



InterviewBit

Array Interview Questions



To view the live version of the page, [click here](#).

© Copyright by Interviewbit

Contents

Array Interview Questions for Freshers

1. Mention some advantages and disadvantages of Arrays.
2. Difference between Array and ArrayList in Java.
3. What will happen if you do not initialize an Array?
4. What is the default value of Array in Java?
5. What is the time complexity for performing basic operations in an array?
6. Can you declare an array without assigning the size of an array?
7. Difference between Array and Object.
8. Where is an Array stored in JVM memory?
9. Find the Target Element in an array.
10. Can a Negative number be passed in Array size?
11. When will we get ArrayStoreException?
12. When will we get ArrayIndexOutOfBoundsException Exception?
13. We know that Arrays are objects so why cannot we write strArray.length()?
14. What is the difference between length and length () in Java?
15. What is a Jagged Array in Java?
16. Can the sizeof operator be used to tell the size of an array passed to a function?

Array Interview Questions for Experienced

17. What do you mean by the terms “Dimension” and “Subscript” when we talk about arrays?
18. Compare Arrays with Linked Lists.

Array Interview Questions for Experienced

(.....Continued)

19. Why is the complexity of fetching from an Array be $O(1)$?
20. How do you find the missing integer in an array of range 1 to 100?
21. How do you remove a particular element from an array?
22. How can you get the index of an array element?
23. How do you merge two sorted arrays into one sorted array?
24. How to check the equality of two arrays?

Array Coding Interview Questions

25. How do you sort an array of 0 and 1?
26. How do you rotate an array?
27. Two sum of an array: In this question you will be given an array arr and a target. You have to return the indices of the two numbers such that they add up to target.
28. Check for balanced parenthesis in an expression using constant space.
29. Find out smallest and largest number in a given Array?

Let's get Started

The Array is a fundamental topic for programming interviews. In your coding journey, you will find that arrays are used in many problems. Arrays are ubiquitous which means no matter which programming language you choose, the usage of arrays will be there. It's available in various programming languages like [C](#), [C++](#), [Java](#), and even in [Python](#), Perl, and Ruby.

What is an Array in Programming?

- An Array is a collection of similar data types stored in contiguous memory locations.
- At the time of declaration of an array, you must specify the type of data with the array name.
- You can access different elements present in an array using their index. For example, if you want to access an element pleasant at the 3rd index(4th element) in an array arr, then you can write arr[3].

How do you declare an Array?

Array declaration syntax in C/C++:

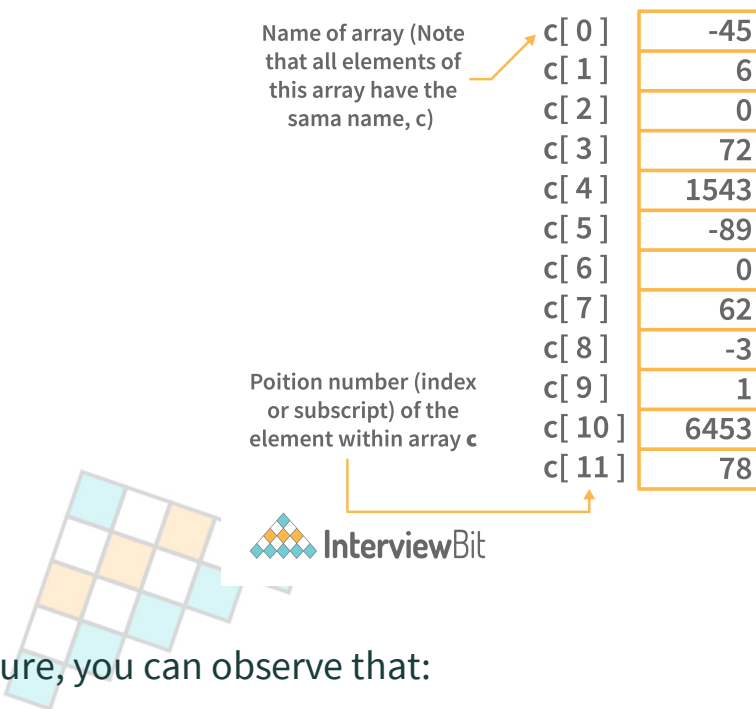
```
DataType ArrayName [size];
```

Array declaration syntax in Java:

```
int [] intArray;
```

- An array is fixed in length i.e static in nature.
- An array can hold primitive types and object references.
- In an array when a reference is made to a nonexistent element, an `IndexOutOfRangeException` occurs.

Note:- Array indexing starts from 0, not 1.



c[0]	-45
c[1]	6
c[2]	0
c[3]	72
c[4]	1543
c[5]	-89
c[6]	0
c[7]	62
c[8]	-3
c[9]	1
c[10]	6453
c[11]	78

From the figure, you can observe that:

- Element 72 is at index 3 but actually, it's the 4th element of the array.
- In an array, the first element is at the lowest address and the last element is at the highest address.

If you are planning to interview for a developer's position, you must prepare arrays very well. In this article, we will be covering a comprehensive list of Array interview questions and answers for freshers and experienced professionals to help you ace the interview:

- [Array Interview Questions for Freshers](#)
- [Array Interview Questions for Experienced](#)
- [Array Coding Interview Questions](#)

Array Interview Questions for Freshers

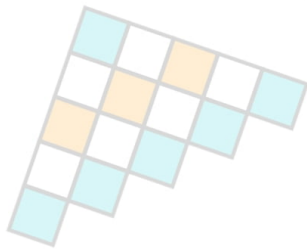
1. Mention some advantages and disadvantages of Arrays.

Advantages:

- Multiple elements of Array can be sorted at the same time.
- Using the index, we can access any element in $O(1)$ time.

Disadvantages:

- You need to specify how many elements you're going to store in your array ahead of time and We can not increase or decrease the size of the Array after creation.
- You have to shift the other elements to fill in or close gaps, which takes worst-case $O(n)$ time.

2. Difference between Array and ArrayList in Java.

Definition	An Array is a collection of similar data types stored in contiguous memory locations.	An ArrayList is a class of Java Collections framework which contains popular classes like Vector, HashMap, etc.
Static/ Dynamic	static in size.	dynamic in size.
Resizable	An array is not resizable as it is a fixed-length data structure.	An ArrayList is a variable-length data structure that can be resized. As an element is being added to an ArrayList, JVM checks to see if it has enough space by calling the ensureCapacity method. If a space exists, it adds the element to the ArrayList, otherwise, it resizes the ArrayList. In the resizing process, an array of a larger size is created, the old array is copied to the new array using Arrays.copyOf, and the new array is then assigned to the existing array.

3. What will happen if you do not initialize an Array?

The array will take default values depending upon the data type.

4. What is the default value of Array in Java?

If we don't specify the values by ourselves, then Java assigns default values in them which are 0 for byte, short, int, and long, 0.0 for float and double, false for boolean, and null for objects respectively.

5. What is the time complexity for performing basic operations in an array?

The Time Complexity of different operations in an array is: For analyzing the real-time complexity you also have to consider the time in bringing the block of memory from an external device to RAM which takes $O(\sqrt{N})$ time.

ARRAY OPERATION	REAL TIME COMPLEXITY	ASSUMED TIME COMPLEXITY
Accessing the i-th element.	$O(\sqrt{N})$	$O(1)$
Traversing all elements.	$O(N + \sqrt{N})$	$O(N)$
Override element at i-th index.	$O(\sqrt{N})$	$O(1)$
Insert an element.	$O(N + \sqrt{N})$	$O(N)$
Delete an element.	$O(N + \sqrt{N})$	$O(N)$

6. Can you declare an array without assigning the size of an array?

No, we cannot declare an array without assigning size. If we declare an array without size, it will throw compile time error.

7. Difference between Array and Object.

- An object represents a thing with characteristics (called a property), whereas an array creates a list of data and stores it in a single variable. Using brackets and dots, we can access, alter, and delete items from objects, while a variety of built-in methods and zero-based indexing allow us to access and modify items in arrays. We can iterate over object properties and array items using various different loops (e.g. for, for...in, for...of, forEach()).
- All Java objects are dynamically allocated on the heap. Unlike C++, where objects can be allocated in memory either on Heap or on Stack. When we use the new() method in C++, the object is allocated on the heap, otherwise on Stack if not global or static.

8. Where is an Array stored in JVM memory?

An Array is an object in java. So, Array is stored in heap memory in Java Virtual Machine.

9. Find the Target Element in an array.

First of all, you will get a number n , which indicates the size of the array. After that, you will get n more inputs corresponding to each index of the array. Then you will be given a target, for which you have to find, at which index of array target is present. Print -1 if target is not present in the array.

Approach:

We will run a for loop on the input array and check if the value equivalent to target is present in the array or not.

If the target is found then we will print the index of this target value and return else we will return -1.

```
import java.io.*;
import java.util.*;
public class InterviewBit {
    public static void main(String[] args) throws Exception {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = scn.nextInt();
        }
        int target = scn.nextInt();
        for (int i = 0; i < arr.length; i++) {
            if (target == arr[i]) {
                System.out.println(i);
                return;
            }
        }
        System.out.println(-1);
    }
}
```

10. Can a Negative number be passed in Array size?

No, a negative number cannot be passed as array size. If you pass a negative number in Array size then you will get the `NegativeArraySizeException` at run time.

11. When will we get `ArrayStoreException`?

- `ArrayStoreException` is a runtime exception.
- **For example**, you will get this exception at run time if you declare a String Array and then try to insert integer elements in the array.

12. When will we get `ArrayIndexOutOfBoundsException` Exception?

`ArrayIndexOutOfBoundsException` is a runtime exception that occurs when the program tries to access the invalid index of an array such as an Index higher than the size of the array or a negative index.

13. We know that Arrays are objects so why cannot we write `strArray.length()`?

We cannot write **strArray.length()** because length is not a method, it's a data item of an array. We can use the methods of Object like **toString()** and **hashCode()** against Array.

14. What is the difference between length and length () in Java?

In Java, the length() is a method of String class whereas length is an instance variable of an array.

- **length in Java**

- The length variable returns the length of an array i.e. a number of elements present in an array.
- After initializing, the length of an array cannot be changed, so the length variable can directly be used to get the length of an array.
- It is used only for an array.
- **Example:**

```
public class InterviewBit {  
    public static void main(String args[]) {  
        int array[] = {1, 2, 3, 4, 5};  
        System.out.println("Length of an array is: " + array.length);  
    }  
}
```

Output

```
Length of an array is: 5
```

- **length() in Java**

- It is a static method of String class.
- The length() returns the number of characters stored in a string object.
- The string class uses this method as the length of a string can be modified using the various operations performed on a string object.
- The String class uses a char[] array internally.
- **Example:**

```
public class InterviewBit {  
    public static void main(String args[]) {  
        String str = "Welcome to InterviewBit";  
        System.out.println("Length of String using length() method is: " + str.length());  
    }  
}
```

Output:

```
Length of String using length() method is: 23
```

15. What is a Jagged Array in Java?

Jagged arrays are multidimensional arrays in which the member arrays are of different sizes. As an example, we can make a 2D array where the first array contains three elements, and the second array consists of four elements. Below is an example demonstrating the concept of jagged arrays.

```
public class InterviewBit {  
    public static void main(String[] args){  
        int[][] 2dArray = new int[2][];  
        2dArray[0] = new int[3];  
        2dArray[1] = new int[4];  
        int counter = 0;  
        for(int row=0; row < 2dArray.length; row++){  
            for(int col=0; col < 2dArray[row].length; col++){  
                2dArray[row][col] = counter++;  
            }  
        }  
        for(int row=0; row < 2dArray.length; row++){  
            System.out.println();  
            for(int col=0; col < 2dArray[row].length; col++){  
                System.out.print(2dArray[row][col] + " ");  
            }  
        }  
    }  
}
```

Output:

```
0 1 2
3 4 5 6
```

16. Can the sizeof operator be used to tell the size of an array passed to a function?

Passing an array as a parameter in C or C++ does not pass information about how many elements there are in the array. Although sizeof() can tell you the size of the pointer and the size of the type it points to, it cannot tell you how many bytes are occupied by the entire array.

Array Interview Questions for Experienced

17. What do you mean by the terms “Dimension” and “Subscript” when we talk about arrays?

- In an array "Dimension" is the number of indices, or subscripts, that you need for specifying an individual element. Dimensions and subscripts may be confusing.
- A subscript is a number, while the dimension is a description of the range of acceptable keys.
- You only need 1 subscript for each dimension of the array.
- For example, arr[10][5] is an array having 2 dimensions:
 - One with size 10 and the other with size 5. You need 2 subscripts to address its elements. One between 0 and 9, inclusive; the other between 0 and 4.

18. Compare Arrays with Linked Lists.

- **Size:** The size of an array cannot be altered at runtime since data can only be stored in contiguous blocks of memory in an array. However, due to the node structure of a linked list, its size can be altered easily since each node points to the next one such that data can exist at scattered (non-contiguous) addresses.
- **Memory allocation:** For arrays, memory is allocated at compile time whereas for linked lists memory is allocated at runtime. But, a dynamically allocated array also allocates memory at runtime.
- **Memory efficiency:** For the same number of elements, the linked list uses more memory due to its node structure since each node stores a reference to the next node along with the data. However, linked lists may use less memory overall compared to arrays when there is uncertainty about size or there are large variations in the size of data elements.
- **Execution time:** In the case of a linked list, all the previous elements must be traversed to reach any element whereas elements in an array can be accessed directly using their index. As a result, some operations such as modifying an element are faster in arrays, while some other operations such as inserting an element are faster in linked lists.

19. Why is the complexity of fetching from an Array be $O(1)$?

In an Array, objects are stored in continuous memory location. So, if you know the address of the base object then you will be able to find the address of the i th object.

```
address(a[i]) = address(a[0]) + i*size(object)
```

This term is independent of n , so the time complexity of fetching from an Array is $O(1)$.

20. How do you find the missing integer in an array of range 1 to 100?

- During an interview, this question is often used to assess your knowledge of how programmers may manipulate or troubleshoot arrays. As the answer may depend on the exact elements or structure of the array, this question may also display your problem-solving abilities. In addition to showing your flexibility and extensive knowledge, providing solutions to all situations can also impress the interviewer.
- A missing integer can be found by calculating the sum of the series using this function: $n(n + 1) / 2$
- This function will work only if the array doesn't contain any duplicates or is missing more than one number. If an array contains duplicate elements, you can sort the array and determine whether there are two equal elements.

21. How do you remove a particular element from an array?

- You can't directly remove elements from the original array, as arrays are fixed sets and the size can't change therefore the interviewer is looking for you to suggest an alternate solution and address the issue that the question presents. The best way to remove an element would be to create a new array. In this array, you could include copies of elements of the first array and omit only the element you want to remove.
- Another approach is searching for the target element in the array and then moving all the elements in one position back which are on the right side of the target element.

22. How can you get the index of an array element?

- You can find the index of an element through a linear or binary search. A linear search is a function in which you loop through each and every element of an array until it finds the match of the desired element. When it finds the matching element, it returns the index. Therefore time complexity of the linear search is $O(n)$. Linear search can be applied to sorted as well as an unsorted array.
- If the array is sorted, you can use a binary search that repeatedly splits the array in half until the median of the interval matches the desired element and returns the index. Therefore time complexity of the binary search is $O(\log n)$.

23. How do you merge two sorted arrays into one sorted array?

During a programming interview, the employer may ask you to demonstrate your ability to work with more complex coding functions, such as merging arrays.

Approach 1:

- To merge two arrays, you can create a new array of size equal to the sum of two arrays. After that, you can copy the elements from the first array into the new one. Then you can traverse the second array and insert elements from it into the new one using insertion sort. This method takes
 - Time complexity: $O(n_1 * n_2)$
 - Extra Space: $O(n_1 + n_2)$.

Approach 2 using Heap:

- Convert the second array to min-heap:
 - As you traverse the first array, compare the current element to the top of the created min_heap.
 - In this case, if the current element in the first array is greater than the heap top, swap the current element of the first array with the root of the heap, and heapify the root of the min_heap.
 - The first array now contains the N first elements of the sorted merged array after performing the above operation for every element of the first array.
- Now, the last M elements of the sorted merged array are in the min_heap or second array.
- Apply in-place heapsort to the second array to sort them.
 - Time Complexity: $O(N * \log M + M * \log N)$
 - Extra Space: $O(1)$

24. How to check the equality of two arrays?

You will be given two arrays and you have to check whether the 2 arrays are equal or not.

First, you have to check the lengths of two given arrays. When the length of both arrays is the same, we compare corresponding elements of both arrays. Both the arrays will be considered equal if all corresponding pairs of elements are equal. If the arrays are big in size, this method will be time-consuming; therefore, this method is not recommended to check the equality of two arrays. You can also use the in-built `equals()` method of Arrays class, but in the interview, the interviewer may ask you to compare two arrays without using in-built functions; therefore, this method will help you at that time.

Array Coding Interview Questions

25. How do you sort an array of 0 and 1?

You are given an array of 0s and 1s in random order and you have to sort this array i.e. Segregate 0s on the left side and 1s on the right side of the array.

Approach:

- Maintain two indexes and initialize the first index as 0 and second index $n-1$.
- Now follow the following algorithm until $left < right$
 - a) Keep incrementing left index while there are 0s at it
 - b) Keep decrementing index right while there are 1s at it
 - c) Whenever $left < right$, exchange `arr[left]` and `arr[right]`

```
class InterviewBit
{
void segregatearray(int arr[], int size)
{
    int left = 0, right = size - 1;
    while (left < right)
    {
        while (arr[left] == 0 && left < right)
            left++;
        while (arr[right] == 1 && left < right)
            right--;
        if (left < right)
        {
            arr[left] = 0;
            arr[right] = 1;
            left++;
            right--;
        }
    }
}

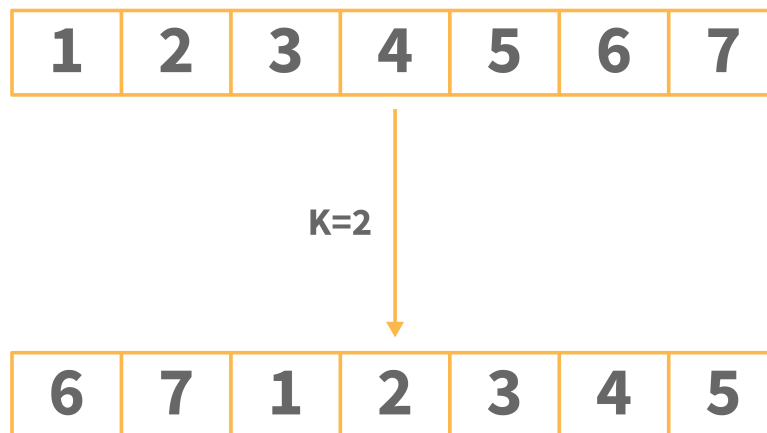
public static void main(String[] args)
{
    Segregate seg = new Segregate();
    int arr[] = new int[]{0, 1, 0, 1, 1, 1};
    int i, arr_size = arr.length;
    seg.segregatearray(arr, arr_size);
    System.out.print("Array after segregation is ");
    for (i = 0; i < 6; i++)
        System.out.print(arr[i] + " ");
}
}
```

Time Complexity : $O(n)$

26. How do you rotate an array?

You will be given an input number N and N numbers following it as the input. You are also given an integer K which represents how many times you have to rotate the array. Rotate the array K values to the right if K is positive and K values to the left if K is negative. If $K=0$, do not rotate the array.

The figure shows how the rotating array will look if $k=2$.

**Approach:**

1. If the value of K is positive then do $K = K \% N$ where N is the length of the input array for eg: if $k = 10$ and $N = 3$ then k will become 1 which means rotating the array 10 times is equivalent to rotating the array 1 time. Similarly, If the value of K is negative, $K = K \% N + N$.
2. Reverse the first part of the array i.e. from 0 to $N - K - 1$ and the second part from $N - K$ to $N - 1$ separately.
3. Reverse the entire array i.e. from 0 to $N - 1$. K determines how the array will be rotated.

Code:

```
import java.io.*;

import java.util.*;

public class Main {
    public static void display(int[] arr) {
        StringBuilder sb = new StringBuilder();

        for (int val : arr) {
            sb.append(val + " ");
        }
        System.out.println(sb);
    }

    public static void reverse(int[] arr, int li, int ri) {
        while (li < ri) {
            int t = arr[li];
            arr[li] = arr[ri];
            arr[ri] = t;

            li++;
            ri--;
        }
    }
}
```

```
public static void rotate(int[] arr, int k) {
    k = k % arr.length;
    if (k < 0) {
        k += arr.length;
    }
    reverse(arr, 0, arr.length - k - 1);
    reverse(arr, arr.length - k, arr.length - 1);
    reverse(arr, 0, arr.length - 1);
}
```

```
public static void main(String[] args) throws Exception {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    int n = Integer.parseInt(br.readLine());
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = Integer.parseInt(br.readLine());
    }
    int k = Integer.parseInt(br.readLine());

    rotate(arr, k);
    display(arr);
}
```

27. Two sum of an array: In this question you will be given an array arr and a target. You have to return the indices of the two numbers such that they add up to target.

In this question, you will be given an array arr and a target. You have to return the indices of the two numbers such that they add up to the target.

Approach:

So the easiest and most efficient solution is using a HashMap. We have to iterate through all the elements of the array and if (target-num) is present in the array then return {lower index, higher index} else push the value at arr[i] and the index i in the hashmap.

```
public class InterviewBit {  
    public int[] twoSum(int[] arr, int target) {  
  
        HashMap<Integer,Integer> hash = new HashMap<Integer,Integer>();  
        for(int i = 0; i <arr.length; i++){  
  
            Integer diff = (Integer)(target - arr[i]);  
            if(hash.containsKey(diff)){  
                int toReturn[] = {hash.get(diff)+1, i+1};  
                return toReturn;  
            }  
  
            hash.put(arr[i], i);  
        }  
        return null;  
    }  
}
```

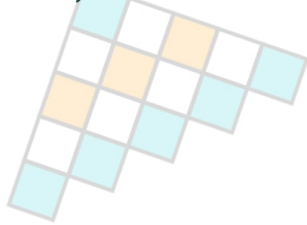
The time Complexity for this approach will be $O(n)$.

28. Check for balanced parenthesis in an expression using constant space.

- You will be given string str containing characters '(', ')', '{', '}', '[', and ']', your task is to determine whether the brackets are balanced or not.
- These conditions should satisfy for balanced brackets:
 - Open brackets must be closed by the same type of brackets.
 - Open brackets must be closed in the correct order.

Approach:

- To keep track of two brackets to be compared keep two variables i and j .
- Maintain a count variable and increment its value on encountering an opening bracket and decrements on encountering a closing bracket.
- Assign values of j and i as- $j = i$, $i = i + 1$ and $count++$ when opening brackets are encountered.
- When Closing brackets are encountered do $count--$ i.e decrement its value and compare brackets at i and j ,
 - If brackets at i and j match, then substitute '#' in the string at i th and j th position. Increment i and decrement j until non '#' value is encountered or $j \geq 0$.
 - If brackets at i and j do not match, return false.
- If $count \neq 0$, return false.



```
import java.util.*;

class InterviewBit
{
    static String str = "[[]]()";
    static int Count = 0;
    static int i = 0;
    static int j = -1;

    static int helperFunc(char compare)
    {
        Count--;
        char temp = str.charAt(j);
        if (j > -1 && temp == compare)
        {
            str = str.replace(str.charAt(i), '#');
            str = str.replace(str.charAt(j), '#');
            temp = str.charAt(j);
            while (j >= 0 && temp == '#')
                j--;
            i++;
            return 1;
        }
        else
            return 0;
    }

    static boolean isValid()
    {
        if (str.length() == 0)
            return true;

        else {
            int result;
            while (i < str.length())
            {
                char temp = str.charAt(i);
                if(temp=='{')
                {
                    result = helperFunc('{');
                    if (result == 0)
                    {
                        return false;
                    }
                }
                else if(temp == ')')
                {
                    result = helperFunc('(');
                    if (result == 0)
                    {
                        return false;
                    }
                }
            }

            else if(temp == ']')
            {

```


29. Find out smallest and largest number in a given Array?

Approach:

- For storing the largest and smallest numbers create two variables named smallest and largest.
- Assign Integer.MAX_VALUE to the variable smallest
- Assign Integer.MIN_VALUE to the variable largest
- In each traversal of for loop, we will compare the current element with the largest and smallest number and update the value accordingly.
- If a number is larger than the largest, then it can not be smaller than the smallest, therefore, we can skip if the first condition holds.

```
import java.util.*;
public class InterviewBit{
    public static void main(String args[]) {

        int[] inputArray = {10,20, 22, 30, 77};
        int largest = inputArray[0];
        int smallest = inputArray[0];

        for( int number : inputArray ) {
            if(number > largest) {
                largest = number;
            }
            else if (smallest > number) {
                smallest = number;
            }
        }
        System.out.println("Largest and Smallest numbers are "
                           + largest + " "+smallest);
    }
}
```

Output :

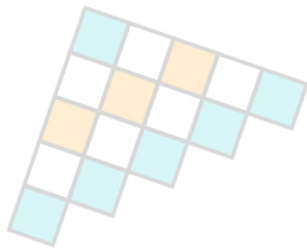
```
Largest and Smallest numbers are 77 10
```

Conclusion

In this article, we have covered the most important and commonly asked interview questions based on arrays. To make the most of all the knowledge available, it is absolutely necessary to practice data structures and algorithms as much as possible. You should keep in mind certain properties of array data structures, for example, array index starts at 0, the elements of an array are stored in contiguous memory locations, etc.

Additional Interview Resources:

- [Data Structure Interview Questions](#)
- [Algorithm Interview Questions](#)
- [DSA Tutorial](#)



Links to More Interview Questions

[C Interview Questions](#)

[Php Interview Questions](#)

[C Sharp Interview Questions](#)

[Web Api Interview Questions](#)

[Hibernate Interview Questions](#)

[Node Js Interview Questions](#)

[Cpp Interview Questions](#)

[Oops Interview Questions](#)

[Devops Interview Questions](#)

[Machine Learning Interview Questions](#)

[Docker Interview Questions](#)

[Mysql Interview Questions](#)

[Css Interview Questions](#)

[Laravel Interview Questions](#)

[Asp Net Interview Questions](#)

[Django Interview Questions](#)

[Dot Net Interview Questions](#)

[Kubernetes Interview Questions](#)

[Operating System Interview Questions](#)

[React Native Interview Questions](#)

[Aws Interview Questions](#)

[Git Interview Questions](#)

[Java 8 Interview Questions](#)

[Mongodb Interview Questions](#)

[Dbms Interview Questions](#)

[Spring Boot Interview Questions](#)

[Power Bi Interview Questions](#)

[Pl Sql Interview Questions](#)

[Tableau Interview Questions](#)

[Linux Interview Questions](#)

[Ansible Interview Questions](#)

[Java Interview Questions](#)

[Jenkins Interview Questions](#)