Einführung in die Programmierung für NF MI

Übung 04

Inhalt

- Arrays
- Einführung in Objekte

Arrays

- Arrays repräsentieren Reihungen von Objekten, z.B. Variablen
- Man kann auf die Elemente eines Arrays einzeln zugreifen
- Die Elemente werden durch ihren Index adressiert
- Alle Elemente in einem Array müssen vom selben Datentyp sein

Arrays

- Beispiel:

Array: ['K', 'U', 'M', 'M']

Index: 0 1 2 3

Array: [11, 17, 91, 33, 84, 4]

Index: 0 1 2 3 4 5

Arrays in Java

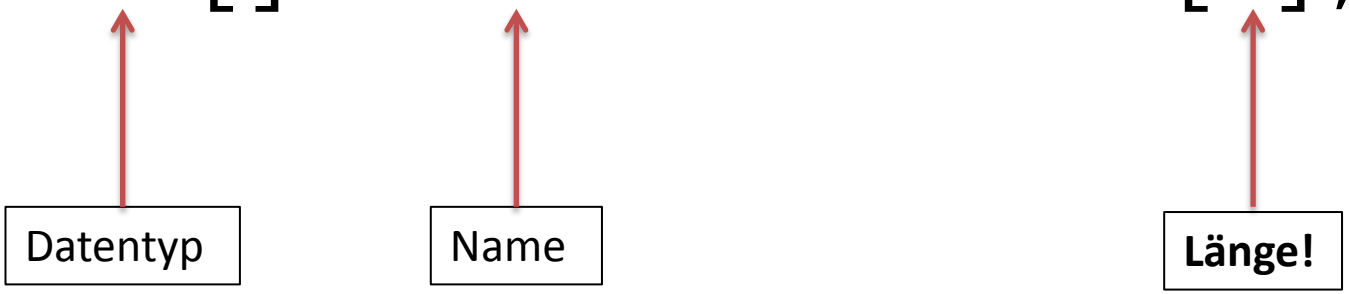
- Deklarieren von Arrays in Java durch `type[] name;`

- Beispiele:
  - `int[] zahlen;`
  - `String[] worte;`
  - `double[] coole_zahlen;`

Arrays in Java

- Deklarieren alleine reicht aber nicht aus
- Arrays sind Objekte und müssen vor ihrer Verwendung erzeugt werden

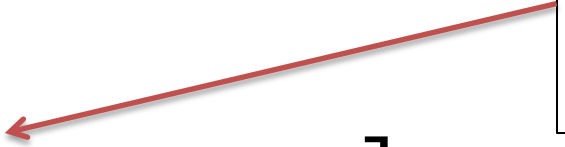
```
int[] zahlen = new int[5];
```



Arrays in Java

- `int[] zahlen = new int[5];`

- erzeugt ein „leeres“ Array der Länge 5, in das nur Integer-Werte geschrieben werden dürfen
- Vorstellbar als



Da es keinen leeren Integerwert gibt, werden die Felder mit „0“ gefüllt

```
zahlen: [   ,   ,   ,   ,   ]
Index:      0   1   2   3   4
```

Arrays in Java

- Inhalte können auch gleich bei der Erzeugung des Arrays eingetragen werden

```
int[] zahlen = new int[]{7,3,98,14,1};
```

- Die Länge muss dabei nicht mehr explizit angegeben werden

Arrays in Java

- Länge und Typ eines Arrays können (in Java) nicht verändert werden!

- Die bei Erzeugung des Arrays angegebenen Werte gelten also für die gesamte Programmlänge

Arrays in Java

- Der Zugriff auf die Elemente im Array erfolgt ebenfalls mit eckigen Klammern []

```
arrayname[index];
```

- Beispiel:

zahlen: `[7,3,98,14,1]`

```
zahlen[2] = 17;
```

zahlen: `[7,3,17,14,1]`

Arrays in Java

- Es gibt einige Funktionen und Attribute, die auf Arrays angewendet werden können
- Das wichtigste Attribut ist dabei **`length`**
- `arrayname.length;` gibt die Länge eines Arrays zurück
- `System.out.println(zahlen.length);`

  gibt 5 auf der Konsole aus

Arrays in Java

- Wie gebe ich ein Array auf der Konsole aus?

- Idee: `System.out.println(zahlen);`
  - Gibt nur die Speicheradresse aus, an der das Array gespeichert ist

- Lösung: Das Array muss Element für Element durchlaufen werden, um diese auszugeben

Arrays in Java

- Wie kann ich die einzelnen Elemente eines Arrays durchlaufen?

- Mit einer for-Schleife!

- Die Indizes des Arrays sind Integer-Zahlen, das Array fängt immer bei Index 0 an und die Länge kann ausgelesen werden

Arrays in Java

```
int[] zahlen = new int[5];

for (int i = 0; i < zahlen.length; i++) {
   System.out.println(zahlen[i]);
}
```

- In diesem Beispiel werden also

  fünf Nuller ausgegeben

Arrays in Java

- Beispiel:

```
int[] zahlen = new int[5];

for (int i = 0; i < zahlen.length; i++) {
   zahlen[i] = i;
}

for (int i = 0; i < zahlen.length; i++) {
   zahlen[i] = zahlen[i]*2;
}
```

Arrays in Java

- Funktion um ein Array zu durchsuchen:

```
public boolean arraysearch (int[] array,int e){

   for (int i = 0; i < array.length; i++) {
      if(array[i] == e){
         return true;
      }
   }
   return false;
}
```

Arrays in Java

- Da Arrays selbst auch Objekte sind, kann ein Array auch aus Arrays bestehen

- Diese Kombination wird als multidimensionales Array bezeichnet

Arrays in Java

- Anlegen eines multidimensionalen Arrays:

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

- Legt ein Array der Länge 5 an
- An jeder Stelle dieses Arrays steht ein Integer-Array der Länge 3

Arrays in Java

- Durchlaufen eines multidimensionalen Arrays mit einer doppelten for-Schleife:

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

for (int i = 0; i < m_array.length; i++) {
   for (int j = 0; j < m_array[i].length; j++) {
      System.out.println(m_array[i][j]);
   }
}
```

Fragen zum Übungsblatt?