**Java Programming Interview Questions**

Java Programming Interview Questions are always deciding factor in any Java interview. Recently I am taking a lot of interview for my organisation. So I was in the search of some java programming interview questions that are little bit tricky also.

## Java Programming Interview Questions

[](https://cdn.journaldev.com/wp-content/uploads/2011/03/java-programming-interview-questions.jpg)

Here I am providing five of the java programming interview questions I found interesting and need a closer look to understand.

The explanation will be provided after the questions. Test your knowledge of java by trying to provide the answer of the below java interview test questions.

### Java Programming Interview Question 1

What is the output of the below statements?

String s1 = "abc";

String s2 = "abc";

System.out.println("s1 == s2 is:" + s1 == s2);

### Java Programming Interview Question 2

What is the output of the below statements?

String s3 = "JournalDev";

int start = 1;

char end = 5;

System.out.println(start + end);

System.out.println(s3.substring(start, end));

### Java Programming Interview Question 3

What is the output of the below statements?

HashSet shortSet = new HashSet();

for (short i = 0; i < 100; i++) {

shortSet.add(i);

shortSet.remove(i - 1);

}

System.out.println(shortSet.size());

### Java Programming Interview Question 4

What will be the boolean "flag" value to reach the finally block?

try {

if (flag) {

while (true) {

}

} else {

System.exit(1);

}

} finally {

System.out.println("In Finally");

}

### Java Programming Interview Question 5

What will be the output of the below statements?

String str = null;

String str1="abc";

System.out.println(str1.equals("abc") | str.equals(null));

### Java Programming Interview Question 6

What will be the output of the below statements?

String x = "abc";

String y = "abc";

x.concat(y);

System.out.print(x);

### Java Programming Interview Question 7

What will be the output of below program?

public class MathTest {

public void main(String[] args) {

int x = 10\*10-10;

System.out.println(x);

}

}

### Java Programming Interview Question 8

What will be the output when below java program is compiled and executed?

public class Test {

public static void main(String[] args) {

try {

throw new IOException("Hello");

}catch(IOException | Exception e) {

System.out.println(e.getMessage());

}

}

}

## Java Programming Interview Questions Answers

I hope you have looked into above questions before looking at the answers and explanation.

### Java Programming Interview Question 1 Answer and Explanation

The given statements output will be "false" because in java + operator precedence is more than == operator. So the given expression will be evaluated to "s1 == s2 is:abc" == "abc" i.e false.

### Java Programming Interview Question 2 Answer and Explanation

The given statements output will be "ourn". First character will be automatically type caste to int. After that since in java first character index is 0, so it will start from 'o' and print till 'n'. Note that in String substring function it leaves the end index.

### Java Programming Interview Question 3 Answer and Explanation

The size of the shortSet will be 100. Java Autoboxing feature has been introduced in JDK 5, so while adding the short to HashSet<Short> it will automatically convert it to Short object. Now "i-1" will be converted to int while evaluation and after that it will autoboxed to Integer object but there are no Integer object in the HashSet, so it will not remove anything from the HashSet and finally its size will be 100.

### Java Programming Interview Question 4 Answer and Explanation

The finally block will never be reached here. If flag will be TRUE, it will go into an infinite loop and if its false its exiting the JVM. So finally block will never be reached here.

### Java Programming Interview Question 5 Answer and Explanation

The given print statement will throw [java.lang.NullPointerException](https://www.journaldev.com/14544/java-lang-nullpointerexception) because while evaluating the OR logical operator it will first evaluate both the literals and since str is null, .equals() method will throw exception. Its always advisable to use short circuit logical operators i.e "||" and "&&" which evaluates the literals values from left and since the first literal will return true, it will skip the second literal evaluation.

### Java Programming Interview Question 6 Answer and Explanation

The statements will print abc. Notice that x.concat(y); will create a new string but it’s not assigned to x, so value of x is not changed.

### Java Programming Interview Question 7 Answer and Explanation

This is a tricky question, it looks like the test is about the order of execution of the mathematical operators and syntax of main method will get overlooked. It will produce Runtime error because main method is not static, something like below.

pankaj:bin pankaj$ java MathTest

Error: Main method is not static in class MathTest, please define the main method as:

public static void main(String[] args)

### Java Programming Interview Question 8 Answer and Explanation

No, it won't print Hello. It will be a compile time error as The exception IOException is already caught by the alternative Exception.

I hope that the above scenarios will help a bit in understanding some of the java concepts. Please try these java programming interview questions before going to the solution and comment to let me know your score.

### Java Tricky Interview Question 1

What is the output of the below program?

public class Test {

public static void main(String[] args) {

method(null);

}

public static void method(Object o) {

System.out.println("Object impl");

}

public static void method(String s) {

System.out.println("String impl");

}

}

### Java Tricky Programming Question 2

What will below statements print?

long longWithL = 1000\*60\*60\*24\*365L;

long longWithoutL = 1000\*60\*60\*24\*365;

System.out.println(longWithL);

System.out.println(longWithoutL);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Java Tricky Interview Question 1 Answer with Explanation

As we know that we can assign null to any object, so doesn’t compiler complains about this program? According to java specs, in case of overloading, compiler picks the most specific function. Obviously String class is more specific that Object class, hence it will print “String impl”.  
What if we have another method in the class like below:

public static void method(StringBuffer i){

System.out.println("StringBuffer impl");

}

In this case, java compiler will throw error as “The method method(String) is ambiguous for the type Test” because String and StringBuffer, none of them are more specific to others. A method is more specific than another if any invocation handled by the first method could be passed on to the other one without a compile-time type error. We can pass String as parameter to Object argument and String argument but not to StringBuffer argument method.

### Java Tricky Programming Question 2 Answer with Explanation

The output of the code snippet will be:

31536000000

1471228928

In case of first variable, we are explicitly making it a long by placing a “L” at the end, so compiler will treat this at long and assign it to first variable.  
In second case, compiler will do the calculation and treat it as a 32-bit integer, since the output is outside the range of integer max value (2147483647), compiler will truncate the most significant bits and then assign it to the variable.

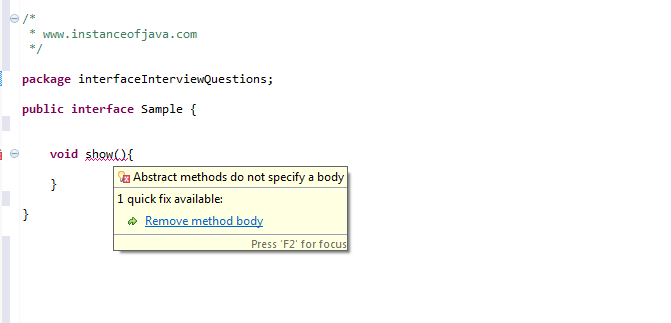
Binary equivalent of 1000\*60\*60\*24\*365L = 011101010111101100010010110000000000 (36 bits)  
Removing 4 most significant bits to accommodate in 32-bit int, value = 01010111101100010010110000000000 (32 bits)  
Which is equal to 1471228928 and hence the output.

|  |
| --- |
| **Program: How to get distinct elements from an array by avoiding duplicate elements?** |
| package com.java2novice.algos;  public class MyDisticntElements {     public static void printDistinctElements(int[] arr){   for(int i=0;i<arr.length;i++){      boolean isDistinct = false;      for(int j=0;j<i;j++){  if(arr[i] == arr[j]){      isDistinct = true;      break;  }      }      if(!isDistinct){  System.out.print(arr[i]+" ");      }  }      }   public static void main(String a[]){    int[] nums = {5,2,7,2,4,7,8,2,3};  MyDisticntElements.printDistinctElements(nums);      }  } |

**Java interface interview programs part 1: interface programming java**  
  
**Program #1: what will happen if we define normal methods in interface**

1. package interfaceinverviewprograms.instanceofjava;
2. public interface A{
4. /\*\*
5. \* @java interface  interview programming  questions and answers for freshers and experienced
6. \*/
8. void show() {
10. System.out.println("Hello world");
12. }
13. }

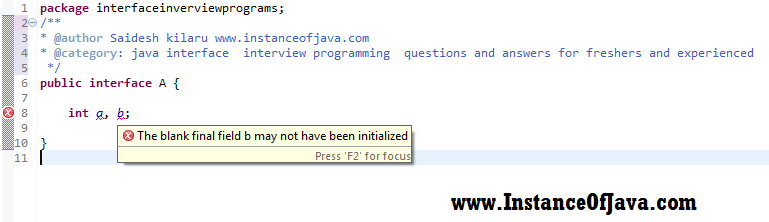
Click for Output 

[](https://3.bp.blogspot.com/-WlTDjb2dQVg/VtWXfP3-RDI/AAAAAAAAAnk/I_iVg-c-vI0jubZGXA3xNgQngdWWsEgNQCKgB/s1600/interface+concrete+method.png)

**Program #2:java interview programs to practice: Non static variables in interface**

1. package interfaceinverviewprograms.instanceofjava;
2. public interface A{
4. /\*\*
5. \* @java interface  interview programming  questions and answers for freshers and experienced
6. \*/
8. int a,b;
9. }

Click for Output 

[](https://4.bp.blogspot.com/-_lQlucEx54s/V5To3CgUilI/AAAAAAAABI4/I2wt5C1C8f0n28YUNXxXlrjW0_ZuKn8fwCKgB/s1600/non+static+variables+in+interfaces.png)

**Program #3:java interview programs to practice: which modifiers interface allows**

1. package interfaceinverviewprograms.instanceofjava;
2. public interface A{
4. /\*\*
5. \* @java interface  interview programming  questions and answers for freshers and experienced
6. \*/
8. private int x;
9. protected int y;
10. }

Click for Output 

1. compile time error: Illegal modifier for the interface field A.x; only
2. public, static & final are permitted
3. compile time error: Illegal modifier for the interface field A.y; only
4. public, static & final are permitted

**Program #4:java interview programs to practice: interface allows constructor?**

1. package interfaceinverviewprograms.instanceofjava;
2. public interface A{
4. /\*\*
5. \* @java interface  interview programming  questions and answers for freshers and experienced
6. \*/
7. A(){
9. }
10. }

Click for Output 

1. Compile time error: interface cannot allow constructor

================Kishori=============================

Find the point where maximum intervals overlap

[**3.4**](https://www.geeksforgeeks.org/medium/)

Consider a big party where a log register for guest’s entry and exit times is maintained. Find the time at which there are maximum guests in the party. Note that entries in register are not in any order.

Example:

Input: arrl[] = {1, 2, 9, 5, 5}

exit[] = {4, 5, 12, 9, 12}

First guest in array arrives at 1 and leaves at 4,

second guest arrives at 2 and leaves at 5, and so on.

Output: 5

There are maximum 3 guests at time 5.

Below is a **Simple Method** to solve this problem.

1) Traverse all intervals and find min and max time (time at which first guest arrives and time at which last guest leaves)

2) Create a count array of size ‘max – min + 1’. Let the array be count[].

3) For each interval [x, y], run a loop for i = x to y and do following in loop.  
     count[i – min]++;

4) Find the index of maximum element in count array. Let this index be ‘max\_index’, return max\_index + min.

Above solution requires O(max-min+1) extra space. Also time complexity of above solution depends on lengths of intervals. In worst case, if all intervals are from ‘min’ to ‘max’, then time complexity becomes O((max-min+1)\*n) where n is number of intervals.

An **Efficient Solution** is to use sorting n O(nLogn) time. The idea is to consider all events (all arrivals and exits) in sorted order. Once we have all events in sorted order, we can trace the number of guests at any time keeping track of guests that have arrived, but not exited.

Consider the above example.

arr[] = {1, 2, 10, 5, 5}

dep[] = {4, 5, 12, 9, 12}

Below are all events sorted by time. Note that in sorting, if two

events have same time, then arrival is preferred over exit.

Time Event Type Total Number of Guests Present

------------------------------------------------------------

1 Arrival 1

2 Arrival 2

4 Exit 1

5 Arrival 2

5 Arrival 3 // Max Guests

5 Exit 2

9 Exit 1

10 Arrival 2

12 Exit 1

12 Exit 0

Total number of guests at any time can be obtained by subtracting  
total exits from total arrivals by that time.

So maximum guests are three at time 5.

Following is the implementation of above approach. Note that the implementation doesn’t create a single sorted list of all events, rather it individually sorts arr[] and dep[] arrays, and then uses merge process of merge sort to process them together as a single sorted array.

* Java

|  |
| --- |
| // Java Program to find maximum guest  // at any time in a party  import java.util.\*;    class GFG {        static void findMaxGuests(int arrl[], int exit[],                                            int n)      {      // Sort arrival and exit arrays      Arrays.sort(arrl);      Arrays.sort(exit);        // guests\_in indicates number of guests at a time      int guests\_in = 1, max\_guests = 1, time = arrl[0];      int i = 1, j = 0;        // Similar to merge in merge sort to process      // all events in sorted order      while (i < n && j < n)      {          // If next event in sorted order is arrival,          // increment count of guests          if (arrl[i] <= exit[j])          {              guests\_in++;                // Update max\_guests if needed              if (guests\_in > max\_guests)              {                  max\_guests = guests\_in;                  time = arrl[i];              }              i++; //increment index of arrival array          }          else // If event is exit, decrement count          { // of guests.              guests\_in--;              j++;          }      }        System.out.println("Maximum Number of Guests = "+                      max\_guests + " at time " + time);      }        // Driver program to test above function      public static void main(String[] args)      {          int arrl[] = {1, 2, 10, 5, 5};          int exit[] = {4, 5, 12, 9, 12};          int n = arrl.length;          findMaxGuests(arrl, exit, n);      }  } |

Output:

Maximum Number of Guests = 3 at time 5

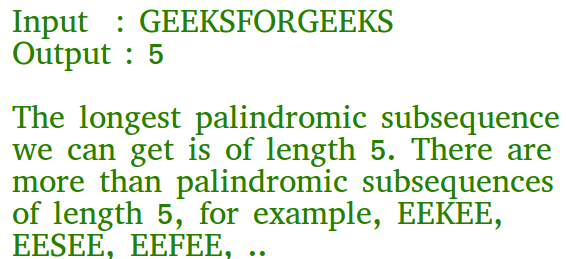
Time Complexity of this method is O(nLogn).

Thanks to Gaurav Ahirwar for suggesting this method.

# Longest palindrome subsequence with O(n) space

[**3.3**](https://www.geeksforgeeks.org/medium/)

Given a sequence, find the length of the longest palindromic subsequence in it.



More examples

Input : abbaab

Output : 4

Input : geeksforgeeks

Output : 5

## [Recommended: Please try your approach on *{IDE}* first, before moving on to the solution.](https://ide.geeksforgeeks.org/)

We have discussed a [Dynamic Programming solution for Longest Palindromic Subsequence](https://www.geeksforgeeks.org/dynamic-programming-set-12-longest-palindromic-subsequence/)which is based on below recursive formula.

*// Every single character is a palindrome of length 1  
L(i, i) = 1 for all indexes i in given sequence*

*// IF first and last characters are not same  
If (X[i] != X[j]) L(i, j) = max{L(i + 1, j), L(i, j – 1)}*

*// If there are only 2 characters and both are same  
Else if (j == i + 1) L(i, j) = 2*

*// If there are more than two characters, and first  
// and last characters are same  
Else L(i, j) = L(i + 1, j – 1) + 2*

The solution discussed above takes O(n2) extra space. In this post a space optimized solution is discussed that requires O(n) extra space. The idea is to create a one dimensional array a[] of same size as given string. We make sure that a[i] stores length of longest palindromic subsequence of prefix ending with i (or substring s[0..i]).

* C++
* Java
* Python3

|  |
| --- |
| // A Space optimized Dynamic Programming  // based Java program for LPS problem    class GFG {        // Returns the length of the longest      // palindromic subsequence in str      static int lps(String s)      {          int n = s.length();        // a[i] is going to store length      // of longest palindromic subsequence      // of substring s[0..i]          int a[] = new int[n];            // Pick starting point          for (int i = n - 1; i >= 0; i--)              {              int back\_up = 0;        // Pick ending points and see if s[i]      // increases length of longest common      // subsequence ending with s[j].      for (int j = i; j < n; j++) {        // similar to 2D array L[i][j] == 1      // i.e., handling substrings of length      // one.          if (j == i)          a[j] = 1;        // Similar to 2D array L[i][j] = L[i+1][j-1]+2      // i.e., handling case when corner characters      // are same.      else if (s.charAt(i) == s.charAt(j))          {          int temp = a[j];          a[j] = back\_up + 2;          back\_up = temp;          }        // similar to 2D array L[i][j] = max(L[i][j-1],      // a[i+1][j])        else              {                  back\_up = a[j];                  a[j] = Math.max(a[j - 1], a[j]);              }            }      }      return a[n - 1];      }    /\* Driver program to test above functions \*/      public static void main(String[] args)      {          String str = "GEEKSFORGEEKS";          System.out.println(lps(str));      }  }    //This article is contributed by prerna saini. |

Run on IDE

5

Time Complexity : O(n\*n)  
Auxiliary Space : O(n)