Einführung in die Informatik

Arrays & Matrices

Arrays & Matrizen

Wolfram Burgard
Cyrill Stachniss

Arrays

- Der **Array** ist ebenso wie die primitiven Datentypen ein eingebauter Datentyp für Kollektionen.
- **Arrays** haben verschiedene Gemeinsamkeiten mit **ArrayListen**:
  - Er enthält mehrere Elemente,
  - auf jedes Element kann durch einen Index zugegriffen werden,
  - die erste Position ist `0`,
  - Arrays werden durch die `new`-Operation erzeugt,
  - ein Array ist ein Objekt und
  - für Arrays werden Referenzvariablen verwendet.

Unterschiede Array und ArrayList

- **Für Arrays gibt keine Klasse**.
- **Arrays sind** — ebenso wie primitive Datentypen — **in die Sprache eingebaut**.
- **Es gibt keine Methoden für Arrays**.
- Jedem Array ist eine Variable **`length`** zugeordnet, welche als Wert die **Anzahl der Felder des Arrays** enthält.
- **Arrays können** — im Gegensatz zu ArrayListen — **primitive Datentypen** wie z.B. `int` **enthalten**.
- **Arrays haben eine feste Größe**. Ihre Länge wächst nicht automatisch wie die von ArrayListen.
- **Arrays sind**, da sie von der Sprache direkt zur Verfügung gestellt werden, **effizienter als ArrayListen**.

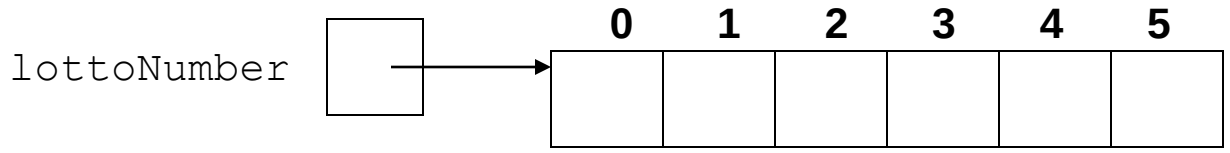
Deklaration und Erzeugung von Arrays

- Bei der **Deklaration einer Referenzvariable für ein Array**, geben wir ebenfalls den Typ der Elemente an:

```
int[] lottoNumber;
String[] winner;
Employee[] emp;
```

- Um ein **Array zu erzeugen**, verwenden wir ebenfalls den **`new`-Operator**. Dabei geben wir den Typ der Elemente und ihre Anzahl an:

```
lottoNumber = new int[6];
winner = new String[100];
emp = new Employee[1000];
```



Zugriff auf die Elemente eines Arrays

Um das Element an Position `k` eines Arrays auf einen bestimmten Wert zu setzen, verwenden wir eine Wertzuweisung der Form:

```
lottoNumber[k] =value;
```

Um auf ein Element eines Arrays zuzugreifen verwendet man die eckigen Klammern:

```
n = lottoNumber[3];
```

Zugriffe auf die Elemente eines Arrays lassen sich natürlich auch **schachteln**:

```
s = winner[z[3]];
```

Mehrdimensionale Arrays

- Arrays können nicht nur eindimensional, sondern auch **mehrdimensional** sein.
- Für ein zweidimensionales Feld von `double`-Werten wird beispielsweise folgende Deklaration verwendet.

```
public double[][] value;
```

- Das zweidimensionale Feld wird dann mit dem Statement

```
value = new double[m][n];
```

  erzeugt.

- Der Zugriff auf die Elemente eines zweidimensionalen Arrays wird folgendermaßen durchgeführt:

```
value[i][j] = 3.0;
```

Matrizen: Anwendung zweidimensionaler Arrays

Eine Matrix ist die Anordnung von $m*n$ Werten in einer Tabelle von m Zeilen und n Spalten. Dabei heißt eine Matrix quadratisch, falls $m==n$.

Eine $m \times n$ Matrix hat die Form:

$$\begin{pmatrix} a[0][0] & \cdots & a[0][n-1] \\ \vdots & \ddots & \vdots \\ a[m-1][0] & \cdots & a[m-1][n-1] \end{pmatrix}$$

Eine typische Matrizenoperation ist das Transponieren, d.h. das Vertauschen der Zeilen und Spalten einer Matrix.

Das Element an Position `a[i][j]` der Transponierten entspricht dem Element `a[j][i]` der Originalmatrix.

Eine einfache Klasse für Matrizen

```
class Matrix {
    public Matrix(int m, int n) {
        this.value = new double[m][n];
        this.m = m;
        this.n = n;
    }
    public Matrix transposed(){
        Matrix mat = new Matrix(this.n, this.m);
        for (int i = 0; i < this.m; i++)
            for (int j = 0; j < this.n; j++)
                mat.value[j][i] = this.value[i][j];
            return mat;
    }
    public void print(){
        for (int i = 0; i < this.m; i++){
            for (int j = 0; j < this.n; j++)
                System.out.print(this.value[i][j] + " ");
            System.out.println();
        }
    }
    public double[][] value;
    public int m;
    public int n;
}
```

Eine kleine Beispielanwendung

```
class UseMatrix {
    public static void main(String [] args) {
        Matrix m = new Matrix(2,2);
        m.value[0][0] = m.value[1][0] = m.value[1][1] = 0.0;
        m.value[0][1] = 1.0;

        m.print();
        System.out.println();
        m.transposed().print();
    }
}
```

Dieses Programm erzeugt die Ausgabe

```
0.0 1.0
0.0 0.0

0.0 0.0
1.0 0.0
```

Zusammenfassung

- Arrays enthalten Kollektionen ähnlich wie **`ArrayList`-Objekte.**
- Die Objekte habe in dem Array eine bestimmte Ordnung.
- Der **Index der Elemente in ArrayListen und Arrays beginnt bei** `0`.
- Mit Arrays lassen sich leicht mehrdimensionale Datenstrukturen erzeugen wie beispielsweise Matrizen