwaltung der Noten einer Schulklasse.

Das Programm hat vier Funktionen.

1. Noten der Klausur einlesen lassen, wobei die Klassengröße interaktiv in Erfahrung gebracht wird
2. Notenliste der Klausur ausgeben lassen
3. Durchschnittsnote der Klasse ermitteln lassen
4. Programm beenden

Die Funktionen werden über ein immer wiederkehrendes Menü angesteuert (Tipp: switch und while).



### Aufgabe 5 – Standardbelegung der Arrays

Im Theorieteil hast du gelernt, dass die Arrays nach dem Anlegen mit

```
name = new Datentyp [anzahl];
```

schon mit einem Standardwert initialisiert sind. Beim Datentyp int ist das die 0.

Finde heraus, welcher Wert je nach Datentyp für die Standardbelegung gewählt wird! Trage deine Ergebnisse hier ein:

int: 0

double

---

char

---

String

---



### THEORIE

Die Arrays, die du kennengelernt hast, sind **eindimensionale Arrays**. Jetzt wirst du lernen, **zweidimensionale Arrays** einzusetzen. Zweidimensionale Arrays sind **Tabellen** zum Speichern von Werten des gleichen Datentyps. Schaue dir zunächst das Beispiel an:

- y ist ein zweidimensionales Array für Integerwerte.
- y hat 3 Zeilen und 4 Spalten.
- Die Speicherzelle in der Zeile 1 und der Spalte 2 enthält eine 12.

	y			
	0	1	2	3
0	1	-9	34	-8
1	6	-6	12	0
2	9	89	0	0

Wie realisiert man zweidimensionale Arrays in einem Programm?

### Deklaration eines zweidimensionalen Arrays

```
Datentyp name [] [];
```

An den **zwei** eckigen Klammern wird erkannt, dass es sich um ein zweidimensionales Array handelt.

## Anlegen eines zweidimensionalen Arrays

```
name = new Datentyp [zeilenanzahl] [spaltenanzahl];
```

Anzahl der Zeilen, die in dem Array zur Verfügung stehen sollen. Die Zeilen des Arrays werden von 0 bis `zeilenanzahl-1` durchnummeriert.

Anzahl der Spalten, die in dem Array zur Verfügung stehen sollen. Die Spalten des Arrays werden von 0 bis `spaltenanzahl-1` durchnummeriert.

## Speichern und Auslesen bei zweidimensionalen Arrays

Jede einzelne Speicherzelle kann mit dem Namen des Arrays, der Nummer seiner Spalte und der Nummer seiner Zeile angesprochen werden.

```
name [zeile] [spalte]
```

### zeile

- ist eine ganze Zahl
- $0 \leq \text{zeile} \leq \text{spaltenanzahl}-1$
- Nummer der Zeile des Arrays, die angesprochen werden soll.

### spalte

- ist eine ganze Zahl
- $0 \leq \text{spalte} \leq \text{spaltenanzahl}-1$
- Nummer der Spalte des Arrays, die angesprochen werden soll.

## Beispiel zu dem oben angegebenen Array

	y			
	0	1	2	3
0	1	-9	34	-8
1	6	-6	12	0
2	9	89	0	0

## Programm ZweidimensionalesArray

```
public class ZweidimensionalesArray{  
    public static void main (String [] arguments){  
  
        //Deklaration von y als zweidimensionales  
        //int-Array:  
        int y[][];  
  
        //Anlegen von y mit 3 Zeilen und 4 Spalten  
        y = new int [3][4];  
  
        // Speichern der Werte in Zeile 0  
        y[0][0] = 1;  
        y[0][1] = -9;  
        y[0][2] = 34;  
        y[0][3] = -8;  
  
        // Speichern der Werte in Zeile 1  
        y[1][0] = 6;  
        y[1][1] = -6;  
        y[1][2] = 12;  
        y[1][3] = 0;  
  
        // Speichern der Werte in Zeile 2  
        y[2][0] = 9;  
        y[2][1] = 89;  
        y[2][2] = 0;  
        y[2][3] = 0;  
    }  
}
```



---

### SICHERUNGSPHASE

#### Aufgabe 6 – Stundenplanmanager



Schreibe einen Stundenplanmanager! Bearbeite die folgenden Schritte alle im gleichen Programmquellcode der Reihe nach. Gehe immer erst zum nächsten Schritt über, wenn der aktuelle gemeistert ist.

1. Lege ein zweidimensionales Array an, das sich dazu eignet, einen Stundenplan zu speichern (z. B. 5 Spalten, 6 Zeilen).
2. Programmiere eine entsprechende Ausgabe für den Stundenplan (also für das in 1 angelegte Array).

**TIPP:** \t innerhalb eines Strings bewirkt bei der Ausgabe eine Einrückung wie bei der Nutzung der Tabulatortaste.

Beispiel:

Aktueller Stundenplan				
Mo	Di	Mi	Do	Fr
-----				
M	If	M	E	Ph
M	Sp	M	E	Ph
E	D	Sp	M	E
E	D	Sp	F	E
Ph	F	K	If	K
	F		If	K

3. Erweitere dein Programm um die Möglichkeit Eintragungen in den Stundenplan vornehmen zu lassen.

Beispiel:

```

** Stunden eintragen **
Tage 1=Mo  2=Di  3=Mi  4=Do oder 5=Fr
Stunden 1 - 6
Welche Stunde soll eingetragen werden?
Tag: 2
```

4. Erweitere dein Programm so, dass man die Möglichkeit hat, in einem Menü aus den folgenden drei Punkten auszuwählen:
  - a. Stundenplan ausgeben
  - b. Stunde eintragen
  - c. Programm beenden

Nach Ausführung des entsprechenden Menüpunktes wird zum Menü zurückgekehrt.

**TIPP:** switch und while!

Beispiel:

```
***** Stundenplanmanager *****
(1) Stundenplan ausgeben
(2) Stunde eintragen
(3) Programm beenden
Bitte waehle einen Menuepunkt: 2

**Stunden eintragen **
Tage 1=Mo 2=Di 3=Mi 4=Do oder 5=Fr
Stunden 1 - 6
Welche Stunde soll eingetragen werden?
Tag: 2
Stunde: 1
Fach: If
```



#### LERNKONTROLLE

Um den Kapiteltest zu bestehen, musst du den Umgang mit ein- und zweidimensionalen Arrays beherrschen.



#### ADDITUM – NOTENVERWALTUNG PLUS

Erweitere dein Programm aus Aufgabe 4 so, dass die Noten mehrerer Klausuren eingelesen werden können.

Die Anzahl der Notenlisten, die eingegeben werden sollen, wird vorher interaktiv erfragt. Es steht die Möglichkeit zur Verfügung, für jeden Schüler eine Durchschnittsnote seiner bisher erzielten Noten zu ermitteln.



#### ADDITUM – MATRIZEN

Man kann ein zweidimensionales Array auch als Matrix auffassen. Wenn du nicht weißt, was eine Matrix ist, dann schaue im Internet nach.

1. Finde im Internet heraus, wie zwei Matrizen multipliziert werden.

---

---

2. Multipliziere!

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 0 \end{pmatrix} = \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}$$

3. Schreibe ein Programm, das

- zwei Matrizen als zweidimensionale Arrays speichert,
- multipliziert und
- das Ergebnis in einem weiteren zweidimensionalen Array speichert.

4. Überprüfe deine Lösung aus 2. mit dem Programm.