150+ Core Java Interview Questions and Answers for Freshers and Experienced in 2023

simplilearn.com/tutorials/java-tutorial/java-interview-questions

Table of Contents

View More

Java is the most widely used programming language in the current IT industry. One major reason for the vast number of beginners and professionals in the field of programming is the career potential that Java knowledge comes with. This article is dedicated to the same purpose. Here is a complete guide on how to help you crack the most frequently asked Core Java Interview questions.

Java Interview Questions for Freshers

So let's get started with the first set of basic core Java technical interview questions which is primarily useful for freshers.

1. What are the differences between C++ and Java?

Concept.

C++ is not platform-independent; the principle behind C++ programming is "write once, compile anywhere."

In contrast, because the byte code generated by the Java compiler is platform-independent, it can run on any machine, Java programs are written once and run everywhere.

Also Read: Learn C++ Programming

Languages Compatibility.

C++ is a programming language that is based on the C programming language. Most other high-level languages are compatible with C++.

Most of the languages of Java are incompatible. Java is comparable to those of C and C++.

Interaction with the library.

It can access the native system libraries directly in C++. As a result, it's better for programming at the system level.

Java's native libraries do not provide direct call support. You can use Java Native Interface or access the libraries.

Characteristics.

C++ distinguishes itself by having features that are similar to procedural and object-oriented languages. The characteristic that sets Java apart is automatic garbage collection. Java doesn’t support destructors at the moment.

The semantics of the type.

Primitive and object types in C++ have the same kind of semantics. The primitive and object and classes of Java, on the other hand, are not consistent.

In the context of Compiler and Interpreter.

Java refers to a compiled and interpreted language. In contrast, C++ is only a compiled language.

In Java, the source code is the compiled output is a platform-independent byte code.

In C++, the source program is compiled into an object code that is further executed to produce an output.

2. List the features of the Java Programming language?

A few of the significant features of Java Programming Language are:

Easy: Java is a language that is considered easy to learn. One fundamental concept of OOP Java has a catch to understand.

Secured Feature: Java has a secured feature that helps develop a virus-free and tamper-free system for the users.

OOP: OOP stands for Object-Oriented Programming language. OOP signifies that, in Java, everything is considered an object.

Independent Platform: Java is not compiled into a platform-specific machine; instead, it is compiled into platform-independent bytecode. This code is interpreted by the Virtual Machine on which the platform runs.

Get the Must-Have Skills of a Web Developer

Caltech Coding BootcampExplore Program

3. What do you get in the Java download file? How do they differ from one another?

We get two major things along with the Java Download file.

JDK - Java Development Kit

JRE - Java Runtime Environment

JDK	JRE
Abbreviation for JavaDevelopment Kit	Abbreviation for Java Runtime Environment
JDK is a dedicated kit for solely software development	JRE is a set of software and library designed for executing Java Programs
Unlike JVM, JDK is Platform Dependent	Unlike JVM, JRE is also Platform Dependent
JDK package is a set of tools for debugging and Developing	JRE Package is one that only supports files and libraries for a runtime environment
JDK package will be provided with an installer file	JRE Package does not get an installer but has only a runtime environment

4. What is a ClassLoader?

A classloader in Java is a subsystem of Java Virtual Machine, dedicated to loading class files when a program is executed; ClassLoader is the first to load the executable file.

Java has Bootstrap, Extension, and Application classloaders.

Also Read: What is Bootstrap and How to Embed Bootstrap into Angular?

5. What are the Memory Allocations available in JavaJava?

Java has five significant types of memory allocations.

- Class Memory
- Heap Memory
- Stack Memory
- Program Counter-Memory
- Native Method Stack Memory

6. What are the differences between Heap and Stack Memory in Java?

Stack memory in data structures is the amount of memory allocated to each individual programme. It is a fixed memory space. Heap memory, in contrast, is the portion that was not assigned to the Java code but will be available for use by the Java code when it is required, which is generally during the program's runtime.

7. Will the program run if we write static public void main?

Yes, the program will successfully execute if written so. Because, in Java, there is no specific rule for the order of specifiers

8. What is the default value stored in Local Variables?

Neither the Local Variables nor any primitives and Object references have any default value stored in them.

9. Explain the expected output of the following code segment?

```
public class Simplilearn

{

   public static void main (String args[])

   {

      System.out.println(100 + 100 +“Simplilearn");

      System.out.println(“E-Learning Company" + 100 + 100);

   }

}
```

The answers for the two print statements are as follows.

- 200Simplilearn
- E-Learning Company100100

Here's How to Land a Top Software Developer Job

Full Stack Development-MEANExplore Program

10. What is an Association?

An Association can be defined as a relationship that has no ownership over another. For example, a person can be associated with multiple banks, and a bank can be related to various people, but no one can own the other.

11. What do you mean by aggregation?

The term aggregation refers to the relationship between two classes best described as a “whole/part” and “has-a” relationship. This kind is the most specialized version of an association relationship. It contains the reference to another class and is said to have ownership of that class.

12. Define Copy Constructor in Java

A Copy Constructor in Java is a constructor that initializes an object through another object of the same class.

13. What is a Marker Interface?

An empty interface in Java is referred to as a Marker interface. Serializable and Cloneable are some famous examples of Marker Interface.

14. What is Object Cloning?

An ability to recreate an object entirely similar to an existing object is known as Object Cloning in Java. Java provides a clone() method to clone a current object offering the same functionality as the original object.

15. Can Java be said to be the complete object-oriented programming language

No, Java cannot be treated as a complete object-oriented programming language.

16. What is an object-oriented paradigm?

A Paradigm that is based on the concepts of "Objects." It contains data and code. Data that is in the form of fields, and regulation, that is in the form of procedures. The exciting feature of this paradigm is that the object's procedures can access and often modify the data fields themselves.

17. Define Wrapper Classes in Java.

In Java, when you declare primitive datatypes, then Wrapper classes are responsible for converting them into objects(Reference types).

18. What is a singleton class in Java? And How to implement a singleton class?

A class that can possess only one object at a time is called a singleton class. To implement a singleton class given steps are to be followed:

1.
    1. Make sure that the class has only one object
    2. Give global access to that object

19. Define package in Java.

The package is a collective bundle of classes and interfaces and the necessary libraries and JAR files. The use of packages helps in code reusability.

20. Can you implement pointers in a Java Program?

Java Virtual Machine takes care of memory management implicitly. Java's primary motto was to keep programming simple. So, accessing memory directly through pointers is not a recommended action. Hence, pointers are eliminated in Java.

21. Differentiate between instance and local variables.

For instance, variables are declared inside a class, and the scope of variables in javascript is limited to only a specific object.

A local variable can be anywhere inside a method or a specific block of code. Also, the scope is limited to the code segment where the variable is declared.

22. Explain Java String Pool.

A collection of strings in Java's Heap memory is referred to as Java String Pool. In case you try to create a new string object, JVM first checks for the presence of the object in the pool. If available, the same object reference is shared with the variable, else a new object is created.

23. What is an Exception?

An Exception handling in Java is considered an unexpected event that can disrupt the program's normal flow. These events can be fixed through the process of Exception Handling.

24. What is the final keyword in Java?

The term final is a predefined word in Java that is used while declaring values to variables. When a value is declared using the final keyword, then the variable's value remains constant throughout the program's execution.

25. What happens when the main() isn't declared as static?

When the main method is not declared as static, then the program may be compiled correctly but ends up with a severe ambiguity and throws a run time error that reads "NoSuchMethodError."

26. Why is Java a platform independent language?

One of the most well-known and widely used programming languages is Java. It is a programming language that is independent of platforms. Java doesn't demand that the complete programme be rewritten for every possible platform. The Java Virtual Machine and Java Bytecode are used to support platform independence. Any JVM operating system can run this platform-neutral byte code. The application is run after JVM translates the byte code into machine code. Because Java programmes can operate on numerous systems without having to be individually rewritten for each platform, the language is referred to as "Write Once, Run Anywhere" (WORA).

27. Why is the main method static in Java?

Java's main() function is static by default, allowing the compiler to call it either before or after creating a class object. The main () function is where the compiler begins programme execution in every Java programme. Thus, the main () method needs to be called by the compiler. If the main () method is permitted to be non-static, the JVM must instantiate its class when calling the function.

28. What part of memory - Stack or Heap - is cleaned in the garbage collection process?

On Heap memory, garbage collection is employed to release the memory used by objects with no references. Every object created in the Heap space has access to the entire application and may be referred to from anywhere.

29. What is the difference between the program and the process?

A programme is a non-active entity that includes the collection of codes necessary to carry out a specific operation. When a programme is run, an active instance of the programme called a process is launched. A process is begun by a programme once it has been run. The process carries out the program's specified instructions.

30. What are the differences between constructor and method of a class in Java?

Initializing the state of the object is done by constructors. A function Object () { [native code] }, like methods, contains a group of statements (or instructions) that are carried out when an object is created. A method is a group of statements that work together to complete a certain task and return the outcome to the caller. A method has the option of working without returning anything.

31. Which among String or String Buffer should be preferred when there are a lot of updates required to be done in the data?

Because StringBuilder is quicker than StringBuffer, it is advised to utilize it wherever possible. However, StringBuffer objects are the best choice if thread safety is required.

32. What happens if the static modifier is not included in the main method signature in Java?

The main function is called by the JVM even before the objects are created, thus even if the code correctly compiles, there will still be an error at runtime.

33. Can we make the main() thread a daemon thread?

This technique designates whether the active thread is a user thread or a daemon thread. For instance, tU.setDaemon(true) would convert a user thread named tU into a daemon thread. On the other side, executing tD.setDaemon(false) would convert a Daemon thread, tD, into a user thread.

34. What happens if there are multiple main methods inside one class in Java?

There is no limit to the number of major approaches you can use. Overloading is the ability to have main methods with different signatures than main (String []), and the JVM will disregard those main methods.

35. How does an exception propagate in the code?

In the event that an exception is not caught, it is initially thrown from the top of the stack and then moves down the call stack to the preceding method. The runtime system looks for a way to handle an exception that a method throws. The ordered list of methods that were called to get to the method where the error occurred is the collection of potential "somethings" that can be used to manage the exception. The call stack is the list of methods, and exception propagation is the search technique.

36. How do exceptions affect the program if it doesn't handle them?

If you don't deal with an exception once it occurs, the programme will end abruptly and the code after the line where the exception occurred won't run.

37. Is it mandatory for a catch block to be followed after a try block?

Each attempt block does not necessarily have to be followed by a catch block. Either a catch block or a final block ought to come after it. Additionally, any exceptions that are expected to be thrown should be mentioned in the method's throws clause.

38. Can you call a constructor of a class inside another constructor?

Yes, a class may include any number of constructors, and each function Object () {[native code] } may call the others using the this() function Object() { [native code] } call function [please do not mix the this() function Object() { [native code] } call function with this keyword]. The constructor's first line should be either this () or this(args). Overloading of constructors is what this is called.

39. Contiguous memory locations are usually used for storing actual values in an array but not in ArrayList. Explain.

Primitive data types like int, float, and others are typically present in an array. In such circumstances, the array immediately saves these elements at contiguous memory regions. While an ArrayList does not contain primitive data types. Instead of the actual object, an ArrayList includes the references to the objects' many locations in memory. The objects are not kept in consecutive memory regions because of this.

40. Why does the java array index start with 0?

The distance from the array's beginning is just an offset. There is no distance because the first element is at the beginning of the array. Consequently, the offset is 0.

41. Why is the remove method faster in the linked list than in an array?

Because there is no background scaling of an array, insertion, addition, and removal operations are quicker with a LinkedList. Only references in adjacent items need to update when a new item is added in the middle of the list.

42. How many overloaded add() and addAll() methods are available in the List interface? Describe the need and uses.

List is an interface in the Java Collections Framework. The add() and addAll() methods are the main methods at the List interface. The add() method is used to add an element to the list, while the addAll() method is used to add a collection of elements to the list.

The List interface contains two overloaded versions of the add() method:

The first add() method accepts a single argument of type E, the element to be added to the list.

The second add() method accepts a variable number of arguments of type E, which are the elements to be added to the list.

The List interface also contains two overloaded versions of the addAll() method:

The first addAll() method accepts a single argument of type Collection<? Extends E>, which is the collection of elements to be added to the list.

The second addAll() method accepts a variable number of arguments of type E, which are the elements to be added to the list.

43. How does the size of ArrayList grow dynamically? And also state how it is implemented internally?

A resizable array implementation in Java is called ArrayList. Dynamically expanding array lists make it possible to add new elements at any time. The underlying data structure of the ArrayList is an array of the Object class. The ArrayList class in Java has three constructors. There are available readObject and writeObject methods specific to it. The Object Array in an ArrayList is temporary. There are implemented and Serialization-capable versions of RandomAccess, Cloneable, and java.io (that are Marker Interface in Java).

44. Although inheritance is a popular OOPs concept, it is less advantageous than composition. Explain.

A class's testability is improved through composition over inheritance. If a class is comprised of another class, it is simple to create a mock object to simulate the combined class for testing purposes. This privilege is not given by inheritance. Even while Composition and Inheritance both let you reuse code, Inheritance has the drawback of breaking encapsulation. If the function of the subclass depends on the superclass's action, it suddenly becomes vulnerable. Sub-class functionality may be broken without any alteration on the part of the super-class when its behaviour changes.

45. What are Composition and Aggregation? State the difference.

Aggregation (HAS-A) and composition are its two forms (Belongs-to). In contrast to composition, which has a significant correlation, the aggregation has a very modest association. Aggregation can be thought of as a more confined version of the composition. Since all compositions are aggregates but not all aggregates are compositions, aggregate can be thought of as the superset of composition.

46. How is the creation of a String using new() different from that of a literal?

The new () operator always produces a new object in heap memory when creating a String object. The String pool may return an existing object if we build an object using the String literal syntax, such as "Baeldung," on the other hand.

47. How is the ‘new' operator different from the ‘newInstance()' operator in java?

Both the new operator and the newInstance() method are used to create objects in Java. If we already know the kind of object to create, we can use the new operator; however, if the type of object to create is supplied to us at runtime, we must use the newInstance() function.

48. Is exceeding the memory limit possible in a program despite having a garbage collector?

Yes, even with a garbage collector in place, the programme could still run out of memory. Garbage collection aids in identifying and removing programme objects that are no longer needed in order to release the resources they use. When an object in a programme cannot be reached, trash collection is executed with respect to that object. If there is not enough memory available to create new objects, a garbage collector is used to free up memory for things that have been removed from the scope. When the amount of memory released is insufficient for the creation of new objects, the program's memory limit is exceeded.

49. Why is synchronization necessary? Explain with the help of a relevant example.

Multiple threads trying to access the same resources in a multi-threaded software may frequently result in unexpected and incorrect outcomes. Therefore, it must be ensured through some form of synchronization that only one thread can access the resource at any given time. Java offers a method for setting up threads and synchronizing their operations with the aid of synchronized blocks. The synchronized keyword in Java is used to identify synchronized blocks. In Java, a synchronized block is one that is tied to an object. Only one thread can be running at a time inside synchronized blocks since they are all synchronized on the same object. Until the thread inside the synchronized block exits the block, all other threads trying to enter the block are blocked.

50. Define System.out.println().

System.out.println() in Java outputs the argument that was supplied to it. On the monitor, the println() method displays the findings. An objectname is typically used to call a method.

FREE Java Certification Training

Learn A-Z of Java like never beforeEnroll Now

51. Can you explain the Java thread lifecycle?

A thread can be in any of the following states in Java. These are the states:

- New: A new thread is always in the new state when it is first formed. The function hasn't been run yet, thus it hasn't started to execute for a thread in the new state.
- Active: A thread switches from the new state to the active state when it calls the start() method. The runnable state and the running state are both contained within the active state.
- Blocked or Waiting: A thread is either in the blocked state or the waiting state when it is inactive for a while (but not indefinitely).
- Timed waiting: When we use the sleep () method on a particular thread, we are actually engaging in timed waiting. The thread enters the timed wait state using the sleep () function. The thread awakens when the allotted time has passed and resumes execution where it left off.
- Termination: A thread that has been terminated means it is no longer active in the system. In other words, the thread is inactive and cannot be revived (made active again after being killed).

52. What could be the tradeoff between the usage of an unordered array versus the usage of an ordered array?

When opposed to an unordered array, which has a time complexity of O, an ordered array's search times have a time complexity of O(log n) (n). Due to the need to shift the elements with higher values to create room for the new member, an ordered array has a temporal complexity of O(n) during the insertion process. Instead, an unordered array's insertion operation requires a constant O amount of time (1).

53. Is it possible to import the same class or package twice in Java and what happens to it during runtime?

The same package or class may be imported more than once. Neither the JVM nor the compiler raise an objection. Even if you import the same class several times, the JVM will only internally load it once.

54. In case a package has sub packages, will it suffice to import only the main package? e.g. Does importing of com.myMainPackage.* also import com.myMainPackage.mySubPackage.*?

Sub-packages won't be imported when a package is imported. When you import a package, all of its classes and interfaces—with the exception of those from its sub-packages—are imported.

55. Will the final block be executed if the code System.exit(0) is written at the end of the try block?

The system is established as the last line to be run, after which nothing will happen, therefore both the catch and finally blocks are essentially ignored.

56. Explain the term “Double Brace Initialisation” in Java?

The outer braces of the double-brace initialization construct an anonymous class that is descended from the provided class and gives an initializer block for that class (the inner braces).

57. Why is it said that the length() method of String class doesn't return accurate results?

Since this char [] array is used by the Java String class internally, the length variable cannot be made public.

58. What are the possible ways of making objects eligible for garbage collection (GC) in Java?

If a reference variable for an object is removed from the programme while it is running, the object may be trash collected. They are also referred to as inaccessible objects occasionally. The new operator returns a reference to an object after dynamically allocating memory for it.

59. In the below Java Program, how many objects are eligible for garbage collection?

I don't know about the program, but generally, three objects are eligible for garbage collection.

The first object is created when the program is started and is no longer needed when the program ends.

The second object is created when the user inputs their name and is no longer required when the program ends.

The third object is created when the user inputs their address and is no longer needed when the program ends.

60. What is the best way to inject dependency? Also, state the reason.

Constructor injection. A class requesting its dependencies through its function Object() { [native code] } is the most typical instance of dependency injection. Since the client cannot be constructed without the required dependencies, this guarantees that it is always in a correct state.

61. How we can set the spring bean scope. And what supported scopes does it have?

There are four ways to set the scope of a Spring bean: singleton, prototype, request, and session.

The singleton scope creates a single instance of a bean, which is shared by all objects that request it.

The prototype scope creates a new instance of a bean for each object that requests it.

The request and session scopes are only available in a web-based context. The request scope creates a new bean instance for each HTTP request, and the session scope creates a single instance of a bean shared by all objects in a single HTTP session.

62. What are the different categories of Java Design patterns?

The three categories of Java design patterns are creational, structural, and behavioural design patterns.

63. What is a Memory Leak? Discuss some common causes of it.

A memory leak is the slow degradation of system performance over time brought on by the fragmentation of a computer's RAM as a result of shoddy application design or programming that fails to release memory chunks when they are no longer required. These memory leaks frequently result from session items in excess, insertion into Collection objects without deletion, infinite caches, excessive page switching on the operating system, listener methods that are not called, and bespoke data structures that are poorly written.

64. Assume a thread has a lock on it, calling the sleep() method on that thread will release the lock?

No, the thread might release the locks using notify, notifyAll(), and wait() methods.

65. Write a Java Program to print Fibonacci Series using Recursion.

```
class FibonacciExample2{
 static int n1=0,n2=1,n3=0;
 static void printFibonacci(int count){
    if(count>0){
         n3 = n1 + n2;
         n1 = n2;
         n2 = n3;
         System.out.print(" "+n3);
         printFibonacci(count-1);
     }
 }
 public static void main(String args[]){
  int count=10;
  System.out.print(n1+" "+n2);//printing 0 and 1
  printFibonacci(count-2);//n-2 because 2 numbers are already printed
 }
 }
```

66. Write a Java program to check if the two strings are anagrams.

```
import java.util.Arrays;
public class AnagramString {
    static void isAnagram(String str1, String str2) {
        String s1 = str1.replaceAll("\\s", "");
        String s2 = str2.replaceAll("\\s", "");
        boolean status = true;
        if (s1.length() != s2.length()) {
            status = false;
        } else {
            char[] ArrayS1 = s1.toLowerCase().toCharArray();
            char[] ArrayS2 = s2.toLowerCase().toCharArray();
            Arrays.sort(ArrayS1);
            Arrays.sort(ArrayS2);
            status = Arrays.equals(ArrayS1, ArrayS2);
        }
        if (status) {
            System.out.println(s1 + " and " + s2 + " are anagrams");
        } else {
            System.out.println(s1 + " and " + s2 + " are not anagrams");
        }
    }
    public static void main(String[] args) {
        isAnagram("Keep", "Peek");
        isAnagram("Mother In Law", "Hitler Woman");
    }
}
```

Output

```
Keep and Peek are anagrams
MotherInLaw and HitlerWoman are anagrams
```

67. Write a Java Program to find the factorial of a given number.

4! = 4*3*2*1 = 24

5! = 5*4*3*2*1 = 120

68. Given an array of non-duplicating numbers from 1 to n where one number is missing, write an efficient java program to find that missing number.

Input: arr[] = {1, 2, 4, 6, 3, 7, 8}, N = 8

Output: 5

Explanation: The missing number between 1 to 8 is 5

69. Write a Java Program to check if any number is a magic number or not. A number is said to be a magic number if after doing the sum of digits in each step and in turn doing the sum of digits of that sum, the ultimate result (when there is only one digit left) is 1.

```
 // Java program to check if

// a number is Magic number.

class GFG

{

public static boolean isMagic(int n)

{

        int sum = 0;

        // Note that the loop continues

        // if n is 0 and sum is non-zero.

        // It stops when n becomes 0 and

        // sum becomes single digit.

        while (n > 0 || sum > 9)

        {

                if (n == 0)

                {

                        n = sum;

                        sum = 0;

                }
```

```
                    sum += n % 10;

                    n /= 10;

          }

          // Return true if sum becomes 1.

          return (sum == 1);

}

// Driver code

public static void main(String args[])

          {

          int n = 1234;

          if (isMagic(n))

                    System.out.println("Magic Number");

          else

                    System.out.println("Not a magic Number");

          }

}

class InvalidAgeException  extends Exception

{

   public InvalidAgeException (String str)

   {

// calling the constructor of parent Exception

      super(str);

   }

}
```

70. Write a Java program to create and throw custom exceptions.

```
// class that uses custom exception InvalidAgeException
```

```
public class TestCustomException1

{

    // method to check the age

    static void validate (int age) throws InvalidAgeException{

       if(age < 18){

        // throw an object of user defined exception

        throw new InvalidAgeException("age is not valid to vote");

    }

       else {

        System.out.println("welcome to vote");

        }

     }

    // main method

    public static void main(String args[])

    {

        try

        {

            // calling the method

            validate(13);

        }

        catch (InvalidAgeException ex)

        {

            System.out.println("Caught the exception");

            // printing the message from InvalidAgeException object

            System.out.println("Exception occured: " + ex);

        }
        System.out.println("rest of the code...");

    }

}
```

71. Write a Java program to rotate arrays 90 degree clockwise by taking matrices from user input.

```
public class RotateMatrixClockwise

{

public static void main(String args[])

{

//matrix to rotate

int a[][]= {{1,2,3},{4,5,6},{7,8,9}};

System.out.println("Original Matrix: \n");

//loop for rows

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

{

//loop for columns

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

{

//prints the elements of the original matrix

System.out.print(" "+a[i][j]+"\t");

}

System.out.println("\n");

}

System.out.println("Rotate Matrix by 90 Degrees Clockwise: \n");

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

{

for(int j=2;j>=0;j--)
{

//prints the elements of the rotated matrix

System.out.print(""+a[j][i]+"\t");

}

System.out.println("\n");

}

}

}
```

72. Write a java program to check if any number given as input is the sum of 2 prime numbers.

```
// C program to check if a prime number

// can be expressed as sum of

// two Prime Numbers

#include <stdio.h>

#include <math.h>

#include <stdbool.h>

// Function to check whether a number

// is prime or not

bool isPrime(int n)

{

        if (n <= 1)

                return false;

        for (int i = 2; i <= sqrt(n); i++)

        {

                if (n % i == 0)

                        return false;

        }
        return true;
}
// Function to check if a prime number
// can be expressed as sum of
// two Prime Numbers
bool isPossible(int N)
{
        // if the number is prime,
        // and number-2 is also prime
        if (isPrime(N) && isPrime(N - 2))
                return true;
        else
                return false;
}
// Driver code
int main()
{
        int n = 13;
        if (isPossible(n))
                printf("%s", "Yes");
        else
                printf("%s", "No");
        return 0;
}
```

73. Write a Java program for solving the Tower of Hanoi Problem.

```
 // Java recursive program to solve tower of hanoi puzzle
```

```
class GFG

{

        // Java recursive function to solve tower of hanoi puzzle

        static void towerOfHanoi(int n, char from_rod, char to_rod, char aux_rod)

        {

                if (n == 1)

                {

                        System.out.println("Move disk 1 from rod " + from_rod + " to
rod " +to_rod);

                        return;

                }

                towerOfHanoi(n-1, from_rod, aux_rod, to_rod);

                System.out.println("Move disk " + n + " from rod " + from_rod + " to rod "
+to_rod);

                towerOfHanoi(n-1, aux_rod, to_rod, from_rod);

        }

        // Driver method

        public static void main(String args[])

        {

                int n = 4; // Number of disks

                towerOfHanoi(n, \'A\', \'C\', \'B\'); // A, B and C are names of rods

        }

}
```

74. Implement Binary Search in Java using recursion.

```
// Java Program to Illustrate Recursive Binary Search

// Importing required classes

import java.util.*;
```

```
// Main class

class GFG {

    // Method 1

    // Recursive binary search

    // Returns index of x if it is present

    // in arr[l..r], else return -1

    int binarySearch(int arr[], int l, int r, int x)

    {

        // Restrict the boundary of right index

        // and the left index to prevent

        // overflow of indices

        if (r >= l && l <= arr.length - 1) {

            int mid = l + (r - l) / 2;

            // If the element is present

            // at the middle itself

            if (arr[mid] == x)

                return mid;

            // If element is smaller than mid, then it can

            // only be present in left subarray

            if (arr[mid] > x)

                return binarySearch(arr, l, mid - 1, x);

            // Else the element can only be present

            // in right subarray

            return binarySearch(arr, mid + 1, r, x);

        }

        // We reach here when element is not present in
        // array
        return -1;
}
// Method 2
// Main driver method
public static void main(String args[])
{
        // Creating object of above class
        GFG ob = new GFG();
        // Custom input array
        int arr[] = { 2, 3, 4, 10, 40 };
        // Length of array
        int n = arr.length;
        // Custom element to be checked
        // whether present or not
        int x = 10;
        // Calling above method
        int result = ob.binarySearch(arr, 0, n - 1, x);
        // Element present
        if (result == -1)
                // Print statement
                System.out.println("Element not present");
        // Element not present
        else
               // Print statement
                System.out.println("Element found at index "
                                                        + result);

        }

}
```

75. Is delete, next, main, exit or null keyword in java?

No, these keywords do not exist in Java. Delete, Next, Exit are the operations performed in the Java program, Main is the predefined method, and Null is the default String type.

With this we are done with the first section that is Basic Java Interview Question, Now, lets move on to our next section of Intermediate Java Interview Questions.

Java Interview Coding Questions For Intermediate

Now, let's have a look at some of the most asked Java technical interview questions for intermediate experienced professionals.

76. What is JDK? Mention the variants of JDK?

JDK is an abbreviation for Java Development Kit. It is a combined Package of JRE and Developer tools used for designing Java Applications and Applets. Oracle has the following variants.

- JDK Standard Edition
- JDK Enterprise Edition
- JDK Micro Edition

Get Certified in UI & UX Design with UMass Amherst

Free Webinar | Thursday, 19 January | 10 PM ISTRegister Now

77. What is the difference between JDK, JRE, and JVM?

JVM has a Just in Time (JIT) compiler tool that converts all the Java source code into the low-level compatible machine language. Therefore, it runs faster than the regular application.

JRE has class libraries and other JVM supporting files. But it doesn’t have any tool for java development such as compiler or debugger.

JDK has tools that are required to write Java Programs and uses JRE to execute them. It has a compiler, Java application launcher, and an applet viewer.

78. What is a JIT compiler?

JIT compiler refers to Just in Time compiler. It is the simplest way of executing the computer code that takes in compilation during the execution of a program rather than before performance. It commonly uses bytecode translation to machine code. It is then executed directly.

79. What are Brief Access Specifiers and Types of Access Specifiers?

Access Specifiers are predefined keywords used to help JVM understand the scope of a variable, method, and class. We have four access specifiers.

- Public Access Specifier
- Private Access Specifier
- Protected Access Specifier
- Default Access Specifier

80. How many types of constructors are used in Java?

There are two types of constructors in Java.

Parameterized Constructors: Parameterized constructor accepts the parameters with which users can initialize the instance variables. Users can initialize the class variables dynamically at the time of instantiating the class.

Default constructors: This type doesn’t accept any parameters; rather, it instantiates the class variables with their default values. It is used mainly for object creation.

81. Can a constructor return a value?

Yes, A constructor can return a value. It replaces the class's current instance implicitly; you cannot make a constructor return a value explicitly.

82. Explain ‘this’ keyword in Java.

The term "this" is a particular keyword designated as a reference keyword. The "this" keyword is used to refer to the current class properties like method, instance, variable, and constructors.

83. Explain ‘super’ keyword in Java.

The term "super" is a particular keyword designated as a reference keyword. The "super" keyword refers to the immediate parent class object.

84. Explain Method Overloading in Java.

The process of creating multiple method signatures using one method name is called Method Overloading in Java. Two ways to achieve method overloading are:

1. Varying the number of arguments
2. Changing the return type of the Method

85. Can we overload a static method?

No, Java does not support the Overloading of a static method. The process would throw an error reading "static method cannot be referenced."

86. Define Late Binding.

Binding is a process of unifying the method call with the method's code segment. Late binding happens when the method's code segment is unknown until it is called during the runtime.

87. Define Dynamic Method Dispatch.

The Dynamic method dispatch is a process where the method call is executed during the runtime. A reference variable is used to call the super-class. This process is also known as Run-Time Polymorphism.

88. Why is the delete function faster in the linked list than an array?

Delete Function is faster in linked lists in Java as the user needs to make a minor update to the pointer value so that the node can point to the next successor in the list

89. Give a briefing on the life cycle of a thread.

The life cycle of a thread includes five stages, as mentioned below.

1. New Born State
2. Runnable State
3. Running State
4. Blocked State
5. Dead State

90. Explain the difference between >> and >>> operators.

Although they look similar, there is a massive difference between both.

- >> operator does the job of right shifting the sign bits
- >>> operator is used in shifting out the zero-filled bits

91. Brief the life cycle of an applet.

The life cycle of an applet involves the following.

1. Initialization
2. Start

3. Stop
4. Destroy
5. Paint

92. Why are generics used in Java Programming?

Compile-time type safety is provided by using generics. Compile-time type safety allows users to catch unnecessary invalid types at compile time. Generic methods and classes help programmers specify a single method declaration, a set of related methods, or related types with an available class declaration.

93. Explain the Externalizable interface.

The Externalizable interface helps with control over the process of serialization. An "externalisable" interface incorporates readExternal and writeExternal methods.

94. What is the Daemon Thread?

The Daemon thread can be defined as a thread with the least priority. This Daemon thread is designed to run in the background during the Garbage Collection in Java.

The setDaemon() method creates a Daemon thread in Java.

95. Explain the term enumeration in Java.

Enumeration or enum is an interface in Java. Enum allows the sequential access of the elements stored in a collection in Java.

96. Why is Java is Dynamic?

Java is designed to adapt to an evolving environment. Java programs include a large amount of runtime information that is used to resolve access to objects in real-time.

97. Can you run a code before executing the main method?

Yes, we can execute any code, even before the main method. We will be using a static block of code when creating the objects at the class's load time. Any statements within this static block of code will get executed at once while loading the class, even before creating objects in the main method.

98. How many times is the finalize method called?

The finalize method is called the Garbage collector. For every object, the Garbage Collector calls the finalize() method just for one time.

Java Interview Questions for Experienced

Now, lets move on to our last section of Advanced Core Java Interview Questions which is primarly useful for experienced and working professionals.

99. Can "this" and "super" keywords be used together?

No, "this" and "super" keywords should be used in the first statement in the class constructor. The following code gives you a brief idea.

```
public class baseClass {

    baseClass() {

        super();

        this();

        System.out.println(" baseClass object is created");

    }

    public static void main(String []args){

        baseClass bclass = new baseClass();

    }

}
```

100. What is a JSP page?

JSP is an abbreviation for Java Servlet Page. The JSP page consists of two types of text.

- Static Data
- JSP elements

Find Our Java Training in Top Cities

India	United States	Other Countries
Java Training in Bangalore	Java Training New York	Java Course London
Java Training in Chennai	Java Training San Diego	Java Course Singapore
Java Training in Hyderabad	Java Training Dallas	Java Course Melbourne

101. What is JDBC?

JDBC is an abbreviation for Java Database Connector.

JDBC is an abstraction layer used to establish connectivity between an existing database and a Java application

102. Explain the various directives in JSP.

Directives are instructions processed by JSP Engine. After the JSP page is compiled into a Servlet, Directives set page-level instructions, insert external files, and define customized tag libraries. Directives are defined using the symbols below:

start with "< %@" and then end with "% >"

The various types of directives are shown below:

Include directive

It includes a file and combines the content of the whole file with the currently active pages.

Page directive

Page Directive defines specific attributes in the JSP page, like the buffer and error page.

Taglib

Taglib declares a custom tag library, which is used on the page.

103. What are the observer and observable classes?

Objects that inherit the "Observable class" take care of a list of "observers."

When an Observable object gets upgraded, it calls the update() method of each of its observers.

After that, it notifies all the observers that there is a change of state.

The Observer interface gets implemented by objects that observe Observable objects.

Free Course: JavaScript for Beginners

Learn the Basics of JavaScriptEnroll Now

104. What is Session Management in Java?

A session is essentially defined as the random conversation's dynamic state between the client and the server. The virtual communication channel includes a string of responses and requests from both sides. The popular way of implementing session management is establishing a session ID in the client's communicative discourse and the server.

105. Briefly explain the term Spring Framework.

Spring is essentially defined as an application framework in Java and inversion of control containers for Java. The spring framework creates enterprise applications in Java. Especially useful to keep in mind that the spring framework's central features are essentially conducive to any Java application.

106. How to handle exceptions in Spring MVC Framework?

Spring MVC has two approaches for handling the exceptions:

- Exception handler method: In this kind of exception handling, the user will get the @ExceptionHandler annotation type used to annotate a method to handle exceptions.
- XML Configuration: The user can use the SimpleMappingExceptionResolver bean in Spring's application file and map the exception.

107. What is JCA in Java?

Java Cryptography Architecture gives a platform and provides architecture and application programming interfaces that enable decryption and encryption.

Developers use Java Cryptography Architecture to combine the application with the security applications. Java Cryptography Architecture helps in implementing third party security rules and regulations.

Java Cryptography Architecture uses the hash table, encryption message digest, etc. to implement the security.

108. Explain JPA in Java.

The Java Persistence API enables us to create the persistence layer for desktop and web applications. Java Persistence deals in the following:

1. Java Persistence API
2. Query Language
3. Java Persistence Criteria API
4. Object Mapping Metadata

109. Explain the different authentications in Java Servlets.

Authentication options are available in Servlets: There are four different options for authentication in servlet:

Basic Authentication:

Usernames and passwords are given by the client to authenticate the user.

Form-based authentication:

In this, the login form is made by the programmer by using HTML.

Digest Authentication:

It is similar to basic authentication, but the passwords are encrypted using the Hash formula. Hash Formula makes digest more secure.

Client certificate Authentication:

It requires that each client accessing the resource has a certificate that it sends to authenticate itself. Client Authentication requires the SSL protocol.

110. Explain FailFast iterator and FailSafe iterator along with examples for each.

FailFast iterators and FailSafe iterators are used in Java Collections.

FailFast iterators do not allow changes or modifications to the Java Collections, which means they fail when the latest element is added to the collection or an existing element gets removed from the collection. The FailFast iterators tend to fail and throw an exception called ConcurrentModificationException.

Ex: ArrayList, HashMap

Whereas, on the other hand, FailSafe iterators allow changes or modifications to be done on the Java Collections. It is possible, as the FailSafe iterators usually operate on the cloned copy of the collection. Hence, they do not throw any specific exception.

Ex: CopyOnWriteArrayList

111. How do we reverse a string?

The string can be reversed by using the following program.

```
package simplilearnJava;

public class StringReverse {

public static void main(String args[]) {

String str = "Simplilearn";

String reverse = new StringBuffer(str).reverse().toString();

System.out.printf("Actual Word: %s, Word after reversing %s", str, reverse);

}

public static String reverse(String source) {

if (source == null || source.isEmpty()) {

return source;

}

String reverse = "";
for (int i = source.length() - 1; i >= 0; i--) {

reverse = reverse + source.charAt(i);

}

return reverse;

}

}
```

Expected Output:

```
Actual Word: Simplilearn, Word after reversing nraelilpmiS
```

112. Write a program to find the square root of a number.

The Square root of a number can be found by using the following program.

```
package simplilearnJava;

import java.util.Scanner;

public class SRoot {

public static void main(String args[]) {

try (Scanner sc = new Scanner(System.in)) {

System.out.println("Input a number to find square root: ");

double square = sc.nextDouble();

double squareRoot = Math.sqrt(square);

System.out.printf("The square root is: %f ", squareRoot);

}

}

}
```

Expected Output:

```
Input a number to find square root:

25

The square root is: 5
```

Free Course: Java Hibernate Fundamentals

Learn Java Hibernate Basic Skills for FreeEnroll Now

113. Write a program that detects the duplicate characters in a string.

The program that finds the duplicate elements in a string is written below:

```
package simplilearnJava;

import java.util.HashMap;

import java.util.Map;

import java.util.Set;

public class FindDuplicate {

public static void main(String args[]) {

printDuplicateCharacters("Simplilearn");

}

public static void printDuplicateCharacters(String word) {

char[] characters = word.toCharArray();

Map<Character, Integer> charMap = new HashMap<Character, Integer>();

for (Character ch : characters) {

if (charMap.containsKey(ch)) {

charMap.put(ch, charMap.get(ch) + 1);

} else {

charMap.put(ch, 1);

}

}

Set<Map.Entry<Character, Integer>> entrySet = charMap.entrySet();

System.out.printf("List of duplicate characters in String '%s' %n", word);

for (Map.Entry<Character, Integer> entry : entrySet) {

if (entry.getValue() > 1) {
```

```
System.out.printf("%s: %d %n", entry.getKey(), entry.getValue());

}

}

}

}
```

Expected output:

```
List of duplicate characters in String 'Simplilearn.'

i: 2

l: 2
```

114. Write a Program to remove duplicates in an ArrayList.

The following program can be implemented to remove duplicate elements in an ArrayList

```
package simplilearnJava;

import java.util.ArrayList;

import java.util.LinkedHashSet;

import java.util.List;

import java.util.Set;

public class ArrayDuplicate {

public static void main(String args[]) {

List<Integer> num = new ArrayList<Integer>();

num.add(1);

num.add(2);

num.add(3);

num.add(4);

num.add(5);

num.add(6);

num.add(3);
num.add(4);

num.add(5);

num.add(6);

System.out.println("Your list of elements in ArrayList : " + num);

Set<Integer> primesWithoutDuplicates = new LinkedHashSet<Integer>(num);

num.clear();

num.addAll(primesWithoutDuplicates);

System.out.println("list of original numbers without duplication: " + num);

}

}
```

Expected Output:

```
Your list of elements in ArrayList : [1, 2, 3, 4, 5, 6, 3, 4, 5, 6]

list of original numbers without duplication: [1, 2, 3, 4, 5, 6]
```

115. Find the word count in a string using HashMap Collection.

The following program can be used for word count.

```
package simplilearnJava;

import java.util.HashMap;

public class WordCount {

public static void main(String[] args) {

String str = "Hello World, Welcome to Simplilearn";

String[] split = str.split(" ");

HashMap<String, Integer> map = new HashMap<String, Integer>();

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

if (map.containsKey(split[i])) {

int count = map.get(split[i]);

map.put(split[i], count + 1);
} else {

map.put(split[i], 1);

}

}

System.out.println(map);

}

}
```

Expected Output:

```
{Hello=1, Simplilearn=1, Welcome=1, to=1, World,=1}
```

116. Write a program to find the Second Highest number in an ArrayList

The following program can be used to find the second biggest number in an array list.

```
package simplilearnJava;

public class NextHighest {

public static void main(String[] args)

  {

    int array[] = { 1, 2, 3, 4, 11, 12, 13, 14, 21, 22, 23, 24, 31, 32};

    int high = 0;

    int nextHigh = 0;

    System.out.println("The given array is:");

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

    {

      System.out.print(array[i] + "\t");

    }

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

    {

      if (array[i] > high)
```

```
            {
                nextHigh = high;
                high = array[i];
            }
            else if (array[i] > nextHigh)
            {
                nextHigh = array[i];
            }
        }
        System.out.println("Second Highest is:" + nextHigh);
        System.out.println("Highest Number is: "  +high);
    }
}
```

Expected Output:

```
The given array is:
1 2 3 4 11 12 13 14 21 22 23 24 31 32
Second Highest is:31
The highest number is: 32
```

117. What is the difference between System.out, System.err, and System.in?

System.out and System.err represent the monitor by default and thus can be used to send data or results to the monitor. System.out is used to display normal messages and results. System.eerr is used to display error messages. System.in represents InputStream object which by default represents standard input device, i.e., keyboard.

118. Could you provide some implementation of a Dictionary having a large number of words?

The simplest implementation that can be given is that of a List wherein one can place ordered words and perform a Binary search. The other implementation with a better search performance is HashMap where the key is used as the first character of the word and the value as a LinkedList.

Up another level, there are HashMaps like:

hashmap {

a (key) -> hashmap (key-aa , value (hashmap(key-aaa,value)

b (key) -> hashmap (key-ba , value (hashmap(key-baa,value)

z (key) -> hashmap (key-za , value (hashmap(key-zaa,value)

}

Up to n levels where n is the average size of the word in the dictionary.

119. How would you tackle it if you might have to encounter pattern programs in Java?

Solution - Top 25 Most Frequently asked Pattern Programs in Java

With this, we have come to the end of this Java Interview Questions article. Moving ahead, we will look into the next crucial steps that you could pursue, to master Java.

120. What do you understand by an instance variable and a local variable?

Generally, instance variables are declared in a class but outside methods whereas a local variable is declared within the blocks of code.

//Local Variable

import Java.io.*;

class Main {

public static void main(String[] args)

{

int var = 145;

System.out.println("Local Variable: " + var);

}

}

//Instance variable

import Java.io.*;

class Main {

```
public int value = 12;

public static void main(String[] args)

{

Main va = new Main();

System.out.println("My value is: " + va.value);

}

}
```

121. Can the main method be overloaded?

Yes, the main method can be overloaded as many times as we want. Nevertheless, JVM prefers to call the main method with the help of its predefined calling method.

Example:

```
class Main {

   public static void main(String args[]) {

      System.out.println(" Main Method");

   }

   public static void main(int[] args){

      System.out.println("Overloaded Integer array Main Method");

   }

   public static void main(char[] args){

      System.out.println("Overloaded Character array Main Method");

   }

   public static int main(double[] args){

      System.out.println("Overloaded Double array Main Method");

   }

   public static void main(float args){

      System.out.println("Overloaded float Main Method");

   }
}
```

122. Comment on method overloading and overriding by citing relevant examples.

Method overloading occurs during the compile time, whereas method overriding occurs during the run time. Static binding is used during overloading, whereas dynamic binding is used during methods overriding.

```
//Function overloading

#function1

void addPodium(int a, int b)

{

System.out.println(a + b);

}

#function2

float addPodium(float a, float b, float c)

{

System.out.println(a + b + c);

}

//Function overriding

class Parent {

   void show()

   {

      System.out.println("I am Parent");

   }

}

class Child extends Parent {

   void show()

   {
        System.out.println("I am Child");

    }

}

class Main {

    public static void main(String[] args)

    {

        Parent obja = new Parent();

        obja.show();

        Parent objb = new Child();

        objb.show();

    }

}
```

123. A single try block and multiple catch blocks can co-exist in a Java Program. Explain.

One or more catch blocks can follow a try block. Each catch block must have a unique exception handler. So, if you want to perform multiple tasks in response to various exceptions, use the Java multi-catch block.

124. Do final, finally and finalize keywords have the same function?

No, final, finally and finalize keywords have different functionalities.

Final is used to restrict classes, variables, or methods, the final keyword.

Finally is used to execute the code written inside the block without handling any exceptions.

Finalize is used to call the function of the implementation of cleaning the garbage collection of an object.

125. When can you use the "super" keyword?

Basically, the super keyword is used to refer to the parent class. When there are the same fields in both parent and child classes, then one can use a super keyword to access data members of the parent class.

126. What are shallow copy and deep copy in Java?

In the case of a shallow copy, primitive data types are copied, whereas in the case of a deep copy along with primitive data types the object references are also copied.

127. Using relevant properties highlight the differences between interfaces and abstract classes.

An abstract class can have a combination of both abstract and non-abstract methods, whereas an interface has only abstract methods in it.

128. What are the different ways of thread usage?

There are two ways to define and implement a thread in Java. They are by implementing the runnable interface and extending the thread class.

Extending the Thread class

```
class InterviewBitThreadExample extends Thread{

  public void run(){

     System.out.println("Thread runs...");

  }

  public static void main(String args[]){

     InterviewBitThreadExample ib = new InterviewBitThreadExample();

     ib.start();

  }

}
```

Implementing the Runnable interface

```
class InterviewBitThreadExample implements Runnable{

  public void run(){

     System.out.println("Thread runs...");

  }

  public static void main(String args[]){

     Thread ib = new Thread(new InterviewBitThreadExample());

     ib.start();

  }
}
```

Implementing a thread using the method of Runnable interface is more preferred and advantageous as Java does not have support for multiple inheritances of classes.

start() method is used for creating a separate call stack for the thread execution. Once the call stack is created, JVM calls the run() method for executing the thread in that call stack.

129. What is the difference between the ‘throw' and ‘throws' keyword in Java?

The throw keyword is often used to explicitly throw an exception. It can only throw one exception at a time whereas throws can be used to declare multiple exceptions.

130. Identify the output of the below Java program and Justify your answer.

```
class Main {

   public static void main(String args[]) {

      Scaler s = new Scaler(5);

   }

}

class InterviewBit{

   InterviewBit(){

      System.out.println(" Welcome to InterviewBit ");

   }

}

class Scaler extends InterviewBit{

   Scaler(){

      System.out.println(" Welcome to Scaler Academy ");

   }

   Scaler(int x){

      this();

      super();
```

```
        System.out.println(" Welcome to Scaler Academy 2");

    }

}
```

The above code will throw the compilation error. It is because the super() is used to call the parent class constructor. But there is the condition that super() must be the first statement in the block. Now in this case, if we replace this() with super() then also it will throw the compilation error. Because this() also has to be the first statement in the block. So in conclusion, we can say that we cannot use this() and super() keywords in the same block.

131. Java works as a “pass by value” or “pass by reference” phenomenon?

Java works as a “pass by value” phenomenon, because “pass by reference” needs the help of pointers. But there are no pointers in Java.

132. How to not allow serialization of attributes of a class in Java?

One approach to not allow serialization of attributes of a class in Java is by using writeObject() and readObject() methods in the subclass and throwing a not Serializable exception.

133. What are the default values assigned to variables and instances in Java?

By default, for a numerical value it is 0, for the boolean value it is false and for objects it is NULL.

134. What do you mean by data encapsulation?

Data encapsulation is one of the properties of OOPS concepts, where all the data such as variables and methods are enclosed together as a single unit.

135. Can you tell the difference between equals() method and equality operator (==) in Java?

Equality operator (==) is used to check the equality condition between two variables. But the equals() method is used to check the equality condition between two objects.

136. How is an infinite loop declared in Java?

An infinite loop can be declared in Java by breaking the logic in the instruction block. For example,

```
for(int i = 1; i > 0; i++)
```

```
{

//statements

}
```

The above code forms an infinite loop in Java.

137. Briefly explain the concept of constructor overloading

The concept of constructor overloading refers to having multiple methods in a class with their name being the same as the class name. The difference lies in the set of parameters passed to the functions.

138. Explain the use of the final keyword in variable, method and class.

In Java, one can apply the final keyword to a variable, methods, and class. With the help of the final keyword, the variable turns out to be a constant, the method cannot be inherited and the class cannot be overridden.

139. Is it possible that the ‘finally' block will not be executed? If yes then list the case.

Yes, there is a possibility that the ‘finally’ block cannot get executed. Here are some of the cases where the above situation occurs.

1. During the time of fatal errors such as memory exhaustion, memory access error, etc.
2. During the time of using System.exit()

140. Difference between static methods, static variables, and static classes in Java.

A variable, method, or class can be made static by using the static keyword. A static class cannot be instantiated. When both objects or instances of a class share the same variables, this is referred to as static variables. Static methods are simply methods that refer to the class in which they are written.

141. What is the main objective of garbage collection?

The main goal of using garbage collection is to free the heap memory by eliminating unnecessary objects.

142. Apart from the security aspect, what are the reasons behind making strings immutable in Java?

Because of security, synchronization, concurrency, caching, and class loading, the String is immutable in Java. The reason for making string final would be to destroy its immutability and help stop others from trying to extend it. String objects are cached in the

String pool, making them immutable.

143. Which of the below generates a compile-time error? State the reason.

```
int[] n1 = new int[0];

boolean[] n2 = new boolean[-200];

double[] n3 = new double[2241423798];

char[] ch = new char[20];
```

We get a compile-time error in line 3. The error we will get in Line 3 is - the integer number too large. It is because the array requires size as an integer. And Integer takes 4 Bytes in the memory. And the number (2241423798) is beyond the capacity of the integer. The maximum array size we can declare is - (2147483647).

Because the array requires the size in integer, none of the lines (1, 2, and 4) will give a compile-time error. The program will compile fine. But we get the runtime exception in line 2. The exception is - NegativeArraySizeException.

Here what will happen is - At the time when JVM will allocate the required memory during runtime then it will find that the size is negative. And the array size can't be negative. So the JVM will throw the exception.

144. How would you differentiate between a String, StringBuffer, and a StringBuilder?

The string class is immutable but the other two are mutable in nature. StringBuffer is synchronous whereas the StringBuilder is asynchronous. String uses string pool as memory storage whereas the other two use heap memory for storage purposes.

145. What is a Comparator in Java?

A comparator is an interface, which is used to sort the objects.

146. In Java, static as well as private method overriding is possible. Comment on the statement.

In Java, you could indeed override a private or static method. If you create a similar method in a child class with the same return type and method arguments, it will hide the super class method; this is known as method hiding. Similarly, you cannot override a private method in a subclass because it is not accessible from that.

147. What makes a HashSet different from a TreeSet?

In a HashSet, the elements are unsorted and work faster than a Tree set. It is implemented using a hash table.

148. Why is the character array preferred over string for storing confidential information?

Because Strings are immutable, any change will result in the creation of a new String, whereas char[] allows you to set all of the elements to blank or zero. So storing a password in a character array clearly reduces the security risk of password theft.

149. What are the differences between HashMap and HashTable in Java?

HashMap	HashTable
1. Asynchronous in nature	1. Synchronous in nature
1. Not thread-safe	2. Thread safe
1. It allows one null key and null values	3. It doesn’t allow null keys and values.

150. What is the importance of reflection in Java?

Reflection is a property of Java, enabling the Java code to inspect itself. A Java class, for example, can get the names of all its members and showcase them.

151. What are the different types of Thread Priorities in Java? And what is the default priority of a thread assigned by JVM?

There are different types of thread properties in Java. They are MIN_PRIORITY, MAX_PRIORITY, and NORM_PRIORITY. By default, the thread is assigned NORM_PRIORITY.

152. What is the ‘IS-A ‘ relationship in OOPs Java?

‘IS-A’ relationship is related to the Inheritance property of OOPs Java. It is a kind of parent-child relationship that is established between two classes.

> We offer complete Job Guarantee and money-back if you don't secure a job within 6 months of graduation. Get interview ready with intense and comprehensive career mentoring sessions in our Full Stack Java Developer Program. Enroll TODAY!

Next Steps

Java Interview Questions are really important to go through before attending an interview. It helps you to put yourself on the safer side by getting you ready to be able to answer the questions asked in your interview. Now, the next step is to learn How to Become a Software Developer.

You can also explore and get familiar with interview questions and answers related to other backend languages like PHP, Python and Node.js Interview Questions.

If you're looking for more in-depth knowledge about the Java programming language and information on how to get certified as a web development professional developer, explore our Java training and certification programs, which Simplilearn's experienced industry experts offer in real-time. In particular, check out our Full Stack Developer Course in India today!

If you have any questions about this "Java Interview Questions" article, please leave them in the comments section, and our experts will answer them for you, at the earliest!

Find our Full Stack Java Developer Online Bootcamp in top cities:

Name	Date	Place	
Full Stack Java Developer	Cohort starts on 19th Jan 2023, Weekend batch	Your City	View Details
Full Stack Java Developer	Cohort starts on 8th Feb 2023, Weekend batch	Your City	View Details

About the Author

Ravikiran A S
Ravikiran A S works with Simplilearn as a Research Analyst. He an enthusiastic geek always in the hunt to learn the latest technologies. He is proficient with Java Programming Language, Big Data, and powerful Big Data Frameworks like Apache Hadoop and Apache Spark.

View More