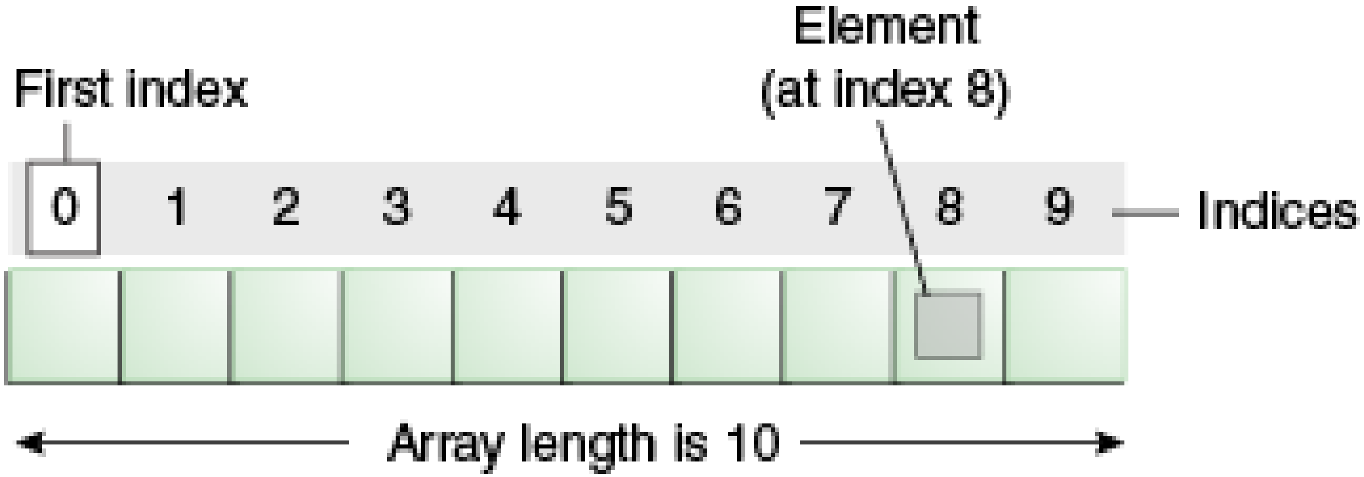
# Arrays

➔An array is a group of similar data type elements.

➔An Array is a collection of elements that share the same type and name.

➔The elements from the array can be accessed by the index. The array indices start at zero and ends at size-1.

➔You can access a specific element in the array by specifying its index within square brackets.


Element
(at index 8)
First index
0
1
2
3
4
5
6
7
8
9
Indices
Array length is 10

# Types of Array

There are three types of array.

1. Single Dimensional Array
2. Multi Dimensional Array

# Single Dimensional Array

➔To create an array, we must first create the array variable of the desired type. The general form of the One Dimensional array is as follows:

**type  arrayname[ ] = new type[size];**

➔Here type declares the base type of the array. This base type determine what type of elements that array can consist.

➔size specifies the number of elements stored in the array.

➔new to allocate memory for an array.

➔The elements in the array allocated by new will automatically be initialized to zero.

## Example: int months[] = new int[12];

Here type is int, the array name is months. All the elements in the months are integers. Since, the base type is int. The elements in the array allocated by new will automatically be initialized to null.

| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Element | null | null | null | null | null | null | null | null | null | null | null | null |

# Initialization of Java Array

We can declare, instantiate and initialize the java array together by:

**datatype arrayname[]={elemements are separated by comma};**

**Example**

```
int a[]={33,3,4,5};
//Java Program to illustrate the use of declaration, instantiation
//and initialization of Java array in a single line
```

```
class Testarray1
        {
            public static void main(String args[])
            {
                int a[]={33,3,4,5};
                for(int i=0;i<a.length;i++)
                    System.out.println(a[i]);
            }
         }
```

**length: length** can be used for int[], double[], String[] to know the length of the arrays.

Syntax

**arrayname.length**

**Example Program: Write a Java Program to read elements into array and display them?**

```
import java.util.*;
class ArrayTest
{
public static void main(String args[])
{
Scanner s=new Scanner(System.in);
int a[] = new int[5];
//read the elements into array
System.out.println("Enter the elements into Array:");
 for(int i=0;i<a.length;i++)
a[i]=s.nextInt();
System.out.println("The elements of Array:");
for(int i=0;i<a.length;i++)
System.out.print(a[i]+"  ");
}
}
```

# Examples:

1. Java Program to copy all elements of one array into another array
2. Java Program to print the elements of an array
3. Java Program to print the elements of an array in reverse order
4. Java Program to print the elements of an array present on even position
5. Java Program to print the elements of an array present on odd position
6. Java Program to print the largest element in an array
7. Java Program to print the smallest element in an array
8. Java Program to print the number of elements present in an array
9. Java Program to print the sum of all the items of the array

# Multidimensional Arrays

**Multidimensional Arrays** can be defined in simple words as array of arrays. Data in multidimensional arrays are stored in tabular form.

**Syntax:**

**datatype**[1st dimension][2nd dimension][]..[Nth dimension] **array_name = new data_type**[size1][size2]….[sizeN];

**where:**

- **data_type**: Type of data to be stored in the array. For example: int, char, etc.
- **dimension**: The dimension of the array created. For example: 1D, 2D, etc.
- **array_name**: Name of the array
- **size1, size2, …, sizeN**: Sizes of the dimensions respectively.

**Examples:**

- Two dimensional array:

```
int[][]  two = new int[10][20];
```

- Three dimensional array:

```
int[][][]  three = new int[10][20][30];
```

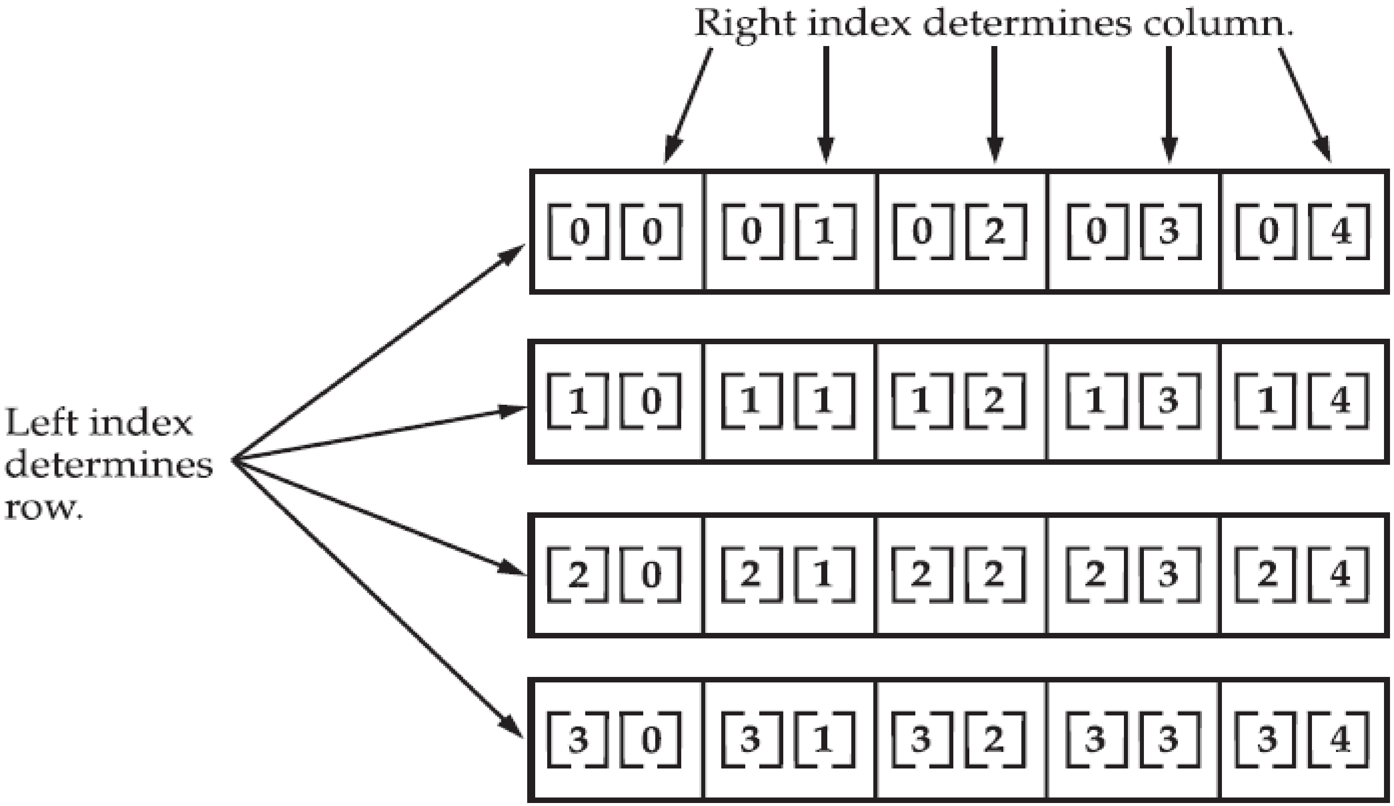
**Size of multidimensional arrays**: The total number of elements that can be stored in a multidimensional array can be calculated by multiplying the size of all the dimensions.

**For example:**

- The array **int[][] x = new int[10][20]** can store a total of (10*20) = 200 elements.
- Similarly, array **int[][][] x = new int[5][10][20]** can store a total of (5*10*20) = 1000 elements.

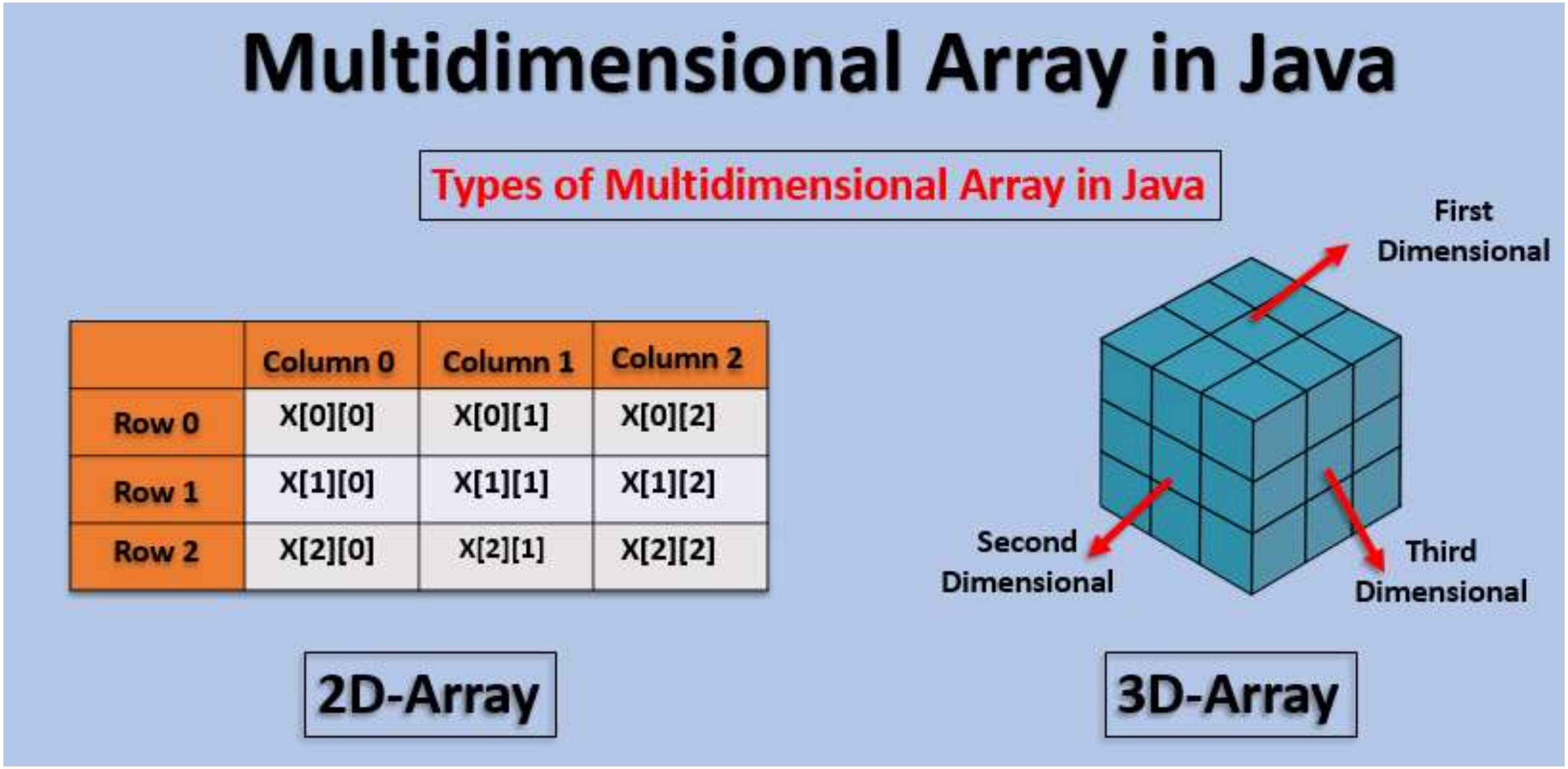
For example, the following declares a two dimensional array variable called **twoD:**

int twoD[][] = new int[4][5];



| | | | | |
|---|---|---|---|---|
| [0][0] | [0][1] | [0][2] | [0][3] | [0][4] |
| [1][0] | [1][1] | [1][2] | [1][3] | [1][4] |
| [2][0] | [2][1] | [2][2] | [2][3] | [2][4] |
| [3][0] | [3][1] | [3][2] | [3][3] | [3][4] |

Given: int twoD [ ] [ ] = new int [4] [5] ;



# Multidimensional Array in Java

## Types of Multidimensional Array in Java

| | Column 0 | Column 1 | Column 2 |
|---|---|---|---|
| Row 0 | X[0][0] | X[0][1] | X[0][2] |
| Row 1 | X[1][0] | X[1][1] | X[1][2] |
| Row 2 | X[2][0] | X[2][1] | X[2][2] |

2D-Array

3D-Array

```
// Read and print the two dimensional array elements
import java.util.Scanner;
class Testarray2 {
public static void main(String args[]) {
int m, n, i, j;
Scanner s = new Scanner(System.in);
System.out.println("Enter the no. of rows and columns of matrix");
m = s.nextInt());
n = s.nextInt());
int[ ][ ] array  = new int[m][n];
System.out.println("Enter the elements of matrix:");
for ( i = 0 ; i < m ; i++ )
for ( j = 0 ; j < n ; j++ )
array[i][j] = s.nextInt();
```

```
System.out.println(“The given matrix elements are:");
for ( i = 0 ; i < m ; i++ ) {
for ( j = 0 ; j < n ; j++ ) {
System.out.print( array[i][j]+“ “);
        }
System.out.println();
    }
}
}
```

```
// Print the three dimensional array elements
class Testarray3 {
int  i, j, k;
public static void main(String args[]) {
int[ ][ ] array  = { {{1,2,3},{4,5,6}} , {{6,5,4},{3,2,1}} }
System.out.println(“The given elements are:");
for ( i = 0 ; i < 2; i++ ) {
  for ( j = 0 ; j < 3 ; j++ ) {
    for ( k = 0 ; k < 3 ; k++ ) {
         System.out.println(“   “ + array[i][j][k]);
       }
    System.out.println();
     }
   System.out.println();
   }
 }
}
```

# Examples:

1. Transpose matrix
2. Java Program to subtract the two matrices
3. Java Program to determine whether a given matrix is an identity matrix
4. Java Program to determine whether a given matrix is a sparse matrix
5. Java Program to determine whether two matrices are equal
6. Java Program to display the lower triangular matrix
7. Java Program to display the upper triangular matrix
8. Java Program to find the frequency of odd & even numbers in the given matrix
9. Java Program to find the product of two matrices
10. Java Program to find the sum of each row and each column of a matrix