**Arrays Question and Answers**

**Q1 What is an Array?**

1. Array is a collection of similar data types. It can not have different data type. It can hold both primitive types (int, float, double) and object references.
2. It is fixed in length i.e static in nature.
3. Arrays are created on the heap memory not on the stack.
4. Accessing an invalid index of an Array will cause exception.

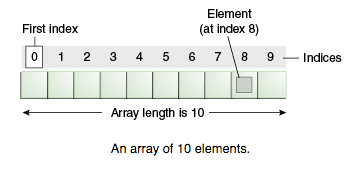
**Q2 How do you declare an Array in java?**

You can declare an Array in java by the following way :  
dataType[]    arrayVariableName  = new dataType[arraySize];  
  
for example for int data type, you can declare an int array as :  
int[]  temp = new int[256]

**Q3 What is the default value of Array for different data types?**

**Data Type                    Default value**byte, short, int, long             0  
float, double                         0.0  
boolean                                false  
Any object                           null

**Q4 Can you change size of Array in java after creation?**  
You can not change the size of Array after creation. Although there are other data-structures which can change size after creation.

[](https://2.bp.blogspot.com/-LrPFeShml8c/XJZJmwIFQ2I/AAAAAAAAA_w/X0HecDBL6ngLkD3kZGoarBqJks_EQDz-ACLcBGAs/s1600/Arrays+in+java.png)

**Q5 Can you pass the negative number in Array size?**  
No, you can not pass the negative number as Array size. If you pass a negative number in Array size then you will not get the compiler error. Instead, you will get the NegativeArraySizeException at run time.  
  
**Q6 Can you declare an Array without Array size?**  
  
No, you can not declare Array without Array size. You will get compile time error.  
  
**Q7 Where does Array stored in JVM memory ?**  
 Array is an object in java. So, Array is stored in heap memory in JVM.  
  
**Q8 Given a primitive Array in java, where does in JVM it is stored?**  
This is a follow-up question of Q7. An Array will always be an object on heap memory, even if the Array is declared to hold primitive elements.  
  
**Q9 What is ArrayStoreException ? When this exception is** **thrown ?**  
  
ArrayStoreException is a runtime exception. Array must contain the same data type elements.  
  
This exception is thrown to indicate that an attempt has been made to store the wrong type of object into an array of objects. In other words, if you want to store the integer Object in an Array of String you will get ArrayStoreException.  
  
The following code throws ArrayStoreException :

public class JavaHungry {

public static void main(String args[]) {

Object x[] = new String[3];

x[0] = new Integer(0);

}

}

**Q10 What is the difference between ArrayStoreException and ArrayOutOfBoundsException ?**  
ArrayStoreException is thrown if you are trying to add incompatible data type. For example, if you try to add an integer object to String Array, then ArrayStoreException is thrown.  
  
ArrayOutOfBoundsException is thrown when an attempt is made to access the Array with illegal index. For example, illegal index means if the index is either negative or greater than or equal to the size of the Array.  
  
**Q11 What are the advantages of Array ?**  
  
a. We can sort multiple elements of Array at the same time.  
  
b. Using index, we can access the element of the Array in O(1) time.  
  
**Q12 What are the disadvantages of Array?**  
a. To create an Array, contiguous memory is required. It is possible in JVM that the memory is available to accommodate Array but memory available is not contiguous.  
  
b. The Array is static data structure. It is of fixed size. We can not increase or decrease the size of the Array after creation.  
  
**Q13 Can we use Generics with an Array?**  
No, we can not use generics with an Array.

**Q14 What is an Anonymous Array in Java ? Give example?**  
  
An array without any name (or reference) is called an Anonymous Array. They are useful for the scenarios where we need one time usage of Array. For example,  
  
Anonymous int array :   
new int[] {2,3,4,5,6,7};  
  
Anonymous String array :  
new String[]{"Java", "Hungry"};  
  
**Q15 Write a program to print elements of Array ?**

public class JavaHungry {

public static void main(String args[]) {

int[] rollNumber = { 23, 17, 20, 29, 30 };

for (int temp : rollNumber)

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

}

}

Output :  
23 17 20 29 30   
  
**Q16 Write a program to sort an Array in Java ?**

You do not need to write quick sort or merge sort algorithm in order to sort an Array. You can sort an Array by using Arrays.sort() method. Check out the program below :

import java.util.\*;

public class JavaHungry {

public static void main(String args[]) {

int[] rollNumber = { 23, 17, 20, 29, 30 };

Arrays.sort(rollNumber);

for (int temp : rollNumber)

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

}

}

Output :  
17 20 23 29 30   
  
**Q17 Write a program to check whether two given Arrays are equal, given both contains same data type and same length ?**  
To check whether two given arrays are equal or not , we can use Arrays.equals() method. Check the program below :

import java.util.\*;

public class JavaHungry {

public static void main(String args[]) {

int[] arr1 = {2, 3, 4};

int[] arr2 = {1, 2, 3};

System.out.println(Arrays.equals(arr1 , arr2));

int[] arr3 = {2, 3, 4};

System.out.println(Arrays.equals(arr1 , arr3));

}

}

Output :  
false  
true  
  
**Q18 Which is legal int[] arr or int arr[] ?**  
Both are legal statements. It is preferred to write int[] arr instead of int arr[].

**Q19 Write a program to throw ArrayOutOfBoundsException?**

public class JavaHungry {

public static void main(String args[]) {

int[] rollNumber = { 23, 17, 20, 29, 30 };

/\* Index below is greater than the size

of the given Array \*/

int element = rollNumber[6];

for (int temp : rollNumber)

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

}

}

Output:  
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 6  
    at JavaHungry.main(JavaHungry.java:8)  
  
**Q20 Write a program to throw ArrayStoreException ?**

public class JavaHungry {

public static void main(String args[]) {

Object x[] = new String[3];

x[0] = new Integer(0);

}

}

Output:  
Exception in thread "main" java.lang.ArrayStoreException: java.lang.Integer  
    at JavaHungry.main(JavaHungry.java:6)

**Q21 How to find the missing number in a given Array from number 1 to 100 ?**

public class JavaHungry {

public static void main(String args[]) {

// Given input Array from 1 to n

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

// Calculate n value

int n = input.length + 1;

// Calculate Sum of N Number

// using Math formula n(n+1)/2

int sumOfNNumbers = sumOfAllNNumbers(n);

// Calculate Sum of all numbers in input array

int sumOfInputArray = sumOfInputArrayNumbers(input);

// Calculate missing number

// using subtraction

int missingNumber = sumOfNNumbers - sumOfInputArray;

// Print the Missing number

System.out.println("Missing number in an array is : "

+ missingNumber);

}

public static int sumOfAllNNumbers(int n){

int sum = (n\*(n+1))/2;

return sum;

}

public static int sumOfInputArrayNumbers(int[] input){

int sum = 0;

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

sum = sum + input[i];

}

return sum;

}

}

Output :  
Missing number in an array is : 8

**Q22 What are jagged arrays in java?**  
  
Arrays containing arrays of different length is known as jagged arrays. Multidimensional arrays are also known as jagged arrays

**Q23 There are two arrays object one containing 100 elements and another containing 50 elements. Both are of same data type. Can we assign one Array to another Array.**  
Yes,  an Array of 100 elements can be assigned to an Array of 50 elements in java. The only criteria is that both arrays of same data type. It is because at the time of assigning the values compiler looks for the data type of Array not the size of Array.

import java.util.\*;

public class JavaHungry {

public static void main(String args[]) {

int[] arr1 = new int[50];

int[] arr2 = new int[100];

arr1 = arr2;

System.out.println(arr1.length);

}

}

Output :  
100  
  
**Q24 What are the different ways to copy one Array from another Array?**  
  
There are four ways by which we can copy an Array.  
  
a. By using for loop  
b. By using clone() method  
c. By using Arrays.copyOf() method  
d. By using System.arraycopy() method  
  
**Q25 Write a program to search a specific element in an Array?**  
  
You can use binarySearch(int[], int) method. This method internally uses binary search algorithm.  
It is prerequisite for binary search algorithm to have elements sorted. In the given example I have taken an already sorted Array.

import java.util.\*;

public class JavaHungry {

public static void main(String args[]) {

// Sorted Array

int[] arr1 = {1, 2, 3, 4};

/\* if element present in Array, binarySearch()

method will return index \*/

System.out.println(Arrays.binarySearch(arr1, 2));

}

}

Output :  
1  
  
**Q26 What will happen if you do not initialize an Array?**  
Array will take default value depending upon the data type.  
  
**Q27 How to find duplicate elements in a given Array?**There are many ways by which you can find the duplicates in an Array. I am sharing two ways  
a. using for loop and compare  
b. using HashSet

import java.util.\*;

public class JavaHungry {

public static void main(String args[]) {

String[] arr1 = {"abc", "java", "javahungry", "java", "javahungry" };

for(int i=0; i < arr1.length-1; i++){

for(int j=i+1; j < arr1.length; j++) {

if(arr1[i].equals(arr1[j]) && i!=j ) {

System.out.println("Duplicate in Array is : "+ arr1[j]);

}

}

}

}

}

Output:  
Duplicate in Array is : java  
Duplicate in Array is : javahungry  
  
Time Complexity O(n^2) i.e quadratic.  
  
HashSet does not allow duplicate elements. You can traverse the array and insert each element into the HashSet. Add elements using add() method. If it returns true then continue to traverse the array. Otherwise if false then print out the duplicate value.

public class JavaHungry {

public static void main(String args[]) {

String[] arr1 = {"abc", "java", "javahungry", "java", "javahungry" };

HashSet<String> set = new HashSet<String>();

for (String val : arr1)

{

if (set.add(val) == false){

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

}

}

}

}

Output :  
java javahungry  
  
Time Complexity O(n)  
  
**Q28 What are the different ways to traverse an Array in java?**a. Using for loop  
b. Using for each loop  
  
**Q29 Is this a legal way to define arrays int[] arr = new int [4]{1, 2, 3, 4};**  
This is invalid way to initialize an Array in Java. You can not provide the size of the Array when you are declaring the elements in it.  
  
**Q30 What is two dimensional Array in java?**  
An Array of an Array in java is called as two dimensional Array.  
  
**Q31 How do you declare a two dimensional Array in java?**int[][] arr = new int[4][4];  
The above statement will create a 4 x 4 matrix.  
  
**Q32 Can we make Array volatile using volatile keyword?**  
Yes, we can make an Array volatile. Only variable representing an Array can be made volatile.  
  
**Q33 Are Array thread-safe ?**  
Reading an Array is a thread-safe operation but modifications are not.  
  
**Q34 What is the time complexity O(n) of different operations of an Array?**a. Access operation : O(1) This means very fast given the index of the element.  
b. Search operation : O(n) where n represents the number of elements in an Array.  
c. Insertion operation : O(n) where n represents the number of elements in an Array.  
b. Deletion operation : O(n) where n represents the number of elements in an Array.

**Q35 How to convert HashSet to Array in java ?**You can convert HashSet to Array using toArray() method.

**Q36 How to convert Array to TreeSet in java ?**   
  
To convert Array to TreeSet in java, first we need to convert Array to List using Arrays class asList() method. After converting Array to List,  pass the list object to TreeSet constructor. That's it , Array has been converted to TreeSet. You can confirm by printing out the values of TreeSet object.

**Q37 Write a program to find the first repeating number in an integer Array?**  
a. Logic is declare a variable minimum. Iterate through the Array in reverse order i.e from last element to first element. Add elements to the HashSet one by one.  
b. If repeated value occurs then store the index of the repeated value to the minimum variable.

import java.util.\*;

public class FirstRepeating {

public static void main(String args[]) {

int[] arr = new int[]{1,2,3,4,5,7,4,9};

firstRepeating(arr);

}

public static void firstRepeating(int[] arr) {

int minimum = -1;

HashSet set = new HashSet();

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

if(set.contains(arr[i]))

minimum = i;

else

set.add(arr[i]);

}

if(minimum != -1){

System.out.println("first repeating element is : "+ arr[minimum]);

}

}

}

Output :  
first repeating element is  : 4  
   
**Q38 Write a program to find the first non-repeating number in an integer Array?**   
Compare one element with rest of the remaining elements. If value does not match then exit the iteration and print the value.

public class FirstNonRepeating {

public static void main(String args[]) {

int[] arr = new int[]{1,2,3,4,5,1,2,3,5,9};

System.out.println(firstNonRepeating(arr,arr.length));

}

public static int firstNonRepeating(int[] arr, int length) {

int j;

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

for(j=0; j < length ;j++) {

if (arr[i]==arr[j] && i!=j) {

break;

}

}

if (j == length)

return arr[i];

}

return -1;

}

}

Output :  
4

**Q39 Differences between Array and ArrayList**

1. An array is basic functionality provided by Java. ArrayList is part of collection framework in Java.

Therefore array members are accessed using [], while ArrayList has a set of methods to access

elements and modify them.

**Example:**

|  |
| --- |
| A Java program to demonstrate differences between array and  ArrayList  import java.util.ArrayList;  import java.util.Arrays;    class Test  {   public static void main(String args[])  {  /\* ........... Normal Array............. \*/  int[] arr = new int[2];  arr[0] = 1;  arr[1] = 2;  System.out.println(arr[0]);  /\*............ArrayList..............\*/   // Create an arrayList with initial capacity 2  ArrayList<Integer> arrL = new ArrayList<Integer>(2);  // Add elements to ArrayList  arrL.add(1);  arrL.add(2);   // Access elements of ArrayList  System.out.println(arrL.get(0));  }  } |

**Output:**

1

1

.

2. Array is a fixed size data structure while ArrayList is not. One need not to mention the size of Arraylist

while creating its object. Even if we specify some initial capacity, we can add more elements.

|  |
| --- |
| **Example:**  A Java program to demonstrate differences between array and ArrayList  import java.util.ArrayList;  import java.util.Arrays;  class Test  {  public static void main(String args[])  {  /\* ........... Normal Array............. \*/  // Need to specify the size for array  int[] arr = new int[3];  arr[0] = 1;  arr[1] = 2;  arr[2] = 3;  // We cannot add more elements to array arr[]  /\*............ArrayList..............\*/  // Need not to specify size  ArrayList<Integer> arrL = new ArrayList<Integer>();  arrL.add(1);  arrL.add(2);  arrL.add(3);  arrL.add(4);  // We can add more elements to arrL  System.out.println(arrL);  System.out.println(Arrays.toString(arr));  }  } |

**Output:**

[1, 2, 3, 4]

[1, 2, 3]

.

3. Array can contain both primitive data types as well as objects of a class depending on the definition of

the array. However, ArrayList only supports object entries, not the primitive data types.

**Note:** When we do arraylist.add(1); : it converts the primitive int data type into an Integer object.

**Sample Code**:

|  |
| --- |
| import java.util.ArrayList;  class Test  {  public static void main(String args[])  {  // allowed  int[] array = new int[3];  // allowed, however, need to be intialized  Test[] array1 = new Test[3];  // not allowed (Uncommenting below line causes  // compiler error)  // ArrayList<char> arrL = new ArrayList<char>();  // Allowed  ArrayList<Integer> arrL1 = new ArrayList<>();  ArrayList<String> arrL2 = new ArrayList<>();  ArrayList<Object> arrL3 = new ArrayList<>();  }  } |

4. Since ArrayList can’t be created for primitive data types, members of ArrayList are always references

to objects at different memory locations. Therefore in ArrayList, the actual objects are never stored at

contiguous locations. References of the actual objects are stored at contiguous locations.In array, it

depends whether the arrays is of primitive type or object type. In case of primitive types,

actual values are contiguous locations, but in case of objects, allocation is similar to ArrayList.

**Q40 Key Differences Between Array and Linked List**

1. An array is the data structure that contains a collection of similar type data elements whereas the Linked list is considered as non-primitive data structure contains a collection of unordered linked elements known as nodes.  
2. In the array the elements belong to indexes, i.e., if you want to get into the fourth element you have to write the variable name with its index or location within the square bracket.  
3. In a linked list though, you have to start from the head and work your way through until you get to the fourth element.  
4. Accessing an element in an array is fast, while Linked list takes linear time, so it is quite a bit slower.  
5. Operations like insertion and deletion in arrays consume a lot of time. On the other hand, the performance of these operations in Linked lists is fast.  
6. Arrays are of fixed size. In contrast, Linked lists are dynamic and flexible and can expand and contract its size.  
7. In an array, memory is assigned during compile time while in a Linked list it is allocated during execution or runtime.  
9. Elements are stored consecutively in arrays whereas it is stored randomly in Linked lists.  
10. The requirement of memory is less due to actual data being stored within the index in the array. As against, there is a need for more memory in Linked Lists due to storage of additional next and previous referencing elements.  
11. In addition memory utilization is inefficient in the array. Conversely, memory utilization is efficient in the linked list.