## **Introduction to array in Java**

**1. Introduction to Arrays**

An array is a **fixed-size** collection of **similar data types** stored in **contiguous memory locations**.

Instead of declaring multiple variables, you can store a group of values in one variable using an array.

**Key Features:**

* Stores multiple values of the same type.
* Can store **primitives** (int, float, char...) or **objects**
* Indexed starting from 0.
* Fixed size (cannot grow or shrink dynamically).
* Fast access using index (O(1) time complexity).

**2. Types of Arrays**

**1. Single-Dimensional Array (1D Array)**

int[] numbers = {126,32,230,21,200};  
or

int[] numbers = new int[5];

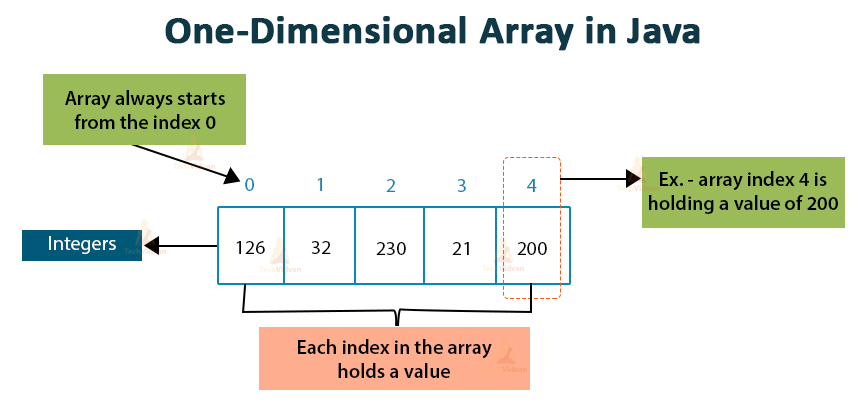
numbers[0]=126;

numbers[1]=32;

numbers[2]=230;

numbers[3]=21;

numbers[4]=200;



int a[] = new int[10];

a[0]=1;

a[1]=2;

a[2]=3;

a[3]=4;

a[4]=5;

a[5]=6;

a[6]=7;

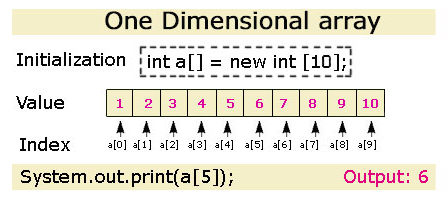
a[7]=8;

a[8]=9;

a[9]=10;

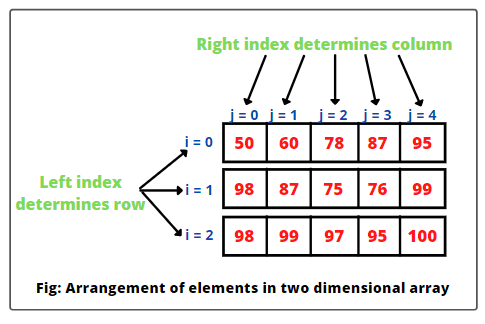
Or

int a[]= {1,2,3,4,5,6,7,8,9,10};



**2. Multi-Dimensional Array (2D, 3D, etc.)**

int[][]= { {50,60,78,87,95}, {98,87,75,76,99}, {98,99,97,95,100} };



**3. Real-World Examples of Arrays**

1. **Student Marks:** int[] marks = {85, 92, 78, 90};
2. **Shopping Cart Items:** String[] cart = {"Milk", "Bread", "Eggs"};
3. **Chess Board:** char[][] chessBoard = new char[8][8];
4. **Temperature Records:** double[] weeklyTemp = {25.5, 26.0, 24.8};

**Practical Real-World Examples Using Arrays**

**1] Store and Print Student Marks**

import java.util.Scanner;

public class StudentMarks {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int[] marks = new int[5];

for (int i = 0; i < marks.length; i++) {

System.out.print("Enter mark " + (i + 1) + ": ");

marks[i] = sc.nextInt();

}

System.out.println("Entered Marks:");

for (int mark : marks) {

System.out.println(mark);

}

}

}

**2] Find Maximum in an Array**

public class MaxInArray {

public static void main(String[] args) {

int[] numbers = {12, 45, 7, 89, 23};

int max = numbers[0];

for (int num : numbers) {

if (num > max) {

max = num;

}

}

System.out.println("Maximum number is: " + max);

}

}

**3] Count Even and Odd Numbers**

public class EvenOddCount {

public static void main(String[] args) {

int[] nums = {1, 4, 5, 8, 9, 10};

int even = 0, odd = 0;

for (int n : nums) {

if (n % 2 == 0) even++;

else odd++;

}

System.out.println("Even: " + even + ", Odd: " + odd);

}

}

**4] 2D Array – Matrix Addition**

public class MatrixAddition {

public static void main(String[] args) {

int[][] a = {

{1, 2},

{3, 4}

};

int[][] b = {

{5, 6},

{7, 8}

};

int[][] sum = new int[2][2];

for (int i = 0; i < 2; i++) {

for (int j = 0; j < 2; j++) {

sum[i][j] = a[i][j] + b[i][j];

}

}

System.out.println("Sum of matrices:");

for (int[] row : sum) {

for (int val : row) {

System.out.print(val + " ");

}

System.out.println();

}

}

}

**🔧 Common Operations on Arrays in Java**

**👉 Operations:**

1. **Insert an element**
2. **Delete an element**
3. **Display the array**
4. **Search for an element**
5. **Sort the array**

📝 Note: Arrays in Java have **fixed size**, so we must manually manage the array content (or use ArrayList for dynamic arrays later).

✅ 1. Insert an Element at a Specific Position

public class InsertElement {

public static void main(String[] args) {

int[] arr = new int[6];

int n = 5;

// Initial array

arr[0] = 10;

arr[1] = 20;

arr[2] = 30;

arr[3] = 40;

arr[4] = 50;

int pos = 2;

int newVal = 25;

for (int i = n; i > pos; i--) {

arr[i] = arr[i - 1];

}

arr[pos] = newVal;

n++;

System.out.println("Array after insertion:");

for (int i = 0; i < n; i++) {

System.out.print(arr[i] + " ");

}

}

}

🧾 Output:

Array after insertion:

10 20 25 30 40 50

✅ 2. Delete an Element from Specific Position  
  
public class DeleteElement {

public static void main(String[] args) {

int[] arr = {10, 20, 30, 40, 50};

int n = 5;

int pos = 2;

for (int i = pos; i < n - 1; i++) {

arr[i] = arr[i + 1];

}

n--;

System.out.println("Array after deletion:");

for (int i = 0; i < n; i++) {

System.out.print(arr[i] + " ");

}

}

}

🧾 Output:

Array after deletion:

10 20 40 50

✅ 3. Display Elements of Array

public class DisplayArray {

public static void main(String[] args) {

int[] arr = {5, 10, 15, 20};

System.out.println("Array elements:");

for (int val : arr) {

System.out.print(val + " ");

}

}

}

🧾 Output:

Array elements:

5 10 15 20

✅ 4. Search an Element in Array

public class SearchArray {

public static void main(String[] args) {

int[] arr = {12, 23, 34, 45, 56};

int key = 34;

boolean found = false;

for (int i = 0; i < arr.length; i++) {

if (arr[i] == key) {

System.out.println(key + " found at index " + i);

found = true;

break;

}

}

if (!found) {

System.out.println(key + " not found.");

}

}

}

🧾 Output:

34 found at index 2

✅ 5. Sort the Array (Ascending Order – Bubble Sort)

public class SortArray {

public static void main(String[] args) {

int[] arr = {25, 10, 45, 20, 5};

for (int i = 0; i < arr.length - 1; i++) {

for (int j = 0; j < arr.length - 1 - i; j++) {

if (arr[j] > arr[j + 1]) {

// Swap

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

System.out.println("Sorted array:");

for (int val : arr) {

System.out.print(val + " ");

}

}

}

🧾 Output:

Sorted array:

5 10 20 25 45

**✅ Summary Table**

| **Operation** | **Java Approach** |
| --- | --- |
| Insert | Shift elements right, then assign |
| Delete | Shift elements left from delete index |
| Display | Loop through array |
| Search | Loop & compare |
| Sort | Use bubble sort or built-in sort |

✅ **Arrays Class in Java – Overview**

import java.util.Arrays;  
  
🛠 Common Operations:

* Arrays.toString() – Print array as string
* Arrays.sort() – Sort array
* Arrays.equals() – Compare arrays
* Arrays.copyOf() – Copy array
* Arrays.fill() – Fill array with a value
* Arrays.binarySearch() – Search in sorted array

**1. Displaying Array using Arrays.toString()**  
import java.util.Arrays;

public class DisplayArray {

public static void main(String[] args) {

int[] arr = {10, 20, 30};

System.out.println("Array: " + Arrays.toString(arr));

}

}

🧾 Output:

Array: [10, 20, 30]

**2. Sorting Array using Arrays.sort()**

import java.util.Arrays;

public class SortExample {

public static void main(String[] args) {

int[] arr = {25, 10, 15, 5};

Arrays.sort(arr);

System.out.println("Sorted Array: " + Arrays.toString(arr));

}

}

🧾 Output:

Sorted Array: [5, 10, 15, 25]

**3. Copy Array using Arrays.copyOf()**

import java.util.Arrays;

public class CopyArray {

public static void main(String[] args) {

int[] original = {10, 20, 30};

int[] copy = Arrays.copyOf(original, 5); // new size 5

System.out.println("Copied Array: " + Arrays.toString(copy));

}

}

🧾 Output:

Copied Array: [10, 20, 30, 0, 0]

**4. Fill Array using Arrays.fill()**

import java.util.Arrays;

public class FillArray {

public static void main(String[] args) {

int[] arr = new int[5];

Arrays.fill(arr, 100);

System.out.println("Filled Array: " + Arrays.toString(arr));

}

}

🧾 Output:

Filled Array: [100, 100, 100, 100, 100]

**5. Search in Array using Arrays.binarySearch()**

import java.util.Arrays;

public class SearchArray {

public static void main(String[] args) {

int[] arr = {5, 10, 15, 20, 25};

int index = Arrays.binarySearch(arr, 15);

System.out.println("Index of 15: " + index);

}

}

🧾 Output:

Index of 15: 2

🔸 Note: binarySearch() **only works on sorted arrays  
  
✅ Guess the Output**

**1] Array Traversal**

int[] nums = {1, 2, 3, 4, 5};

int sum = 0;

for(int i = nums.length - 1; i >= 0; i -= 2) {

sum += nums[i];

}

System.out.println(sum);

**2] Array Default Values**

boolean[] flags = new boolean[3];

int [] marks = new int[10];

System.out.println(flags[1]);

System.out.println(marks [6]);

System.out.println(marks [10]);

**3] Array Modification**

int[] a = {2, 4, 6, 8};

for(int i = 0; i < a.length; i++) {

a[i] = a[i] + 1;

}

System.out.println(a[2]);

**4] Character Array**

char[] letters = {'J', 'a', 'v', 'a'};

System.out.println(letters[0] + letters[1]);

## **Assignments for this week:**

**1] Student Marks Analyzer**

You are given the marks of a student in 5 subjects. Write a program to calculate and print the average, highest, and lowest marks.

**Input:**int[] marks = {78, 85, 62, 90, 71};

**Expected Output:**

Average Marks: 77.2

Highest Marks: 90

Lowest Marks: 62   
  
**2] Reverse an Array (Useful in UI/UX Scrolling)**  
  
Reverse the order of items in an array, such as reversing the order of images or recent activity logs.  
  
**Input:**

int[] activityLog = {101, 102, 103, 104};

**Expected Output:**

Reversed Log: 104 103 102 101

**3] Website Visitor Tracker**

Track the number of visitors on a website for 7 days. Find on which days the traffic was below average.

**Input:**

int[] visitors = {300, 500, 400, 450, 600, 200, 350};

**Expected Output:**

Average Visitors: 400.0

Days with below average traffic: Day 1, Day 6, Day 7

**4] Score Improvement Tracker**

Given a student's test scores over time, determine how many times their score **increased compared to the previous test**.

**Input:**

int[] scores = {70, 75, 73, 80, 85, 82};

Expected Output:

Number of Improvements: 3

Explanation: Score improved from 70→75, 73→80, 80→85.

**5] Check if Array is Sorted in Ascending Order**

Write a method that checks whether the array is sorted in **ascending order**.

Input:

int[] arr = {3, 5, 8, 9, 12};

Expected Output:

Is Sorted: true

**6] Move All Zeros to End**

Move all 0s in an array to the end while maintaining the order of non-zero elements.

**Input:**

int[] arr = {0, 1, 0, 3, 12};

// start=0, end= arr.length-1 arr[start]==0

**Expected Output:**

Output: [1, 3, 12, 0, 0]

**7] Remove Duplicates from a Sorted Array**

Remove duplicates from a **sorted** array. Return the new length and array (ignore extra space after).

Input:

int[] arr = {1, 1, 2, 2, 3, 4, 4};

Expected Output:

Unique Length: 4

Modified Array: [1, 2, 3, 4]