Computer Science for Engineers
Exercise 4
Arrays

Prof. Dr. Dr.-Ing. J. Ovtcharova
Dipl. Wi.-Ing. Dan Gutu
5th of December 2008
Arrays in Java (1)

- **Array**
  - Structure of elements of the same kind
  - The individual elements are called components and can be activated by indices.
  - Arrays are basically dynamic in Java, that means that the number of components can be defined during runtime.
  - The programmer will be informed about the number of components by the element `length` in each array.

**Example**

```java
int[] field;  // Declaration of the variable field
field = new int[8]  // Initialisation with 8 components
for (int i = 0; i < field.length; i++)
    field[i] = i;
```
Arrays in Java (2)

- Arrays can save several values or objects
- Each value or object can be stored in a compartment of the array.
- The number of the compartments is defined with the creation of a new array.
  - Example: Creating the arrays `double_array` with 10 elements:
    ```java
    // Array for the storing of double-values
    double[] double_array = new double[10];
    ```
- The value or object in a certain compartment is accessed by the index of the compartment.
  ```java
  k = double_array[i];
  ```
  - All arrays begin with index 0!
  - Example: Assigning the circle in field i to variable k
One-dimensional arrays (1)

• Arrangements of the same kind of information in one structure
• Syntax of the declaration of an array:

```
Datatype of the component [ ] Array name ;
```

• Arrays are no primitive data-types
• Reservation of disk space and initialisation by using `new`:

```
Array name = new Datatype of the component [ Number of components ] ;
```
One-dimensional arrays (2)

• common: declaration, disk space and initialization in one line
• examples:

```java
double[] aField = new double[5];
int number = 12;
int[] anotherField = new int[number];
```

• access one individual components of an array:

```
Array name [ Valid index ]
```

• examples:

```
aField[0] = 1.2;
anotherField[10] = -42;
```
Example:

array: "field"

Total number of array elements:

Array component index:
• Compute the sum of the components of an array passed to the method as parameter

• Solution:

```java
public double sumOfComponents(double[] aField) {
    double sum = 0.0;
    for (int i = 0; i < aField.length; i++)
        sum = sum + aField[i];
    return sum;
}
```
One-dimensional array – Example 2: Fibonacci-numbers

• Task: Compute the Fibonacci-numbers:

\[ f_0 = 1, \ f_1 = 1, \ f_n = f_{n-2} + f_{n-1}, \text{for } n > 2 \]

Write a method that computes and returns the Fibonacci-numbers. The method receives as parameter the number of Fibonacci-numbers to be computed.

• Solution:

```java
public int[] fibonacci(int number) {
    int[] fibonacci = new int[number];
    fibonacci[0] = 1;
    fibonacci[1] = 1;
    for (int i = 2; i < fibonacci.length; i++)
        fibonacci[i] = fibonacci[i-2] + fibonacci[i-1];
    return fibonacci;
}
```
Multidimensional arrays (1)

• “multidimensional data-types” in JAVA not possible to realize

• alternative: nested, one dimensional arrays:
  in each component of an one dimensional array can be placed in a one dimensional array, again.

• Examples of declarations, reservation of disk space and nested Initialisation:

```java
int[][] valuesOnAFlatGrid = new int[10][9];
double[][][] aStructure = new double[3][4][5];
```

```java
valuesOnAFlatGrid[2][5] = -0.56;
```

• access on components of nested arrays:

```java
valuesOnAFlatGrid[2][5] = -0.56;
```
multidimensional arrays

are displayed in Java by arrays in arrays

On one dimensional fields, you have to distinguish between the absolute declarations and the initialisation with `new` (--> assignment of disk space).

possible syntax

```java
int matrix1[][];
int[] matrix2[];
int[][] matrix3;
```
example for a 3-dimensional array $[i][j][k]$

values for example:

array $[3][0][1] = 20$, array $[1][0][2] = 12$
- also not-rectangular structures are possible: in each component one dimensional errors of different size

- example for direct initialisation:

  ```java
  int[][] aField = {{1, 2}, {3, 4, 5, 6}, {7, 8, 9}};
  aField.length -> 3;    aField[0].length -> 2
  aField[1].length -> 4; aField[2].length -> 3
  ```
Initialisation of multidimensional arrays (1)

```java
// a pair of brackets for each dimension
// to enclose the components
public class Matrix {
    public static void main(String[] args) {
        double[][] matrix = {
            { 0*0, 1*0, 2*0, 3*0 },
            { 0*1, 1*1, 2*1, 3*1 },
            { 0*2, 1*2, 2*2, 3*2 },
            { 0*3, 1*3, 2*3, 3*3 },
        };

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

Output on the screen

```
0 0 0 0
0 1 2 3
0 2 4 6
0 3 6 9
```
// account of components as products of indices
public class ThreeDArray {
    public static void main(String[] args) {
        int[][][] threeD = new int[3][4][5];

        for (int i = 0; i < threeD.length; i++)
            for (int j = 0; j < threeD[i].length; j++)
                for (int k = 0; k < threeD[i][j].length; k++)
                    threeD[i][j][k] = i * j * k;

        for (int i = 0; i < threeD.length; i++) {
            for (int j = 0; j < threeD[i].length; j++) {
                for (int k = 0; k < threeD[i][j].length; k++)
                    System.out.print(threeD[i][j][k] + " ");
                System.out.println();
            }
            System.out.println();
        }
    }
}
example on a 2-dimensional array:

<table>
<thead>
<tr>
<th></th>
<th>1.1</th>
<th>1.2</th>
<th>1.3</th>
<th>1.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5</td>
<td>1.6</td>
<td>1.7</td>
<td>1.8</td>
<td></td>
</tr>
<tr>
<td>1.9</td>
<td>2.0</td>
<td>2.1</td>
<td>2.2</td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td>2.4</td>
<td>2.5</td>
<td>2.6</td>
<td></td>
</tr>
</tbody>
</table>

Display by allocation of the matrix in rows

Row 1: [1.1, 1.2, 1.3, 1.4]

Row 2: [1.5, 1.6, 1.7, 1.8]

Row 3: [1.9, 2.0, 2.1, 2.2]

Row 4: [2.3, 2.4, 2.5, 2.6]
// two dimensional array:
public class TwoDArray {
    public static void main(String[] args) {
        int[][] twoD = new int[4][5];
        int i, j, k = 0;
        for(i=0; i < twoD.length; i++) {
            for(j=0; j < twoD[i].length; j++) {
                twoD[i][j] = k;
                k++;
            }
        }
        for(i=0; i < twoD.length; i++) {
            for(j=0; j < twoD[i].length; j++) {
                System.out.print(twoD[i][j] + " ");
            }
            System.out.println();
        }
    }
}
// example for a nested array
public class TwoDAgain {
    public static void main(String[] args) {
        int[][] twoD = new int[3][];
        twoD[0] = new int[1];
        twoD[1] = new int[2];
        twoD[2] = new int[3];
        int k = 0;
        for(int i=0; i < twoD.length; i++) {
            for(int j=0; j < twoD[i].length; j++) {
                twoD[i][j] = k;
                k++;
            }
        }
        for(int i=0; i < twoD.length; i++) {
            for(int j=0; j < twoD[i].length; j++)
                System.out.print(twoD[i][j] + " ");
            System.out.println();
        }
    }
}

Output on the screen:
0
1 2
3 4 5