**Java Array**

**Array** is a collection of elements of same type. **For example** an int array contains integer elements and a String array contains String elements. The elements of Array are stored in contiguous locations in the memory.

This is how an array looks like:

int number[] = new int[10]

Here number is the array name. The type of the array is integer, which means it can store integer values. The size of the array is 10.

Array works on an **index-based system**. In the above array, number[0] represents the first element of the array, number[1] represents the second element of the array and so on. The index of array starts from 0 and ends at array\_size-1. In the above example, the index of first element is 0 and index of 10th element is 9.

## Advantages of Array

**Better performance:** Since array works on a index based system, it is easier to search an element in the array, thus it gives better performance for various operations.

**Multidimensional:** Unlike ArrayList which is single dimensional, array are multidimensional such as [2D array](https://beginnersbook.com/2014/01/2d-arrays-in-c-example/), 3D array etc.

**Faster access:** Accessing an element is easy in array.

## Disadvantages of Array:

**Fixed Size:** The size of the array is fixed, which cannot be increased later.

**Allows only similar type elements:** Arrays are homogeneous, they don’t allow different type values, for example an int array cannot hold string elements, similarly a String array cannot hold integer elements.

Insertion and delegation requires shifting of elements.

## Declaration, Instantiation and Initialization of Array in Java

This is how we declare, instantiate and initialize an array.

int number[]; //array declaration

number[] = new int[10]; //array instantiation

number[0] = 10; //array Initialization

number[1] = 20; //array Initialization

We can also declare an array like this: All the **three following syntax are valid** for array declaration.

int[] number;

int []number;

int number[];

**Example:**  
The following example demonstrates, how we declared an int array, initialized it with integers and print the elements of the array using for loop.  
**Note:** You can see that we have used length property of array to find the size of the array. The length property of array returns the number of elements present in the array.

public class JavaExample{

public static void main(String args[]){

//array declaration, instantiation and initialization

int number[] = {11, 22, 33, 44, 55};

//print array elements

//length property return the size of the array

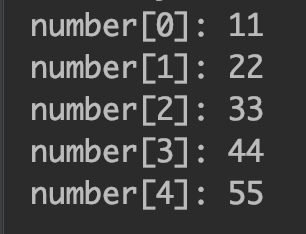
for(int i=0;i<number.length;i++)

System.out.println("number["+i+"]: "+number[i]);

}

}

**Output:**



## Types of array in Java

1. Single Dimensional Array  
2. Multidimensional Array

### 1. Single dimensional array

public class JavaExample{

public static void main(String args[]){

//array declaration

String names[] = new String[5];

//array initialization

names[0]="Chaitanya";

names[1]="Ajeet";

names[2]="Rahul";

names[3]="Shivam";

names[4]="Rohit";

//print array elements

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

System.out.println("names["+i+"]: "+names[i]);

}

}

**Output:**

names[0]: Chaitanya

names[1]: Ajeet

names[2]: Rahul

names[3]: Shivam

names[4]: Rohit

### 2. Multidimensional array

**Multidimensional array declaration:**  
This is how you can declare a multidimensional array: All the four syntax are valid multidimensional array declaration.

int[][] arr;

int [][]arr;

int arr[][];

int []arr[];

**Instantiate Multidimensional Array in Java**  
Number of elements in multidimensional array = number of rows\*number of columns.  
The following array can store upto 2\*3 = 6 elements.

int[][] arr=new int[2][3]; //2 rows and 3 columns

**Initialize Multidimensional Array in Java**

arr[0][0]=11;

arr[0][1]=22;

arr[0][2]=33;

arr[1][0]=44;

arr[1][1]=55;

arr[1][2]=66;

**Example:**

public class JavaExample{

public static void main(String args[]){

//two rows and three columns

int arr[][]={{11,22,33},{44,55,66}};

0 1 2

0 11 22 33

1 44 55 66

//outer loop 0 till number of rows

1

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

//inner loop from 0 till number of columns

1

for(int j=0;j<3;j++){

1 2

System.out.print(arr[i][j]+" "); 11 22 33

} 44 55 66

//new line after each row

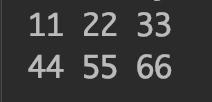
System.out.println();

}

}

}

**Output:**



## Print an Array elements using for-each loop

In the previous examples, we have seen how to print array elements using for loop. There is another way to print Array elements **without using array length property**.

public class JavaExample{

public static void main(String args[]){

//String array

String names[]={"Chaitanya", "Ajeet", "Rahul", "Hari"};

//print array elements using for-each loop

for(String str:names)

System.out.println(str);

//int array

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

//print array elements using for-each loop

for(int num:numbers)

System.out.println(num);

}

}

**Output:**

Chaitanya

Ajeet

Rahul

Hari

1

2

3

4

5

## Exception: ArrayIndexOutOfBoundsException

ArrayIndexOutOfBoundsException occurs when we access an array with an invalid index. This happens when the index is either negative or greater than or equal to the size of the array.

public class JavaExample{

public static void main(String args[]){

int number[]={1, 5, 7, 9, 11};

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

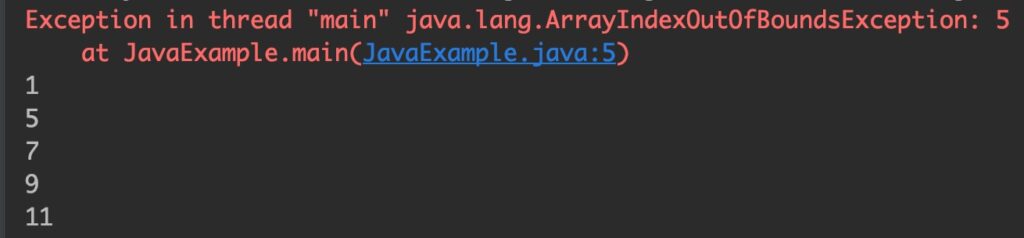
System.out.println(number[i]);

}

}

}

**Output:**



## Example: Program to print number of elements in an array

In this example, we have two arrays. We are finding the count of the elements in both the arrays. The steps to count the number of elements is same in any type of array, you can simply use .length property of the array to quickly find the size of the array.

public class JavaExample {

public static void main(String[] args) {

//Initializing an int array

int [] numbers = new int [] {2, 4, 6, 8, 10, 12};

// We can use length property of array to find the count of elements

System.out.println("Number of elements in the given int array: " + numbers.length);

//Initializing a String array

String [] names = new String [] {"Rick", "Luna", "Steve", "John"};

System.out.println("Number of elements in the given String array: " + names.length);

}

}

**Output:**

Number of elements in the given int array: 6

Number of elements in the given String array: 4

## Example 1: Program to remove duplicate elements from a sorted array

In this example, the given array is a [sorted array](https://beginnersbook.com/2018/10/java-program-to-sort-an-array-in-ascending-order/). To remove the duplicates from an unsorted array, refer the second example.

public class JavaExample{

public static int removeDuplicates(int arr[], int count){

if (count==0 || count==1){

return count;

}

// creating a temporary array to hold non-duplicate elements

int[] temp = new int[count];

int j = 0;

for (int i=0; i<count-1; i++){

if (arr[i] != arr[i+1]){

temp[j++] = arr[i];

}

}

temp[j++] = arr[count-1];

// copying the temp array to the original array

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

arr[i] = temp[i];

}

return j;

}

public static void main (String[] args) {

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

System.out.print("Original Array: ");

int length = arr.length;

for (int i=0; i<length; i++)

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

//getting the new array size after removing duplicates

length = removeDuplicates(arr, length);

//for new line

System.out.println("");

//Displaying array with non-duplicate elements

System.out.print("Array after removing duplicate elements: ");

for (int i=0; i<length; i++)

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

}

}

**Output:**

Original Array: 1 2 2 3 4 5 5 5

Array after removing duplicate elements: 1 2 3 4 5

## Example 2: Program to remove duplicate elements from an unsorted array

This program is same as the above program except that here the given array is **unsorted**. The workaround is pretty simple, we can import java.util.Arrays package and use the Arrays.sort() method to sort the unsorted array and then use the same logic that we have used above to remove duplicate elements.

import java.util.Arrays;

public class JavaExample{

public static int removeDuplicates(int arr[], int count){

if (count==0 || count==1){

return count;

}

// creating a temporary array to hold non-duplicate elements

int[] temp = new int[count];

int j = 0;

for (int i=0; i<count-1; i++){

if (arr[i] != arr[i+1]){

temp[j++] = arr[i];

}

}

temp[j++] = arr[count-1];

// copying the temp array to the original array

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

arr[i] = temp[i];

}

return j;

}

public static void main (String[] args) {

int arr[] = {3, 2, 1, 2, 9, 10, 4, 10, 9};

System.out.print("Original Array: ");

//Sorting the given unsorted array

Arrays.sort(arr);

int length = arr.length;

for (int i=0; i<length; i++)

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

//getting the new array size after removing duplicates

length = removeDuplicates(arr, length);

//for new line

System.out.println("");

//Displaying array with non-duplicate elements

System.out.print("Array after removing duplicate elements: ");

for (int i=0; i<length; i++)

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

}

}

**Output:**

Original Array: 1 2 2 3 4 9 9 10 10

Array after removing duplicate elements: 1 2 3 4 9 10

# Java Program to left rotate the elements of an array

For example, if an array is: {3, 5, 7, 9, 11} and we are left rotating it by 1 then the resulting array after left rotating would be: {5, 7, 9, 11, 3}.

The concept is simple, all the elements are **shifted to the left by the specified number**, in the above example all elements were shifted to the left by 1 position. If this number is 2, then the elements shift to left by two positions and first two elements would be shifted to the end of the array.

Let’s take few example to understand the concept and then we will write a program to implement this:

Array is {2, 4, 6, 8, 10, 12}

Left rotate by 2

After first rotation: {4, 6, 8, 10, 12, 2}

After second rotation: {6, 8, 10, 12, 2, 4}

Output array would be {6, 8, 10, 12, 2, 4}

Note that the first elements are shifted to the end of the array.

Left rotate the same array by 3

Array is {2, 4, 6, 8, 10, 12}

After first rotation: {4, 6, 8, 10, 12, 2}

After second rotation: {6, 8, 10, 12, 2, 4}

After third rotation: {8, 10, 12, 2, 4, 6}

Output array {8, 10, 12, 2, 4, 6}

## Program to left rotate the elements of a given array

In the following example, we have an array initialized and we have a **number n that represents the number of times the array needs to be left rotated**.

We run a loop from 0 till the number n (specifies the left rotation count) and at each iteration of this loop, we shift the element by one to the left.

Which means, the second element would move to 1st position, third element would move to 2nd position and so on. Before moving the second element to 1st position, **we store the 1st element in a different variable firstElement which we add at the end of the array**.

class JavaExample {

public static void main(String[] args) {

//Initializing the array numbers

int [] numbers = new int [] {2, 4, 6, 8, 10, 12};

/\* This number specifies how many times the array needs

\* to be rotated.

\*/

int n = 2;

System.out.println("Given array is: ");

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

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

}

//Left rotate the array by n times

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

int j, firstElement;

//Storing the first element of the array to move to last

firstElement = numbers[0];

for(j = 0; j < numbers.length-1; j++){

//Shifting the element to left by 1 on each iteration of loop

numbers[j] = numbers[j+1];

}

//Adding the first element at the end of the array

numbers[j] = firstElement;

}

System.out.println();

//Printing output array

System.out.println("Array after "+n+" left rotations: ");

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

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

}

}

}

**Output:**

Given array is:

2 4 6 8 10 12

Array after 2 left rotations:

6 8 10 12 2 4

# Java Program to right rotate the elements of an array

we will write a [**java program**](https://beginnersbook.com/2017/09/java-examples/)**to right rotate the elements of an array by a specified number**. For example, if we are right rotate an array {10, 20, 30, 40 ,50} by n =1 then the output array would look like this: {50, 10, 20, 30, 40}.

The concept is similar to the [left rotation](https://beginnersbook.com/2022/06/java-program-to-left-rotate-the-elements-of-an-array/), except that the elements are shifted to right in right rotation. All the elements are **shifted to the right by the specified number** n, in the above example all elements are shifted to the right by 1 position. If **n = 2**, then the elements shift to right by **two positions**and last two elements would be shifted to the beginning of the array.

Let’s take few example to understand the concept and then we will write a program to implement this:

Let's say an Array is {11, 22, 33, 44, 55, 66}

Right rotate by 2

After first rotation: {66, 11, 22, 33, 44, 55}

After second rotation: {55, 66, 11, 22, 33, 44}

Output array would be {55, 66, 11, 22, 33, 44}

Note that the last elements are shifted to the start of the array.

Let's Right rotate the same array by 3

Array is {11, 22, 33, 44, 55, 66}

After first rotation: {66, 11, 22, 33, 44, 55}

After second rotation: {55, 66, 11, 22, 33, 44}

After third rotation: {44, 55, 66, 11, 22, 33}

Output array would be {44, 55, 66, 11, 22, 33}

## Program to Right rotate the elements of a given array

In the following example, we have initialized an array and we have a **number n that represents the number of times the array should be right rotated**.

We run a loop from 0 till the number n (specifies the right rotation count) and at each iteration of this loop, we shift the element to the right by one position.

Which means, the first element would move to 2nd position, second element would move to 3rd position and so on. Before shifting any element, **we store the last element in a different variable lastElement which we add at the beginning of the array**.

class JavaExample {

public static void main(String[] args) {

//Initializing the array numbers

int [] numbers = new int [] {11, 22, 33, 44, 55, 66};

/\* This number specifies how many times the array needs

\* to be right rotated.

\*/

int n = 2;

System.out.println("Given array is: ");

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

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

}

//Right rotate the array by n times

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

int j, lastElement;

//Storing the last element of the array to move to start

lastElement = numbers[numbers.length-1];

for(j = numbers.length-1; j > 0; j--){

//Shifting the element to right by 1 on each iteration of loop

numbers[j] = numbers[j-1];

}

//Adding the last element at the beginning of the array

numbers[0] = lastElement;

}

System.out.println();

//Printing output array

System.out.println("Array after "+n+" right rotations: ");

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

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

}

}

}

**Output:**

Given array is:

11 22 33 44 55 66

Array after 2 right rotations:

55 66 11 22 33 44

# Java program to find smallest number in an array

In the following example, we have an array arr of numbers. We have copied the first element of this array into a variable smallestElement, this is to compare the other elements of the array to it. This is just the initialization of this variable, at the end of the program, this variable will hold the smallest number present in the array.

Running a loop from 0 till arr.length (this represents the size of the array). At every loop iteration, each element of the array is compared with the smallestElement, if any element is smaller than smallestElement, then the value of that element is assigned to smallestElement. This process continues until the whole array is traversed.

In the end, the variable smallestElement holds the smallest element of the given array.

public class JavaExample {

public static void main(String[] args) {

//Initializing an int array

int [] arr = new int [] {3, 8, 1, 12, 7, 99};

//This element will store the smallest element of the array

//Initializing with the first element of the array

int smallestElement = arr[0];

//Running the loop from first element till last element

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

//Compare each elements of array with smallestElement

//If an element is smaller, store the element into smallestElement

if(arr[i] < smallestElement)

smallestElement = arr[i];

}

System.out.println("Smallest element of given array: " + smallestElement);

}

}

**Output:**

Smallest element of given array: 1

## Program to print the largest element of a given array

In the following example, we have an int array arr. The first element of this given array is stored into an int variable largestElement. This is just the initialization of this variable, at the end of the program, this variable will hold the largest element of the array.

Running a loop from 0 till the size of the array. At every loop iteration, each element of the array is compared with the largestElement, if any of the element is found greater than largestElement, then the value of that element is assigned to largestElement. This process continues until the whole array is traversed.

In the end, the variable largestElement holds the largest element of the given array.

public class JavaExample {

public static void main(String[] args) {

//Initializing an int array

int [] arr = new int [] {11, 22, 33, 99, 88, 77};

//This element will store the largest element of the array

//Initializing with the first element of the array

int largestElement = arr[0];

//Running the loop from 1st element till last element

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

//Compare each elements of array with largestElement

//If an element is greater, store the element into largestElement

if(arr[i] > largestElement)

largestElement = arr[i];

}

System.out.println("Largest element of given array: " + largestElement);

}

}

**Output:**

Largest element of given array: 99

# Java Program to count the frequency of each element in array

For example, if an array is {2, 2, 3, 4, 3, 4, 2} then the frequency of element “2” is 3, frequency of element “3” is 2 and frequency of element “4” is 2.

## Java Program to find the frequency of each element in the array

In the following example, we have an array numbers. In this array there are elements that reappeared again and we need to **count the frequency of each element**.

We ran a [nested for loop](https://beginnersbook.com/2015/03/for-loop-in-java-with-example/)(loop inside loop), in such a way that the each element of an array is compared to the all the elements of the same array, and every time there is a match, we increase the value of the count variable that counts the frequency.

Since, elements appeared multiple times, we need to avoid counting the frequency of same element again. For this, we have assigned a -1 value to the already counted elements using counted variable.

In the end, elements and their frequency in the array is displayed, here also we are using the counted variable to avoid printing the frequency of same element again.

public class JavaExample {

public static void main(String[] args) {

//Initializing an array

int [] numbers = new int [] {2, 2, 3, 4, 5, 5, 5, 3, 2, 4};

//This array will store the frequency of each element

int [] frequency = new int [numbers.length];

int counted = -1;

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

int count = 1;

for(int j = i+1; j < numbers.length; j++){

if(numbers[i] == numbers[j]){

count++;

//To avoid counting the frequency of same element again

frequency[j] = counted;

}

}

if(frequency[i] != counted)

frequency[i] = count;

}

//Printing the frequency of each element

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

if(frequency[i] != counted)

System.out.println("Element: "+numbers[i] + " Frequency: " + frequency[i]);

}

}}

**Output:**

Element: 2 Frequency: 3

Element: 3 Frequency: 2

Element: 4 Frequency: 2

Element: 5 Frequency: 3

## Program to print duplicate elements of the given int array

For example, if an array is {1, 2, 3, 2, 3, 5} then the program should print the elements {2, 3} as these elements appeared in the **array more than once**.

In this program, we are running a nested for loop, in such a way that each element of the given array is compared with all other elements of the same array and if there is a match found, print that element.

To compare the elements we are using equal to == operator as the given array is an int type array. If you want to find out the duplicate elements in a String array, refer the next program.

public class JavaExample {

public static void main(String[] args) {

//Initializing an int array

int [] numbers = new int [] {2, 4, 6, 8, 4, 6, 10, 10};

System.out.println("Duplicate elements in given array are: ");

//Comparing each element of the array with all other elements

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

for(int j = i + 1; j < numbers.length; j++) {

if(numbers[i] == numbers[j]) {

//printing duplicate elements

System.out.println(numbers[j]);

}

}

}

}

}

**Output:**

Duplicate elements in given array are:

4

6

10

## Program to print duplicate elements of the String array

Here, we are displaying the duplicate elements of a String array. In the following example, we have an array that contains strings as elements. The logic we are using to print the duplicate elements is same as the above example, except that here we are using [equals() method](https://beginnersbook.com/2013/12/java-string-equals-and-equalsignorecase-methods-example/) to compare elements as the elements are Strings.

public class JavaExample {

public static void main(String[] args) {

//Initializing an int array

String [] names = new String [] {"Tom", "Steve", "Rick", "Steve", "Rick"};

System.out.println("Duplicate elements in the given array: ");

//Comparing each element of the array with all other elements

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

for(int j = i + 1; j < names.length; j++) {

if(names[i].equals(names[j])){

//printing duplicate elements

System.out.println(names[j]);

}

}

}

}

}

**Output:**

Duplicate elements in the given array:

Steve

Rick

# Java Program to copy all elements of one array into another array

 This can be easily done by using any loop such as for, while or do-while loop. We just need to run a loop from 0 to the array length (size of an array) and at every iteration, read the element of the given array and write the same value in another array.

For example:

array1 is {3, 5, 7, 9}

array2[] is {}

Then after copying all the elements of array1[] to array2[]

the elements in array2[] will be = {3, 5, 7, 9}

## Steps to copy all elements from one array to another array

1. Initialize the first array.
2. Create another array with the same size as of the first array
3. Run a loop from 0 till the length of first array
   * Read the element of first array
   * Copy the element to the second array
4. Repeat the step 3 until the complete array is traversed
5. Run a loop from 0 to the length of any array
   * Read and print the elements of second array
6. Repeat the step 5 until the second array is completely traversed.
7. End of Program.

## Java Program to copy all elements of one array into another array

public class JavaExample {

public static void main(String[] args) {

//Initializing an array

int [] firstArray = new int [] {3, 5, 7, 9, 11};

/\* Creating another array secondArray with same size

\* of first array using firstArray.length as it returns

\* the size of array firstArray.

\*/

int secondArray[] = new int[firstArray.length];

//Displaying elements of first array

System.out.println("Elements of First array: ");

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

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

}

//Copying all elements of firstArray to secondArray

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

secondArray[i] = firstArray[i];

}

//Displaying elements of secondArray

System.out.println();

System.out.println("Elements of Copied array: ");

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

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

}

}

}

**Output:**

Elements of First array:

3 5 7 9 11

Elements of Copied array:

3 5 7 9 11

# Java Program to print the elements of an array present on odd position

Here, we will write a [java program](https://beginnersbook.com/2017/09/java-examples/) to **print the elements of an array present on odd position**. For example, if an array is {2, 12, 23, 7, 6, 15} then we need to display the elements 2, 23 and 6 as they are present in the array on **odd positions**.

## Steps to find even position elements of an Array

1. Initialize the array.
2. Run a loop starting from 0 till the length of the given array. Use +2 in the increment part of the loop, to only traverse the odd positions in the array.
   * Print the current element of the array
3. End of the program.

## Program to display array elements on even positions

Please go through the comments mentioned in the following program to understand why we are starting the [loop](https://beginnersbook.com/2015/03/for-loop-in-java-with-example/) from 0 and why are using i=i+2 in the increment part of the loop.

public class JavaExample {

public static void main(String[] args) {

//Initializing the array

int [] numbers = new int [] {1, 3, 5, 7, 9, 11, 13};

System.out.println("Array Elements on odd Positions: ");

/\* Note we are using i = i+2 as we are only traversing odd positions

\* Important point here is that the array indices start with 0, which

\* means the odd positions such as 1st, 3rd and 5th positions are having

\* indices 0, 2, 4 and so on. That's why numbers[0] prints 1st position

\* element of the array.

\*/

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

System.out.println(numbers[i]);

}

}

}

**Output:**

Array Elements on odd Positions:

1

5

9

13

# Java Program to print the elements of an array present on even position

BY CHAITANYA SINGH | FILED UNDER: [JAVA EXAMPLES](https://beginnersbook.com/category/java-examples/)

Here, we will write a [java program](https://beginnersbook.com/2017/09/java-examples/) to print the elements of an array present on even position. For example, if an array is {1, 4, 8, 3, 9, 17} then we need to display the elements 4, 3 and 17 as they are present in the array on even positions.

## Steps to find even position elements of an Array

1. Initialize the array.
2. Run a loop starting from 1 till the length of the given array. Use +2 in the increment part of the loop, to only traverse the even positions in the array.
3. Display current array element
4. End of the program.

## Program to display array elements on even positions

Please go through the comments mentioned in the following program to understand why we are starting the [loop](https://beginnersbook.com/2015/03/for-loop-in-java-with-example/) from 1 and why are using i=i+2 in the increment part of the loop.

public class JavaExample {

public static void main(String[] args) {

//Initializing the array

int [] numbers = new int [] {5, 12, 16, 3, 9, 7, 1, 100};

System.out.println("Array Elements on even Positions: ");

/\* Note we are using i = i+2 as we are only traversing even positions

\* Important point here is that the array indices start with 0, which

\* means the even positions such as 2nd, 4th and 6th positions are having

\* indices 1, 3, 5 and so on. That's why numbers[1] prints 2nd position

\* element of the array.

\*/

for (int i = 1; i < numbers.length; i = i+2) {

System.out.println(numbers[i]);

}

}

}

**Output:**

Array Elements on even Positions:

12

3

7

100

# Java Program to Add Two Matrix using Multi-dimensional Arrays

BY CHAITANYA SINGH | FILED UNDER: [JAVA EXAMPLES](https://beginnersbook.com/category/java-examples/)

Here we will write a **java program to add two given matrices** and display the output matrix that will be the sum of given matrices.

## Example: Java Program to add two given Matrices

In the following example we have two matrices MatrixA and MatrixB, we have declared these matrices as multi-dimensional arrays.

**Two matrices can only be added** or subtracted only if they have same dimension which means they must have the **same number of rows and columns**. Here we have two MatrixA and MatrixB which have same rows and columns. The addition of these matrices will have same rows and columns.

This is how we declare a matrix as multi-dimensional array:  
**Matrix:** This matrix has two rows and four columns.

| 1 1 1 1 |

| 2 3 5 2 |

**The declaration of this matrix as 2D array:**

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

We are using [for loop](https://beginnersbook.com/2015/03/for-loop-in-java-with-example/) to add the corresponding elements of both the matrices and store the addition values in sum matrix. For example: sum[0][0] = MatrixA[0][0] + MatrixB[0][0], similarly sum[0][1] = MatrixA[0][1] + MatrixB[0][1] and so on.

public class JavaExample {

public static void main(String[] args) {

int rows = 2, columns = 4;

// Declaring the two matrices as multi-dimensional arrays

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

int[][] MatrixB = { {2, 3, 4, 5}, {2, 2, 4, -4} };

/\* Declaring a matrix sum, that will be the sum of MatrixA

\* and MatrixB, the sum matrix will have the same rows and

\* columns as the given matrices.

\*/

int[][] sum = new int[rows][columns];

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

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

sum[i][j] = MatrixA[i][j] + MatrixB[i][j];

}

}

// Displaying the sum matrix

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

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

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

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

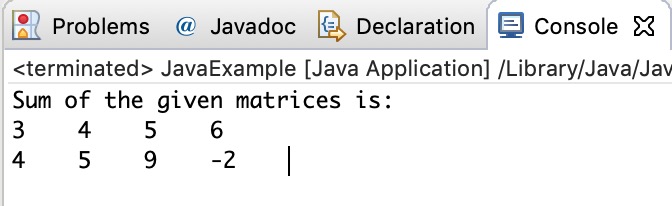
}

System.out.println();

}

}

}

**Output:**  


# Java program to check whether two matrices are equal

Here we will write a **java program to determine whether two matrices are equal or not**. Two matrices are said to be equal if the dimensions (row and column count) of both the matrices are same and all the corresponding elements are equal. For example if two matrices m1 & m2 are equal then dimensions of m1 & m2 should be same and all the elements should match, m1[0][0] = m2[0][0], m1[0][1] = m2[0][1] and so on.

## Program to determine whether two matrices are equal

Logic that we are following in the program below is:

if row and column count of m1 & m2 doesn’t match, no need to do further processing as we can simply say that matrices are not equal.

if row and column count matches then we are checking each element one by one and if any of the element doesn’t match we are storing the boolean value “false” in the boolean variable flag. Later we are displaying “Given matrices are not equal” if the flag value is false.

public class JavaExample

{

public static void main(String[] args) {

int rowCount1, colCount1, rowCount2, colCount2;

boolean flag = true;

//Input matrix m1

int m1[][] = {

{19, 8, 17},

{6, 14, 16},

{4, 15, 17}

};

//Input matrix m2

int m2[][] = {

{19, 8, 17},

{6, 14, 16},

{4, 15, 17}

};

// Finding row and column count for matrix m1

rowCount1 = m1.length;

colCount1 = m1[0].length;

// Finding row and column count for matrix m2

rowCount2 = m2.length;

colCount2 = m2[0].length;

//Checking if size of both the matrices m1 & m2are equal

if(rowCount1 != rowCount2 || colCount1 != colCount2){

System.out.println("Given matrices are not equal");

}

else {

// Here we are matching each element of m1 & m2 if

// any element doesn't match we are setting the flag to false

for(int i = 0; i < rowCount1; i++){

for(int j = 0; j < colCount1; j++){

if(m1[i][j] != m2[i][j]){

flag = false;

break;

}

}

}

//If flag is not true it means the matrices are not equal

if(flag)

System.out.println("Given matrices are equal");

else

System.out.println("Given matrices are not equal");

}

}

}

**Output:**

Given matrices are equal

# Java Program to reverse the Array

This program reverse the array. For example if user enters the array elements as 1, 2, 3, 4, 5 then the program would reverse the array and the elements of array would be 5, 4, 3, 2, 1. To understand this program, you should have the knowledge of following [Java Programming](https://beginnersbook.com/java-tutorial-for-beginners-with-examples/) topics:

1. [Arrays in Java](https://beginnersbook.com/2013/05/java-arrays/)
2. [Java For loop](https://beginnersbook.com/2015/03/for-loop-in-java-with-example/)
3. [Java While loop](https://beginnersbook.com/2015/03/while-loop-in-java-with-examples/)

## Example: Program to reverse the array

import java.util.Scanner;

public class Example

{

public static void main(String args[])

{

int counter, i=0, j=0, temp;

int number[] = new int[100];

Scanner scanner = new Scanner(System.in);

System.out.print("How many elements you want to enter: ");

counter = scanner.nextInt();

/\* This loop stores all the elements that we enter in an

\* the array number. First element is at number[0], second at

\* number[1] and so on

\*/

for(i=0; i<counter; i++)

{

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

number[i] = scanner.nextInt();

}

/\* Here we are writing the logic to swap first element with

\* last element, second last element with second element and

\* so on. On the first iteration of while loop i is the index

\* of first element and j is the index of last. On the second

\* iteration i is the index of second and j is the index of

\* second last.

\*/

j = i - 1;

i = 0;

scanner.close();

while(i<j)

{

temp = number[i];

number[i] = number[j];

number[j] = temp;

i++;

j--;

}

System.out.print("Reversed array: ");

for(i=0; i<counter; i++)

{

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

}

}

}

Output:

How many elements you want to enter: 5

Enter Array Element1: 11

Enter Array Element2: 22

Enter Array Element3: 33

Enter Array Element4: 44

Enter Array Element5: 55

Reversed array: 55 44 33 22 11

# Random shuffling of an array in Java

BY CHAITANYA SINGH | FILED UNDER: [JAVA](https://beginnersbook.com/category/learn-java/)

import java.util.Arrays;

import java.util.Collections;

import java.util.List;

class ShuffleArrayExample {

public static void main(String[] args) {

// String Array

String[] stringArray =

new String[] { "FF", "PP", "AA", "OO", "DD" };

// converting array to a List

List<String> list = Arrays.asList(stringArray);

// Shuffling list elements

Collections.shuffle(list);

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

for (String str : list) {

System.out.println(str);

}

Integer[] intArray = new Integer[]{11, 22, 33, 44, 55};

/\* Rememeber: If you define the array like this:

\* int[] array = new int[]{1, 2, 3, 4}; then the below

\* method won't work as it doesn't work for primitive types

\*/

List<Integer> list2 = Arrays.asList(intArray);

// Shuffling list elements

Collections.shuffle(list2);

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

for (int num : list2) {

System.out.println(num);

}

}

}

**Output:**

String Array:

FF

OO

PP

DD

AA

Int Array:

33

55

11

22

44

# Java Program to Sort an Array in Ascending Order

BY CHAITANYA SINGH | FILED UNDER: [JAVA EXAMPLES](https://beginnersbook.com/category/java-examples/)

In this java [tutorial](https://beginnersbook.com/java-tutorial-for-beginners-with-examples/), we are sorting an array in ascending order using temporary variable and nested for loop. We are using Scanner class to get the input from user.

## Java Example: Program to Sort an Array in Ascending Order

In this program, user is asked to enter the number of elements that he wish to enter. Based on the input we have declared an int array and then we are accepting all the [numbers input by user](https://beginnersbook.com/2014/07/java-program-to-get-input-from-user/) and storing them in the array.

Once we have all the numbers stored in the array, we are sorting them using [nested for loop](https://beginnersbook.com/2015/03/for-loop-in-java-with-example/).

import java.util.Scanner;

public class JavaExample

{

public static void main(String[] args)

{

int count, temp;

//User inputs the array size

Scanner scan = new Scanner(System.in);

System.out.print("Enter number of elements you want in the array: ");

count = scan.nextInt();

int num[] = new int[count];

System.out.println("Enter array elements:");

for (int i = 0; i < count; i++)

{

num[i] = scan.nextInt();

}

scan.close();

for (int i = 0; i < count; i++)

{

for (int j = i + 1; j < count; j++) {

if (num[i] > num[j])

{

temp = num[i];

num[i] = num[j];

num[j] = temp;

}

}

}

System.out.print("Array Elements in Ascending Order: ");

for (int i = 0; i < count - 1; i++)

{

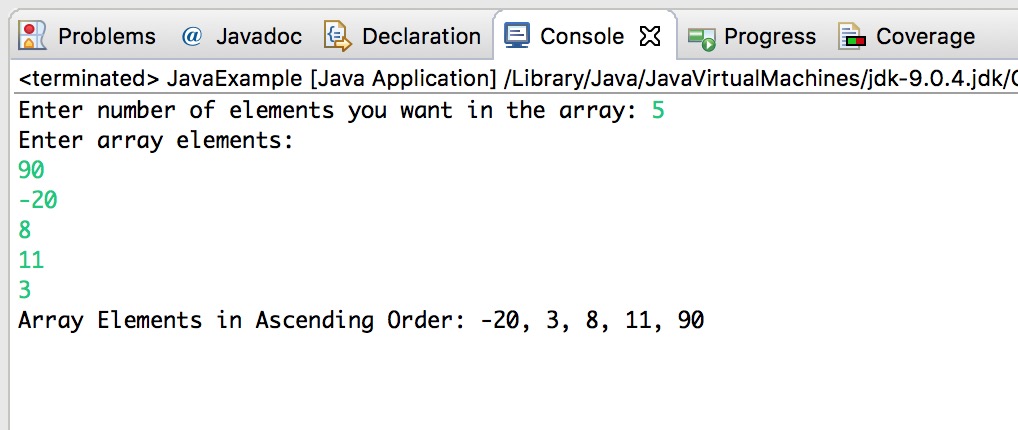
System.out.print(num[i] + ", ");

}

System.out.print(num[count - 1]);

}

}

Output:  


# Java Program to Calculate average using Array

BY CHAITANYA SINGH | FILED UNDER: [JAVA EXAMPLES](https://beginnersbook.com/category/java-examples/)

We will see two programs to find the average of numbers using array. First Program finds the average of specified array elements. The second programs takes the value of n (number of elements) and the numbers provided by user and finds the average of them using array.

To understand these programs you should have the knowledge of following [Java Programming](https://beginnersbook.com/java-tutorial-for-beginners-with-examples/) concepts:  
1) [Java Arrays](https://beginnersbook.com/2013/05/java-arrays/)  
2) [For loop](https://beginnersbook.com/2015/03/for-loop-in-java-with-example/)

## Example 1: Program to find the average of numbers using array

public class JavaExample {

public static void main(String[] args) {

double[] arr = {19, 12.89, 16.5, 200, 13.7};

double total = 0;

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

total = total + arr[i];

}

/\* arr.length returns the number of elements

\* present in the array

\*/

double average = total / arr.length;

/\* This is used for displaying the formatted output

\* if you give %.4f then the output would have 4 digits

\* after decimal point.

\*/

System.out.format("The average is: %.3f", average);

}

}

Output:

The average is: 52.418

## Example 2: Calculate average of numbers entered by user

In this example, we are using Scanner to get the value of n and all the numbers from user.

import java.util.Scanner;

public class JavaExample {

public static void main(String[] args) {

System.out.println("How many numbers you want to enter?");

Scanner scanner = new Scanner(System.in);

int n = scanner.nextInt();

/\* Declaring array of n elements, the value

\* of n is provided by the user

\*/

double[] arr = new double[n];

double total = 0;

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

System.out.print("Enter Element No."+(i+1)+": ");

arr[i] = scanner.nextDouble();

}

scanner.close();

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

total = total + arr[i];

}

double average = total / arr.length;

System.out.format("The average is: %.3f", average);

}

}

**Output:**

How many numbers you want to enter?

5

Enter Element No.1: 12.7

Enter Element No.2: 18.9

Enter Element No.3: 20

Enter Element No.4: 13.923

Enter Element No.5: 15.6

The average is: 16.225

# Sorting char array in Java example

BY CHAITANYA SINGH | FILED UNDER: [JAVA](https://beginnersbook.com/category/learn-java/)

In this example we are sorting a char array. We have demonstrated two types of sorting in the program 1) Complete sorting using [sort(char[] a)](http://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html#sort(char[])) method 2) Sorting specified range of characters only using [sort(char[] a, int fromIndex, int toIndex)](http://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html#sort(char[],%20int,%20int)) method.

import java.util.Arrays;

class SortCharArray {

public static void main(String[] args) {

// Creating a Char Array

char[] charArray = new char[] { 'A', 'Q', 'S', 'Z', 'P' };

// Displaying Array before Sorting

System.out.println("\*\*Char Array Before Sorting\*\*");

for (char ch: charArray){

System.out.println(ch);

}

// Sorting the Array

Arrays.sort(charArray);

System.out.println("\*\*Char Array After Sorting\*\*");

for (char ch: charArray){

System.out.println(ch);

}

// Another Char Array

char[] charArray2 =

new char[] { 'D', 'F', 'V', 'J', 'U', 'M', 'C' };

// Selective Sorting

/\* public static void sort(char[] a, int fromIndex,

\* int toIndex): Sorts the specified range of the

\* array into ascending order. The range to be sorted

\* extends from the index fromIndex, inclusive, to the

\* index toIndex, exclusive. If fromIndex == toIndex,

\* the range to be sorted is empty.

\*/

Arrays.sort(charArray2, 2, 5);

// Displaying array after selective sorting

System.out.println("\*\*Selective Sorting\*\*");

for (char ch: charArray2){

System.out.println(ch);

}

}

}

**Output:**

\*\*Char Array Before Sorting\*\*

A

Q

S

Z

P

\*\*Char Array After Sorting\*\*

A

P

Q

S

Z

\*\*Selective Sorting\*\*

D

F

J

U

V

M

C

**Matrix Multiplication**

import [java](https://ecomputernotes.com/java/what-is-java/what-is-java-explain-basic-features-of-java-language).io.\*;   
 class MultiplicationofMatrix   
 {   
               
             public static void main(String args[]) throws IOException   
             {   
                         BufferedReader bf=new BufferedReader(new InputStreamReader(System.in));   
                         int m1[][]=new int[2][4];   
                         int m2[][]=new int[4][3];   
                         int m3[][]=new int[2][3];   
                         int i,j,k;   
                         String m;   
                         System.out.println("Enter elements of first matrix of order 2 x 4");   
                         for(i=0;i<=1;i++)   
                         {   
                                     for(j=0;j<=3;j++)   
                                     {   
                                                 m=bf.readLine();   
                                                 m1[i][j]=Integer.parseInt(m);   
                                     }   
                         }   
                         System.out.println("Enter elements of second matrix of order 4 x 3");   
                         for(i=0;i<=3;i++)   
                         {   
                                     for(j=0;j<=2;j++)   
                                     {   
                                                 m=bf.readLine();   
                                                 m2[i][j]=Integer.parseInt(m);   
                                     }   
                         }   
                         for(i=0;i<=1;i++)   
                         {   
                                     for(j=0;j<=2;j++)   
                                     {   
                                                 m3[i][j]=0;   
                                                 for(k=0;k<=3;k++)   
                                                 {   
                                                             m3[i][j]=m3[i][j]+m1[i][k]\*m2[k][j];   
                                                 }                        
                                     }   
                         }   
                         System.out.println("The first matrix entered is ");   
                         for(i=0;i<=1;i++)   
                         {   
                                     for(j=0;j<=3;j++)   
                                     {   
                                                 System.out.print(m1[i][j]+"\t");   
                                     }   
                                     System.out.println();   
                         }   
                         System.out.println("The second matrix entered is ");   
                         for(i=0;i<=3;i++)   
                         {   
                                     for(j=0;j<=2;j++)   
                                     {   
                                                 System.out.print(m2[i][j]+"\t");   
                                     }   
                                     System.out.println();   
                         }   
                         System.out.println("The multiplication of matrix is ");   
                         for(i=0;i<=1;i++)   
                         {   
                                     for(j=0;j<=2;j++)   
                                     {   
                                                 System.out.print(m3[i][j]+"\t");   
                                     }   
                                     System.out.println();   
                         }   
             }   
 }

# 2×4 Matrix

import [java](https://ecomputernotes.com/java/what-is-java/what-is-java-explain-basic-features-of-java-language).io.\*;   
 class Matrix2x4   
 {   
             public static void main(String args[])   
             throws IOException   
             {   
                         BufferedReader k=new BufferedReader(new InputStreamReader(System.in));   
                         int m[][]=new int[2][4];   
                         int i,j;   
                         String s;   
                         System.out.println("Enter Elements of a Matrix of order 2 x 4");   
                         for(i=0;i<=1;i++)   
                         {   
                                     for(j=0;j<=3;j++)   
                                     {   
                                                 s=k.readLine();   
                                                 m[i][j]=Integer.parseInt(s);   
                                     }   
                         }   
                         System.out.println("The Original Matrix entered is ");   
                         for(i=0;i<=1;i++)   
                         {   
                                     for(j=0;j<=3;j++)   
                                     {   
                                                 System.out.print(m[i][j]+"\t");   
                                     }   
                                     System.out.println();   
                         }   
                         System.out.println("The Transpose of the Matrix is ");   
                         for(i=0;i<=3;i++)   
                         {   
                                     for(j=0;j<=1;j++)   
                                     {   
                                                 System.out.print(m[j][i]+"\t");   
                                     }   
                                     System.out.println();   
                         }   
             }   
 }

# 3×3 Matrix

import [java](https://ecomputernotes.com/java/what-is-java/what-is-java-explain-basic-features-of-java-language).io.\*;  
class Matrix3x3  
{  
    public static void main(String args[]) throws IOException  
    {  
        BufferedReader BR=new BufferedReader(new InputStreamReader (System.in));  
        int Number[][]=new int[3][3];  
        int i,j;  
        String m;  
        System.out.println("Enter Elements for Matrix 3x3 :");  
        for(i=0;i<=2;i++)  
        {  
            for(j=0;j<=2;j++)  
            {  
                m=BR.readLine();  
                Number[i][j]=Integer.parseInt(m);  
            }  
        }  
        System.out.println("Elements in Matrix are : ");  
                                System.out.println("");  
        for(i=0;i<=2;i++)  
        {  
            for(j=0;j<=2;j++)  
            {  
                System.out.print(Number[i][j]+"\t");  
            }  
            System.out.println();  
        }  
    }  
}

# nxn Matrix

import [java](https://ecomputernotes.com/java/what-is-java/what-is-java-explain-basic-features-of-java-language).io.\*;   
 class NxNMatrix   
 {   
             public static void main(String args[]) throws IOException   
             {   
                         BufferedReader gg=new BufferedReader(new InputStreamReader(System.in));   
                         int i,j,r,c;   
                         String m;   
                         System.out.print("How many Rows ? ");   
                         m=gg.readLine();   
                         r=Integer.parseInt(m);   
                         System.out.print("How many Columns ? ");   
                         m=gg.readLine();   
                         c=Integer.parseInt(m);   
                         int h[] []=new int[r] ;   
                         System.out.println("Enter elements for matrix of order "+r+" x "+c);   
                         for(i=0;i<=r-1;i++)   
                         {   
                                     for(j=0;j<=c-1;j++)   
                                     {   
                                                 m=gg.readLine();   
                                                 h[i] [j]=Integer.parseInt(m);   
                                     }   
                         }   
                         System.out.println("Elements in matrix are ");   
                         for(i=0;i<=r-1;i++)   
                         {   
                                     for(j=0;j<=c-1;j++)   
                                     {   
                                                 System.out.print(h[i][j]+"\t");   
                                     }   
                                     System.out.println();   
                         }   
             }   
 }

# Transpose of a Matrix

**Algorithm for Transpose of a Matrix:**

step 1: read r, c

step 2: create an integer array a[][] of size r,c

step 3: create an integer array b[][] of size c,r

step 4: initialize i=0

step 5: repeat through step-10 while (i < r)

step 6: initialize j=0

step 7: repeat through step-9 while (j < c)

step 8: read a[i][j]

step 9: j=j+ 1

step 10: i=i+ 1

step 11: reset i=0

step 12: repeat through step-17 while (i < r)

step 13: reset j=0

step 14: repeat through step-16 while (j < c)

step 15: b[j][i]=a[i][j]

step 16: j=j+ 1

step 17: i=i+ 1

step 18: initialize i=0

step 19: repeat through step-25 while (i < c)

step 20: move to new line

step 21: initialize j=0

step 22: repeat through step-24 while (j < r)

step 23: print b[i][j]

step 24: j=j+ 1

step 25: i=i+ 1

step 26: Exit

**Here is the**[**Java**](https://ecomputernotes.com/java/what-is-java/what-is-java-explain-basic-features-of-java-language)**Example for Transpose of a Matrix:**

import [java](https://ecomputernotes.com/java/what-is-java/what-is-java-explain-basic-features-of-java-language).util.Scanner;  
 class TransposeMatrix  
 {  
             public static void main(String args[])  
        {  
             int i,j,t,r,c ;  
             Scanner s1=new Scanner(System.in);  
             System.out.print("Enter the size of Rows : ");  
             r=s1.nextInt();  
             System.out.println("Enter the size of Columns :");  
             c=s1.nextInt();  
             System.out.println("\nThe Elements into the Array:->");  
             int a[][]=new int[r];  
             int b[][]=new int[r];  
             for(i=0;i<r;i++)  
             for(j=0;j<c;j++)  
             a[i][j]=s1.nextInt();  
             System.out.println("\nThe elements of Array\n");  
             for(i=0;i<r;i++)  
                 {  
                    System.out.print("\n");  
             for(j=0;j<c;j++)  
                 {  
                    System.out.print(a[i][j]+"\t");  
                 }  
                    for(i=0;i<r;i++)  
                   {  
                    for(j=0;j<c;j++)  
                       {  
                          b[j][i]=a[i][j];  
                       }  
                    }  
             System.out.println("\nAfter Transfering print the data :->");  
                     for(i=0;i<c;i++)  
                        {  
                           System.out.print("\n");  
                           for(j=0;j<r;j++)  
                               {  
                                 System.out.print(b[i][j]+"\t");  
                                }  
                         }  
                     }  
        }  
 }

# https://www.javatpoint.com/array-in-java