Arrays

Problem

Define more than one variable for similar purpose

Example:

Grades of a student group – define 30 variable for them

Solution:

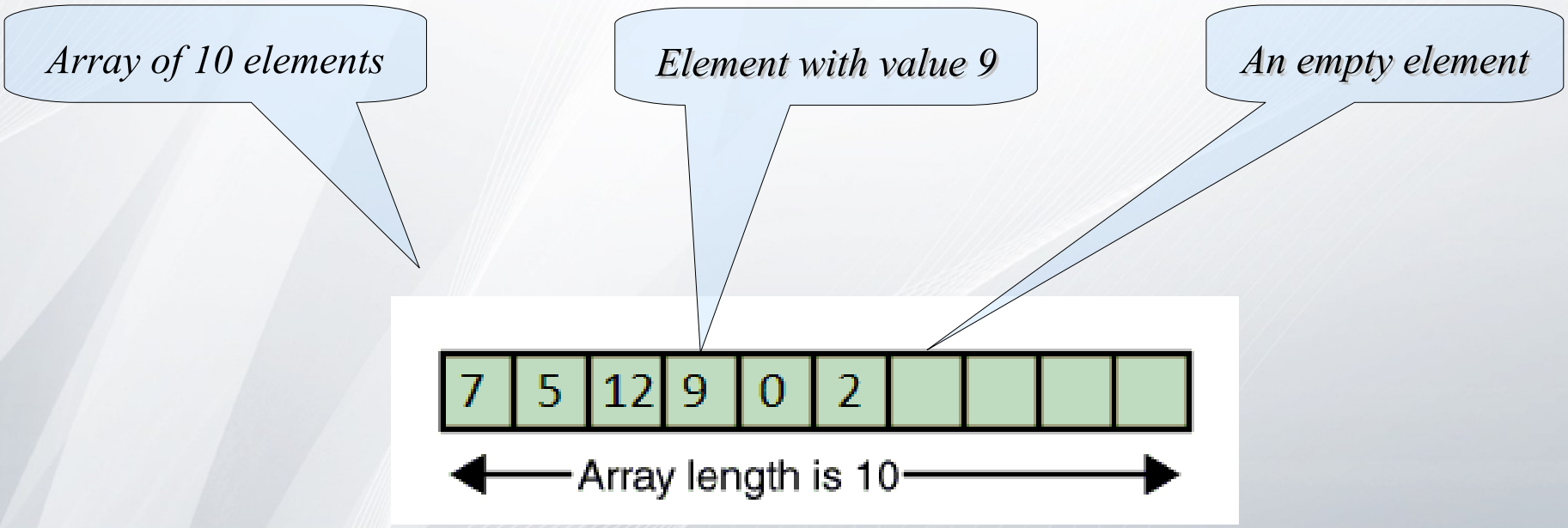
Define 30 variable of type double to hold the information

Is this so rational?

What is an array?

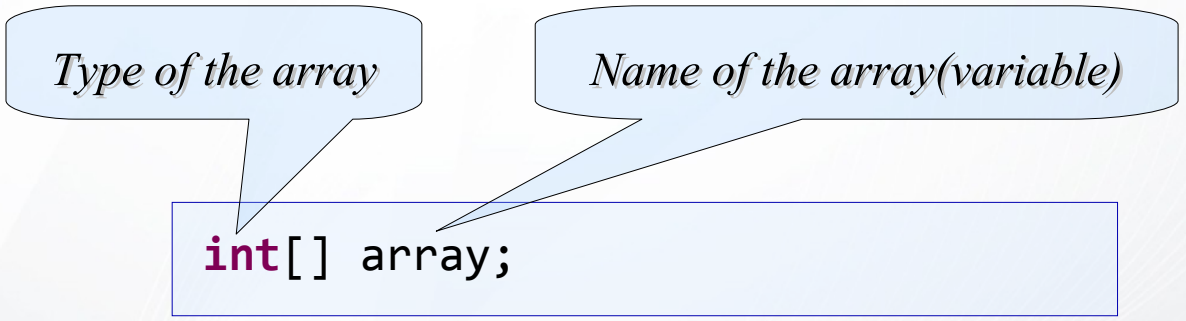
- An array is a sequence of elements
- Arrays keep variables of only one type
- The order of the elements remains the same
- Arrays have a fixed length
- The access to the elements is direct
- The elements are accessed through an index

What is an array?



Declaration and initialization

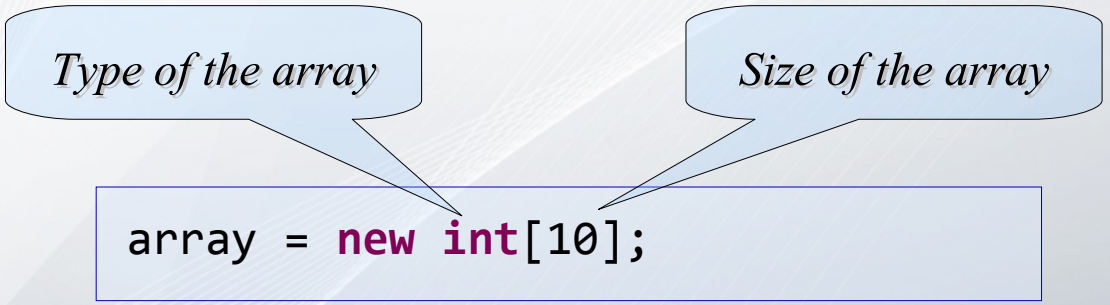
Declaration




```
int[] array;
```

```
int array[];
```

Initialization




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

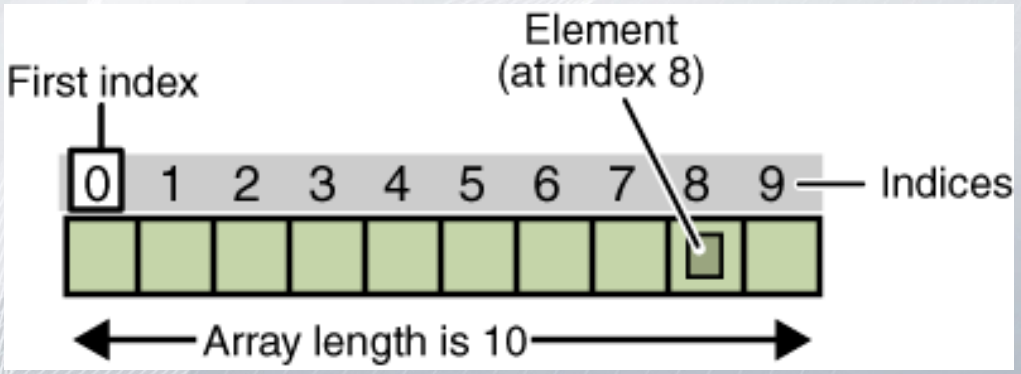
Declaration and initialization

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

```
int[] array = { 5, 7, -2, 12, 0, 4 };
```

Accessing the elements

- Elements are accessed by index
- The index of the first is 0
- The index of the last is equal to the length – 1
- The elements can be read and changed



Accessing the elements

array[i] returns the element with index i

```
System.out.println(array[0]);
```

Print the fisrt element

```
System.out.println(array[1]);
```

Print the second element

```
array[2] = 100;
```

Change the value of the third element

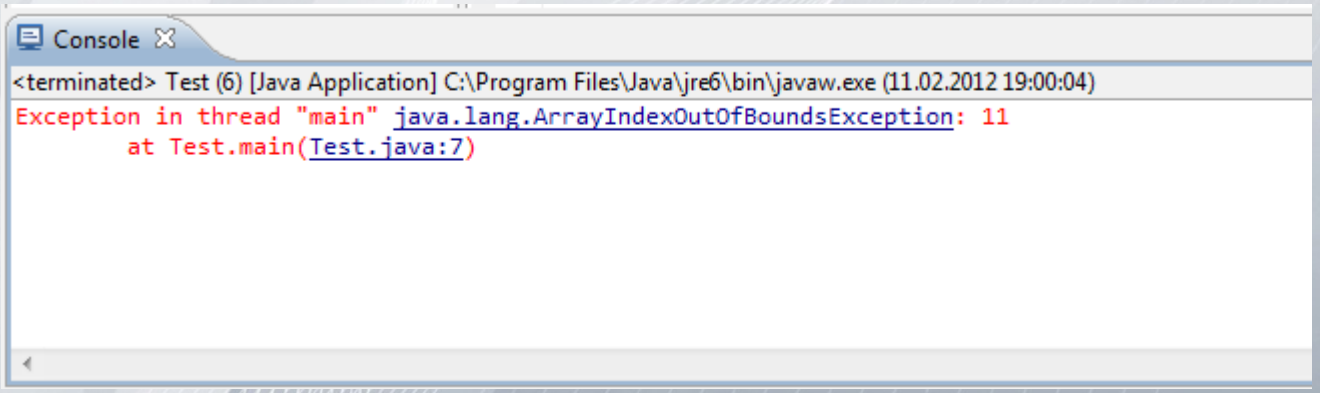
More about the arrays

- array.length returns the length

```
System.out.println(array.length);
```

- Getting an element beyond the size will result in a runtime error

```
array[11] = 20;
```



Exercise

- Create an array with 10 elements
- Set some values to the first 3 elements
- Print the second element
- Multiply the 3rd element with 2 and then print it

Iterating the array

- Usually using loop
- The most common case is using a **for** loop

```
public static void main(String[] args) {
    int[] array = new int[10];
    for (int i = 0; i < array.length; i++) {
        array[i] = 7;
    }
}
```

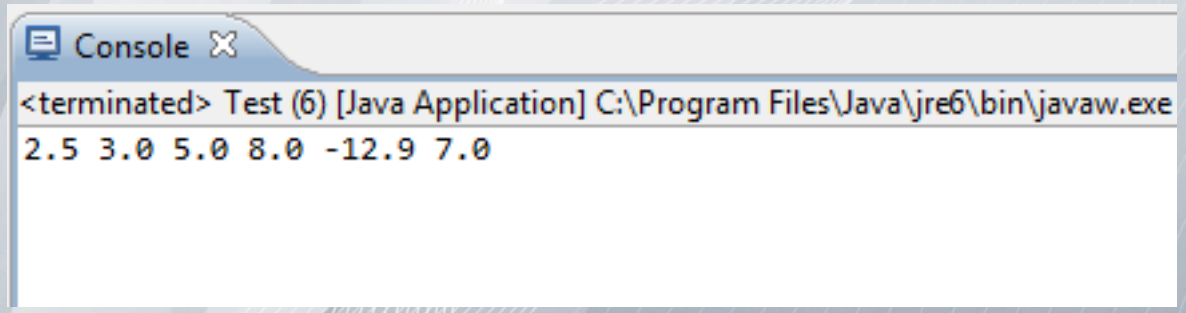
- You can iterate array with **while** loop and any other

```
public static void main(String[] args) {
    int[] array = new int[10];
    int i = 0;
    while (i < array.length) {
        array[i] = 7;
        i++;
    }
}
```

Printing to console

- The array is iterated
- The value of the current element is printed using System.out.print()

```
double[] array = { 2.5 ,3, 5, 8, -12.9, 7.0 };

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



Add a whitespace to separate the elements

Reading from console

- The array is iterated
- Use scanner to read the value from the console
- Assign the read value to the current element

```
public static void main(String[] args) {
    int[] array = new int[10];

    Scanner sc = new Scanner(System.in);

    for (int i = 0; i < array.length; i++) {
        System.out.println("Enter value:");
        array[i] = sc.nextInt();
    }
}
```

Declaration and initialization of the array

*Create **Scanner***

*Iterate the array using **for** loop and read the value for each element from the console*

Exercise

- Create an array with user-specified size
- Enter its elements from the console
- Find its max number
- Print the array to the console
- Print the max number

Exercise (Advanced)

- Invert given array using two approaches:
  - Using second array
  - Without second array

- Write a program, which reads an array from the console and then creates a new array, which is with 1 element bigger than the original one. All elements, except the first one, should be the previous element, multiplied with the current index. For example:

```
[9 2 4 -3 7 5]
[6 9 4 12 -12 35 30]
```

Comparing arrays

- Arrays are referred types and can't be compared using == operator
- To compare two arrays, you have to iterate them and compare their elements respectively

Try it!

```
double[] array = { 2.5 , 3, 5.8 };
double[] array2 = new double[3];
array2[0] = 2.5;
array2[1] = 3;
array2[2] = 5.8;
...
```

Copying arrays

```
int[] newArray = oldArray;
```

is not exactly what you want

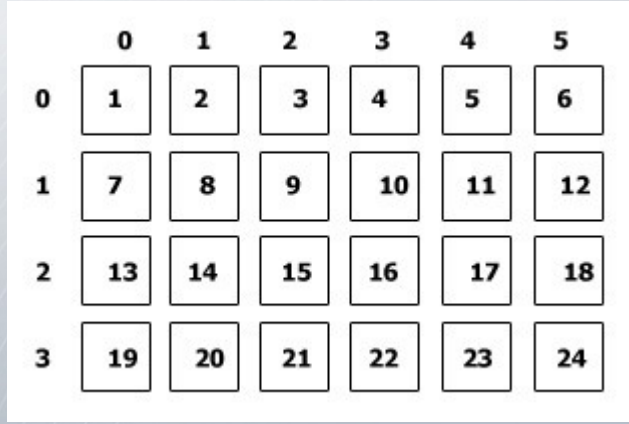
```
public static void main(String[] args) {
    int[] oldArray = { 1, 2, 3};
    int[] newArray = oldArray;

    oldArray[0] = -10;
    System.out.println(newArray[0]);
}
```

What will be printed to the console?

Multidimensional Arrays

- Have more than one dimension (2, 3, 4, …)
- The 2-dimensional arrays are called matrices
- A matrix is an array in which each element is an array

A matrix with 4 rows and 6 columns



	0	1	2	3	4	5
0	1	2	3	4	5	6
1	7	8	9	10	11	12
2	13	14	15	16	17	18
3	19	20	21	22	23	24

Creating and iterating matrix

The multidimensional arrays use the same concept as an ordinary arrays

```
public static void main(String[] args) {
    int[][] matrix = new int[3][4];

    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix[0].length; j++) {
            matrix[i][j] = 10;
        }
    }

    matrix[0][0] = 1;
    matrix[2][3] = 100;
}
```

Create matrix with 3 rows and 4 columns

Set 1 to the element in the upper left corner

Set 100 to the element in the bottom right corner

Summary

- What is array
- How to declare and initialize array
- How to access and change elements
- How to get the array length
- How to read an array from the console
- How to copy an array
- What is a matrix