**Java Arrays**

Normally, an array is a collection of similar type of elements which have a contiguous memory location.

Array in Java is **index-based**, the first element of the array is stored at **the 0th index**, 2nd element is stored on **1st index** and so on.

Unlike C/C++, we can get the **length** of the array using the **length member**. In C/C++, we need to use the **sizeof** operator.

In Java, array is an object of a dynamically generated class. Java array inherits the Object class, and implements the Serializable as well as Cloneable interfaces. We can store primitive values or objects in an array in Java. Like C/C++, we can also create **single dimentional** or **multidimentional** arrays in Java.

Moreover, Java provides the feature of **anonymous** **arrays** which is not available in C/C++.



### Advantages

* **Code Optimization:** It makes the code optimized, we can retrieve or sort the data efficiently.
* **Random access:** We can get any data located at an index position.

### Disadvantages

* **Size Limit:** We can store only the **fixed size of elements** in the array. It doesn't grow its size at runtime. To solve this problem, **collection framework** is used in Java which **grows automatically**.

### Types of Array in java

There are two types of array.

* Single Dimensional Array
* Multidimensional Array

## Single Dimensional Array in Java

**Syntax to Declare an Array in Java**

1. dataType[] arr; (or)
2. dataType []arr; (or)
3. dataType arr[];

**Instantiation of an Array in Java**

1. arrayRefVar=new datatype[size];

### Example of Java Array

Let's see the simple example of java array, where we are going to declare, instantiate, initialize and traverse an array.

1. //Java Program to illustrate how to declare, instantiate, initialize
2. //and traverse the Java array.
3. class Testarray{
4. public static void main(String args[]){
5. int a[]=new int[5];//declaration and instantiation
6. a[0]=10;//initialization
7. a[1]=20;
8. a[2]=70;
9. a[3]=40;
10. a[4]=50;
11. //traversing array
12. for(int i=0;i<a.length;i++)//length is the property of array
13. System.out.println(a[i]);
14. }}
15. Output:
16. 10
17. 20
18. 70
19. 40
20. 50

## Declaration, Instantiation and Initialization of Java Array

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

1. int a[]={33,3,4,5};//declaration, instantiation and initialization

Let's see the simple example to print this array.

1. //Java Program to illustrate the use of declaration, instantiation
2. //and initialization of Java array in a single line
3. class Testarray1{
4. public static void main(String args[]){
5. int a[]={33,3,4,5};//declaration, instantiation and initialization
6. //printing array
7. for(int i=0;i<a.length;i++)//length is the property of array
8. System.out.println(a[i]);
9. }}
10. Output:
11. 33
12. 3
13. 4
14. 5

## For-each Loop for Java Array

We can also print the Java array using [**for-each loop**](https://www.javatpoint.com/for-each-loop). The Java for-each loop prints the array elements one by one. It holds an array element in a variable, then executes the body of the loop.

The syntax of the for-each loop is given below:

1. for(data\_type variable:array){
2. //body of the loop
3. }

Let us see the example of print the elements of Java array using the for-each loop.

1. //Java Program to print the array elements using for-each loop
2. class Testarray1{
3. public static void main(String args[]){
4. int arr[]={33,3,4,5};
5. //printing array using for-each loop
6. for(int i:arr)
7. System.out.println(i);
8. }}

Output:

33

3

4

5

## Passing Array to a Method in Java

We can pass the java array to method so that we can reuse the same logic on any array.

Let's see the simple example to get the minimum number of an array using a method.

1. //Java Program to demonstrate the way of passing an array
2. //to method.
3. class Testarray2{
4. //creating a method which receives an array as a parameter
5. static void min(int arr[]){
6. int min=arr[0];
7. for(int i=1;i<arr.length;i++)
8. if(min>arr[i])
9. min=arr[i];
11. System.out.println(min);
12. }
14. public static void main(String args[]){
15. int a[]={33,3,4,5};//declaring and initializing an array
16. min(a);//passing array to method
17. }}

## Anonymous Array in Java

Java supports the feature of an anonymous array, so you don't need to **declare** the array while **passing** an array to the **method**.

1. //Java Program to demonstrate the way of passing an anonymous array
2. //to method.
3. public class TestAnonymousArray{
4. //creating a method which receives an array as a parameter
5. static void **printArray**(int arr[]){
6. for(int i=0;i<arr.length;i++)
7. System.out.println(arr[i]);
8. }
10. public static void main(String args[]){
11. printArray(new int[]{10,22,44,66});//passing anonymous array to method
12. }}
13. Output:
14. 10
15. 22
16. 44
17. 66

## Returning Array from the Method

We can also return an array from the method in Java.

1. //Java Program to return an array from the method
2. class TestReturnArray{
3. //creating method which returns an array
4. static int[] get(){
5. return new int[]{10,30,50,90,60};
6. }
8. public static void main(String args[]){
9. //calling method which returns an array
10. int arr[]=get();
11. //printing the values of an array
12. for(int i=0;i<arr.length;i++)
13. System.out.println(arr[i]);
14. }}
15. Output:
16. 10
17. 30
18. 50
19. 90

## ArrayIndexOutOfBoundsException

The Java Virtual Machine (JVM) throws an ArrayIndexOutOfBoundsException if length of the array in negative, equal to the array size or greater than the array size while traversing the array.

1. //Java Program to demonstrate the case of
2. //ArrayIndexOutOfBoundsException in a Java Array.
3. public class TestArrayException{
4. public static void main(String args[]){
5. int arr[]={50,60,70,80};
6. for(int i=0;i<=arr.length;i++){
7. System.out.println(arr[i]);
8. }
9. }}
10. Output:
11. Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4
12. at TestArrayException.main(TestArrayException.java:5)
13. 50
14. 60
15. 70
16. 80

## Multidimensional Array in Java

In such case, data is stored in row and column based index (also known as matrix form).

**Syntax to Declare Multidimensional Array in Java**

1. dataType[][] arrayRefVar; (or)
2. dataType [][]arrayRefVar; (or)
3. dataType arrayRefVar[][]; (or)
4. dataType []arrayRefVar[];

**Example to instantiate Multidimensional Array in Java**

1. int[][] arr=new int[3][3];//3 row and 3 column

**Example to initialize Multidimensional Array in Java**

1. arr[0][0]=1;
2. arr[0][1]=2;
3. arr[0][2]=3;
4. arr[1][0]=4;
5. arr[1][1]=5;
6. arr[1][2]=6;
7. arr[2][0]=7;
8. arr[2][1]=8;
9. arr[2][2]=9;

### Example of Multidimensional Java Array

Let's see the simple example to declare, instantiate, initialize and print the 2Dimensional array.

1. //Java Program to illustrate the use of multidimensional array
2. class Testarray3{
3. public static void main(String args[]){
4. //declaring and initializing 2D array
5. int arr[][]={{1,2,3},{2,4,5},{4,4,5}};
6. //printing 2D array
7. for(int i=0;i<3;i++){
8. for(int j=0;j<3;j++){
9. System.out.print(arr[i][j]+" ");
10. }
11. System.out.println();
12. }
13. }}
14. Output:
15. 1 2 3
16. 2 4 5
17. 4 4 5

## Jagged Array in Java

If we are creating odd number of columns in a 2D array, it is known as a jagged array. In other words, it is an array of arrays with different number of columns.

1. //Java Program to illustrate the jagged array
2. class TestJaggedArray{
3. public static void main(String[] args){
4. //declaring a 2D array with odd columns
5. int arr[][] = new int[3][];
6. arr[0] = new int[3];
7. arr[1] = new int[4];
8. arr[2] = new int[2];
9. //initializing a jagged array
10. int count = 0;
11. for (int i=0; i<arr.length; i++)
12. for(int j=0; j<arr[i].length; j++)
13. arr[i][j] = count++;
15. //printing the data of a jagged array
16. for (int i=0; i<arr.length; i++){
17. for (int j=0; j<arr[i].length; j++){
18. System.out.print(arr[i][j]+" ");
19. }
20. System.out.println();//new line
21. }
22. }
23. }
24. Output:
25. 0 1 2
26. 3 4 5 6
27. 7 8

## What is the class name of Java array?

In Java, an array is an object. For array object, a proxy class is created whose name can be obtained by getClass().getName() method on the object.

1. //Java Program to get the class name of array in Java
2. class Testarray4{
3. public static void main(String args[]){
4. //declaration and initialization of array
5. int arr[]={4,4,5};
6. //getting the class name of Java array
7. Class c=arr.getClass();
8. String name=c.getName();
9. //printing the class name of Java array
10. System.out.println(name);
12. }}
13. Output:
14. I

## Copying a Java Array

We can copy an array to another by the arraycopy() method of System class.

**Syntax of arraycopy method**

1. public static void arraycopy(
2. Object src, int srcPos,Object dest, int destPos, int length
3. )

### Example of Copying an Array in Java

1. //Java Program to copy a source array into a destination array in Java
2. class TestArrayCopyDemo {
3. public static void main(String[] args) {
4. //declaring a source array
5. char[] copyFrom = { 'd', 'e', 'c', 'a', 'f', 'f', 'e',
6. 'i', 'n', 'a', 't', 'e', 'd' };
7. //declaring a destination array
8. char[] copyTo = new char[7];
9. //copying array using System.arraycopy() method
10. System.arraycopy(copyFrom, 2, copyTo, 0, 7);
11. //printing the destination array
12. System.out.println(String.valueOf(copyTo));
13. }
14. }
15. Output:
16. caffein

## Cloning an Array in Java

Since, Java array implements the Cloneable interface, we can create the clone of the Java array. If we create the clone of a single-dimensional array, it creates the deep copy of the Java array. It means, it will copy the actual value. But, if we create the clone of a multidimensional array, it creates the shallow copy of the Java array which means it copies the references.

1. //Java Program to clone the array
2. class Testarray1{
3. public static void main(String args[]){
4. int arr[]={33,3,4,5};
5. System.out.println("Printing original array:");
6. for(int i:arr)
7. System.out.println(i);
9. System.out.println("Printing clone of the array:");
10. int carr[]=arr.clone();
11. for(int i:carr)
12. System.out.println(i);
14. System.out.println("Are both equal?");
15. System.out.println(arr==carr);
17. }}

Output:

Printing original array:

33

3

4

5

Printing clone of the array:

33

3

4

5

Are both equal?

false

# Find the two repeating elements in a given array

You are given an array of n+2 elements. All elements of the array are in range 1 to n. And all elements occur once except two numbers which occur twice. Find the two repeating numbers.

For example, array = {4, 2, 4, 5, 2, 3, 1} and n = 5

The above array has n + 2 = 7 elements with all elements occurring once except 2 and 4 which occur twice. So the output should be 4 2.

## Program to print the duplicate elements of an array.

### Explanation

In this program, we need to print the duplicate elements present in the array. This can be done through two loops. The first loop will select an element and the second loop will iteration through the array by comparing the selected element with other elements. If a match is found, print the duplicate element.

Program to print the duplicate elements of an array

In the above array, the first duplicate will be found at the index 4 which is the duplicate of the element (2) present at index 1. So, duplicate elements in the above array are 2, 3 and 8.

### Algorithm

1. Declare and initialize an array.
2. Duplicate elements can be found using two loops. The outer loop will iterate through the array from 0 to length of the array. The outer loop will select an element. The inner loop will be used to compare the selected element with the rest of the elements of the array.
3. If a match is found which means the duplicate element is found then, display the element.
4.  #Initialize array
5.  arr = [1, 2, 3, 4, 2, 7, 8, 8, 3];
6. 
7.  print("Duplicate elements in given array: ");
8.  #Searches for duplicate element
9.  for i in range(0, len(arr)):
10.      for j in range(i+1, len(arr)):
11.          if(arr[i] == arr[j]):
12.              print(arr[j]);

**Write a program to remove duplicates from array in Java?**