

Deklaration von Arrays

- mit Arrays kann man beliebig viele Variablen von einem Datentyp in einer Zeile deklarieren
- Voraussetzung ist jedoch, dass alle Variablen den gleichen Datentyp besitzen
- Arrays besitzen immer eine vorbestimmte Länge
 - => man muss bei der Deklaration schon festlegen, wie viele sogenannte Elemente das Array letztendlich besitzen soll
- jede Variable eines Arrays bezeichnet man als ein Element des Arrays
- es gibt zwei Möglichkeiten, ein Array zu deklarieren:
 - entweder man deklariert es
 - oder man initialisiert es direkt bei der Deklaration mit Werten

```
//Deklaration von Array (Elemente besitzen noch keine Werte)  
//=> Das Array ist also noch "leer"  
//=> Array besitzt insgesamt 50 Elemente
```

```
String[] nameArr = new String[50];
```

```
//Deklaration von Array, welches direkt initialisiert wird  
//=> Array besitzt insgesamt 6 Elemente
```

```
int[] hausNrArr = {1, 2, 3, 4, 5, 6};
```

Auf die einzelnen Array Elemente zugreifen

- man kann jedem Element eines Arrays auch während des Programmverlaufs einen Wert zuweisen
- hierfür verwendet man den sogenannten Subskript Operator
- Zahlen zwischen den eckigen Klammern (Subskript Operator) beziehen sich immer auf den Index des Arrays
- Index startet immer bei 0 und der größte Index ist die Anzahl der Elemente des Arrays minus 1

```
String[] nameArr = new String[50];
```

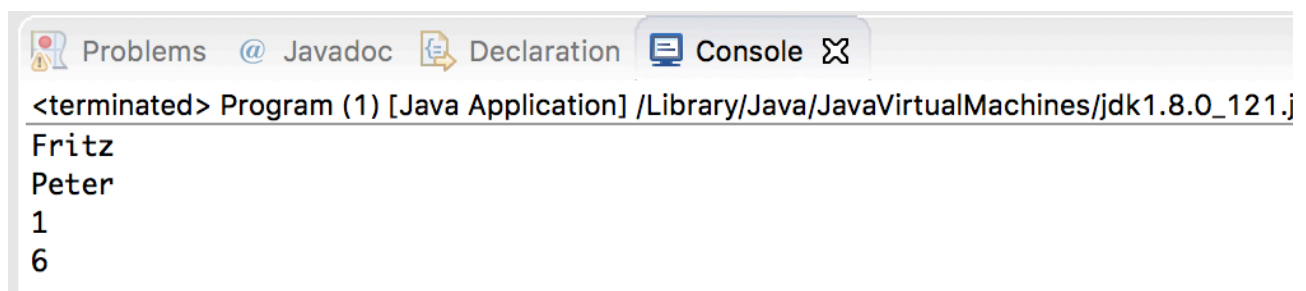
```
//Erstes Element des Arrays ansprechen  
nameArr[0] = "Fritz";
```

```
//Letztes Element des Arrays ansprechen  
nameArr[49] = "Peter";
```

```
//Ausgabe vom ersten und letzten Element des Arrays  
System.out.println(nameArr[0]);  
System.out.println(nameArr[49]);
```

```
int[] hausNrArr = {1, 2, 3, 4, 5, 6};
```

```
//Ausgabe vom ersten und letzten Element des bereits initialisierten Arrays  
System.out.println(hausNrArr[0]);  
System.out.println(hausNrArr[5]);
```



```
<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.j  
Fritz  
Peter  
1  
6
```

Arrays und Schleifen geschickt kombinieren

- Arrays können mithilfe einer for Schleife sehr schnell gefüllt werden
- hierzu ein paar Beispiele:

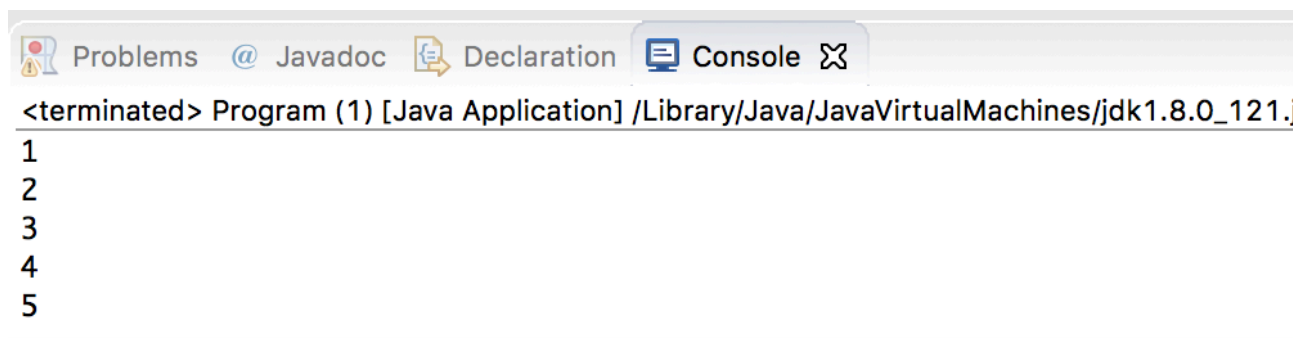
```
boolean[] testArray1 = new boolean[50];
```

```
for(int i = 0; i < 50; i++) {  
    testArray1[i] = true;  
}
```

```
int[] testArray2 = new int[5];
```

```
for(int i = 0; i < 5; i++) {  
    testArray2[i] = i + 1;  
}
```

```
for(int i = 0; i < 5; i++) {  
    System.out.println(testArray2[i]);  
}
```



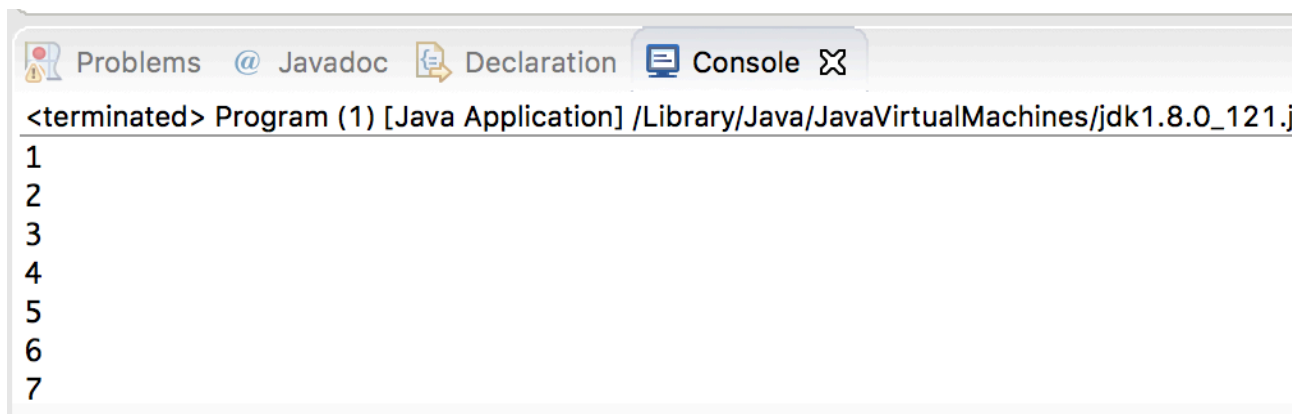
The screenshot shows an IDE window with a tab labeled 'Console'. The console output displays the numbers 1 through 5, each on a new line, which corresponds to the values stored in the testArray2 array in the code above. The window title bar includes icons for Problems, Javadoc, Declaration, and Console.

- falls man mit einer for Schleife über ein Array iteriert, so wie in den obigen Beispielen, dann sollte man eine schönere Variante wählen
- statt die Bedingung der Schleife mit einer Konstanten festzulegen, arbeitet man mit dem length Attribut/Eigenschaft
- => fügt automatisch die Länge des Arrays ein, also die Anzahl der Elemente
- auch hierzu ein kurzes Beispiel:

```
int[] testArray2 = new int[7];

for(int i = 0; i < testArray2.length; i++) {
    testArray2[i] = i + 1;
}

for(int i = 0; i < testArray2.length; i++) {
    System.out.println(testArray2[i]);
}
```



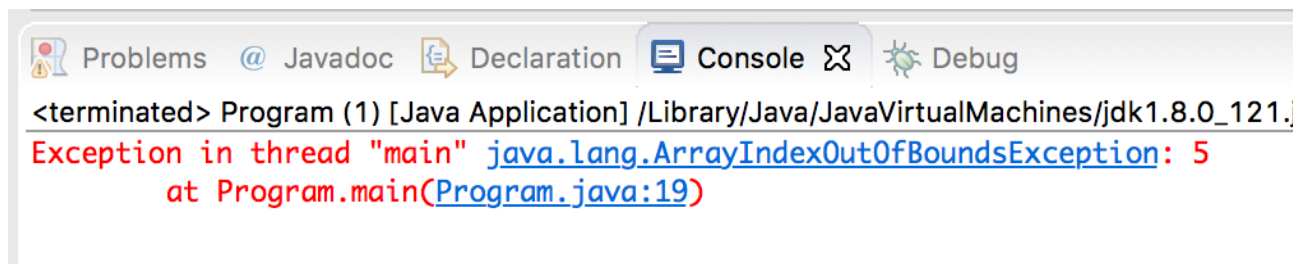
The screenshot shows an IDE window with a tab labeled "Console". The console output displays the numbers 1 through 7, each on a new line, indicating the successful execution of the provided Java code. The window title bar includes icons for Problems, Javadoc, Declaration, and Console, along with a close button. The console text starts with "<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.j" followed by the numbers 1, 2, 3, 4, 5, 6, and 7.

```
<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.j
1
2
3
4
5
6
7
```

Einschub: Speicherbereich von Arrays überschreiten

- was passiert wenn man den Speicherbereich eines Arrays überschreitet?
- beispielsweise wenn man einen Wert an ein Array Element zuweist, welches gar nicht existiert:

```
int[] array = new int[5];  
array[5] = 10;
```



- zu unserem Schutz wird eine sogenannte „IndexOutOfBoundsException“ geschmissen
- dadurch wird verhindert, dass wir einen fremden Speicherbereich überschreiben

Arrays mithilfe der foreach Schleife durchlaufen

-statt der for Schleife kann man auch eine sogenannte foreach Schleife benutzen, wenn man durch Arrays iteriert

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

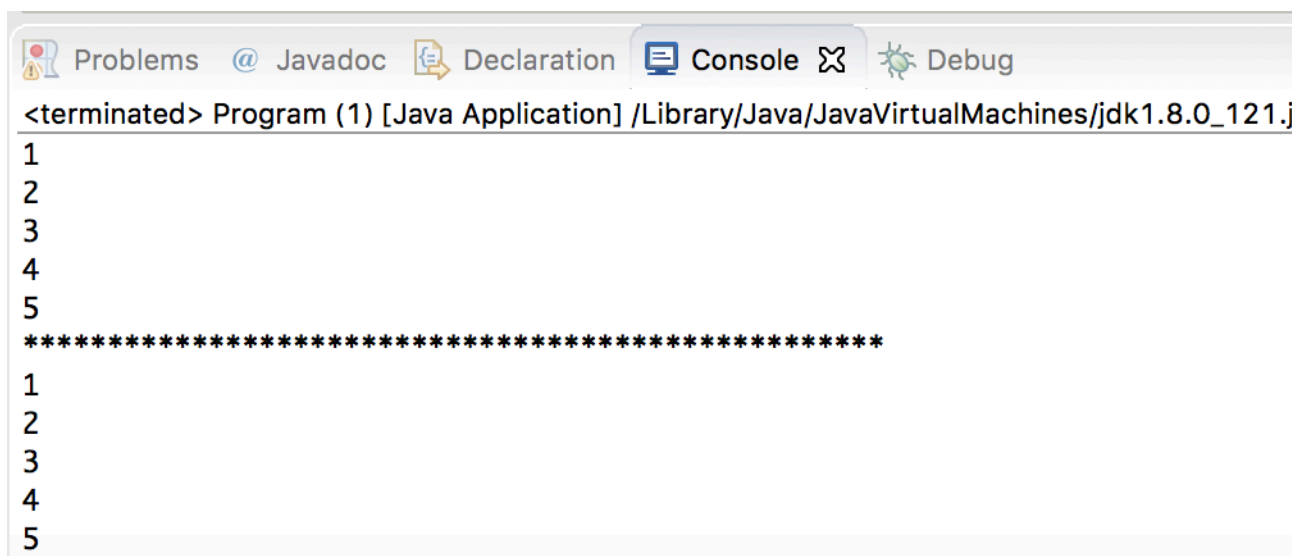
```
//Normale for Schleife
```

```
for(int i = 0; i < array.length; i++) {  
    array[i] = i + 1;  
    System.out.println(array[i]);  
}
```

```
System.out.println("*****");
```

```
//Foreach Schleife
```

```
for(int arrayElement: array) {  
    System.out.println(arrayElement);  
}
```



```
<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.j  
1  
2  
3  
4  
5  
*****  
1  
2  
3  
4  
5
```

Arbeiten mit mehrdimensionalen Arrays

- neben eindimensionalen Arrays gibt es auch mehrdimensionale Arrays
- für jede zusätzliche Dimension, einfach bei der Deklaration ein eckiges Klammerpaar mehr angeben
- bei einem zweidimensionalen Array ist die erste eckige Klammern für die Zeilen und die zweite für die Spalten verantwortlich

//Zweidimensionale Arrays deklarieren

```
int[][] field = new int[4][6];
```

//Dem Feld 1/2 den Wert 100 zuweisen

```
field[1][2] = 100;
```

//Zweidimensionales Array direkt bei Deklaration initialisieren

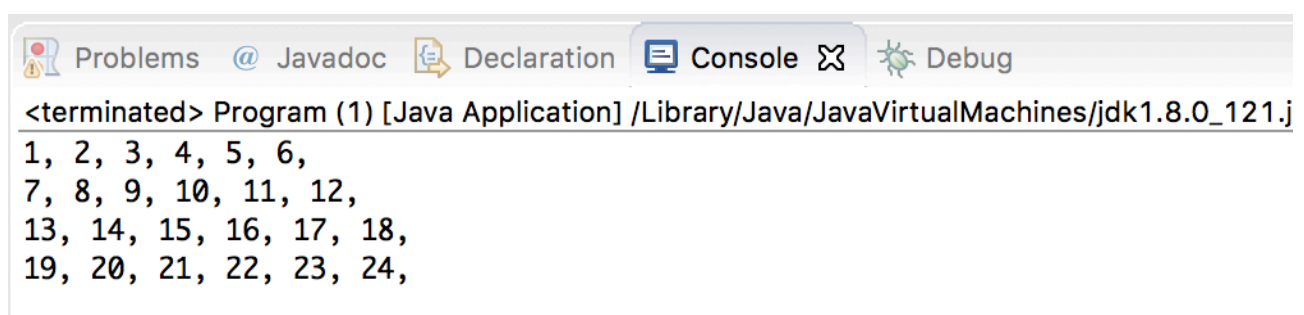
```
int field2[][] = {{1, 2, 3, 4, 5, 6} ,           //1te Zeile
                  {7, 8, 9, 10, 11, 12},         //2te Zeile
                  {13, 14, 15, 16, 17, 18},       //3te Zeile
                  {19, 20, 21, 22, 23, 24}};      //4te Zeile
```

- mithilfe einer verschachtelten for-Schleife kann auch der Inhalt eines zwei dimensional Arrays problemlos auf der Konsole ausgegeben werden

//Zweidimensionales Array direkt bei Deklaration initialisieren

```
int field2[][] = {{1, 2, 3, 4, 5, 6} ,           //1te Zeile
                  {7, 8, 9, 10, 11, 12},         //2te Zeile
                  {13, 14, 15, 16, 17, 18},       //3te Zeile
                  {19, 20, 21, 22, 23, 24}};      //4te Zeile
```

```
for(int i = 0; i < field2.length; i++) {
    for(int j = 0; j < field2[i].length; j++) {
        System.out.print(field2[i][j] + ", ");
    }
    System.out.println();
}
```



```
<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.j
1, 2, 3, 4, 5, 6,
7, 8, 9, 10, 11, 12,
13, 14, 15, 16, 17, 18,
19, 20, 21, 22, 23, 24,
```