By-
Tanu Singh
Lecturer

Java Programming → Arrays

Introduction to Array

* An array is a group of like-typed variables that are refferd to by a common name. Arrays of any type can be created and may have one or more dimensions. A specific element in an array is accessed by its index.

"OR"

An array is a group of continous or related data items that share a common name

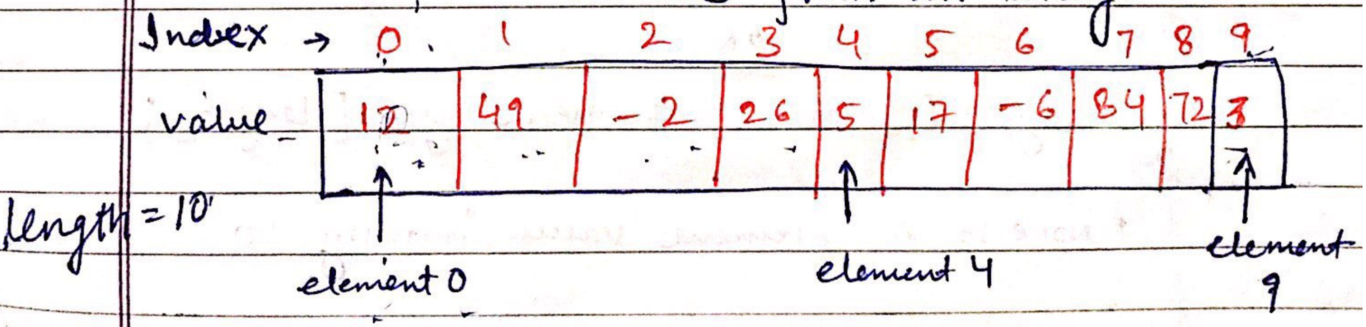
Syntax :- array-name [value];

Example :- salary [10];

"OR"

Array : object that stores many values of the same type.

- * Element : One value in an array
- * Length : Number of elements in an array
- * Index : Starts from 0 to access an elements from an array.



- By
Tanu Singh

Date :
Page No.

Declaration of Arrays

- Arrays in Java may be declared in two forms :

Form 1 : type array-name [];

form 2 : Type [] array-name ;

Examples:

```
int      number [];
float    average [];
int[]    counter ;
float[]  marks ;
```

Creation of Arrays

- Java allows to create arrays using new operator only.

array-name = new type[size];

Examples:

```
number  = new int [5];
average = new float [10];
```

"OR"

type [] name = new type [length];

- Note :- All elements value initially '0'.

Scanned with CamScanner

– By
Tanu Singh

Date :

Page No.

```
int [] numbess = new int [6];
```

index	0	1	2	3	4	5
value	0	0	0	0	0	0

[illegible]

```
type [] name = { value , value , ...value };

        int [] numbess = { 12, 49, -2, 26, 5, 17 };
```

index	0	1	2	3	4	5
value	12	49	-2	26	5	17

Accessing elements

```
name[index]            // access
name[index] = value ;  //modify
```

Example :

```
numbess [0] = 27;
numbess [3] = -6;

System.out.printen (numbess[0]);
  if (numbess [3] < 0)
  {
     System.out.printen ("Element 3 is -ve");
  }
```

Index	0	1	2	3	4	5	6	7	8	9
Value	27	0	0	-6	0	0	0	0	0	0

- By
Tanu Singh

Date :

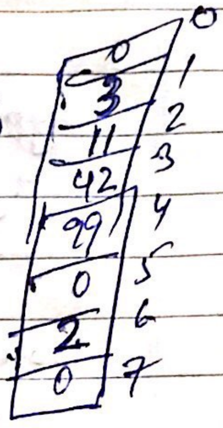
Page No.

Accessing array elements

Example :-

```
int[] numbers = new int[8];
numbers[1] = 3;
numbers [4] = 99;
numbers [6] = 2;

int x = numbers[1];
   numbers [x] = 42;
   numbers [numbers [6]] = 11; //use
      number(6) as index
```



Arrays of other type (Accessing elements)

*
```
double [] results = new double[5];
     results [2] = 3.4 ;
     results [4] = -0.5 ;
```

index	0	1	2	3	4
value	0.0	0.0	3.4	0.0	-0.5

*
```
boolean [] tests = new boolean [6];
        test [3] = true ;
```

index	0	1	2	3	4	5
value	False	False	False	True	False	False

Scanned with CamScanner

-By
Tanu Singh

Date :
Page No.

Arrays and for Loop

*
```
for (int i=0; i<8; i++)
    { numbers [i] = 2 * i; }
```

index	0	1	2	3	4	5	6	7
value	0	2	4	6	8	10	12	14

Out - of - Bounds

* Legal Indexes : b/w 0 and the array length-1
  - Reading or writing any index outside this range will throw an "ArrayIndexOutofBoundException".

Example :

```
int [] data = new int[10];
System.out.println (data [0]); //okay
System.out.println (data[9]); //okay
System.out.println (data [-1]); //exception
System.out.println (data [10]); //exception
```

Array . to String

* Array. to string accepts an array as a parameter and returns a string representation of its elements.

Note* must import java.util.*;

```
              0  1  2  3  4
int [] e = {0, 2, 4, 6, 8};
e [1] = e [3] + e [4];
System.out.println ("e is" + Arrays.toString(e));
```

O/P : `e is [0, 14, 4, 6, 8]`

-By
Tanu Singh

Date :
Page No

The Array Class

- Class Arrays in package java.util has useful static methods for manipulating arrays :

Method Name	Description
binarysearch (array, value)	- search if the array is sorted, also returns the index of a given value in a sorted array
copyof (array, length)	- returns a new copy of an array
equals (array1, array2)	- returns true if the two arrays contain same elements in the same order.
fill (array, value)	- sets every element to a given value
sort (array)	- arranges the elements into sorted order
tostring (array)	- returns a string representing the array such as "[10, 30, -25, 17]".

Syntax: Arrays. methodname (parameters).

– By
Tanu Singh

Date :
Page No.

Two-dimensional Arrays

* In this, the first index selects 'Row' and the second index selects the 'Column' within that ~~Row~~ row.

* For creating two-dimensional array, same steps are to be followed as that of one-dimensional arrays.

Example :

```
int myArray [][];
myArray = new int [3][4];
```

	Column 1	Column 2	Column 3	Column 4
Row 1	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 2	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 3	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Initializing 2D array in Java

```
int [][] a = {
             { 1, 2, 3 },
             { 4, 5, 6, 9 },
             { 7 },
             };
```

	Column 1	Column 2	Column 3	Column 4
Row 1	1	2	3	
Row 2	4	5	6	9
Row 3	7			