Arrays

Exercises and solutions

1. What is an array? Name the property of an array that gives you the number of elements in the array.

**Answer:**

An array is a fixed-length data structure to store data items of the same data type. The length property of an array contains the number of elements in the array.

1. What is the index of the first element of an array?

**Answer:**

The index of the first element of an array is zero.

1. Write the code to initialize an int array in two ways. The array contains elements 10, 20, and 30.

**Solution:**

// Create an empty array, assign values

int[] arr = new int[3];

arr[0] = 10;

arr[1] = 20;

arr[2] = 30;

// Explict array initialization while instantiating the array

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

1. You have to store values in an array, but you do not know the number of elements in advance. How will you code this situation, so you get all elements in an array in the end?

**Answer:**

Use a java.util.ArrayList to store the elements in the beginning. Once you have all the elements, call the toArray() method of the ArrayList to get an array of all elements.

1. Complete the following snippet of code that prints the class name of an array object:  
     
   String[] names = {"Corky", "Bryce", "Paul", "Tony"};  
   String className = names./\* Your code goes here \*/;  
   System.out.println("Class Name: " + className);

**Solution:**

String[] names = {"Corky", "Bryce", "Paul", "Tony"};

String className = names.getClass().getName();

System.out.println("Class Name: " + className);

1. Consider the following declaration of a method named test() that takes an int[] array as an argument:  
     
   public static void test(int[] num) {  
    if(num.length > 0) {  
    num[0] = 100;  
    }  
    num = new int[]{1000, 2000};  
    }  
     
   Write the output when the following code is executed:  
     
   int[] num = {2, 4, 3, 1};  
   System.out.println("num[0] = " + num[0]);  
   test(num);  
   System.out.println("num[0] = " + num[0]);

**Answer:**

num[0] = 2

num[0] = 100

1. Which of the following statements declare a two-dimensional int array:  
     
   int[][] y;  
   int z[][];  
   int[] x[];  
   int[] x = {2, 2};

**Answer:**

Following declare a two-dimensional int array

int[][] y;  
int z[][];  
int[] x[];

1. Declare a two-dimensional array of 3 rows and 3 columns named table. Demonstrate how you will initialize all elements of the array with a value 10 during declaration and using a for loop.

**Solution:**

// Using a for loop

int[][] table = new int[3][3];

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

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

table[i][j] = 10;

}

}

// EUsing an explicit array initialization

int[][] table = {{10, 10, 10}, {10, 10, 10}, {10, 10, 10}};

1. Consider the following declaration for an array:  
     
   int[] x = {10, 20, 30, 40};  
     
   Write a for loop and a for-each loop to print each element's value in the array on a single line on the standard output.

**Solution:**

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

// Using a for loop

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

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

}

// Using a for-each loop

for (int e : x) {

System.out.println(e);

}

1. Consider the following declaration for an array:  
     
   int[] x = {10, 20, 30, 40};  
   System.out.println(x[5]);  
     
   What will happen when this snippet of code is executed.

**Answer:**

A java.lang.ArrayIndexOutOfBoundsException is thrown because the second statement accesses a non-existent sixth element in the array using x[5] and the array contain only four elements.

1. What method of the Arrays class will you use to sort a large array: sort() or parallelSort()?

**Answer:**

The parallelSort() method is suitable for bigger arrays.

1. Name the method in the Arrays class that converts an array to its string representation.

Answer:

The toString() method of the Arrays class is overloaded. It takes an array and returns a string representation of the elements in the array.

1. The Arrays class contains a binarySearch() method that lets you search a value in the array, What condition the array must meet before you should use the binarySearch() method?

**Answer:**

The array must be sorted.

1. Write and explain the output of the following snippet of code:  
     
   int[][] table1 = {{1, 2, 3}, {10, 20, 30}};  
   int[][] table2 = {{1, 2, 3}, {10, 20, 30}};  
     
   boolean equal1 = Arrays.equals(table1, table2);  
   boolean equal2 = Arrays.deepEquals(table1, table2);  
     
   System.out.println(equal1);  
   System.out.println(equal2);

**Output:**

false

true

Two arrays are equal according to the Arrays.equals() method if they contain the same elements in the same order. The equals() method returns false since table1 and table2 contain different int arrays even though those in arrays contain the same elements. For example, the two arrays, {1, 2, 3} and {10, 20, 30}, in table1 are different than the two arrays, {1, 2, 3} and {10, 20, 30}, in table2. Each of these four arrays is a different array object. One way to test this is using the following snippet of code, which will print the references of these four arrays:

for(int[] a : table1) {

System.out.println(a);

}

for(int[] a : table2) {

System.out.println(a);

}

Two array references are considered deeply equal according to the Arrays.deepEquals() method if both are null or they refer to arrays that contain the same number of elements and all corresponding pairs of elements in the two arrays are deeply equal. The Arrays.deepEquals() method returns true since table1 and table2 contain same numbers of elements in the same order at the lowest level.

1. Consider the following snippet of code that is meant to copy the contents of a two-dimensional array named table1 to another two-dimensional array named table2. Help the author of this code to complete the missing logic. You need to write two lines of code.  
     
   int[][] table1 = {{1, 2, 3}, {10, 20, 30}};  
   int[][] table2 = new int[table1.length][];  
     
   // Complete missing logic  
   for(int i = 0; i < table1.length; i++) {  
    /\* Your one line code goes here \*/  
    for(int j = 0; j < table1[i].length; j++) {  
    /\* Your one line code goes here \*/  
    }  
   }  
     
   boolean equal = Arrays.deepEquals(table1, table2);

System.out.println(equal);  
System.out.println(Arrays.deepToString(table1));  
System.out.println(Arrays.deepToString(table2));  
  
This snippet of code is supposed to have the following output:  
  
true  
[[1, 2, 3], [10, 20, 30]]  
[[1, 2, 3], [10, 20, 30]]

**Solution:**

int[][] table1 = {{1, 2, 3}, {10, 20, 30}};

int[][] table2 = new int[table1.length][];

// Complete missing logic

for (int i = 0; i < table1.length; i++) {

**table2[i] = new int[table1[i].length];**

for (int j = 0; j < table1[i].length; j++) {

**table2[i][j] = table1[i][j];**

}

}

boolean equal = Arrays.deepEquals(table1, table2);

System.out.println(equal);

System.out.println(Arrays.deepToString(table1));

System.out.println(Arrays.deepToString(table2));