Top 50 Java Programs from Coding Interviews

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Coding is an integral part of any programming job interviews, and Java development interviews are no exception. I would even suggest you should never hire anyone without testing their coding skill, coding is also an art, and more often than a good code is an excellent developer as well. If you look at tech giants like Amazon, Facebook, and Google they thoroughly test the coding skill of any developer they hire, notably Amazon who first send online coding exercises to filter Java programmers who can code. This online test usually gives you requirements and ask you to write a program in a limited time, usually 2 to 3 hours. The application should meet the output provided by the exercise itself. These type of tasks are very tough to crack if you don't have excellent coding skill.

Btw, the most crucial question is how do you develop that kind of coding skill in the first place? Well, things always start small, and if you pay attention, there are many Java job interviews where you would have been asked to write small programs.

They are simple, but yet they give a good indication of the coding skill of prospective candidates. They are usually preferred by many companies because it often requires 10 to 20 minutes to write the solution and discuss them.

In this list, I am going to share 50 of such small programs from Java Programming interviews. These programs are from various Data Structure and Algorithm topics like an array, string, linked list, binary tree, etc. If you don't have a good knowledge of Data structure and algorithm, I suggest you to first read a good book on Data Structure and Algorithms like Introduction to Algorithms by Thomas H. Cormen.

But, If you find the book challenging, you can also join a comprehensive online course on Data structure and algorithms like **Data Structures and Algorithms: Deep Dive Using Java** on Udemy to learn better and more comfortable.

Top 50 Java Programs from Coding Interviews

Here is a big list of Java programs for Job Interviews. As I said, it includes questions from problem-solving, linked list, array, string, matrix, bitwise operators and other miscellaneous parts of programming. Once you are gone through these questions,

you can handle a good number of questions on real Job interviews.

1. Fibonacci series (solution)
Write a simple Java program which will print Fibonacci series, e.g. 1 1 2 3 5 8 13 ... . up to a given number. We prepare for cross questions like using iteration over recursion and how to optimize the solution using caching and memoization.

2. A prime number (solution)
Write a Java program to check if a given number is prime or not. Remember, a prime number is a number which is not divisible by any other number, e.g. 3, 5, 7, 11, 13, 17, etc. Be prepared for cross, e.g. checking till the square root of a number, etc.

3. String Palindrome (solution)
You need to write a simple Java program to check if a given String is palindrome or not. A Palindrome is a String which is equal to the reverse of itself, e.g., "Bob" is a palindrome because of the reverse of "Bob" is also "Bob." Though be prepared with both recursive and iterative solution of this problem. The interviewer may ask you to solve without using any library method, e.g. indexOf() or `subString()` so be prepared for that.

4. Integer Palindrome (solution)
This is generally asked as follow-up or alternative of the previous program. This time you need to check if given Integer is palindrome or not. An integer is called palindrome if it's equal to its reverse, e.g. 1001 is a palindrome, but 1234 is not because the reverse of 1234 is 4321 which is not equal to 1234. You can use divide by 10 to reduce the number and modulus 10 to get the last digit. This trick is used to solve this problem.

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

5. Armstrong number (solution)
A number is called an Armstrong number if it is equal to the cube of its every digit. For example, 153 is an Armstrong number because of 153= 1+ 125+27, which is equal to 1^3+5^3+3^3. You need to write a program to check if the given number is Armstrong number or not.

6. Avoiding deadlock in Java (solution)
This is one of the interesting programs from Java Interviews, mostly asked to 2 to 3 years of experienced programmers or higher. The interviewer simply asked you to write code where a resource is accessed by multiple threads. You need to write code in such a way that no deadlock should occur. The trick to solving this problem is acquiring resources in order and release them in reverse order, e.g. first acquire resource R1 and only if you have got R1 to go for R2. This way, you can avoid deadlock.

7. Factorial (solution)
This is one of the simplest programs you can expect in interviews. It is generally asked to see if you can code or not. Sometimes interviewer may also ask about changing a recursive solution to iterative one or vice-versa.

8. Reverse a String (solution)
This problem is similar to the String Palindrome problem we have discussed above. If you can solve that problem, you can solve this as well. You can use `indexOf()` or `substring()` to reverse a String or alternatively, convert the problem to reverse an array by operating on character array instead of String. If you want to brush up your data structure skill you can also check Data Structures and Algorithms: Deep Dive Using Java course on Udemy before solving this question.

9. Remove duplicates from an array (solution)
Write a program to remove duplicates from an array in Java without using the Java Collection API. The array can be an array of String, Integer or Character, your solution should be independent of the type of array. If you want to practice more array-based questions, then see this list of top 30 array interview questions from

Java interviews.

10. Printing patterns (solutions)

11. Print repeated characters of String? (solution)

12. GCD of two numbers (solution)

13. The square root of a number (solution)
You need to write a program to calculate the square root of a number without using the Math.sqrt() function of JDK. You need to write your logic and method to calculate the square root. You can though use the popular algorithm, like Newton's method.

14. Reverse array in place (solution)

15. Reverse words of a sentence (solution)

16. Leap year (solution)

17. Binary search (solution)

18. String Anagram (solution)
Write a program to check if two given String is Anagram of each other. Your function should return true if two Strings are Anagram, false otherwise. A string is said to be an anagram if it contains the same characters and same length, but in a different order, e.g. army and Mary are anagrams. You can ignore cases for this problem, but you should clarify that from your interview.

19. Design a Vending Machine (solution)
This one of the popular OOAD (object-oriented analysis and design) question from Java Interviews. You will be given 3 hours to design and code a vending machine satisfying some of the business requirements. You also need to write unit tests to prove your code satisfy those requirements. You can see this article for more object-oriented analysis questions.

20. Reverse a number (solution)

21. The first non-repeated character of String (solution)

22. Finding Middle element of linked list in one pass (solution)

23. Pre-order traversal (solution)

24. Pre-order traversal without recursion (solution)

25. In order traversal (solution)

26. In order traversal without recursion (solution)

27. Post-order traversal (solution)

28. Postorder traversal without recursion (solution)

29. Print all leaves of a binary tree (solution)

30. Sort array using quicksort (solution)
You need to write a Java program to sort an array of integers using a quick sort algorithm. You cannot use any library method, e.g. JDK or a third party library, which means, you need to first implement the quicksort algorithm and then sort the array.

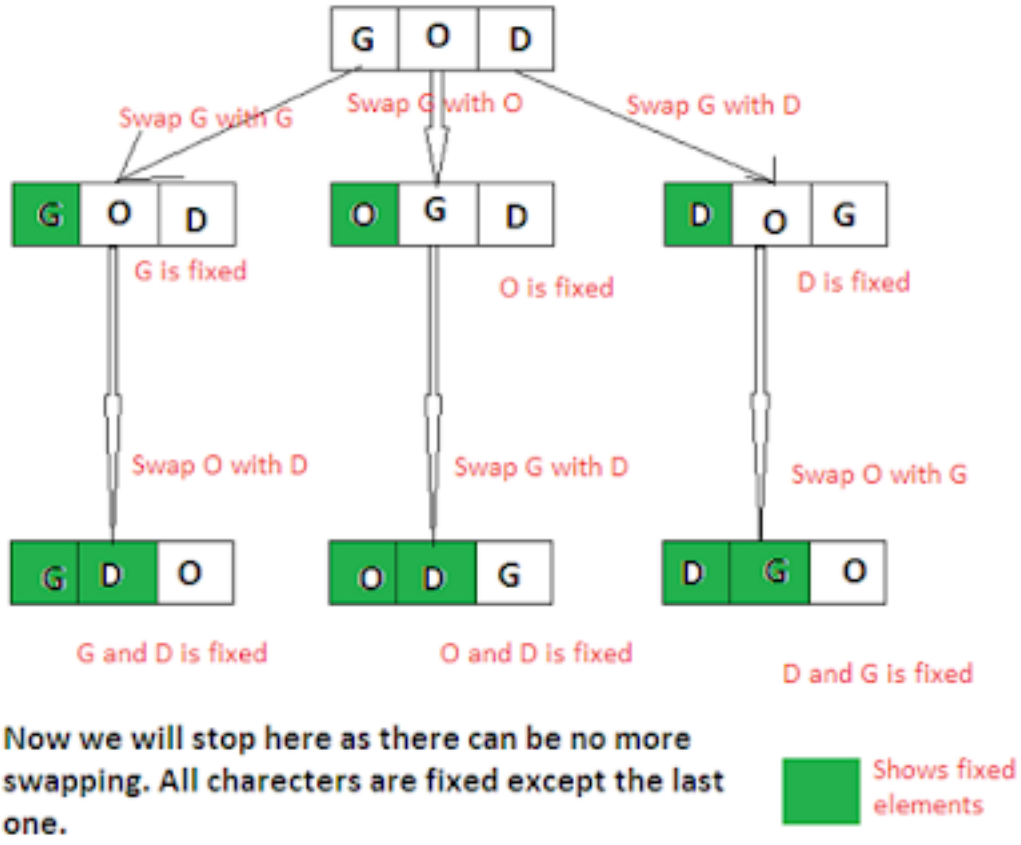
31. Insertion sort (solution)
Write a program to implement the insertion sort algorithm in Java. The program should take an unsorted array and sort it using insertion sort algorithm Also explain the best case and worst case time and space complexity of the Insertion sort algorithm.

32. Bubble sort (solution)
Write a program to implement the bubble sort algorithm in Java. You can use basic

operators and functions, but sorting functions from Java API is not allowed.

33. Transpose a matrix (solution)

34. Print all permutations of String (solution)
Write a Java program to print all permutations of a given String. For example, if given String is "GOD" then your program should print all 6 permutations of this string, e.g. "GOD," "OGD," "DOG," "GDO," "ODG," and "DGO."



35. Reverse a String in place (solution)

36. Adding two matrices in Java (solution)

37. Matrix multiplication (solution)

AJ's Guide To

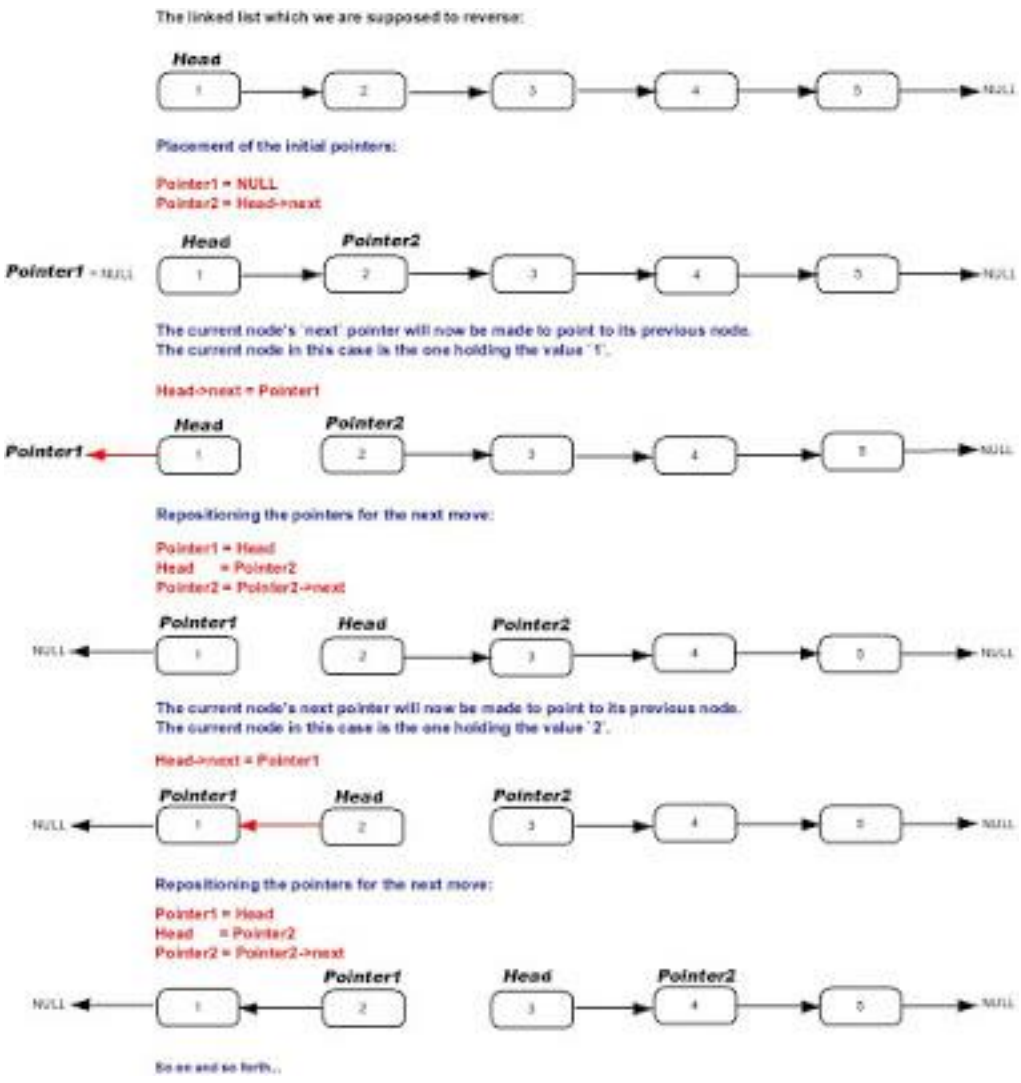
Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

38. Removal all white space from String (solution)

39. Reverse a linked list (solution)

Write a program to reverse a singly linked list in Java. You can use iteration and recursion to solve this problem, but you should reverse a linked list in place.



40. Find the length of the linked list (solution)

Just write a program in Java to find the length of a singly linked list in one pass, i.e. in just one iteration of a singly linked list.

41. Check if a linked list has a loop (solution)

Write a program to check if given linked list has a loop or not. Sometimes a linked list get corrupt, and two nodes point to the same node, which forms the loop or cycle in the linked list.

42. Find the start of loop in a linked list (solution)

43. Find the middle element of a linked list (solution)

44. Find the 3rd element from the tail linked list (solution)
You need to write a program to find the 3rd element from the tail of a singly linked list. You need to solve this problem without iterating twice. If you want more linked list questions, you can see the article about frequently asked linked list interview questions.

44. Convert a linked list to a binary tree (solution)
It's possible to convert a doubly-linked list to a binary tree, you need to write a Java program which takes a doubly-linked list and returns a binary tree.

45. Sort a linked list (solution)
You need to given an unsorted linked list, and you need to write a program in Java to sort them in ascending order of the values in each node.

46. Iterative Quicksort (solution)

You need to write a Java program to implement quicksort sorting algorithm without recursion. You can use essential JDK classes and programming constructs, but recursion is not allowed.

46. Bucket sort (solution)

This program is increasingly getting popular on Java interview because it sorts a given array in linear time. Though there is a lot of prerequisites, e.g. you must know the maximum value present in the array, it is a very interesting problem from interview point of view. You need to write a program to implement a bucket sort algorithm in Java. If you are not familiar with Bucket sort or any other linear sorting algorithm, I suggest you to first read a good on algorithms, e.g. Introduction to Algorithms by Thomas H. Cormen.

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

47. Counting sort (solution)

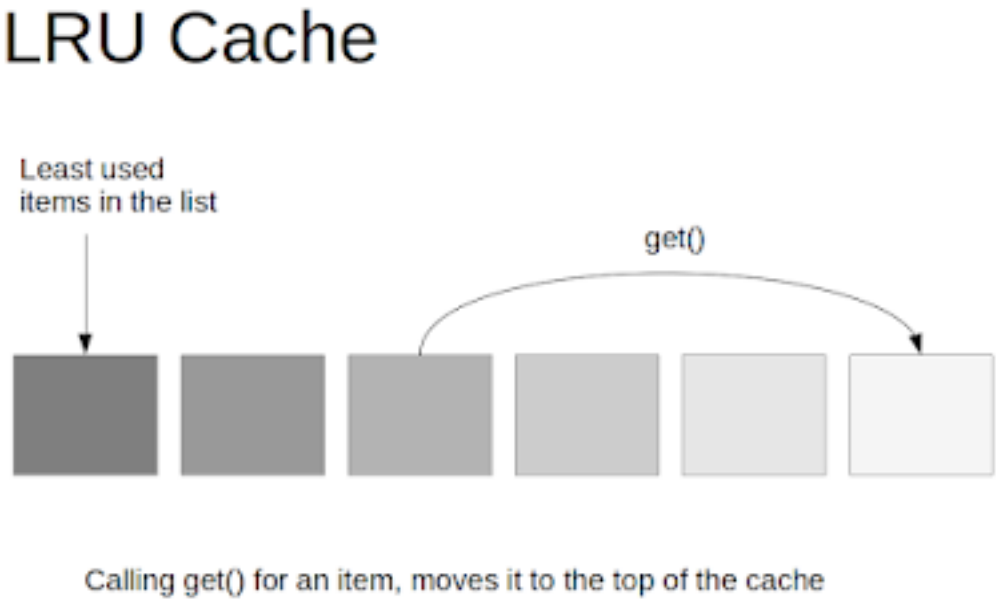
This is another problem which is similar to the previous one because counting sort is also a linear sorting algorithm. Just remember that bucket sort, and counting sort are different algorithms, so it's also good to state how they are different.

48. Check if two string rotation of each other

Write a program which accepts two given String and checks if they are the rotation of each. If they then return true otherwise return false. A String is said to be a rotation of other string if they contain same characters and the sequence is rotated across any character, e.g. "`dabc`" is a rotation of "`abcd`" but "dbac" is not. If you want to practice more string-based questions, you can also see my list of 20 String-based algorithm questions from Java interviews.

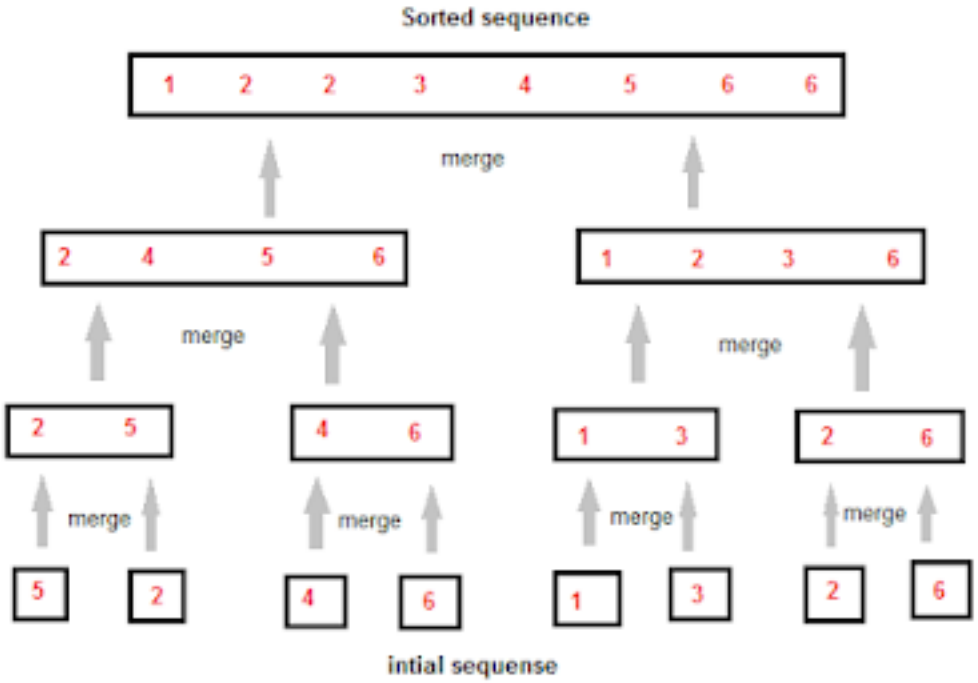
49. LRU cache in Java (solution)

Write a program to implement an LRU cache in Java. An LRU cache means Least Recently Used Cache which removes the least recently used element if the cache is full. You can use LinkedHashMap to implement LRU cache in Java.



50. Merge sort

Implement the merge sort algorithm in Java. You can write a recursive or iterative solution, whichever you like. You also need to explain the time and space complexity for the best, worst, and average case.



That's all about **top 50 programs from Java interviews**. You can practice these Java programs even if you are not preparing for any Job interview. They not only help you to do well on your programming job interviews but also on learning how to code and developing your programming skill and coding sense.

These small programs touch several important areas, e.g. popular data structures like an array, linked list, binary tree, binary search tree, string, etc., popular algorithms, e.g. sieve of the Eratosthenes algorithm for generating primes, the Euclidean algorithm for calculating LCM and GCF, Fibonacci sequence, printing patterns and so on.

These programs also touch base on useful operators like bit-shift and bitwise operators, modulus operators, and others. Overall, you get a good understanding of the Java programming world by practicing these Java programs from coding interviews.

Top 20 Java Interview Programs For Programming And Coding Interview

Last Updated:August 21, 2019

In this tutorial, we have provided a big list of basic Java interview programs with actual logical code examples asked in Programming and Coding Interviews for freshers and experienced candidates.

Important and basic Java programs that are generally asked in the technical round of Java and Automation Interviews.

This has now become a general practice by the interviewers to ask basic Java Programs in interviews rather than just focusing on theoretical aspects.
For this, we have come up with an idea of listing down a few very important Java Programs along with the proper explanation of each program.

Moreover, we have also included the respective outputs which will give you a fair idea about how that program worked. The flow of the program and the concepts are properly explained wherever possible throughout this article.

Most Popular Java Programming Interview Questions

A list of the most popular Java Programming interview questions and answers are explained below and these questions will help you to clear any Automation Interview successfully.

Recommended Read => We have covered Core Java Interview Questions in earlier articles here.

Q #1) Write a Java Program to reverse a string without using String inbuilt function.

Answer: Here, we are initializing a string variable str and are making use of the string builder class.

The object of the string builder class str2 will be further used to append the value stored in the string variable str.

Thereafter, we are using the inbuilt function of string builder (reverse()) and storing the new reversed string in str2.

Finally, we are printing str2.

```
1 public class FinalReverseWithoutUsingStringMethods {
2
3           public static void main(String[] args) {
4                   // TODO Auto-generated method stub
5                   String str = "Automation";
6                   StringBuilder str2 = new StringBuilder();
7                   str2.append(str);
8                   str2 = str2.reverse();      // used string builder to
 reverse
9                    System.out.println(str2);
10           }
11
12 }
```

Output:

noitamotuA

Q #2) Write a Java Program to reverse a string without using String inbuilt function reverse().

Answer:

Method 1:

There are several ways with which you can reverse your string if you are allowed to use the other string inbuilt functions.

In this method, we are initializing a string variable called str with the value of your given string. Then, we are converting that string into character array with toCharArray() function. Thereafter, we are using for loop to iterate between each character in reverse order and printing each character.

```
1 public class FinalReverseWithoutUsingInbuiltFunction {
2     public static void main(String[] args) {
3         String str = "Saket Saurav";
```

```
4         char chars[] = str.toCharArray();  // converted to character
  array and printed in reverse order
5         for(int i= chars.length-1; i>=0; i--) {
6             System.out.print(chars[i]);
7         }
8     }
9 }
```

Output:
varuaS tekaS

Method 2:
This is another method in which you are declaring your string variable str and then using Scanner class to declare an object with predefined standard input object.

This program will accept the string value through the command line (when executed).

We have used nextLine() which will read the input with the spaces between the words of a string. Thereafter, we have used a split() method to split the string into its substrings(no delimiter given here). Finally, we have printed the string in reverse order using for loop.

```
1 import java.util.Scanner;
2
3 public class ReverseSplit {
4
5     public static void main(String[] args) {
          // TODO Auto-generated method
6   stub
7         String str;
8         Scanner in = new Scanner(System.in);
9          System.out.println("Enter your String");
10         str = in.nextLine();
           String[] token = str.split("");    //used split method to print
11    in reverse order
12         for(int i=token.length-1; i>=0; i--)
13         {
14             System.out.print(token[i] + "");
15         }
16
17     }
18
19 }
```

Output:

```
Enter your String
Softwaretestinghelp
plehgnitseterawtfoS
```

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Method 3:

This is almost like method 2, but here we did not use the split() method. We have used the scanner class and nextLine() for reading the input string. Then, we have declared an integer length which has the length of the input string.

Thereafter, we have printed the string in the reverse order using for loop. However, we have used charAt(index) method which will return the character at any specific index. After each iteration, the character will be concatenated to reverse the string variable.

Finally, we have printed the reverse string variable.

```
1 import java.util.Scanner;
2
3 public class Reverse {
4
5     public static void main(String[] args) {
          // TODO Auto-generated method
6 stub
7         String original, reverse = "";
8         System.out.println("Enter the string to be reversed");
9          Scanner in = new Scanner(System.in);
10         original = in.nextLine();
11         int length = original.length();
12         for(int i=length-1; i>=0; i--) {
              reverse = reverse + original.charAt(i);    //used inbuilt
13 method charAt() to reverse the string
14         }
15         System.out.println(reverse);
16     }
17
18 }
```

Output:

Enter the string to be reversed
automation testing
gnitset noitamotua

Q #3) Write a Java Program to swap two numbers with using the third variable.

Answer: In this Example, we have made use of the Scanner class to declare an object with a predefined standard input object. This program will accept the values of x and y through the command line (when executed).
We have used nextInt() which will input the value of an integer variable ‘x' and ‘y' from the user. A temp variable is also declared.

Now, the logic of the program goes like this – we are assigning temp or third variable with the value of x, and then we are assigning x with the value of y and again we are assigning y with the value of temp. So, after the first complete iteration, the temp will have a value of x, x will have a value of y and y will have a value of temp (which is x).

```
1 import java.util.Scanner;
2
3 public class SwapTwoNumbers {
4
5     public static void main(String[] args) {
        // TODO Auto-generated method
6 stub
7         int x, y, temp;
8         System.out.println("Enter x and y");
9          Scanner in = new Scanner(System.in);
10         x = in.nextInt();
11         y = in.nextInt();
12         System.out.println("Before Swapping" + x + y);
13         temp = x;
14         x = y;
15         y = temp;
16         System.out.println("After Swapping" + x + y);
17
18     }
19
20 }
```

Output:
Enter x and y
45
98
Before Swapping4598
After Swapping9845

Q #4) Write a Java Program to swap two numbers without using the third variable.
Answer: Rest all things will be the same as the above program. Only the logic will change. Here, we are assigning x with the value x + y which means x will have a sum of both x and y.

Then, we are assigning y with the value x – y which means we are subtracting the value of y from the sum of (x + y). Till here, x still has the sum of both x and y. But y has the value of x.

Finally, in the third step, we are assigning x with the value x – y which means we are subtracting y (which has the value of x) from the total (x + y). This will assign x with the value of y and vice versa.

```
1 import java.util.Scanner;
2
3 class SwapTwoNumberWithoutThirdVariable
4 {
5    public static void main(String args[])
6    {
7       int x, y;
8       System.out.println("Enter x and y");
9        Scanner in = new Scanner(System.in);
10
11       x = in.nextInt();
12       y = in.nextInt();
13
14       System.out.println("Before Swapping\nx = "+x+"\ny = "+y);
15
16       x = x + y;
17       y = x - y;
18       x = x - y;
19
20       System.out.println("After Swapping without third variable\nx =
"+x+"\ny = "+y);
21    }
22 }
```

Output:

Enter x and y
45
98
Before Swapping
x = 45
y = 98
After Swapping without a third variable
x = 98
y = 45

Q #5) Write a Java Program to count the number of words in a string using HashMap.

Answer: This is a collection class program where we have used HashMap for storing the string.
First of all, we have declared our string variable called str. Then we have used split() function delimited by single space so that we can split multiple words in a string.

Thereafter, we have declared HashMap and iterated using for loop. Inside for loop, we have an if else statement
in which wherever at a particular position, the map contains a key, we set the counter at that position and add the object to the map.

Each time, the counter is incremented by 1. Else, the counter is set to 1.

Finally, we are printing the HashMap.

Note: The same program can be used to count the number of characters in a string. All you need to do is to remove one space (remove space delimited in split method) in String[] split = str.split(“”);

```
1  import java.util.HashMap;
2
3  public class FinalCountWords {
4
5      public static void main(String[] args) {
           // TODO Auto-generated method
6  stub
7          String str = "This this is is done by Saket Saket";
8          String[] split = str.split(" ");
9           HashMap<String,Integer> map = new HashMap<String,Integer>();
10          for (int i=0; i<split.length-1; i++) {
11              if (map.containsKey(split[i])) {
12                  int count = map.get(split[i]);
13                  map.put(split[i], count+1);
14              }
15              else {
16                  map.put(split[i], 1);
17              }
18          }
19          System.out.println(map);
20     }
21
22 }
```

Output:

{Saket=1, by=1, this=1, This=1, is=2, done=1}

Q #6) Write a Java Program to iterate HashMap using While and advance for loop.

Answer: Here we have inserted three elements in HashMap using put() function.
The size of the map can get using size() method. Thereafter, we have used While loop for iterating through the map which contains one key-value pair for each element. Keys and Values can be retrieved through getKey() and getValue().

Likewise, we have used advanced for loop where we have "me2" object for the HashMap.

```
1 import java.util.HashMap;
2 import java.util.Iterator;
3 import java.util.Map;
4
5 public class HashMapIteration {
6
7     public static void main(String[] args) {
          // TODO Auto-generated method
8 stub
9          HashMap<Integer,String> map = new HashMap<Integer,String>();
10         map.put(2, "Saket");
11         map.put(25, "Saurav");
12         map.put(12, "HashMap");
13         System.out.println(map.size());
14         System.out.println("While Loop:");
15         Iterator itr = map.entrySet().iterator();
16         while(itr.hasNext()) {
17             Map.Entry me = (Map.Entry) itr.next();
18             System.out.println("Key is " + me.getKey() + " Value is " +
   me.getValue());
19         }
20         System.out.println("For Loop:");
21         for(Map.Entry me2: map.entrySet()) {
22             System.out.println("Key is: " + me2.getKey() + " Value is:
   " + me2.getValue());
23         }
24     }
25
26 }
```

Output:

3
While Loop:
Key is 2 Value is Saket
Key is 25 Value is Saurav
Key is 12 Value is HashMap
For Loop:
Key is: 2 Value is: Saket
Key is: 25 Value is: Saurav
Key is: 12 Value is: HashMap

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Q #7) Write a Java Program to find whether a number is prime or not.

Answer: Here, we have declared two integers temp and num and used Scanner class with nextInt(as we have integer only).
One boolean variable isPrime is set to true. Thereafter, we have used for loop starting from 2, less than half of the number are entered and incremented by 1 for each iteration. Temp will have remainder for every iteration. If the remainder is 0, then isPrime will be set to False.

Based on isPrime value, we are coming to the conclusion that whether our number is prime or not.

```
1 import java.util.Scanner;
2
3 public class Prime {
4
5     public static void main(String[] args) {
          // TODO Auto-generated method
6 stub
7         int temp, num;
8         boolean isPrime = true;
9          Scanner in = new Scanner(System.in);
10         num = in.nextInt();
11         in.close();
12         for (int i = 2; i<= num/2; i++) {
13             temp = num%i;
14             if (temp == 0) {
15                 isPrime = false;
16                 break;
17             }
18         }
19         if(isPrime)
20             System.out.println(num + "number is prime");
21             else
22                 System.out.println(num + "number is not a prime");
23
24
25     }
26
```

```
27 }
```

Output:
445
445number is not a prime

Q #8) Write a Java Program to find whether a string or number is palindrome or not.
Answer: You can use any of the reverse string program explained above to check whether the number or string
is palindrome or not.
What you need to do is to include one if-else statement. If the original string is equal to a reversed string then the number is a palindrome, otherwise not.

```
1 import java.util.Scanner;
2
3 public class Palindrome {
4     public static void main (String[] args) {
5         String original, reverse = "";
6         Scanner in = new Scanner(System.in);
7         int length;
8         System.out.println("Enter the number or String");
9         original = in.nextLine();
10                        length = original.length();
11         for (int i =length -1; i>=0; i--) {
12             reverse = reverse + original.charAt(i);
13         }
14         System.out.println("reverse is:" +reverse);
15
16         if(original.equals(reverse))
17             System.out.println("The number is palindrome");
18         else
19             System.out.println("The number is not a palindrome");
20
21     }
22 }
```

Output:
For String-
Enter the number or String
vijay
reverse is:yajiv
The number is not a palindrome

For Number-

Enter the number or String
99
reverse is:99
The number is palindrome

Q #9) Write a Java Program for Fibonacci series.

Answer: Fibonacci series is a series of numbers where after the initial two numbers, every occurring number is the sum of two preceding numbers.
For Example 0,1,1,2,3,5,8,13,21……
In this program, we have used Scanner class again with nextInt (discussed above). Initially, we are entering (through command line) the number of times the Fibonacci has to iterate. We have declared integer num and initialized a,b with zero and c with one. Then, we have used for loop to iterate.

The logic goes like a is set with the value of b which is 0, then b is set with the value of c which is 1. Then, c is set with the sum of both a and b.

```
1 import java.util.Scanner;
2
3 public class Fibonacci {
4     public static void main(String[] args) {
5         int num, a = 0,b=0, c =1;
6         Scanner in = new Scanner(System.in);
7         System.out.println("Enter the number of times");
8         num = in.nextInt();
9          System.out.println("Fibonacci Series of the number is:");
10         for (int i=0; i<=num; i++) {
11             a = b;
12             b = c;
13             c = a+b;
14             System.out.println(a + "");    //if you want to print on
 the same line, use print()
15         }
16     }
17 }
```

Output:
Enter the number of times
9
Fibonacci Series of the number is:
0
1
1
2

3
5
8
13
21
34

Q #10) Write a Java Program to iterate ArrayList using for-loop, while-loop, and advance for-loop.

Answer: In this program, we have inserted three elements and printed the size of the ArrayList.
Then, we have used While Loop with an iterator. Whenever the iterator has (next) element, it will display that element until we reach the end of the list. So it will iterate three times.

Likewise, we have done for Advanced For Loop where we have created an object called obj for the ArrayList called list. Then printed the object.

Thereafter, we have put the condition of For Loop where the iterator i is set to 0 index, then it is incremented by 1 until the ArrayList limit or size is reached. Finally, we have printed each element using a get(index) method for each iteration of For Loop.

```
1 import java.util.*;
2
3 public class arrayList {
4     public static void main(String[] args) {
5         ArrayList list = new ArrayList();
6         list.add("20");
7         list.add("30");
8         list.add("40");
9         System.out.println(list.size());
10                          System.out.println("While Loop:");
11         Iterator itr = list.iterator();
12         while(itr.hasNext()) {
13             System.out.println(itr.next());
14         }
15         System.out.println("Advanced For Loop:");
16         for(Object obj : list) {
17             System.out.println(obj);
18     }
19         System.out.println("For Loop:");
20         for(int i=0; i<list.size(); i++) {
21             System.out.println(list.get(i));
22         }
23 }
```

```
24 }
```

Output:
3
While Loop:
20
30
40
Advanced For Loop:
20
30
40
For Loop:
20
30
40

Q #11) Write a Java Program to demonstrate explicit wait condition check.

Answer: There are two main types of wait – implicit and explicit. (We are not considering Fluent wait in this program)
Implicit wait is those waits which are executed irrespective of any condition. In the below program, you can see that it is for Google Chrome and we have used some inbuilt methods to set the property, maximizing window, URL navigation, and web element locating.

```
1 WebDriverWait wait = new WebDriverWait(driver, 20);
  WebElement element2 =
2 wait.until(ExpectedConditions.visibilityOfElementLocated(By.partialLinkTe
  xt("Software testing - Wikipedia")));
3 element2.click();
```

In the above piece of code, you can see that we have created an object wait for WebDriverWait and then we have searched for WebElement called element2.

The condition is set in such a way that webdriver will have to wait until we see the link "Software testing – Wikipedia" on a web page. It won't execute if it does not find this link. If it does, then it will do a mouse click on that link.

```
1 package Codes;
2 import java.util.concurrent.TimeUnit;
3 import org.openqa.selenium.By;
4 import org.openqa.selenium.WebDriver;
5 import org.openqa.selenium.WebElement;
6 import org.openqa.selenium.chrome.ChromeDriver;
7 import org.openqa.selenium.chrome.ChromeOptions;
8 import org.openqa.selenium.support.ui.ExpectedConditions;
```

```
import org.openqa.selenium.support.ui.WebDriverWait;

public class explicitWaitConditionCheck {

            public static void main(String[] args) {
                        // TODO Auto-generated method stub
                       System.setProperty("webdriver.chrome.driver", "C:\\webdriver\\chromedriver.exe");
                      ChromeOptions options = new ChromeOptions();
                       options.addArguments("--disable-arguments");
                       WebDriver driver = new ChromeDriver();
                       driver.manage().window().maximize();
                       driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
                       driver.navigate().to("https://www.google.com");
                       WebElement element = driver.findElement(By.name("q"));
                        element.sendKeys("Testing");
                        element.submit();
                        WebDriverWait wait = new WebDriverWait(driver, 20);

                         WebElement element2 = wait.until(ExpectedConditions.visibilityOfElementLocated(By.partialLinkText("Software testing - Wikipedia")));
                element2.click();
}}
```

Q #12) Write a Java Program to demonstrate Scroll up/ Scroll down.

Answer: All the lines of codes are easily relatable as we have discussed in our previous example.
However, in this program, we have included our JavascriptExecutor js which will do the scrolling. If you see the last line of the code, we have passed window.scrollBy(arg1,arg2).

If we want to scroll up then pass some value in arg1 if you want to scroll down then pass some value in arg2.

```
package Codes;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
```

```
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
                    import org.openqa.selenium.chrome.ChromeDriver;

public class ScrollDown {

          public static void main(String[] args) {
                    // TODO Auto-generated method stub
                    System.setProperty("webdriver.chrome.driver", "C:\\webdriver\\chromedriver.exe");
                    WebDriver driver = new ChromeDriver();
                    JavascriptExecutor js = (JavascriptExecutor) driver;
                    driver.manage().window().maximize();
                    driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
                    driver.get("https://www.google.com");
                    WebElement element = driver.findElement(By.name("q"));
                    element.sendKeys("SoftwareTestingHelp");
                    element.sendKeys(Keys.ENTER);
                     js.executeScript("window.scrollBy(0,1000)");

}

}
```

Q #13) Write a Java Program to open all links of gmail.com.

Answer: It is a typical example of advanced for loop which we have seen in our previous programs.

Once you have opened a website such as Gmail using get() or navigate().to(), you can use a tagName locator to find the tag name of a website which will return all the tags.

We have advanced for loop where we have created a new WebElement link2 for a link(which already has located
all the tags), then we have got all the links through getAttribute("href") and got all the texts through getText().

```
package Codes;

```

```
3 import java.util.concurrent.TimeUnit;
4
5 import org.openqa.selenium.By;
6 import org.openqa.selenium.WebDriver;
7 import org.openqa.selenium.WebElement;
8 import org.openqa.selenium.chrome.ChromeDriver;
9
10 public class openAllLinks {
11
12         public static void main(String[] args) {
1
3          // TODO Auto-generated method stub
1          System.setProperty("webdriver.chrome.drive", "C:\\webdriver\\chr
4 omedriver.exe");
15         WebDriver driver = new ChromeDriver();
16         driver.manage().timeouts().implicitlyWait(20,
  TimeUnit.SECONDS);
17         driver.manage().window().maximize();
18         driver.get("https://www.gmail.com/");
19         java.util.List<WebElement> link =
  driver.findElements(By.tagName("a"));
20        System.out.println(link.size());
21
22        for (WebElement link2: link) {
23
24        //print the links i.e. http://google.com or
  https://www.gmail.com
25       System.out.println(link2.getAttribute("href"));
26
27       //print the links text
28      System.out.println(link2.getText());
29 }
30 }
31 }
```

Output:

```
Starting ChromeDriver 2.38.551601 (edb21f07fc70e9027c746edd3201443e011a61ed) on
port 16163
Only local connections are allowed.
4
https://support.google.com/chrome/answer/6130773?hl=en-GB
Learn more
```

https://support.google.com/accounts?hl=en-GB
Help
https://accounts.google.com/TOS?loc=IN&hl=en-GB&privacy=true
Privacy
https://accounts.google.com/TOS?loc=IN&hl=en-GB
Terms

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Q #14) Write a Selenium code to switch to the previous tab.

Answer: We have demonstrated the use of the Robot class. We see this as an important third party because we can achieve the different navigation within a browser and its tabs if you know the shortcut keys.

For example, if you have three tabs open in your chrome and you want to go to the middle tab, then you have to press control + 2 from your keyboard. The same thing can be achieved through the code as well.

Observe the following code (just after we see the instantiation of Robot class). we have used Robot class object called a robot with two inbuilt methods keyPress(KeyEvenet.VK_*) and keyRelease(KeyEvenet.VK_*).

```
1 package Codes;
2
3 import java.awt.AWTException;
4 import java.awt.Robot;
5 import java.awt.event.KeyEvent;
6 import java.util.concurrent.TimeUnit;
7 import org.openqa.selenium.By;
8 import org.openqa.selenium.Keys;
9 import org.openqa.selenium.WebDriver;
10                  import org.openqa.selenium.WebElement;
11 import org.openqa.selenium.chrome.ChromeDriver;
12 public class PreviousTab {
13          public static void main(String[] args) throws AWTException {
14               // TODO Auto-generated method stub
1               System.setProperty("webdriver.chrome.driver", "C:\\webdriv
5 er\\chromedriver.exe");
1
6              WebDriver driver = new ChromeDriver();
17             driver.manage().window().maximize();
18             driver.manage().timeouts().implicitlyWait(20,
  TimeUnit.SECONDS);
19             driver.get("https://www.google.com");
20             WebElement element1 = driver.findElement(By.name("q"));
21             element1.sendKeys("software testing help");
22             element1.sendKeys(Keys.ENTER);
23             String a = Keys.chord(Keys.CONTROL,Keys.RETURN);
```

```
24              driver.findElement(By.partialLinkText("Software Testing
Help - A Must Visit Software Testing Portal")).sendKeys(a);
25              Robot robot = new Robot(); // instantiated robot class
26              robot.keyPress(KeyEvent.VK_CONTROL); // with robot class
you can easily achieve anything if you know the shortcut keys
27              robot.keyPress(KeyEvent.VK_2); // here, we have just
pressed ctrl+2
28              robot.keyRelease(KeyEvent.VK_CONTROL); // once we press
and release ctrl+2, it will go to the second tab.
29              robot.keyRelease(KeyEvent.VK_2); //if you again want to go
back to first tab press and release vk_1
30              }
31 }
```

Q #15) Write a Java Program to find the duplicate characters in a string.

Answer: In this program, we have created a string variable str and initialized an integer count with zero.

Then, we have created a character array to convert our string variable to the character. With the help of for loop, we are performing a comparison between different character at different indexes.

If two character of consecutive index matches, then it will print that character and the counter will be incremented by 1 after each iteration.

```
1 public class DuplicateCharacters {
2
3           public static void main(String[] args) {
4                    // TODO Auto-generated method stub
5                   String str = new String("Sakkett");
6                   int count = 0;
7                   char[] chars = str.toCharArray();
8                   System.out.println("Duplicate characters are:");
9                   for (int i=0; i<str.length();i++) {
10                                                  for(int j=i+1;
                      j<str.length();j++) {
11                                              if (chars[i] == chars[j]) {
1                                                   System.out.println(c
2 hars[j]);
13                                                  count++;
14                                                  break;
15                                               }
16                                      }
17                  }
18          }
```

```
19
20 }
```

Output:
Duplicate characters are:
k
t

Q #16) Write a Java Program to find the second highest number in an array.

Answer: In this program, we have initialized an array with 10 random elements out of which we are going to
find the second highest number. Here, we have two integers- the largest and second largest. Both set to the first
index of the element. Then, we have printed all the elements using for loop.
Now the logic is when the element at the 0th index is greater than the largest, then assign arr[0] to largest
and secondLargest to largest. Again, if the element at the 0th index is greater than the secondLargest, then
assign secondLargest to arr[0].

This will be repeated for each iteration and ultimately after comparing or completing iterations up to array length will give you the secondLargest element.

```
1 public class SecondHighestNumberInArray {
2 public static void main(String[] args)
3     {
4         int arr[] = { 14, 46, 47, 94, 94, 52, 86, 36, 94, 89 };
5         int largest = arr[0];
6         int secondLargest = arr[0];
7         System.out.println("The given array is:");
8         for (int i = 0; i < arr.length; i++)
9         {
10                System.out.print(arr[i] + "\t");
11         }
12         for (int i = 0; i < arr.length; i++)
13         {
14             if (arr[i] > largest)
15             {
16                 secondLargest = largest;
17                 largest = arr[i];
18             }
19             else if (arr[i] > secondLargest && arr[i] != largest)
20             {
```

```
21                  secondLargest = arr[i];
22              }
23          }
24          System.out.println("\nSecond largest number is:" +
    secondLargest);
25      }
26  }
```

Output:
The given array is:
14 46 47 45 92 52 48 36 66 85
Second largest number is:85

Q #17) Write a Java Program to check Armstrong number.

Answer: First of all we need to understand what Armstrong Number is. Armstrong number is the number
which is the sum of the cubes of all its unit, tens and hundred digits for three digit number.
153 = 1*1*1 + 5*5*5 + 3*3*3 = 1 + 125 + 27 = 153

If you have a four digit number lets say

1634 = 1*1*1*1 + 6*6*6*6 + 3*3*3*3 + 4*4*4*4 = 1 + 1296 + 81 + 256 = 1634

Now, in this program, we have a temp and integers declared. We have initialized c with value 0. Then, we need to
assign the integer value which we are going to check for Armstrong (in our case, let us say 153). Then we have assigned our temp variable with that number which we are going to check.

Thereafter, we have used while conditional check where the remainder is assigned to a and the number is divided by 10 and assigned to n. Now, our c variable which was set to zero initially is assigned with c+(a*a*a). Suppose we have to evaluate a four-digit number then c should be assigned with c + (a*a*a*a).

Lastly, we have put an if-else statement for conditional checking where we have compared the value contained in c against temp(which has the actual number stored at this point). If it matches, then the number is Armstrong otherwise not.

```
1 class Armstrong{
2  public static void main(String[] args)  {
3    int c=0,a,temp;
4    int n=153;//It is the number to check Armstrong
5    temp=n;
6    while(n>0)
```

```
7    {
8    a=n%10;
9    n=n/10;
10           c=c+(a*a*a);
11     }
12     if(temp==c)
13     System.out.println("armstrong number");
14     else
15         System.out.println("Not armstrong number");
16    }
17 }
```

Output:

armstrong number

Q #18) Write a Java Program to remove all white spaces from a string with using replace().

Answer: This is a simple program where we have our string variable str1. Another string variable str2 is initialized with the replaceAll option which is an inbuilt method to remove n number of whitespaces. Ultimately, we have printed str2 which has no whitespaces.

```
1 class RemoveWhiteSpaces
2 {
3     public static void main(String[] args)
4     {
5         String str1 = "Saket Saurav        is a QualityAna    list";
6
7         //1. Using replaceAll() Method
8
9          String str2 = str1.replaceAll("\\s", "");
10
11         System.out.println(str2);
12
13            }
14 }
15 }
```

Output:

SaketSauravisaQualityAnalist

Q #19) Write a Java Program to remove all white spaces from a string without using replace().

Answer: This is another approach to removing all white spaces. Again, we have one string variable str1 with some
value. Then, we have converted that string into a character array using toCharArray(). Then, we have one StringBuffer object sb which will be used to append the value stored at chars[i] index after we have included for loop and one if condition.

If the condition is set such that then the element at i index of the character array should not be equal to space or tab. Finally, we have printed our StringBuffer object sb.

```
1 class RemoveWhiteSpaces
2 {
3     public static void main(String[] args)
4     {
5         String str1 = "Saket Saurav        is an Autom ation Engi
ne      er";
6
7         char[] chars = str1.toCharArray();
8
9          StringBuffer sb = new StringBuffer();
10
11         for (int i = 0; i < chars.length; i++)
12         {
13             if( (chars[i] != ' ') && (chars[i] != '\t') )
14             {
15                 sb.append(chars[i]);
16             }
17         }
18         System.out.println(sb);           //Output :
CoreJavajspservletsjdbcstrutshibernatespring
19     }
20 }
```

Output:
SaketSauravisanAutomationEngineer

Q #20) Write a Java Program to read an excel.

Answer: These types of programs are generally used in Selenium framework. We have added detailed comments for every step to make the program more understandable.
The logic starts after we have loaded the sheet in which the data is stored. We are trying to import email and password. For this, we are retrieving the cell using getRow() and getCell() method. Let's say we have email and passwords at the 1st and 2nd cell.

Then we are setting the type of cell to string. Thereafter we are carrying out a normal web element locator operation (By.id) where we have passed unique locator values such as “email” and “password” which will identify these elements.

Finally, we are sending keys using element.sendKeys where cell.getStringCellValue() is the key. This will return you the value stored at cell number 1 and 2 respectively.

```
@Test
 public void ReadData()  throws IOException
 {
      // Import excel sheet from a webdriver directory which is inside c
drive.
      //DataSource is the name of the excel
      File src=new File("C:\\webdriver\\DataSource.xls");

      //This step is for loading the file. We have used FileInputStream
as
      //we are reading the excel. In case you want to write into the
file,
                          //you need to use FileOutputStream. The path of
                    the file is passed as an argument to FileInputStream
       FileInputStream finput = new FileInputStream(src);

       //This step is to load the workbook of the excel which is done by
global HSSFWorkbook in which we have
       //passed finput as an argument.
      workbook = new HSSFWorkbook(finput);

       //This step is to load the sheet in which data is stored.
       sheet= workbook.getSheetAt(0);

       for(int i=1; i<=sheet.getLastRowNum(); i++)
       {
           // Import data for Email.
           cell = sheet.getRow(i).getCell(1);
           cell.setCellType(Cell.CELL_TYPE_STRING);
           driver.findElement(By.id("email")).sendKeys(cell.getStringCellV
alue());

           // Import data for the password.
           cell = sheet.getRow(i).getCell(2);
           cell.setCellType(Cell.CELL_TYPE_STRING);
```

```
3           driver.findElement(By.id("password")).sendKeys(cell.getStringCe
0  llValue());
31
32          }
33    }
```

Summary

In this article, we have discussed all the important basic Java Interview Programs with code examples that are being asked in any Java Programming Interviews.
We have learned all the Java Tricks like basic manipulation of String, Integer and Characters, Selenium Codes, Reading Data from a File, Mathematical Series through codes and now you do have ample idea of how to go about any Java Interview.

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Why is Java called the 'Platform Independent Programming Language'?

Platform independence means that execution of your program does not dependent on type of operating system(it could be any : Linux, windows, Mac ..etc). So compile code only once and run it on any System (In C/C++, we need to compile the code for every machine on which we run it). Java is both compiler(javac) and interpreter(jvm) based lauguage. Your java source code is first compiled into byte code using javac compiler. This byte code can be easily converted to equivalent machine code using JVM. JVM(Java Virtual Machine) is available in all operating systems we install. Hence, byte code generated by javac is universal and can be converted to machine code on any operating system, this is the reason why java is platform independent.

Explain Final keyword in java?

Final keyword in java is used to restrict usage of variable, class and method.

Variable: Value of Final variable is constant, you can not change it.
Method: you can't override a Final method.
Class: you can't inherit from Final class.

Refer this for details

When is the super keyword used?

super keyword is used to refer:

- immediate parent class constructor,
- immediate parent class variable,
- immediate parent class method.

Refer this for details.

What is the difference between StringBuffer and String?

String is an Immutable class, i.e. you can not modify its content once created. While StringBuffer is a mutable class, means you can change its content later. Whenever we alter content of String object, it creates a new string and refer to that,it does not modify the existing one. This is the reason that the performance with StringBuffer is better than with String.
Refer this for details.

Why multiple inheritance is not supported in java?

Java supports multiple inheritance but not through classes, it supports only through its interfaces. The reason for not supporting multiple inheritance is to avoid the conflict and complexity arises due to it and keep Java a Simple Object Oriented Language. If we recall this in C++, there is a special case of multiple inheritance

(diamond problem) where you have a multiple inheritance with two classes which have methods in conflicts. So, Java developers decided to avoid such conflicts and didn't allow multiple inheritance through classes at all.

Can a top level class be private or protected?

Top level classes in java can't be private or protected, but inner classes in java can. The reason for not making a top level class as private is very obvious, because nobody can see a private class and thus they can not use it. Declaring a class as protected also doesn't make any sense. The only difference between default visibility and protected visibility is that we can use it in any package by inheriting it. Since in java there is no such concept of package inheritance, defining a class as protected is no different from default.

What is the difference between 'throw' and 'throws' in Java Exception Handling?

Following are the differences between two:

- throw keyword is used to throw Exception from any method or static block whereas throws is used to indicate that which Exception can possibly be thrown by this method
- If any method throws checked Exception, then caller can either handle this exception(using try catch block)or can re throw it by declaring another 'throws' clause in method declaration.
- throw clause can be used in any part of code where you feel a specific exception needs to be thrown to the calling method

E.g.
throw
throw new Exception("You have some exception")
throw new IOException("Connection failed!!")
throws
throws IOException, NullPointerException, ArithmeticException

What is finalize() method?

Unlike c++ , we don't need to destroy objects explicitly in Java. 'Garbage Collector' does that automatically for us. Garbage Collector checks if no references to an object exist, that object is assumed to be no longer required, and the memory occupied by the object can be freed. Sometimes an object can hold non-java resources such as file handle or database connection, then you want to make sure these resources are also released before object is destroyed. To perform such operation Java provide protected void finalize() in object class. You can override this method in your class and do the required tasks. Right before an object is freed, the java run time calls the finalize() method on that object. Refer this for more details.

Difference in Set and List interface?

Set and List both are child interface of Collection interface. There are following two main differences between them

- List can hold duplicate values but Set doesn't allow this.
- In List interface data is present in the order you inserted but in the case of Set insertion order is not preserved.

What will happen if you put System.exit(0) on try or catch block? Will finally block execute?

By Calling System.exit(0) in try or catch block, we can skip the finally block. System.exit(int) method can throw a SecurityException. If Sysytem.exit(0) exits the JVM without throwing that exception then finally block will not execute. But, if System.exit(0) does throw security exception then finally block will be executed.

Commonly Asked C Programming Interview Questions | Set 1

What is the difference between declaration and definition of a variable/function
Ans: Declaration of a variable/function simply declares that the variable/function exists somewhere in the program but the memory is not allocated for them. But the declaration of a variable/function serves an important role. And that is the type of the variable/function. Therefore, when a variable is declared, the program knows the data type of that variable. In case of function declaration, the program knows what are the arguments to that functions, their data types, the order of arguments and the return type of the function. So that's all about declaration. Coming to the definition, when we define a variable/function, apart from the role of declaration, it also allocates memory for that variable/function. Therefore, we can think of definition as a super set of declaration. (or declaration as a subset of definition). From this explanation, it should be obvious that a variable/function can be declared any number of times but it can be defined only once. (Remember the basic principle that you can't have two locations of the same variable/function).

```
   // This is only declaration. y is not allocated memory by this
statement
  extern int y;


  // This is both declaration and definition, memory to x is allocated
by this statement.
  int x;
```

What are different storage class specifiers in C?
Ans: auto, register, static, extern

What is scope of a variable? How are variables scoped in C?
Ans: Scope of a variable is the part of the program where the variable may directly be accessible. In C, all identifiers are lexically (or statically) scoped. See this for more details.

How will you print "Hello World" without semicolon?
Ans:

filter_none
edit
play_arrow
brightness_4

```
#include <stdio.h>
int main(void)
{
    if (printf("Hello World")) {
    }
}
```

See print "Geeks for Geeks" without using a semicolon for answer.

When should we use pointers in a C program?
1. To get address of a variable
2. *For achieving pass by reference in C:* Pointers allow different functions to share and modify their local variables.
3. *To pass large structures* so that complete copy of the structure can be avoided.
4. *To implement "linked" data structures* like linked lists and binary trees.

What is NULL pointer?
Ans: NULL is used to indicate that the pointer doesn't point to a valid location. Ideally, we should initialize pointers as NULL if we don't know their value at the time of declaration. Also, we should make a pointer NULL when memory pointed by it is deallocated in the middle of a program.

What is Dangling pointer?
Ans: Dangling Pointer is a pointer that doesn't point to a valid memory location. Dangling pointers arise when an object is deleted or deallocated, without modifying the value of the pointer, so that the pointer still points to the memory location of the deallocated memory. Following are examples.

filter_none
edit
play_arrow
brightness_4

```
// EXAMPLE 1
int* ptr = (int*)malloc(sizeof(int));
.........................free(ptr);

// ptr is a dangling pointer now and operations like following are invalid
*ptr = 10; // or printf("%d", *ptr);
```

filter_none
edit
play_arrow
brightness_4

```
// EXAMPLE 2
int* ptr = NULL
{
    int x = 10;
    ptr = &x;
}
// x goes out of scope and memory allocated to x is free now.
// So ptr is a dangling pointer now.
```

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

What is memory leak? Why it should be avoided

Ans: Memory leak occurs when programmers create a memory in heap and forget to delete it. Memory leaks are particularly serious issues for programs like daemons and servers which by definition never terminate.

filter_none
edit
play_arrow
brightness_4

```
/* Function with memory leak */
#include <stdlib.h>

void f()
{
    int* ptr = (int*)malloc(sizeof(int));

    /* Do some work */

    return; /* Return without freeing ptr*/
}
```

What are local static variables? What is their use?

Ans:A local static variable is a variable whose lifetime doesn't end with a function call where it is declared. It extends for the lifetime of complete program. All calls to the function share the same copy of local static variables. Static variables can be used to count the number of times a function is called. Also, static variables get the default value as 0. For example, the following program prints "0 1"

filter_none
edit
play_arrow
brightness_4

```
#include <stdio.h>
void fun()
{
    // static variables get the default value as 0.
    static int x;
    printf("%d ", x);
    x = x + 1;
}

int main()
{
    fun();
    fun();
    return 0;
}
// Output: 0 1
```

What are static functions? What is their use?
Ans: In C, functions are global by default. The "static" keyword before a function name makes it static. Unlike global functions in C, access to static functions is restricted to the file where they are declared. Therefore, when we want to restrict access to functions, we make them static. Another reason for making functions static can be reuse of the same function name in other files. See this for examples and more details.

1. How to reverse Singly Linked List?
2. Find out duplicate number between 1 to N numbers.
3. Find out middle index where sum of both ends are equal.
4. Write a singleton class.
5. Write a program to create deadlock between two threads.
6. Write a program to reverse a string using recursive algorithm.
7. Write a program to reverse a number.
8. Write a program to convert decimal number to binary format.
9. Write a program to find perfect number or not.
10. Write a program to implement ArrayList.
11. Write a program to find maximum repeated words from a file.
12. Wrie a program to find out duplicate characters in a string.
13. Write a program to find top two maximum numbers in a array.
14. Write a program to sort a map by value.
15. Write a program to find common elements between two arrays.
16. How to swap two numbers without using temporary variable?
17. Write a program to print fibonacci series.
18. Write a program to find sum of each digit in the given number using recursion.
19. Write a program to check the given number is a prime number or not?
20. Write a program to find the given number is Armstrong number or not?
21. Write a program to convert binary to decimal number.
22. Write a program to check the given number is binary number or not?
23. Write a program for Bubble Sort in java.
24. Write a program for Insertion Sort in java.
25. Write a program to implement hashcode and equals.
26. How to get distinct elements from an array by avoiding duplicate elements?
27. Write a program to get distinct word list from the given file.
28. Write a program to get a line with max word count from the given file.
29. Write a program to convert string to number without using Integer.parseInt() method.
30. Write a program to find two lines with max characters in descending order.
31. Write a program to find the sum of the first 1000 prime numbers.
32. Find longest substring without repeating characters.
33. Write a program to remove duplicates from sorted array.
34. How to sort a Stack using a temporary Stack?
35. Write a program to print all permutations of a given string.
36. Implement Binary Search Tree (BST)
37. Find min and max value from Binary Search Tree (BST)
38. Find height of a Binary Search Tree (BST)
39. Implement Binary Search Tree (BST) Level order traversal (breadth first).
40. Implement Binary Search Tree (BST) pre-order traversal (depth first).
41. Implement Binary Search Tree (BST) in-order traversal (depth first).
42. Implement Binary Search Tree (BST) post-order traversal (depth first).
43. How to check the given Binary Tree is Binary Search Tree (BST) or not?
44. How to delete a node from Binary Search Tree (BST)?
45. Write a program to find common integers between two sorted arrays.
46. Write a program to find given two trees are mirror or not.
47. HackerRank stack problem - Find maximum element.
48. HackerRank stack problem - Balanced Brackets.
49. HackerRank stack problem - Equal Stacks.
50. HackerRank stack problem - Game Of Two Stacks.

100+ Java Interview Questions You Must Prepare In 2019

Aayushi Johari

Java Interview Questions

In this Java Interview Questions blog, I am going to list some of the most important Java Interview Questions and Answers which will set you apart in the interview process. Java is used by approx 10 Million developers worldwide to develop applications for 15 Billion devices supporting Java. It is also used to create applications for trending technologies like Big Data to household devices like Mobiles and DTH boxes. And hence today, **Java is used everywhere!** This is the reason why ***Java Certification*** is the most in-demand certification in programming domain.

Let us start by taking a look at some of the most frequently asked Java interview questions,

Q1. Explain JDK, JRE and JVM?
Q2. Explain public static void main(String args[]) in Java
Q3. Why Java is platform independent?
Q4. Why Java is not 100% Object-oriented?
Q5. What are wrapper classes in Java?
Q6. What are constructors in Java?
Q7. What is singleton class in Java and how can we make a class singleton?
Q8. What is the difference between Array list and vector in Java?
Q9. What is the difference between equals() and == in Java?
Q10. What are the differences between Heap and Stack Memory in Java?

We have compiled a list of top Java interview questions which are classified into 7 sections, namely:

1.
2.
3.
4.
5.
6.
7.

Java Interview Questions and Answers | Edureka

As a Java professional, it is essential to know the right buzzwords, learn the right technologies and prepare the right answers to commonly asked Java Interview Questions. Here's a definitive list of top Java Interview Questions that will guarantee a breeze-through to the next level.

*In case you attended any Java interview recently, or have additional questions beyond what we covered, we encourage you to post them in our **QnA Forum**. Our expert team will get back to you at the earliest.*

So let's get started with the first set of basic Java Interview Questions.

Basic Java Interview Questions

Q1. Explain JDK, JRE and JVM?

JDK vs JRE vs JVM

JDK	JRE	JVM
It stands for Java Development Kit.	It stands for Java Runtime Environment.	It stands for Java Virtual Machine.
It is the tool necessary to compile, document and package Java programs.	JRE refers to a runtime environment in which Java bytecode can be executed.	It is an abstract machine. It is a specification that provides a run-time environment in which Java bytecode can be executed.
It contains JRE + development tools.	It's an implementation of the JVM which physically exists.	JVM follows three notations: Specification, **Implementation,** and **Runtime Instance**.

Q2. Explain public static void main(String args[]) in Java.

main() in Java is the entry point for any Java program. It is always written as **public static void main(String[] args)**.

- **public**: Public is an access modifier, which is used to specify who can access this method. Public means that this Method will be accessible by any Class.
- **static**: It is a keyword in java which identifies it is class-based. main() is made static in Java so that it can be accessed without creating the instance of a Class. In case, main is not made static then the compiler will throw an error as **main**() is called by the

JVM before any objects are made and only static methods can be directly invoked via the class.

- **void**: It is the return type of the method. Void defines the method which will not return any value.
- **main**: It is the name of the method which is searched by JVM as a starting point for an application with a particular signature only. It is the method where the main execution occurs.
- **String args[]**: It is the parameter passed to the main method.

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Q3. Why Java is platform independent?

Java is called platform independent because of its byte codes which can run on any system irrespective of its underlying operating system.

Q4. Why Java is not 100% Object-oriented?

Java is not 100% Object-oriented because it makes use of eight primitive data types such as 63ehavio, byte, char, int, float, double, long, short which are not objects.

Q5. What are wrapper classes in Java?

Wrapper classes convert the Java primitives into the reference types (objects). Every primitive data type has a class dedicated to it. These are known as wrapper classes because they "wrap" the primitive data type into an object of that class. Refer to the below image which displays different primitive type, wrapper class and constructor argument.

Q6. What are constructors in Java?

In Java, constructor refers to a block of code which is used to initialize an object. It must have the same name as that of the class. Also, it has no return type and it is automatically called when an object is created.

There are two types of constructors:

1. **Default Constructor:** In Java, a default constructor is the one which does not take any inputs. In other words, default constructors are the no argument constructors which will be created by default in case you no other constructor is defined by the user. Its main purpose is to initialize the instance variables with the default values. Also, it is majorly used for object creation.
2. **Parameterized Constructor:** The parameterized constructor in Java, is the constructor which is capable of initializing the instance variables with the provided values. In other words, the constructors which take the arguments are called parameterized constructors.

Q7. What is singleton class in Java and how can we make a class singleton?

Singleton class is a class whose only one instance can be created at any given time, in one JVM. A class can be made singleton by making its constructor private.

Q8. What is the difference between Array list and vector in Java?

ArrayList	Vector

Array List is not synchronized.	Vector is synchronized.
Array List is fast as it's non-synchronized.	Vector is slow as it is thread safe.
If an element is inserted into the Array List, it increases its Array size by 50%.	Vector defaults to doubling size of its array.
Array List does not define the increment size.	Vector defines the increment size.
Array List can only use Iterator for traversing an Array List.	Vector can use both Enumeration and Iterator for traversing.

Q9. What is the difference between equals() and == in Java?

Equals() method is defined in Object class in Java and used for checking equality of two objects defined by business logic.

"==" or equality operator in Java is a binary operator provided by Java programming language and used to compare primitives and objects. *Public 64ehavio equals(Object o)* is the method provided by the Object class. The default implementation uses == operator to compare two objects. For example: method can be overridden like String class. Equals() method is used to compare the values of two objects.

Q10. What are the differences between Heap and Stack Memory in Java?

The major difference between Heap and Stack memory are:

Features	Stack	Heap
Memory	Stack memory is used only by one thread of execution.	Heap memory is used by all the parts of the application.
Access	Stack memory can't be accessed by other threads.	Objects stored in the heap are globally accessible.
Memory Management	Follows LIFO manner to free memory.	Memory management is based on the generation associated with each object.
Lifetime	Exists until the end of execution of the thread.	Heap memory lives from the start till the end of application execution.
Usage	Stack memory only contains local primitive and reference variables to objects in heap space.	Whenever an object is created, it's always stored in the Heap space.

Q11. What is a package in Java? List down various advantages of packages.

Packages in Java, are the collection of related classes and interfaces which are bundled together. By using packages, developers can easily modularize the code and optimize its reuse. Also, the code within the packages can be imported by other classes and reused. Below I have listed down a few of its advantages:

- Packages help in avoiding name clashes
- They provide easier access control on the code
- Packages can also contain hidden classes which are not visible to the outer classes and only used within the package
- Creates a proper hierarchical structure which makes it easier to locate the related classes

Q12. Why pointers are not used in Java?

Java doesn't use pointers because they are unsafe and increases the complexity of the program. Since, Java is known for its simplicity of code, adding the concept of pointers will be contradicting. Moreover, since JVM is responsible for implicit memory allocation, thus in order to avoid direct access to memory by the user, pointers are discouraged in Java.

Q13. What is JIT compiler in Java?

JIT stands for Just-In-Time compiler in Java. It is a program that helps in converting the Java bytecode into instructions that are sent directly to the processor. By default, the JIT compiler is enabled in Java and is activated whenever a Java method is invoked. The JIT compiler then compiles the bytecode of the invoked method into native machine code, compiling it "just in time" to execute. Once the method has been compiled, the JVM summons the compiled code of that method directly rather than interpreting it. This is why it is often responsible for the performance optimization of Java applications at the run time.

Q14. What are access modifiers in Java?

In Java, access modifiers are special keywords which are used to restrict the access of a class, constructor, data member and method in another class. Java supports four types of access modifiers:

1. *Default*
2. *Private*
3. *Protected*
4. *Public*

Modifier	**Default**	**Private**	**Protected**	**Public**
Same class	YES	YES	YES	YES
Same Package subclass	YES	NO	YES	YES
Same Package non-subclass	YES	NO	YES	YES
Different package subclass	NO	NO	YES	YES
Different package non-subclass	NO	NO	NO	YES

Q15. Define a Java Class.

A class in Java is a blueprint which includes all your data. A class contains fields (variables) and methods to describe the 66ehaviour of an object. Let's have a look at the syntax of a class.

```
1  class Abc {
2  member variables
3  methods}
```

Q16. What is an object in Java and how is it created?

An object is a real-world entity that has a state and 66ehaviour. An object has three characteristics:

1. State
2. Behavior
3. Identity

An object is created using the 'new' keyword. For example:

ClassName obj = new ClassName();

Q17. What is Object Oriented Programming?

Object-oriented programming or popularly known as OOPs is a programming model or approach where the programs are organized around objects rather than logic and functions. In other words, OOP mainly focuses on the objects that are required to be manipulated instead of logic. This approach is ideal for the programs large and complex codes and needs to be actively updated or maintained.

Q18. What are the main concepts of OOPs in Java?

Object-Oriented Programming or OOPs is a programming style that is associated with concepts like:

1. *Inheritance:* Inheritance is a process where one class acquires the properties of another.
2. *Encapsulation:* Encapsulation in Java is a mechanism of wrapping up the data and code together as a single unit.
3. *Abstraction:* Abstraction is the methodology of hiding the implementation details from the user and only providing the functionality to the users.

4. *Polymorphism:* Polymorphism is the ability of a variable, function or object to take multiple forms.

Q19. What is the difference between a local variable and an instance variable?

In Java, a **local variable** is typically used inside a method, constructor, or a **block** and has only local scope. Thus, this variable can be used only within the scope of a block. The best benefit of having a local variable is that other methods in the class won't be even aware of that variable.

Example

```
1  if(x > 100)
2  {
3  String test = "Edureka";
4  }
```

Whereas, an **instance variable** in Java, is a variable which is bounded to its object itself. These variables are declared within a **class**, but outside a method. Every object of that class will create it's own copy of the variable while using it. Thus, any changes made to the variable won't reflect in any other instances of that class and will be bound to that particular instance only.

```
1  class Test{
2  public String EmpName;
3  public int empAge;
4  }
```

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Q20. Differentiate between the constructors and methods in Java?

Methods	Constructors
1. Used to represent the behavior of an object	1. Used to initialize the state of an object
2. Must have a return type	2. Do not have any return type
3. Needs to be invoked explicitly	3. Is invoked implicitly
4. No default method is provided by the compiler	4. A default constructor is provided by the compiler if the class has none
5. Method name may or may not be same as class name	5. Constructor name must always be the same as the class name

Q21. What is final keyword in Java?

final is a special keyword in Java that is used as a non-access modifier. A final variable can be used in different contexts such as:

- **final variable**

When the final keyword is used with a variable then its value can't be changed once assigned. In case the no value has been assigned to the final variable then using only the class constructor a value can be assigned to it.

- *final method*

When a method is declared final then it can't be overridden by the inheriting class.

- *final class*

When a class is declared as final in Java, it can't be extended by any subclass class but it can extend other class.

Q22. What is the difference between break and continue statements?

break	continue
1. Can be used in switch and loop (for, while, do while) statements	1. Can be only used with loop statements
2. It causes the switch or loop statements to terminate the moment it is executed	2. It doesn't terminate the loop but causes the loop to jump to the next iteration
3. It terminates the innermost enclosing loop or switch immediately	3. A continue within a loop nested with a switch will cause the next loop iteration to execute

Example break:

```
1  for (int i = 0; i < 5; i++)
2  {
3  if (i == 3)
4  {
5  break;
6  }
7  System.out.println(i);
8  }
```

Example continue:

```
1  for (int i = 0; i < 5; i++)
2  {
3  if(i == 2)
4  {
5  continue;
6  }
7  System.out.println(i);
8  }
```

Q23.What is an infinite loop in Java? Explain with an example.

An infinite loop is an instruction sequence in Java that loops endlessly when a functional exit isn't met. This type of loop can be the result of a programming error or may also be a deliberate action based on the application behavior. An infinite loop will terminate automatically once the application exits.

For example:

```
1  public class InfiniteForLoopDemo
```

```
2  {
3  public static void main(String[] arg) {
4  for(;;)
5  System.out.println("Welcome to Edureka!");
6  }
7  }
8
```

Q24. What is the difference between this() and super() in Java?

In Java, super() and this(), both are special keywords that are used to call the constructor.

this()	super()
1. this() represents the current instance of a class	1. super() represents the current instance of a parent/base class
2. Used to call the default constructor of the same class	2. Used to call the default constructor of the parent/base class
3. Used to access methods of the current class	3. Used to access methods of the base class
4. Used for pointing the current class instance	4. Used for pointing the superclass instance
5. Must be the first line of a block	5. Must be the first line of a block

Q25. What is Java String Pool?

Java String pool refers to a collection of Strings which are stored in heap memory. In this, whenever a new object is created, String pool first checks whether the object is already present in the pool or not. If it is present, then the same reference is returned to the variable else new object will be created in the String pool and the respective reference will be returned.

Q26. Differentiate between static and non-static methods in Java.

Static Method	Non-Static Method
1. *The static* keyword must be used before the method name	1. No need to use the *static* keyword before the method name
2. It is called using the class (className.methodName)	2. It is can be called like any general method
3. They can't access any non-static instance variables or methods	3. It can access any static method and any static variable without creating an instance of the class

Q27. What is constructor chaining in Java?

In Java, constructor chaining is the process of calling one constructor from another with respect to the current object. Constructor chaining is possible only through legacy where a subclass constructor is responsible for invoking the superclass' constructor first. There could be any number of classes in the constructor chain. Constructor chaining can be achieved in two ways:

1. Within the same class using this()
2. From base class using super()

Q28. Difference between String, String Builder, and String Buffer.

Factor	String	String Builder	String Buffer
Storage Area	Constant String Pool	Heap Area	Heap Area

Mutability	Immutable	Mutable	Mutable
Thread Safety	Yes	Yes	No
Performance	Fast	Slow	Fast

Q29. What is a classloader in Java?

The **Java ClassLoader** is a subset of JVM (Java Virtual Machine) that is responsible for loading the class files. Whenever a Java program is executed it is first loaded by the classloader. Java provides three built-in classloaders:

1. Bootstrap ClassLoader
2. Extension ClassLoader
3. System/Application ClassLoader

Q30. Why Java Strings are immutable in nature?

In Java, string objects are immutable in nature which simply means once the String object is created its state cannot be modified. Whenever you try to update the value of that object instead of updating the values of that particular object, Java creates a new string object. Java String objects are immutable as String objects are generally cached in the String pool. Since String literals are usually shared between multiple clients, action from one client might affect the rest. It enhances security, caching, synchronization, and performance of the application.

Q31. What is the difference between an array and an array list?

Array	ArrayList
Cannot contain values of different data types	Can contain values of different data types.
Size must be defined at the time of declaration	Size can be dynamically changed
Need to specify the index in order to add data	No need to specify the index
Arrays are not type parameterized	Arraylists are type
Arrays can contain primitive data types as well as objects	Arraylists can contain only objects, no primitive data types are allowed

Q32. What is a Map in Java?

In Java, Map is an interface of Util package which maps unique keys to values. The Map interface is not a subset of the main Collection interface and thus it behaves little different from the other collection types. Below are a few of the characteristics of Map interface:

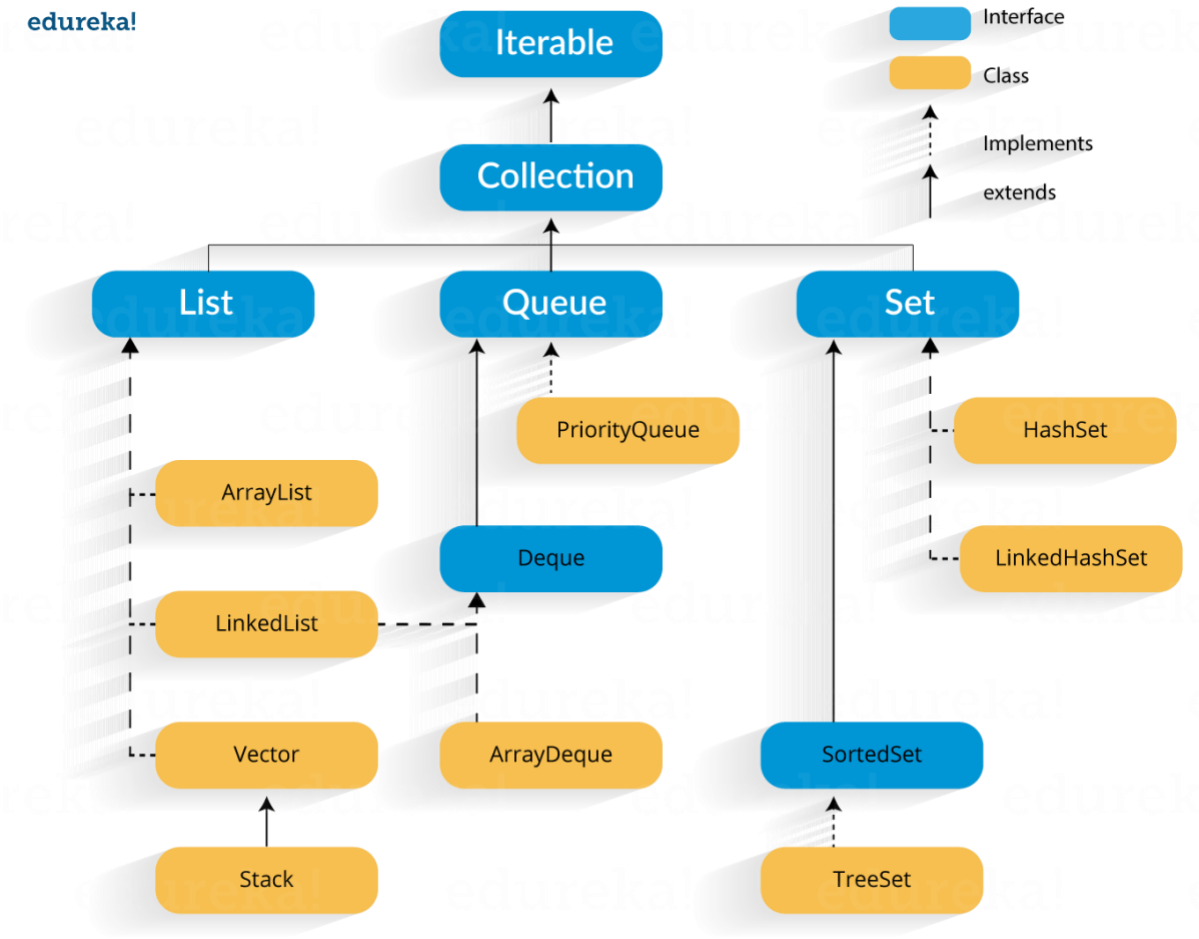
1. Map doesn't contain duplicate keys.
2. Each key can map at max one value.

Q33. What is collection class in Java? List down its methods and interfaces.

In Java, the collection is a framework that acts as an architecture for storing and manipulating a group of objects. Using Collections you can perform various tasks like searching, sorting, insertion, manipulation, deletion, etc. Java collection framework includes the following:

- Interfaces
- Classes
- Methods

The below image shows the complete hierarchy of the Java Collection.

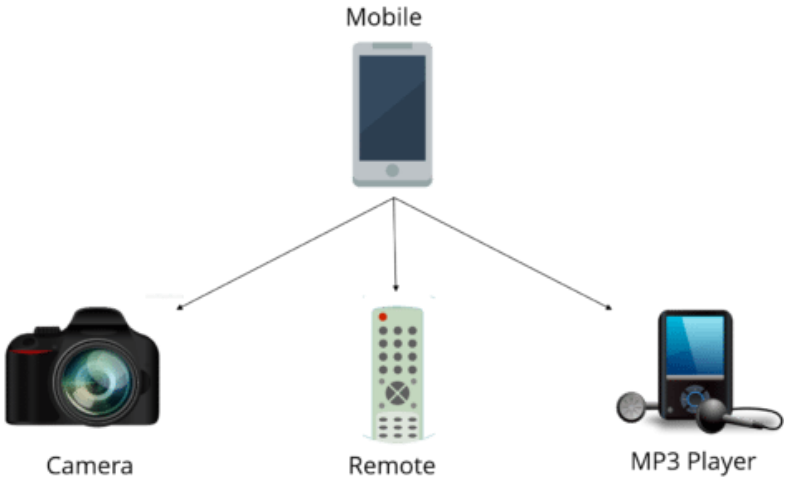


In case you are facing any challenges with these java interview questions, please comment on your problems in the section below.

OOPS Java Interview Questions

Q1. What is Polymorphism?

Polymorphism is briefly described as “one interface, many implementations”. Polymorphism is a characteristic of being able to assign a different meaning or usage to something in different contexts – specifically, to allow an entity such as a variable, a function, or an object to have more than one form. There are two types of polymorphism:



1. Compile time polymorphism
2. Run time polymorphism

Compile time polymorphism is method overloading whereas Runtime time polymorphism is done using inheritance and interface.

Q2. What is runtime polymorphism or dynamic method dispatch?

In Java, runtime polymorphism or dynamic method dispatch is a process in which a call to an overridden method is resolved at runtime rather than at compile-time. In this process, an overridden method is called through the reference variable of a superclass. Let’s take a look at the example below to understand it better.

```
1  class Car {
2  void run()
3  {
4  System.out.println(“car is running”);
5  }
6  }
```

```
7  class Audi extends Car {
8  void run()
9  {
10 System.out.prinltn("Audi is running safely with 100km");
11 }
12 public static void main(String args[])
13 {
14 Car b= new Audi();
15 b.run();
16 }
17 }
```

Q3. What is abstraction in Java?

Abstraction refers to the quality of dealing with ideas rather than events. It basically deals with hiding the details and showing the essential things to the user. Thus you can say that abstraction in Java is the process of hiding the implementation details from the user and revealing only the functionality to them. Abstraction can be achieved in two ways:

1. **Abstract Classes** (0-100% of abstraction can be achieved)
2. **Interfaces** (100% of abstraction can be achieved)

Q4. What do you mean by an interface in Java?

An interface in Java is a blueprint of a class or you can say it is a collection of abstract methods and static constants. In an interface, each method is public and abstract but it does not contain any constructor. Thus, interface basically is a group of related methods with empty bodies. Example:

```
public interface Animal {

  public void eat();

  public void sleep();

  public void run();
```

}

Q5. What is the difference between abstract classes and interfaces?

Abstract Class	Interfaces
An abstract class can provide complete, default code and/or just the details that have to be overridden	An interface cannot provide any code at all, just the signature
In the case of an abstract class, a class may extend only one abstract class	A Class may implement several interfaces
An abstract class can have non-abstract methods	All methods of an Interface are abstract
An abstract class can have instance variables	An Interface cannot have instance variables
An abstract class can have any visibility: public, private, protected	An Interface visibility must be public (or) none
If we add a new method to an abstract class then we have the option of providing default implementation and therefore all the existing code might work properly	If we add a new method to an Interface then we have to track down all the implementations of the interface and define implementation for the new method
An abstract class can contain constructors	An Interface cannot contain constructors
Abstract classes are fast	Interfaces are slow as it requires extra indirection to find the corresponding method in the actual class

Q6. What is inheritance in Java?

Inheritance in Java is the concept where the properties of one class can be inherited by the other. It helps to reuse the code and establish a relationship between different classes. Inheritance is performed between two types of classes:

1. Parent class (Super or Base class)
2. Child class (Subclass or Derived class)

A class which inherits the properties is known as Child Class whereas a class whose properties are inherited is known as Parent class.

Q7. What are the different types of inheritance in Java?

Java supports four types of inheritance which are:

1. **Single Inheritance:** In single inheritance, one class inherits the properties of another i.e there will be only one parent as well as one child class.

2. **Multilevel Inheritance:** When a class is derived from a class which is also derived from another class, i.e. a class having more than one parent class but at different levels, such type of inheritance is called Multilevel Inheritance.
3. **Hierarchical Inheritance:** When a class has more than one child classes (subclasses) or in other words, more than one child classes have the same parent class, then such kind of inheritance is known as hierarchical.
4. **Hybrid Inheritance:** Hybrid inheritance is a combination of two *or more types* of inheritance.

Q8. What is method overloading and method overriding?

Method Overloading :

- In Method Overloading, Methods of the same class shares the same name but each method must have a different number of parameters or parameters having different types and order.
- Method Overloading is to “add” or “extend” more to the method’s behavior.
- It is a compile-time polymorphism.
- The methods must have a different signature.
- It may or may not need inheritance in Method Overloading.

Let’s take a look at the example below to understand it better.

```
1  class Adder {
2  Static int add(int a, int b)
3  {
4  return a+b;
5  }
6  Static double add( double a, double b)
7  {
8  return a+b;
9  }
10 public static void main(String args[])
11 {
```

```
12  System.out.println(Adder.add(11,11));
13  System.out.println(Adder.add(12.3,12.6));
14  }}
```

Method Overriding:

- In Method Overriding, the subclass has the same method with the same name and exactly the same number and type of parameters and same return type as a superclass.
- Method Overriding is to “Change” existing behavior of the method.
- It is a run time polymorphism.
- The methods must have the same signature.
- It always requires inheritance in Method Overriding.

Let’s take a look at the example below to understand it better.

```
1  class Car {
2  void run(){
3  System.out.println(“car is running”);
4  }
5  Class Audi extends Car{
6  void run()
7  {
8  System.out.prinltn("Audi is running safely with 100km");
9  }
10 public static void main( String args[])
11 {
12 Car b=new Audi();
13 b.run();
14 }
15 }
```

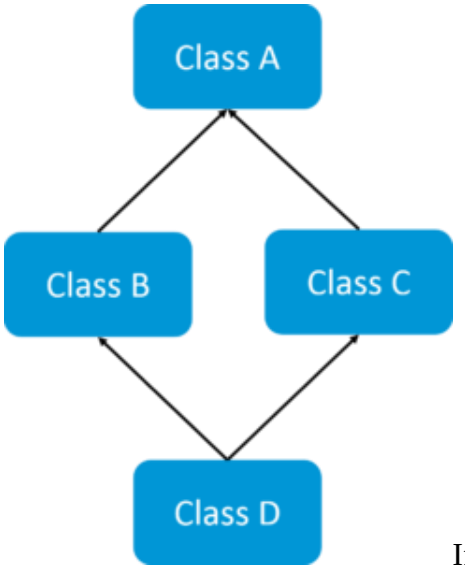
Q9. Can you override a private or static method in Java?

You cannot override a private or static method in Java. If you create a similar method with the same return type and same method arguments in child class then it will hide the superclass method; this is known as method hiding. Similarly, you cannot override a private method in subclass because it's not accessible there. What you can do is create another private method with the same name in the child class. Let's take a look at the example below to understand it better.

```
1  class Base {
2  private static void display() {
3  System.out.println("Static or class method from Base");
4  }
5  public void print() {
6  System.out.println("Non-static or instance method from Base");
7  }
8  class Derived extends Base {
9  private static void display() {
10 System.out.println("Static or class method from Derived");
11 }
12 public void print() {
13 System.out.println("Non-static or instance method from Derived");
14 }
15 public class test {
16 public static void main(String args[])
17 {
18 Base obj= new Derived();
19 obj1.display();
```

```
20  obj1.print();
21  }
22  }
```

Q10. What is multiple inheritance? Is it supported by Java?



If a child class inherits the property from multiple classes is known as multiple inheritance. Java does not allow to extend multiple classes.

The problem with multiple inheritance is that if multiple parent classes have the same method name, then at runtime it becomes difficult for the compiler to decide which method to execute from the child class.

Therefore, Java doesn't support multiple inheritance. The problem is commonly referred to as Diamond Problem.

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Q11. What is encapsulation in Java?

Encapsulation is a mechanism where you bind your data(variables) and code(methods) together as a single unit. Here, the data is hidden from the outer world and can be accessed only via current class methods. This helps in protecting the data from any unnecessary modification. We can achieve encapsulation in Java by:

- Declaring the variables of a class as private.
- Providing public setter and getter methods to modify and view the values of the variables.

Q12. What is an association?

Association is a relationship where all object have their own lifecycle and there is no owner. Let's take the example of Teacher and Student. Multiple students can associate with a single teacher and a single student can associate with multiple teachers but there is no ownership between the objects and both have their own lifecycle. These relationships can be one to one, one to many, many to one and many to many.

Q13. What do you mean by aggregation?

An aggregation is a specialized form of Association where all object has their own lifecycle but there is ownership and child object can not belong to another parent object. Let's take an example of Department and teacher. A single teacher can not belong to multiple departments, but if we delete the department teacher object will not destroy.

Q14. What is composition in Java?

Composition is again a specialized form of Aggregation and we can call this as a "death" relationship. It is a strong type of Aggregation. Child object does not have their lifecycle and if parent object deletes all child object will also be deleted. Let's take again an example of a relationship between House and rooms. House can contain multiple rooms there is no independent life of room and any room can not belongs to two different houses if we delete the house room will automatically delete.

Q15. What is a marker interface?

A Marker interface can be defined as the interface having no data member and member functions. In simpler terms, an empty interface is called the Marker interface. The most common examples of Marker interface in Java are Serializable, Cloneable etc. The marker interface can be declared as follows.

```
1  public interface Serializable{

2  }
```

Q16. What is object cloning in Java?

Object cloning in Java is the process of creating an exact copy of an object. It basically means the ability to create an object with a similar state as the original object. To achieve this, Java provides a method **clone()** to make use of this functionality. This method creates a new instance of the class of the current object and then initializes all its fields with the exact same contents of corresponding fields. To object clone(), the marker interface **java.lang.Cloneable** must be implemented to avoid any runtime exceptions. One thing you must note is Object clone() is a protected method, thus you need to override it.

Q17. What is a copy constructor in Java?

Copy constructor is a member function that is used to initialize an object using another object of the same class. Though there is no need for copy constructor in Java since all objects are passed by reference. Moreover, Java does not even support automatic pass-by-value.

Q18. What is a constructor overloading in Java?

In Java, constructor overloading is a technique of adding any number of constructors to a class each having a different parameter list. The compiler uses the number of parameters and their types in the list to differentiate the overloaded constructors.

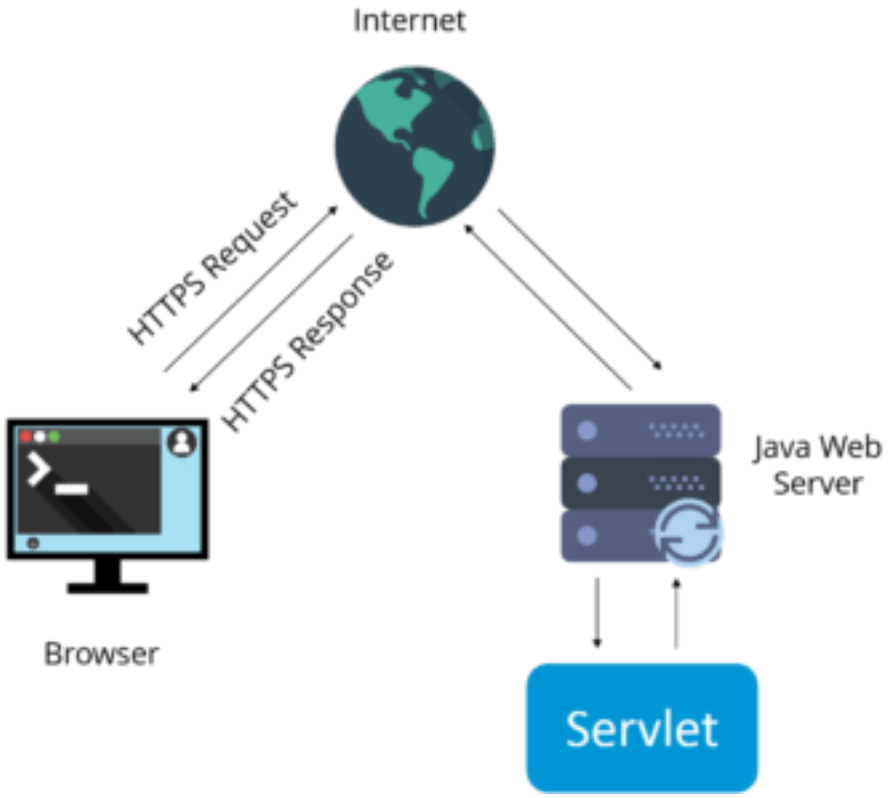
```
1
2   class Demo
3   {
4   int i;
5   public Demo(int a)
6   {
7   i=k;
8   }
9   public Demo(int a, int b)
10  {
11  }
12  }
```

In case you are facing any challenges with these java interview questions, please comment on your problems in the section below. Apart from this Java Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for a structured training from edureka!

Servlets Interview Questions

Q1. What is a servlet?

- Java Servlet is server-side technologies to extend the capability of web servers by providing support for dynamic response and data persistence.
- The javax.servlet and javax.servlet.http packages provide interfaces and classes for writing our own servlets.
- All servlets must implement the javax.servlet.Servlet interface, which defines servlet lifecycle methods. When implementing a generic service, we can extend the GenericServlet class provided with the Java Servlet API. The HttpServlet class provides methods, such as doGet() and doPost(), for handling HTTP-specific services.
- Most of the times, web applications are accessed using HTTP protocol and thats why we mostly extend HttpServlet class. Servlet API hierarchy is shown in below image.



Q2. What are the differences between Get and Post methods?

Get	Post

Limited amount of data can be sent because data is sent in header.	Large amount of data can be sent because data is sent in body.
Not Secured because data is exposed in URL bar.	Secured because data is not exposed in URL bar.
Can be bookmarked	Cannot be bookmarked
Idempotent	Non-Idempotent
It is more efficient and used than Post	It is less efficient and used

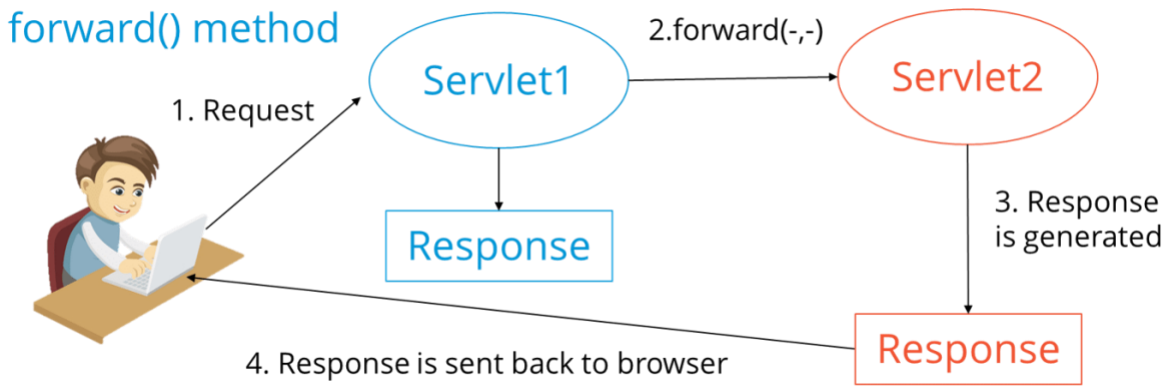
Q3. What is Request Dispatcher?

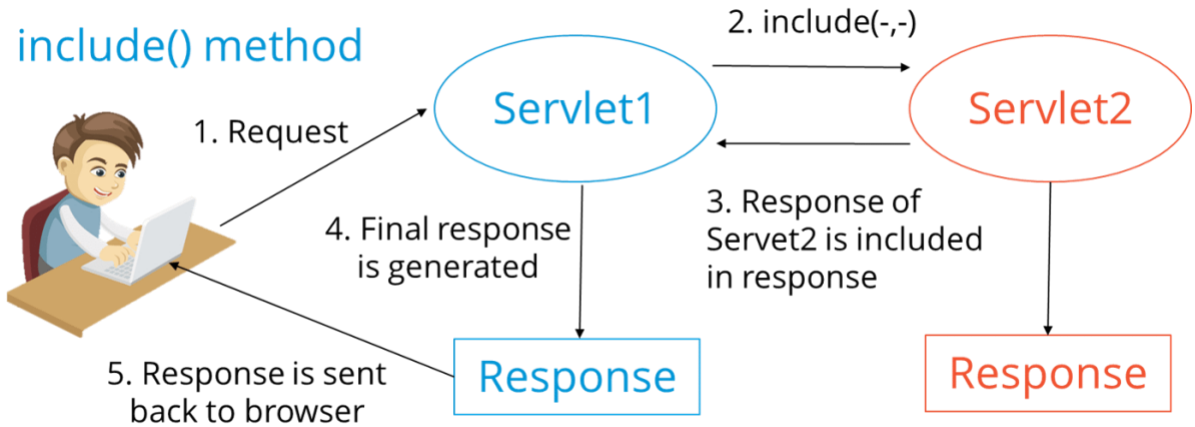
RequestDispatcher interface is used to forward the request to another resource that can be HTML, JSP or another servlet in same application. We can also use this to include the content of another resource to the response.

There are two methods defined in this interface:

1.void forward()

2.void include()





Q4. What are the differences between forward() method and sendRedirect() methods?

forward() method	SendRedirect() method
forward() sends the same request to another resource.	sendRedirect() method sends new request always because it uses the URL bar of the browser.
forward() method works at server side.	sendRedirect() method works at client side.
forward() method works within the server only.	sendRedirect() method works within and outside the server.

Q5. What is the life-cycle of a servlet?

There are 5 stages in the lifecycle of a servlet:

1. Servlet is loaded
2. Servlet is instantiated
3. Servlet is initialized
4. Service the request
5. Servlet is destroyed

Q6. How does cookies work in Servlets?

- Cookies are text data sent by server to the client and it gets saved at the client local machine.
- Servlet API provides cookies support through javax.servlet.http.Cookie class that implements Serializable and Cloneable interfaces.
- HttpServletRequest getCookies() method is provided to get the array of Cookies from request, since there is no point of adding Cookie to request, there are no methods to set or add cookie to request.
- Similarly HttpServletResponse addCookie(Cookie c) method is provided to attach cookie in response header, there are no getter methods for cookie.

Q7. What are the differences between ServletContext vs ServletConfig?

The difference between ServletContext and ServletConfig in Servlets JSP is in below tabular format.

ServletConfig	ServletContext
Servlet config object represent single servlet	It represent whole web application running on particular JVM and common for all the servlet
Its like local parameter associated with particular servlet	Its like global parameter associated with whole application
It's a name value pair defined inside the servlet section of web.xml file so it has servlet wide scope	ServletContext has application wide scope so define outside of servlet tag in web.xml file.
getServletConfig() method is used to get the config object	getServletContext() method is used to get the context object.
for example shopping cart of a user is a specific to particular user so here we can use servlet config	To get the MIME type of a file or application session related information is stored using servlet context object.

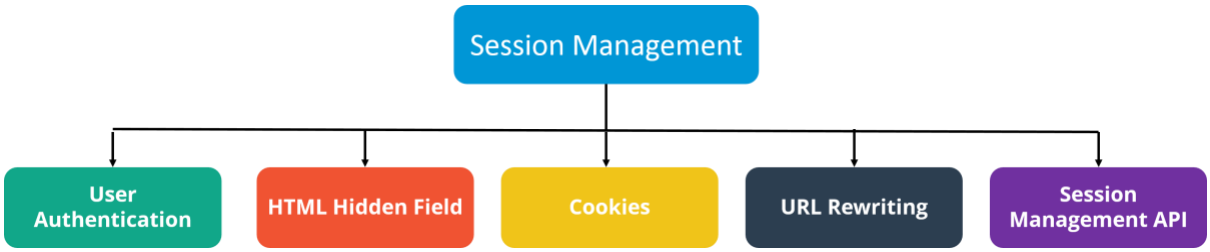
Q8. What are the different methods of session management in servlets?

Session is a conversational state between client and server and it can consists of multiple request and response between client and server. Since HTTP and Web Server both are stateless, the only way to maintain a session is when some unique information about the session (session id) is passed between server and client in every request and response.

Some of the common ways of session management in servlets are:

1. User Authentication
2. HTML Hidden Field
3. Cookies

4. URL Rewriting
5. Session Management API



In case you are facing any challenges with these java interview questions, please comment your problems in the section below. Apart from this Java Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for a structured training from edureka! Click below to know more.

JDBC Interview Questions

1. What is JDBC Driver?

JDBC Driver is a software component that enables java application to interact with the database. There are 4 types of JDBC drivers:

1. JDBC-ODBC bridge driver
2. Native-API driver (partially java driver)
3. Network Protocol driver (fully java driver)

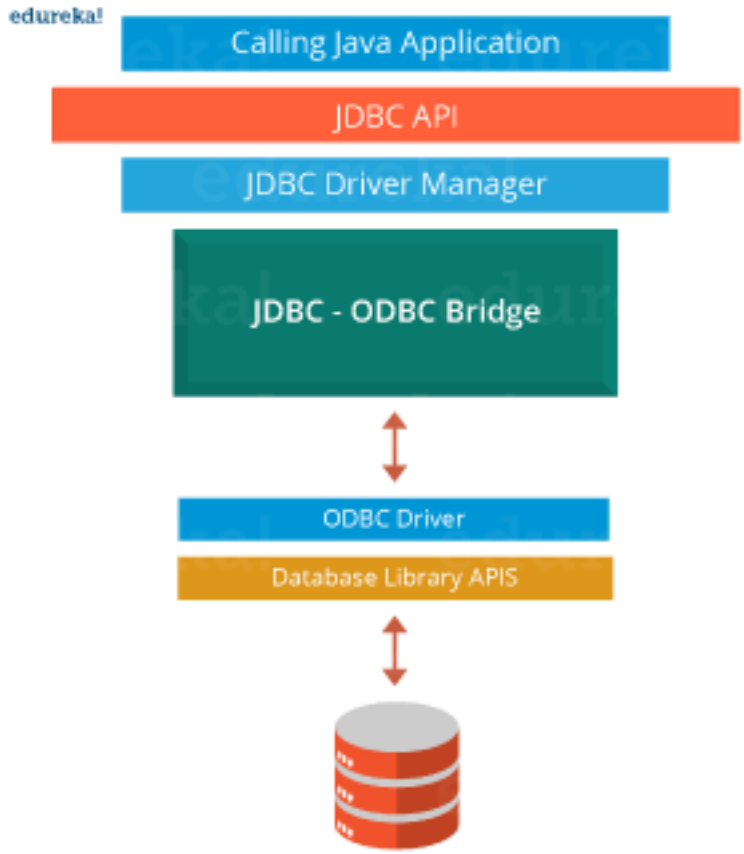
4. Thin driver

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

5.
6. (fully java driver)



2. What are the steps to connect to a database in java?

- Registering the driver class
- Creating connection
- Creating statement
- Executing queries
- Closing connection

3. What are the JDBC API components?

The java.sql package contains interfaces and classes for JDBC API.

Interfaces:

- Connection
- Statement
- PreparedStatement
- ResultSet

- ResultSetMetaData
- DatabaseMetaData
- CallableStatement etc.

Classes:

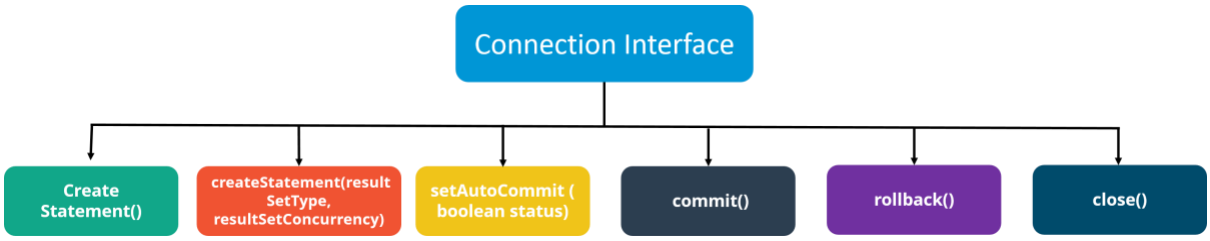
- DriverManager
- Blob
- Clob
- Types
- SQLException etc.

4. What is the role of JDBC DriverManager class?

The DriverManager *class* manages the registered drivers. It can be used to register and unregister drivers. It provides factory method that returns the instance of Connection.

5. What is JDBC Connection interface?

The Connection interface maintains a session with the database. It can be used for transaction management. It provides factory methods that returns the instance of Statement, PreparedStatement, CallableStatement and DatabaseMetaData.



6. What is the purpose of JDBC ResultSet interface?

The ResultSet object represents a row of a table. It can be used to change the cursor pointer and get the information from the database.

7. What is JDBC ResultSetMetaData interface?

The ResultSetMetaData interface returns the information of table such as total number of columns, column name, column type etc.

8. What is JDBC DatabaseMetaData interface?

The DatabaseMetaData interface returns the information of the database such as username, driver name, driver version, number of tables, number of views etc.

9. What do you mean by batch processing in JDBC?

Batch processing helps you to group related SQL statements into a batch and execute them instead of executing a single query. By using batch processing technique in JDBC, you can execute multiple queries which makes the performance faster.

10. What is the difference between execute, executeQuery, executeUpdate?

Statement ***execute(String query)*** is used to execute any SQL query and it returns TRUE if the result is an ResultSet such as running Select queries. The output is FALSE when there is no ResultSet object such as running Insert or Update queries. We can use *getResultSet()* to get the ResultSet and *getUpdateCount()* method to retrieve the update count.

Statement ***executeQuery(String query)*** is used to execute Select queries and returns the ResultSet. ResultSet returned is never null even if there are no records matching the query. When executing select queries we should use executeQuery method so that if someone tries to execute insert/update statement it will throw java.sql.SQLException with message "executeQuery method can not be used for update".

Statement **executeUpdate(String query**) is used to execute Insert/Update/Delete (DML) statements or DDL statements that returns nothing. The output is int and equals to the row count for SQL Data Manipulation Language (DML) statements. For DDL statements, the output is 0.

You should use execute() method only when you are not sure about the type of statement else use executeQuery or executeUpdate method.

Q11. What do you understand by JDBC Statements?

JDBC statements are basically the statements which are used to send SQL commands to the database and retrieve data back from the database. Various methods like execute(), executeUpdate(), executeQuery, etc. are provided by JDBC to interact with the database.

JDBC supports 3 types of statements:

1. *Statement:* Used for general purpose access to the database and executes a static SQL query at runtime.
2. *PreparedStatement:* Used to provide input parameters to the query during execution.
3. *CallableStatement:* Used to access the database stored procedures and helps in accepting runtime parameters.

In case you are facing any challenges with these java interview questions, please comment your problems in the section below. Apart from this Java Interview Questions Blog, if you

want to get trained from professionals on this technology, you can opt for a structured training from edureka!

Spring Interview Questions

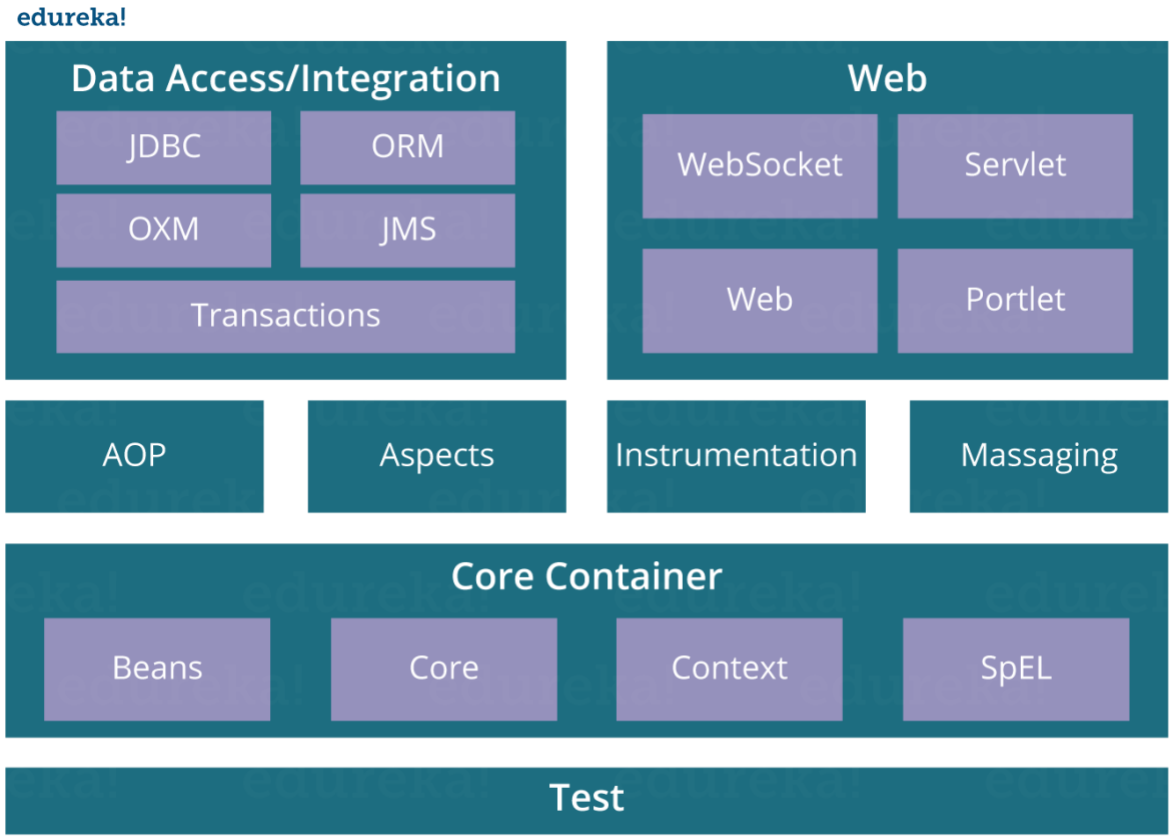
Q1. What is Spring?

Wikipedia defines the Spring framework as "an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform." Spring is essentially a lightweight, integrated framework that can be used for developing enterprise applications in java.

Q2. Name the different modules of the Spring framework.

Some of the important Spring Framework modules are:

- Spring Context – for dependency injection.
- Spring AOP – for aspect oriented programming.
- Spring DAO – for database operations using DAO pattern
- Spring JDBC – for JDBC and DataSource support.
- Spring ORM – for ORM tools support such as Hibernate
- Spring Web Module – for creating web applications.
- Spring MVC – Model-View-Controller implementation for creating web applications, web services etc.



Q3. List some of the important annotations in annotation-based Spring configuration.

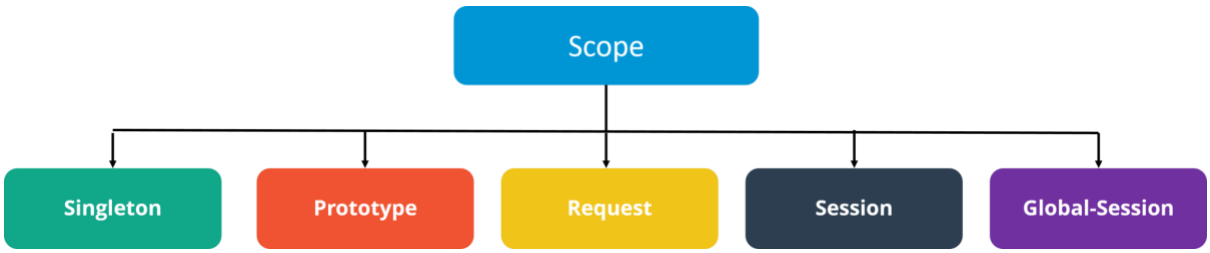
The important annotations are:

- @Required
- @Autowired
- @Qualifier
- @Resource
- @PostConstruct
- @PreDestroy

Q4. Explain Bean in Spring and List the different Scopes of Spring bean.

Beans are objects that form the backbone of a Spring application. They are managed by the Spring IoC container. In other words, a bean is an object that is instantiated, assembled, and managed by a Spring IoC container.

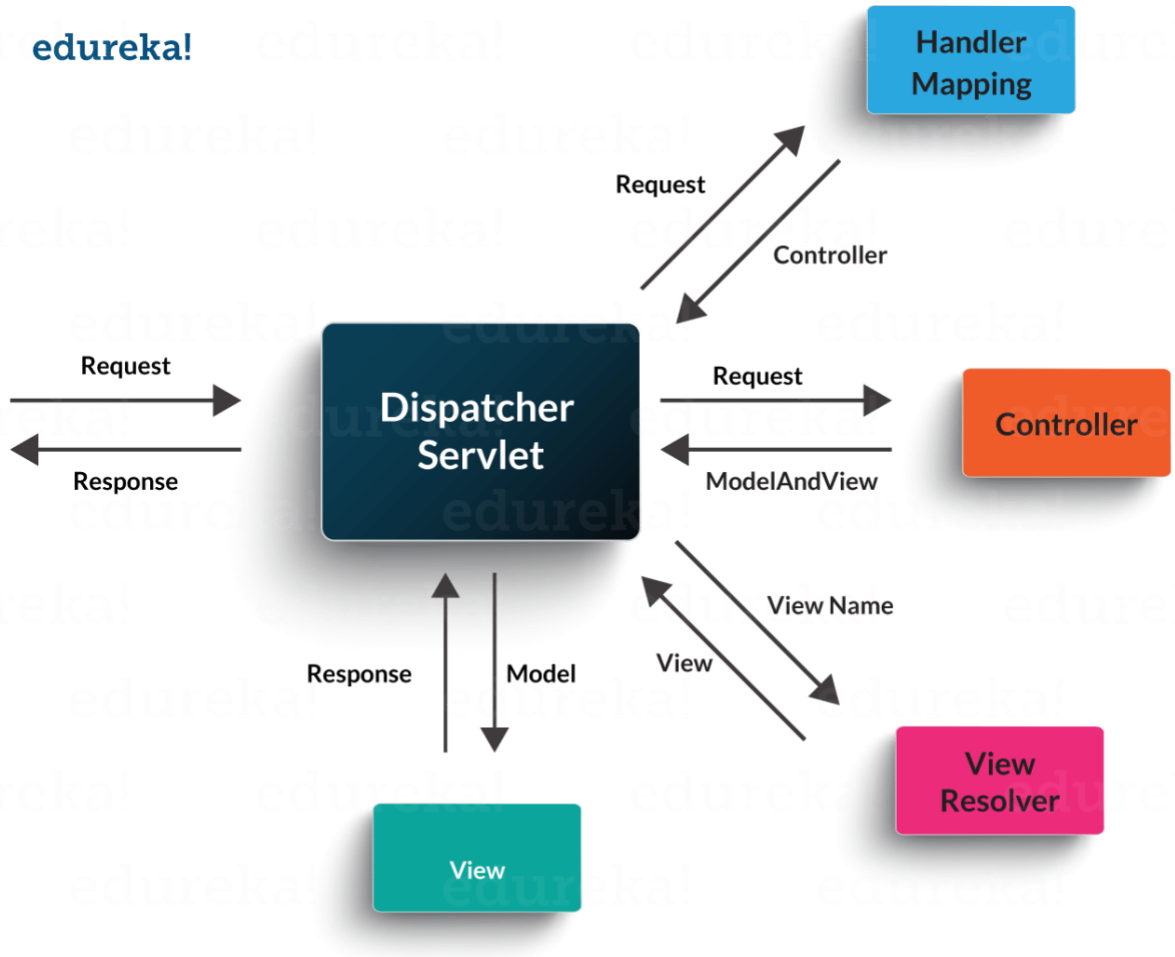
There are five Scopes defined in Spring beans.



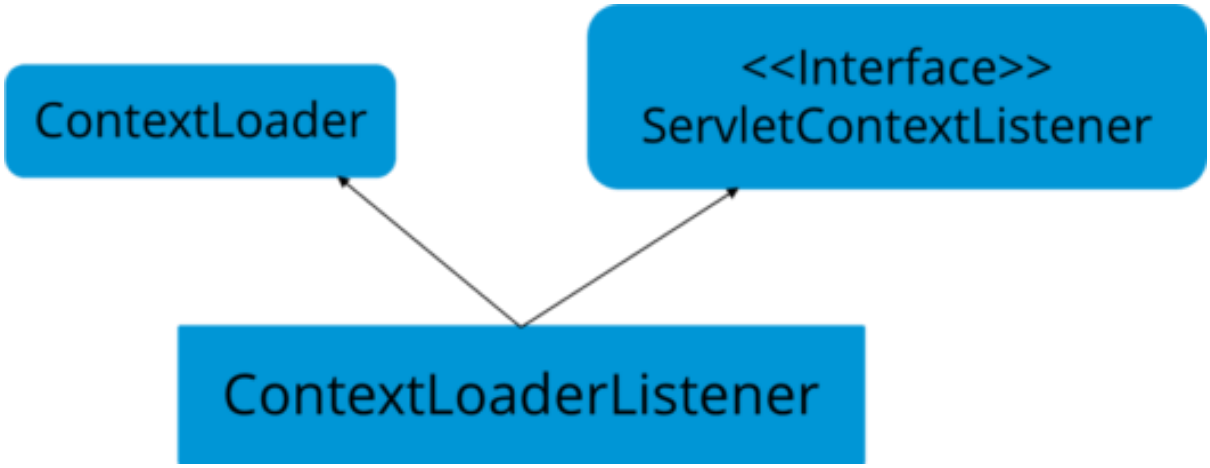
- **Singleton**: Only one instance of the bean will be created for each container. This is the default scope for the spring beans. While using this scope, make sure spring bean doesn't have shared instance variables otherwise it might lead to data inconsistency issues because it's not thread-safe.
- **Prototype**: A new instance will be created every time the bean is requested.
- **Request**: This is same as prototype scope, however it's meant to be used for web applications. A new instance of the bean will be created for each HTTP request.
- **Session**: A new bean will be created for each HTTP session by the container.
- **Global-session**: This is used to create global session beans for Portlet applications.

Q5. Explain the role of DispatcherServlet and ContextLoaderListener.

DispatcherServlet is basically the front controller in the Spring MVC application as it loads the spring bean configuration file and initializes all the beans that have been configured. If annotations are enabled, it also scans the packages to configure any bean annotated with @Component, @Controller, @Repository or @Service annotations.



ContextLoaderListener, on the other hand, is the listener to start up and shut down the WebApplicationContext in Spring root. Some of its important functions includes tying up the lifecycle of Application Context to the lifecycle of the ServletContext and automating the creation of ApplicationContext.



Q6. What are the differences between constructor injection and setter injection?

No.	Constructor Injection	Setter Injection
1)	No Partial Injection	Partial Injection
2)	Doesn't override the setter property	Overrides the constructor property if both are defined.
3)	Creates a new instance if any modification occurs	Doesn't create a new instance if you change the property value
4)	Better for too many properties	Better for a few properties.

Q7. What is autowiring in Spring? What are the autowiring modes?

Autowiring enables the programmer to inject the bean automatically. We don't need to write explicit injection logic. Let's see the code to inject bean using dependency injection.

1. <bean id="emp" class="com.javatpoint.Employee" autowire="byName"/>

The autowiring modes are given below:

No.	Mode	Description
1)	no	this is the default mode, it means autowiring is not enabled.
2)	byName	Injects the bean based on the property name. It uses setter method.
3)	byType	Injects the bean based on the property type. It uses setter method.
4)	constructor	It injects the bean using constructor

Q8. How to handle exceptions in Spring MVC Framework?

Spring MVC Framework provides the following ways to help us achieving robust exception handling.

Controller Based:

We can define exception handler methods in our controller classes. All we need is to annotate these methods with @ExceptionHandler annotation.

Global Exception Handler:

Exception Handling is a cross-cutting concern and Spring provides @ControllerAdvice annotation that we can use with any class to define our global exception handler.

HandlerExceptionResolver implementation:

For generic exceptions, most of the times we serve static pages. Spring Framework provides HandlerExceptionResolver interface that we can implement to create global exception handler. The reason behind this additional way to define global exception handler is that Spring framework also provides default implementation classes that we can define in our spring bean configuration file to get spring framework exception handling benefits.

Q9. What are some of the important Spring annotations which you have used?

Some of the Spring annotations that I have used in my project are:

@Controller – for controller classes in Spring MVC project.

@RequestMapping – for configuring URI mapping in controller handler methods. This is a very important annotation, so you should go through Spring MVC RequestMapping Annotation Examples

@ResponseBody – for sending Object as response, usually for sending XML or JSON data as response.

@PathVariable – for mapping dynamic values from the URI to handler method arguments.

@Autowired – for autowiring dependencies in spring beans.

@Qualifier – with @Autowired annotation to avoid confusion when multiple instances of bean type is present.

@Service – for service classes.

@Scope – for configuring the scope of the spring bean.

@Configuration, @ComponentScan and @Bean – for java based configurations.

AspectJ annotations for configuring aspects and advices , @Aspect, @Before, @After, @Around, @Pointcut, etc.

Q10. How to integrate Spring and Hibernate Frameworks?

We can use Spring ORM module to integrate Spring and Hibernate frameworks if you are using Hibernate 3+ where SessionFactory provides current session, then you should avoid using HibernateTemplate or HibernateDaoSupportclasses and better to use DAO pattern with dependency injection for the integration.

Also, Spring ORM provides support for using Spring declarative transaction management, so you should utilize that rather than going for hibernate boiler-plate code for transaction management.

Q11. Name the types of transaction management that Spring supports.

Two types of transaction management are supported by Spring. They are:

1. **Programmatic transaction management:** In this, the transaction is managed with the help of programming. It provides you extreme flexibility, but it is very difficult to maintain.
2. **Declarative transaction management:** In this, transaction management is separated from the business code. Only annotations or XML based configurations are used to manage the transactions.

In case you are facing any challenges with these java interview questions, please comment your problems in the section below. Apart from this Java Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for structured training from edureka!

Hibernate Interview Questions

1. What is Hibernate Framework?

Object-relational mapping or ORM is the programming technique to map application domain model objects to the relational database tables. Hibernate is Java-based ORM tool that provides a framework for mapping application domain objects to the relational database tables and vice versa.

Hibernate provides a reference implementation of Java Persistence API, that makes it a great choice as ORM tool with benefits of loose coupling. We can use the Hibernate persistence API for CRUD operations. Hibernate framework provide option to map plain old java objects to traditional database tables with the use of JPA annotations as well as XML based configuration.

Similarly, hibernate configurations are flexible and can be done from XML configuration file as well as programmatically.

2. What are the important benefits of using Hibernate Framework?

Some of the important benefits of using hibernate framework are:

1. Hibernate eliminates all the boiler-plate code that comes with JDBC and takes care of managing resources, so we can focus on business logic.
2. Hibernate framework provides support for XML as well as JPA annotations, that makes our code implementation independent.
3. Hibernate provides a powerful query language (HQL) that is similar to SQL. However, HQL is fully object-oriented and understands concepts like inheritance, polymorphism, and association.
4. Hibernate is an open source project from Red Hat Community and used worldwide. This makes it a better choice than others because learning curve is small and there are tons of online documentation and help is easily available in forums.
5. Hibernate is easy to integrate with other Java EE frameworks, it's so popular that Spring Framework provides built-in support for integrating hibernate with Spring applications.
6. Hibernate supports lazy initialization using proxy objects and perform actual database queries only when it's required.
7. Hibernate cache helps us in getting better performance.
8. For database vendor specific feature, hibernate is suitable because we can also execute native sql queries.

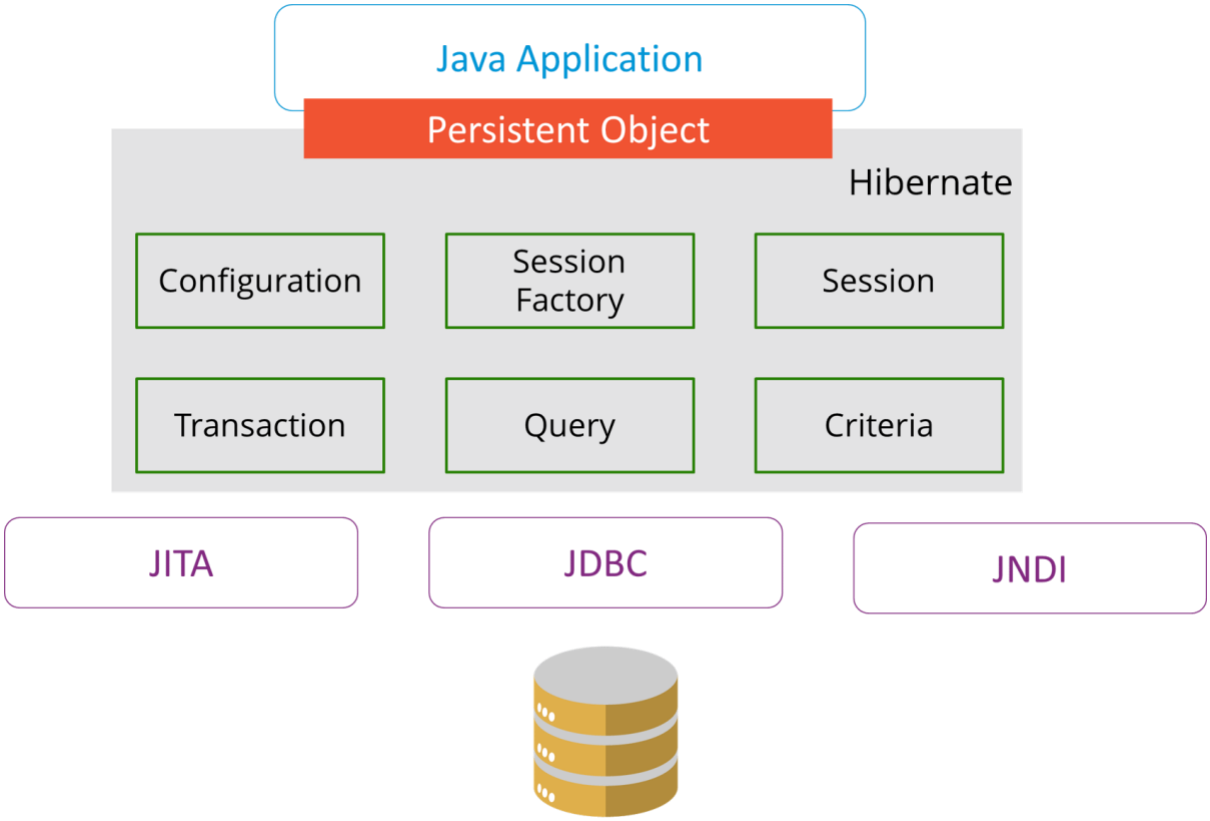
Overall hibernate is the best choice in current market for ORM tool, it contains all the features that you will ever need in an ORM tool.

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

3. Explain Hibernate architecture.



4. What are the differences between get and load methods?

The differences between get() and load() methods are given below.

No.	get()	load()
1)	Returns null if object is not found.	Throws ObjectNotFoundExceptionif an object is not found.
2)	get() method always hit the database.	load() method doesn't hit the database.
3)	It returns a real object, not a proxy.	It returns a proxy object.
4)	It should be used if you are not sureabout the existence of instance.	It should be used if you are surethat the instance exists.

5. What are the advantages of Hibernate over JDBC?

Some of the important advantages of Hibernate framework over JDBC are:

1. Hibernate removes a lot of boiler-plate code that comes with JDBC API, the code looks cleaner and readable.

2. Hibernate supports inheritance, associations, and collections. These features are not present with JDBC API.
3. Hibernate implicitly provides transaction management, in fact, most of the queries can't be executed outside transaction. In JDBC API, we need to write code for transaction management using commit and rollback.
4. JDBC API throws SQLException that is a checked exception, so we need to write a lot of try-catch block code. Most of the times it's redundant in every JDBC call and used for transaction management. Hibernate wraps JDBC exceptions and throw JDBCException or HibernateException un-checked exception, so we don't need to write code to handle it. Hibernate built-in transaction management removes the usage of try-catch blocks.
5. Hibernate Query Language (HQL) is more object-oriented and close to Java programming language. For JDBC, we need to write native SQL queries.
6. Hibernate supports caching that is better for performance, JDBC queries are not cached hence performance is low.
7. Hibernate provides option through which we can create database tables too, for JDBC tables must exist in the database.
8. Hibernate configuration helps us in using JDBC like connection as well as JNDI DataSource for the connection pool. This is a very important feature in enterprise application and completely missing in JDBC API.
9. Hibernate supports JPA annotations, so the code is independent of the implementation and easily replaceable with other ORM tools. JDBC code is very tightly coupled with the application.

In case you are facing any challenges with these Java interview questions, please comment on your problems in the section below. Apart from this Java Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for structured training from edureka!

Java Interview Questions: JSP

1. What are the life-cycle methods for a jsp?

Methods	Description
public void jspInit()	It is invoked only once, same as init method of servlet.
public void _jspService(ServletRequest request,ServletResponse)throws ServletException,IOException	It is invoked at each request, same as service() method of servlet.
public void jspDestroy()	It is invoked only once, same as destroy() method of servlet.

2. What are the JSP implicit objects?

JSP provides 9 implicit objects by default. They are as follows:

Object	Type
1) out	JspWriter
2) request	HttpServletRequest
3) response	HttpServletResponse
4) config	ServletConfig
5) session	HttpSession
6) application	ServletContext
7) pageContext	PageContext
8) page	Object
9) exception	Throwable

3. What are the differences between include directive and include action?

include directive	include action
The include directive includes the content at page translation time.	The include action includes the content at request time.
The include directive includes the original content of the page so page size increases at runtime.	The include action doesn't include the original content rather invokes the include() method of Vendor provided class.
It's better for static pages.	It's better for dynamic pages.

4. How to disable caching on back button of the browser?

```
<%
response.setHeader("Cache-Control","no-store");
response.setHeader("Pragma","no-cache");
response.setHeader ("Expires", "0");                    //prevents caching at the proxy server
%>
```

5. What are the different tags provided in JSTL?

There are 5 type of JSTL tags.

1. core tags
2. sql tags
3. xml tags
4. internationalization tags

5. functions tags

6. How to disable session in JSP?

1. <%@ page session="false" %>

7. How to delete a Cookie in a JSP?

The following code explains how to delete a Cookie in a JSP :

```
1
2
3
Cookie mycook = new Cookie("name1","value1");
4
response.addCookie(mycook1);
5
Cookie killmycook = new Cookie("mycook1","value1");
6
killmycook . set MaxAge ( 0 );
7
killmycook . set Path ("/");
8
killmycook . addCookie ( killmycook 1 );
9
10
11
```

8. Explain the jspDestroy() method.

jspDestry() method is invoked from **javax.servlet.jsp.JspPage** interface whenever a JSP page is about to be destroyed. Servlets destroy methods can be easily overridden to perform cleanup, like when closing a database connection.

9. How is JSP better than Servlet technology?

JSP is a technology on the server's side to make content generation simple. They are document-centric, whereas servlets are programs. A Java server page can contain fragments of Java program, which execute and instantiate Java classes. However, they occur inside an HTML template file. It provides the framework for the development of a Web Application.

10. Why should we not configure JSP standard tags in web.xml?

We don't need to configure JSP standard tags in web.xml because when container loads the web application and find TLD files, it automatically configures them to be used directly in the application JSP pages. We just need to include it in the JSP page using taglib directive.

11. How will you use JSP EL in order to get the HTTP method name?

Using pageContext JSP EL implicit object you can get the request object reference and make use of the dot operator to retrieve the HTTP method name in the JSP page. The JSP EL code for this purpose will look like ${pageContext.request.method}.

In case you are facing any challenges with these java interview questions, please comment on your problems in the section below. Apart from this Java Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for structured training from edureka!

Exception and Thread Java Interview Questions

Q1. What is the difference between Error and Exception?

An error is an irrecoverable condition occurring at runtime. Such as OutOfMemory error. These JVM errors you cannot repair them at runtime. Though error can be caught in the catch block but the execution of application will come to a halt and is not recoverable.

While exceptions are conditions that occur because of bad input or human error etc. e.g. FileNotFoundException will be thrown if the specified file does not exist. Or a NullPointerException will take place if you try using a null reference. In most of the cases it is possible to recover from an exception (probably by giving the user feedback for entering proper values etc.

Q2. How can you handle Java exceptions?

There are five keywords used to handle exceptions in Java:

1. try
2. catch
3. finally
4. throw
5. throws

Q3. What are the differences between Checked Exception and Unchecked Exception?

Checked Exception

- The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions.
- Checked exceptions are checked at compile-time.
- Example: IOException, SQLException etc.

Unchecked Exception

- The classes that extend RuntimeException are known as unchecked exceptions.
- Unchecked exceptions are not checked at compile-time.
- Example: ArithmeticException, NullPointerException etc.

Q4. What purpose do the keywords final, finally, and finalize fulfill?

Final:

Final is used to apply restrictions on class, method, and variable. A final class can't be inherited, final method can't be overridden and final variable value can't be changed. Let's take a look at the example below to understand it better.

```
1  class FinalVarExample {
2  public static void main( String args[])
3  {
4  final int a=10;
5  a=50;
6  }
```

Finally

Finally is used to place important code, it will be executed whether the exception is handled or not. Let's take a look at the example below to understand it better.

```
1  class FinallyExample {
2  public static void main(String args[]){
```

```
3  try {
4  int x=100;
5  }
6  catch(Exception e) {
7  System.out.println(e);
8  }
9  finally {
10 System.out.println("finally block is executing");}
11 }}
12 }
```

Finalize

Finalize is used to perform clean up processing just before the object is garbage collected. Let's take a look at the example below to understand it better.

```
1  class FinalizeExample {
2  public void finalize() {
3  System.out.println("Finalize is called");
4  }
5  public static void main(String args[])
6  {
7  FinalizeExample f1=new FinalizeExample();
8  FinalizeExample f2=new FinalizeExample();
9  f1= NULL;
10 f2=NULL;
11 System.gc();
```

```
12  }
13  }
```

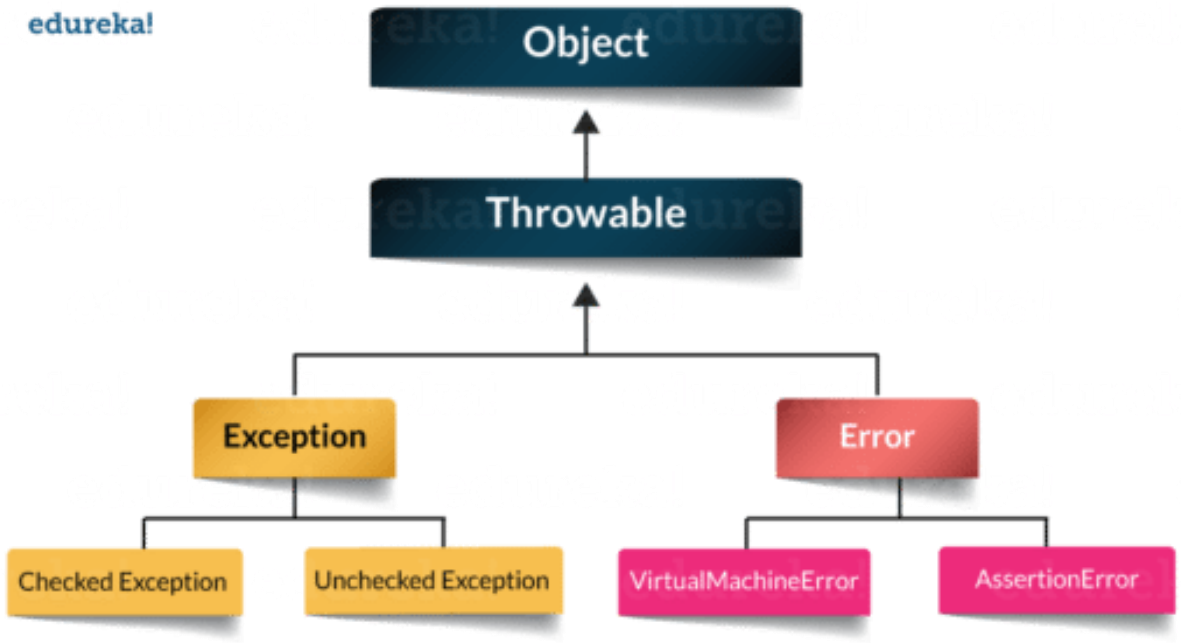
Q5. What are the differences between throw and throws?

throw keyword	**throws keyword**
Throw is used to explicitly throw an exception.	Throws is used to declare an exception.
Checked exceptions can not be propagated with throw only.	Checked exception can be propagated with throws.
Throw is followed by an instance.	Throws is followed by class.
Throw is used within the method.	Throws is used with the method signature.
You cannot throw multiple exception	You can declare multiple exception e.g. public void method()throws IOException,SQLException.

Q6. What is exception hierarchy in java?

The hierarchy is as follows:

Throwable is a parent class of all Exception classes. There are two types of Exceptions: Checked exceptions and UncheckedExceptions or RunTimeExceptions. Both type of exceptions extends Exception class whereas errors are further classified into Virtual Machine error and Assertion error.



Q7. How to create a custom Exception?

To create you own exception extend the Exception class or any of its subclasses.

- class New1Exception extends Exception { } // this will create Checked Exception
- class NewException extends IOException { } // this will create Checked exception
- class NewException extends NullPonterExcpetion { } // this will create UnChecked exception

Q8. What are the important methods of Java Exception Class?

Exception and all of it's subclasses doesn't provide any specific methods and all of the methods are defined in the base class Throwable.

1. **String getMessage()** – This method returns the message String of Throwable and the message can be provided while creating the exception through it's constructor.
2. **String getLocalizedMessage()** – This method is provided so that subclasses can override it to provide locale specific message to the calling program. Throwable class implementation of this method simply use getMessage() method to return the exception message.
3. **Synchronized Throwable getCause()** – This method returns the cause of the exception or null id the cause is unknown.
4. **String toString()** – This method returns the information about Throwable in String format, the returned String contains the name of Throwable class and localized message.
5. **void printStackTrace()** – This method prints the stack trace information to the standard error stream, this method is overloaded and we can pass PrintStream or PrintWriter as an argument to write the stack trace information to the file or stream.

Q9. What are the differences between processes and threads?

	Process	**Thread**
Definition	An executing instance of a program is called a process.	A thread is a subset of the process.
Communication	Processes must use inter-process communication to communicate with sibling processes.	Threads can directly communicate with other threads of its process.
Control	Processes can only exercise control over child processes.	Threads can exercise considerable control over threads of the same process.

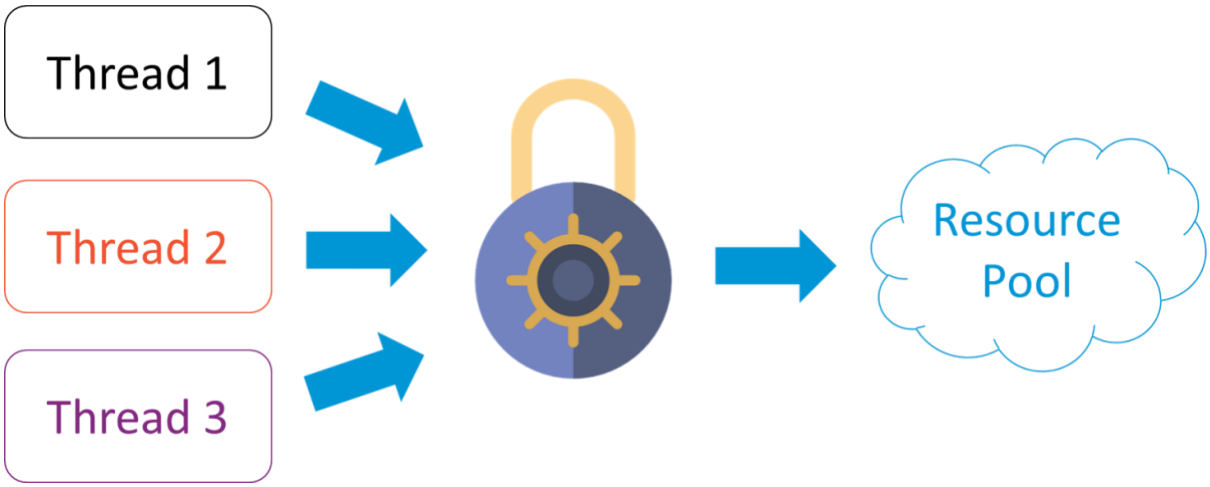
Changes	Any change in the parent process does not affect child processes.	Any change in the main thread may affect the behavior of the other threads of the process.
Memory	Run in separate memory spaces.	Run in shared memory spaces.
Controlled by	Process is controlled by the operating system.	Threads are controlled by programmer in a program.
Dependence	Processes are independent.	Threads are dependent.

Q10. What is a finally block? Is there a case when finally will not execute?

Finally block is a block which always executes a set of statements. It is always associated with a try block regardless of any exception that occurs or not.
Yes, finally will not be executed if the program exits either by calling System.exit() or by causing a fatal error that causes the process to abort.

Q11. What is synchronization?

Synchronization refers to multi-threading. A synchronized block of code can be executed by only one thread at a time. As Java supports execution of multiple threads, two or more threads may access the same fields or objects. Synchronization is a process which keeps all concurrent threads in execution to be in sync. Synchronization avoids memory consistency errors caused due to inconsistent view of shared memory. When a method is declared as synchronized the thread holds the monitor for that method's object. If another thread is executing the synchronized method the thread is blocked until that thread releases the monitor.



Q12. Can we write multiple catch blocks under single try block?

Yes we can have multiple catch blocks under single try block but the approach should be from specific to general. Let's understand this with a programmatic example.

```
1  public class Example {
2  public static void main(String args[]) {
3  try {
4  int a[]= new int[10];
5  a[10]= 10/0;
6  }
7  catch(ArithmeticException e)
8  {
9  System.out.println("Arithmetic exception in first catch block");
10 }
11 catch(ArrayIndexOutOfBoundsException e)
12 {
13 System.out.println("Array index out of bounds in second catch
   block");
14 }
15 catch(Exception e)
16 {
17 System.out.println("Any exception in third catch block");
18 }
19 }
```

Q13. What are the important methods of Java Exception Class?

Methods are defined in the base class Throwable. Some of the important methods of Java exception class are stated below.

1. **String getMessage()** – This method returns the message String about the exception. The message can be provided through its constructor.
2. **public StackTraceElement[] getStackTrace() –** This method returns an array containing each element on the stack trace. The element at index 0 represents the

top of the call stack whereas the last element in the array represents the method at the bottom of the call stack.

3. **Synchronized Throwable getCause()** – This method returns the cause of the exception or null id as represented by a Throwable object.
4. **String toString()** – This method returns the information in String format. The returned String contains the name of Throwable class and localized message.
5. **void printStackTrace()** – This method prints the stack trace information to the standard error stream.

Q14. What is OutOfMemoryError in Java?

OutOfMemoryError is the subclass of java.lang.Error which generally occurs when our JVM runs out of memory.

Q15. What is a Thread?

A thread is the smallest piece of programmed instructions which can be executed independently by a scheduler. In Java, all the programs will have at least one thread which is known as the main thread. This main thread is created by the JVM when the program starts its execution. The main thread is used to invoke the main() of the program.

Q16. What are the two ways to create a thread?

In Java, threads can be created in the following two ways:-

- By implementing the Runnable interface.
- By extending the Thread

Q17. What are the different types of garbage collectors in Java?

Garbage collection in Java a program which helps in implicit memory management. Since in Java, using the new keyword you can create objects dynamically, which once created will consume some memory. Once the job is done and there are no more references left to the object, Java using garbage collection destroys the object and relieves the memory occupied by it. Java provides four types of garbage collectors:

- Serial Garbage Collector
- Parallel Garbage Collector
- CMS Garbage Collector
- G1 Garbage Collector

In case you are facing any challenges with these java interview questions, please comment your problems in the section below. Apart from this Java Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for structured training from edureka!

So this brings us to the end of the Java interview questions blog. The topics that you learned in this Java Interview Questions blog are the most sought-after skill sets that recruiters look for in a Java Professional. These set of Java Interview Questions will definitely help you ace your job interview. **Good luck with your interview!**

*Check out the **Java Training** by Edureka, a trusted online learning company with a network of more than 250,000 satisfied learners spread across the globe. We are here to help you with every step on your journey, for becoming a besides this java interview questions, we come up with a curriculum which is designed for students and professionals who want to be a Java Developer. The course is designed to give you a head start into Java programming and train you for both core and advanced Java concepts along with various Java frameworks like Hibernate & Spring.*

*Got a question for us? Please mention it in the comments section of this "*Java Interview Questions*" and we will get back to you as soon as possible.*

100+ Core JAVA Interview Questions & Answers

Q1. What is the difference between an Inner Class and a Sub-Class?

Ans: An Inner class is a class which is nested within another class. An Inner class has access rights for the class which is nesting it and it can access all variables and methods defined in the outer class.

A sub-class is a class which inherits from another class called super class. Sub-class can access all public and protected methods and fields of its super class.

Q2. What are the various access specifiers for Java classes?

Ans: In Java, access specifiers are the keywords used before a class name which defines the access scope. The types of access specifiers for classes are:

1. Public : Class,Method,Field is accessible from anywhere.

2. Protected:Method,Field can be accessed from the same class to which they belong or from the sub-classes,and from the class of same package,but not from outside.

3. Default: Method,Field,class can be accessed only from the same package and not from outside of it's native package.

4. Private: Method,Field can be accessed from the same class to which they belong.

Q3. What's the purpose of Static methods and static variables?

Ans: When there is a requirement to share a method or a variable between multiple objects of a class instead of creating separate copies for each object, we use static keyword to make a method or variable shared for all objects.

Q4. What is data encapsulation and what's its significance?

Ans: Encapsulation is a concept in Object Oriented Programming for combining properties and methods in a single unit.

Encapsulation helps programmers to follow a modular approach for software development as each object has its own set of methods and variables and serves its functions independent of other objects. Encapsulation also serves data hiding purpose.

Q5. What is a singleton class? Give a practical example of its usage.

A singleton class in java can have only one instance and hence all its methods and variables belong to just one instance. Singleton class concept is useful for the situations when there is a need to limit the number of objects for a class.

The best example of singleton usage scenario is when there is a limit of having only one connection to a database due to some driver limitations or because of any licensing issues.

Q6. What are Loops in Java? What are three types of loops?

Ans: Looping is used in programming to execute a statement or a block of statement repeatedly. There are three types of loops in Java:

1) For Loops

For loops are used in java to execute statements repeatedly for a given number of times. For loops are used when number of times to execute the statements is known to programmer.

2) While Loops

While loop is used when certain statements need to be executed repeatedly until a condition is fulfilled. In while loops, condition is checked first before execution of statements.

3) Do While Loops

Do While Loop is same as While loop with only difference that condition is checked after execution of block of statements. Hence in case of do while loop, statements are executed at least once.

Q7: What is an infinite Loop? How infinite loop is declared?

Ans: An infinite loop runs without any condition and runs infinitely. An infinite loop can be broken by defining any breaking logic in the body of the statement blocks.

Infinite loop is declared as follows:

```
for (;;)
{
   // Statements to execute

   // Add any loop breaking logic
}
```

Q8. What is the difference between continue and break statement?

Ans: break and continue are two important keywords used in Loops. When a break keyword is used in a loop, loop is broken instantly while when continue keyword is used, current iteration is broken and loop continues with next iteration.

In below example, Loop is broken when counter reaches 4.

```
for (counter = 0; counter & lt; 10; counter++)
   system.out.println(counter);

if (counter == 4) {

   break;
}

}
```

In the below example when counter reaches 4, loop jumps to next iteration and any statements after the continue keyword are skipped for current iteration.

```
for (counter = 0; counter < 10; counter++)
   system.out.println(counter);

if (counter == 4) {

   continue;
}
system.out.println("This will not get printed when counter is 4");
}
```

Q9. What is the difference between double and float variables in Java?

Ans: In java, float takes 4 bytes in memory while Double takes 8 bytes in memory. Float is single precision floating point decimal number while Double is double precision decimal number.

Q10. What is Final Keyword in Java? Give an example.

Ans: In java, a constant is declared using the keyword Final. Value can be assigned only once and after assignment, value of a constant can't be changed.

In below example, a constant with the name const_val is declared and assigned avalue:

Private Final int const_val=100

When a method is declared as final,it can NOT be overridden by the subclasses.This method are faster than any other method,because they are resolved at complied time.

When a class is declares as final,it cannot be subclassed. Example String,Integer and other wrapper classes.

Q11. What is ternary operator? Give an example.

Ans: Ternary operator , also called conditional operator is used to decide which value to assign to a variable based on a Boolean value evaluation. It's denoted as ?

In the below example, if rank is 1, status is assigned a value of "Done" else "Pending".

```
public class conditionTest {
   public static void main(String args[]) {
      String status;
      int rank = 3;
      status = (rank == 1) ? "Done" : "Pending";
      System.out.println(status);
   }
}
```

Q12: How can you generate random numbers in Java?

Ans:

- Using Math.random() you can generate random numbers in the range greater than or equal to 0.1 and less than 1.0
- Using Random class in package java.util

Q13. What is default switch case? Give example.

Ans: In a switch statement, default case is executed when no other switch condition matches. Default case is an optional case .It can be declared only once all other switch cases have been coded.

In the below example, when score is not 1 or 2, default case is used.

```
public class switchExample {
   int score = 4;
   public static void main(String args[]) {
      switch (score) {
         case 1:
            system.out.println("Score is 1");
            break;
```

```
        case 2:
          system.out.println("Score is 2");
          break;
        default:
          system.out.println("Default Case");
    }
  }
}
```

Q14. What's the base class in Java from which all classes are derived?

Ans: java.lang.object

Q15. Can main() method in Java can return any data?

Ans: In java, main() method can't return any data and hence, it's always declared with a void return type.

Q16. What are Java Packages? What's the significance of packages?

Ans: In Java, package is a collection of classes and interfaces which are bundled together as they are related to each other. Use of packages helps developers to modularize the code and group the code for proper re-use. Once code has been packaged in Packages, it can be imported in other classes and used.

Q17. Can we declare a class as Abstract without having any abstract method?

Ans: Yes we can create an abstract class by using abstract keyword before class name even if it doesn't have any abstract method. However, if a class has even one abstract method, it must be declared as abstract otherwise it will give an error.

Q18. What's the difference between an Abstract Class and Interface in Java?

Ans: The primary difference between an abstract class and interface is that an interface can only possess declaration of public static methods with no concrete implementation while an abstract class can have members with any access specifiers (public, private etc) with or without concrete implementation.

Another key difference in the use of abstract classes and interfaces is that a class which implements an interface must implement all the methods of the interface while a class which inherits from an abstract class doesn't require implementation of all the methods of its super class.

A class can implement multiple interfaces but it can extend only one abstract class.

Q19. What are the performance implications of Interfaces over abstract classes?

Ans: Interfaces are slower in performance as compared to abstract classes as extra indirections are required for interfaces. Another key factor for developers to take into consideration is that any class can extend only one abstract class while a class can implement many interfaces.

Use of interfaces also puts an extra burden on the developers as any time an interface is implemented in a class; developer is forced to implement each and every method of interface.

Q20. Does Importing a package imports its sub-packages as well in Java?

Ans: In java, when a package is imported, its sub-packages aren't imported and developer needs to import them separately if required.

For example, if a developer imports a package university.*, all classes in the package named university are loaded but no classes from the sub-package are loaded. To load the classes from its sub-package (say department), developer has to import it explicitly as follows:

Import university.department.*

Q21. Can we declare the main method of our class as private?

Ans: In java, main method must be public static in order to run any application correctly. If main method is declared as private, developer won't get any compilation error however, it will not get executed and will give a runtime error.

Q22. How can we pass argument to a function by reference instead of pass by value?

Ans: In java, we can pass argument to a function only by value and not by reference.

Q23. How an object is serialized in java?

Ans: In java, to convert an object into byte stream by serialization, an interface with the name Serializable is implemented by the class. All objects of a class implementing serializable interface get serialized and their state is saved in byte stream.

Q24. When we should use serialization?

Ans: Serialization is used when data needs to be transmitted over the network. Using serialization, object's state is saved and converted into byte stream .The byte stream is transferred over the network and the object is re-created at destination.

Q25. Is it compulsory for a Try Block to be followed by a Catch Block in Java for Exception handling?

Ans: Try block needs to be followed by either Catch block or Finally block or both. Any exception thrown from try block needs to be either caught in the catch block or else any specific tasks to be performed before code abortion are put in the Finally block.

Q26. Is there any way to skip Finally block of exception even if some exception occurs in the exception block?

Ans: If an exception is raised in Try block, control passes to catch block if it exists otherwise to finally block. Finally block is always executed when an exception occurs and the only way to avoid execution of any statements in Finally block is by aborting the code forcibly by writing following line of code at the end of try block:

```
System.exit(0);
```

Q27. When the constructor of a class is invoked?

Ans: The constructor of a class is invoked every time an object is created with new keyword.

For example, in the following class two objects are created using new keyword and hence, constructor is invoked two times.

```
public class const_example {

  const_example() {

    system.out.println("Inside constructor");
  }
  public static void main(String args[]) {

    const_example c1 = new const_example();

    const_example c2 = new const_example();
  }
}
```

Q28. Can a class have multiple constructors?

Ans: Yes, a class can have multiple constructors with different parameters. Which constructor gets used for object creation depends on the arguments passed while creating the objects.

Q29. Can we override static methods of a class?

Ans: We cannot override static methods. Static methods belong to a class and not to individual objects and are resolved at the time of compilation (not at runtime).Even if we try

to override static method,we will not get an complitaion error,nor the impact of overriding when running the code.

Q30. In the below example, what will be the output?

```
public class superclass {

   public void displayResult() {

      system.out.println("Printing from superclass");

   }

}

public class subclass extends superclass {

   public void displayResult() {

      system.out.println("Displaying from subClass");

      super.displayResult();

   }

   public static void main(String args[]) {

      subclass obj = new subclass();

      obj.displayResult();

   }

}
```

Ans: Output will be:

Displaying from subclass

Displaying from superclass

Q31. Is String a data type in java?

Ans: String is not a primitive data type in java. When a string is created in java, it's actually an object of Java.Lang.String class that gets created. After creation of this string object, all built-in methods of String class can be used on the string object.

Q32. In the below example, how many String Objects are created?

```
String s1="I am Java Expert";

String s2="I am C Expert";

String s3="I am Java Expert";
```

Ans: In the above example, two objects of Java.Lang.String class are created. s1 and s3 are references to same object.

Q33. Why Strings in Java are called as Immutable?

Ans: In java, string objects are called immutable as once value has been assigned to a string, it can't be changed and if changed, a new object is created.

In below example, reference str refers to a string object having value "Value one".

```
String str="Value One";
```

When a new value is assigned to it, a new String object gets created and the reference is moved to the new object.

```
str="New Value";
```

Q34. What's the difference between an array and Vector?

Ans: An array groups data of same primitive type and is static in nature while vectors are dynamic in nature and can hold data of different data types.

Q35. What is multi-threading?

Ans: Multi threading is a programming concept to run multiple tasks in a concurrent manner within a single program. Threads share same process stack and running in parallel. It helps in performance improvement of any program.

Q36. Why Runnable Interface is used in Java?

Ans: Runnable interface is used in java for implementing multi threaded applications. Java.Lang.Runnable interface is implemented by a class to support multi threading.

Q37. What are the two ways of implementing multi-threading in Java?

Ans: Multi threaded applications can be developed in Java by using any of the following two methodologies:

1. By using Java.Lang.Runnable Interface. Classes implement this interface to enable multi threading. There is a Run() method in this interface which is implemented.

2. By writing a class that extend Java.Lang.Thread class.

Q38. When a lot of changes are required in data, which one should be a preference to be used? String or StringBuffer?

Ans: Since StringBuffers are dynamic in nature and we can change the values of StringBuffer objects unlike String which is immutable, it's always a good choice to use StringBuffer when data is being changed too much. If we use String in such a case, for every data change a new String object will be created which will be an extra overhead.

Q39. What's the purpose of using Break in each case of Switch Statement?

Ans: Break is used after each case (except the last one) in a switch so that code breaks after the valid case and doesn't flow in the proceeding cases too.

If break isn't used after each case, all cases after the valid case also get executed resulting in wrong results.

Q40. How garbage collection is done in Java?

Ans: In java, when an object is not referenced any more, garbage collection takes place and the object is destroyed automatically. For automatic garbage collection java calls either System.gc() method or Runtime.gc() method.

Q41. How we can execute any code even before main method?

Ans: If we want to execute any statements before even creation of objects at load time of class, we can use a static block of code in the class. Any statements inside this static block of code will get executed once at the time of loading the class even before creation of objects in the main method.

Q42. Can a class be a super class and a sub-class at the same time? Give example.

Ans: If there is a hierarchy of inheritance used, a class can be a super class for another class and a sub-class for another one at the same time.

In the example below, continent class is sub-class of world class and it's super class of country class.

```
public class world {
```

```
.....

}
public class continenet extends world {

.....

}
public class country extends continent {

.....

}
```

Q43. How objects of a class are created if no constructor is defined in the class?

Ans: Even if no explicit constructor is defined in a java class, objects get created successfully as a default constructor is implicitly used for object creation. This constructor has no parameters.

Q44. In multi-threading how can we ensure that a resource isn't used by multiple threads simultaneously?

Ans: In multi-threading, access to the resources which are shared among multiple threads can be controlled by using the concept of synchronization. Using synchronized keyword, we can ensure that only one thread can use shared resource at a time and others can get control of the resource only once it has become free from the other one using it.

Q45. Can we call the constructor of a class more than once for an object?

Ans: Constructor is called automatically when we create an object using new keyword. It's called only once for an object at the time of object creation and hence, we can't invoke the constructor again for an object after its creation.

Q46. There are two classes named classA and classB. Both classes are in the same package. Can a private member of classA can be accessed by an object of classB?

Ans: Private members of a class aren't accessible outside the scope of that class and any other class even in the same package can't access them.

Q47. Can we have two methods in a class with the same name?

Ans: We can define two methods in a class with the same name but with different number/type of parameters. Which method is to get invoked will depend upon the parameters passed.

For example in the class below we have two print methods with same name but different parameters. Depending upon the parameters, appropriate one will be called:

```
public class methodExample {

  public void print() {

    system.out.println("Print method without parameters.");

  }

  public void print(String name) {

    system.out.println("Print method with parameter");

  }

  public static void main(String args[]) {

    methodExample obj1 = new methodExample();

    obj1.print();

    obj1.print("xx");

  }

}
```

Q48. How can we make copy of a java object?

Ans: We can use the concept of cloning to create copy of an object. Using clone, we create copies with the actual state of an object.

Clone() is a method of Cloneable interface and hence, Cloneable interface needs to be implemented for making object copies.

Q49. What's the benefit of using inheritance?

Ans: Key benefit of using inheritance is reusability of code as inheritance enables sub-classes to reuse the code of its super class. Polymorphism (Extensibility) is another great benefit which allow new functionality to be introduced without effecting existing derived classes.

Q50. What's the default access specifier for variables and methods of a class?

Ans: Default access specifier for variables and method is package protected i.e variables and class is available to any other class but in the same package,not outside the package.

Q51. Give an example of use of Pointers in Java class.

Ans: There are no pointers in Java. So we can't use concept of pointers in Java.

Q52. How can we restrict inheritance for a class so that no class can be inherited from it?

Ans: If we want a class not to be extended further by any class, we can use the keyword **Final** with the class name.

In the following example, Stone class is Final and can't be extend

```
public Final Class Stone {
        // Class methods and Variables
}
```

Q53. What's the access scope of Protected Access specifier?

Ans: When a method or a variable is declared with Protected access specifier, it becomes accessible in the same class,any other class of the same package as well as a sub-class.

Modifier	Class	Package	Subclass	Worl
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no modifier	Y	Y	N	N
private	Y	N	N	N

Q54. What's difference between Stack and Queue?

Ans: Stack and Queue both are used as placeholder for a collection of data. The primary difference between a stack and a queue is that stack is based on Last in First out (LIFO) principle while a queue is based on FIFO (First In First Out) principle.

Q55. In java, how we can disallow serialization of variables?

Ans: If we want certain variables of a class not to be serialized, we can use the keyword **transient** while declaring them. For example, the variable trans_var below is a transient variable and can't be serialized:

```
public class transientExample {
        private transient trans_var;
  // rest of the code
}
```

Q56. How can we use primitive data types as objects?

Ans: Primitive data types like int can be handled as objects by the use of their respective wrapper classes. For example, Integer is a wrapper class for primitive data type int. We can apply different methods to a wrapper class, just like any other object.

Q57. Which types of exceptions are caught at compile time?

Ans: Checked exceptions can be caught at the time of program compilation. Checked exceptions must be handled by using try catch block in the code in order to successfully compile the code.

Q58. Describe different states of a thread.

Ans: A thread in Java can be in either of the following states:

- Ready: When a thread is created, it's in Ready state.
- Running: A thread currently being executed is in running state.
- Waiting: A thread waiting for another thread to free certain resources is in waiting state.
- Dead: A thread which has gone dead after execution is in dead state.

Q59. Can we use a default constructor of a class even if an explicit constructor is defined?

Ans: Java provides a default no argument constructor if no explicit constructor is defined in a Java class. But if an explicit constructor has been defined, default constructor can't be invoked and developer can use only those constructors which are defined in the class.

Q60. Can we override a method by using same method name and arguments but different return types?

Ans: The basic condition of method overriding is that method name, arguments as well as return type must be exactly same as is that of the method being overridden. Hence using a different return type doesn't override a method.

Q61.What will be the output of following piece of code?

```
public class operatorExample {

  public static void main(String args[]) {
```

```
    int x = 4;

    system.out.println(x++);
            }
}
```

Ans: In this case postfix ++ operator is used which first returns the value and then increments. Hence it's output will be 4.

Q61. A person says that he compiled a java class successfully without even having a main method in it? Is it possible?

Ans: main method is an entry point of Java class and is required for execution of the program however; a class gets compiled successfully even if it doesn't have a main method. It can't be run though.

Q62. Can we call a non-static method from inside a static method?

Ans: Non-Static methods are owned by objects of a class and have object level scope and in order to call the non-Static methods from a static block (like from a static main method), an object of the class needs to be created first. Then using object reference, these methods can be invoked.

Q63. What are the two environment variables that must be set in order to run any Java programs?

Ans: Java programs can be executed in a machine only once following two environment variables have been properly set:

1. PATH variable
2. CLASSPATH variable

Q64. Can variables be used in Java without initialization?

Ans: In Java, if a variable is used in a code without prior initialization by a valid value, program doesn't compile and gives an error as no default value is assigned to variables in Java.

Q65. Can a class in Java be inherited from more than one class?

Ans: In Java, a class can be derived from only one class and not from multiple classes. Multiple inheritances is not supported by Java.

Q66. Can a constructor have different name than a Class name in Java?

Ans: Constructor in Java must have same name as the class name and if the name is different, it doesn't act as a constructor and compiler thinks of it as a normal method.

Q67. What will be the output of Round(3.7) and Ceil(3.7)?

Ans: Round(3.7) returns 4 and Ceil(3.7) returns 4.

Q68: Can we use goto in Java to go to a particular line?

Ans: In Java, there is not goto keyword and java doesn't support this feature of going to a particular labeled line.

Q69. Can a dead thread be started again?

Ans: In java, a thread which is in dead state can't be started again. There is no way to restart a dead thread.

Q70. Is the following class declaration correct?

Ans:

```
public abstract final class testClass {
        // Class methods and variables
}
```

Ans: The above class declaration is incorrect as an abstract class can't be declared as Final.

Q71. Is JDK required on each machine to run a Java program?

Ans: JDK is development Kit of Java and is required for development only and to run a Java program on a machine, JDK isn't required. Only JRE is required.

Q72. What's the difference between comparison done by equals method and == operator?

Ans: In Java, equals() method is used to compare the contents of two string objects and returns true if the two have same value while == operator compares the references of two string objects.

In the following example, equals() returns true as the two string objects have same values. However == operator returns false as both string objects are referencing to different objects:

```
public class equalsTest {

  public static void main(String args[]) {
```

```
    String str1 = new String("Hello World");

    String str2 = new String("Hello World");

    if (str1.equals(str2))

    { // this condition is true

       System.out.println("str1 and str2 are equal in terms of values");

    }

    if (str1 == str2) {

       //This condition is true

       System.out.println("Both strings are referencing same object");

    } else

    {

       // This condition is NOT true

       System.out.println("Both strings are referencing different objects");

    }

  }

}
```

Q73. Is it possible to define a method in Java class but provide it's implementation in the code of another language like C?

Ans: Yes, we can do this by use of native methods. In case of native method based development, we define public static methods in our Java class without its implementation and then implementation is done in another language like C separately.

Q74. How are destructors defined in Java?

Ans: In Java, there are no destructors defined in the class as there is no need to do so. Java has its own garbage collection mechanism which does the job automatically by destroying the objects when no longer referenced.

Q75. Can a variable be local and static at the same time?

Ans: No a variable can't be static as well as local at the same time. Defining a local variable as static gives compilation error.

Q76. Can we have static methods in an Interface?

Ans: Static methods can't be overridden in any class while any methods in an interface are by default abstract and are supposed to be implemented in the classes being implementing the interface. So it makes no sense to have static methods in an interface in Java.

Q77. In a class implementing an interface, can we change the value of any variable defined in the interface?

Ans: No, we can't change the value of any variable of an interface in the implementing class as all variables defined in the interface are by default public, static and Final and final variables are like constants which can't be changed later.

Q78. Is it correct to say that due to garbage collection feature in Java, a java program never goes out of memory?

Ans: Even though automatic garbage collection is provided by Java, it doesn't ensure that a Java program will not go out of memory as there is a possibility that creation of Java objects is being done at a faster pace compared to garbage collection resulting in filling of all the available memory resources.

So, garbage collection helps in reducing the chances of a program going out of memory but it doesn't ensure that.

Q79. Can we have any other return type than void for main method?

Ans: No, Java class main method can have only void return type for the program to get successfully executed.

Nonetheless , if you absolutely must return a value to at the completion of main method , you can use System.exit(int status)

Q80. I want to re-reach and use an object once it has been garbage collected. How it's possible?

Ans: Once an object has been destroyed by garbage collector, it no longer exists on the heap and it can't be accessed again. There is no way to reference it again.

Q81. In Java thread programming, which method is a must implementation for all threads?

Ans: Run() is a method of Runnable interface that must be implemented by all threads.

Q82. I want to control database connections in my program and want that only one thread should be able to make database connection at a time. How can I implement this logic?

Ans: This can be implemented by use of the concept of synchronization. Database related code can be placed in a method which hs **synchronized** keyword so that only one thread can access it at a time.

Q83. How can an exception be thrown manually by a programmer?

Ans: In order to throw an exception in a block of code manually, **throw** keyword is used. Then this exception is caught and handled in the catch block.

```
public void topMethod() {
   try {
      excMethod();
   } catch (ManualException e) {}
}

public void excMethod {
   String name = null;
   if (name == null) {
      throw (new ManualException("Exception thrown manually ");
      }
   }
```

Q84. I want my class to be developed in such a way that no other class (even derived class) can create its objects. How can I do so?

Ans: If we declare the constructor of a class as private, it will not be accessible by any other class and hence, no other class will be able to instantiate it and formation of its object will be limited to itself only.

Q85. How objects are stored in Java?

Ans: In java, each object when created gets a memory space from a heap. When an object is destroyed by a garbage collector, the space allocated to it from the heap is re-allocated to the heap and becomes available for any new objects.

Q86. How can we find the actual size of an object on the heap?

Ans: In java, there is no way to find out the exact size of an object on the heap.

Q87. Which of the following classes will have more memory allocated?

Class A: Three methods, four variables, no object

Class B: Five methods, three variables, no object

Ans: Memory isn't allocated before creation of objects. Since for both classes, there are no objects created so no memory is allocated on heap for any class.

Q88. What happens if an exception is not handled in a program?

Ans: If an exception is not handled in a program using try catch blocks, program gets aborted and no statement executes after the statement which caused exception throwing.

Q89. I have multiple constructors defined in a class. Is it possible to call a constructor from another constructor's body?

Ans: If a class has multiple constructors, it's possible to call one constructor from the body of another one using **this()**.

Q90. What's meant by anonymous class?

Ans: An anonymous class is a class defined without any name in a single line of code using new keyword.

For example, in below code we have defined an anonymous class in one line of code:

```
public java.util.Enumeration testMethod()

{

  return new java.util.Enumeration()

  {

    @Override

    public boolean hasMoreElements()

    {

      // TODO Auto-generated method stub

      return false;

    }

    @Override
```

```
    public Object nextElement()
    {
      // TODO Auto-generated method stub
      return null;
    }
}
```

Q91. Is there a way to increase the size of an array after its declaration?

Ans: Arrays are static and once we have specified its size, we can't change it. If we want to use such collections where we may require a change of size (no of items), we should prefer vector over array.

Q92. If an application has multiple classes in it, is it okay to have a main method in more than one class?

Ans: If there is main method in more than one classes in a java application, it won't cause any issue as entry point for any application will be a specific class and code will start from the main method of that particular class only.

Q93. I want to persist data of objects for later use. What's the best approach to do so?

Ans: The best way to persist data for future use is to use the concept of serialization.

Q94. What is a Local class in Java?

Ans: In Java, if we define a new class inside a particular block, it's called a local class. Such a class has local scope and isn't usable outside the block where its defined.

Q95. String and StringBuffer both represent String objects. Can we compare String and StringBuffer in Java?

Ans: Although String and StringBuffer both represent String objects, we can't compare them with each other and if we try to compare them, we get an error.

Q96. Which API is provided by Java for operations on set of objects?

Ans: Java provides a Collection API which provides many useful methods which can be applied on a set of objects. Some of the important classes provided by Collection API include ArrayList, HashMap, TreeSet and TreeMap.

Q97. Can we cast any other type to Boolean Type with type casting?

Ans: No, we can neither cast any other primitive type to Boolean data type nor can cast Boolean data type to any other primitive data type.

Q98. Can we use different return types for methods when overridden?

Ans: The basic requirement of method overriding in Java is that the overridden method should have same name, and parameters.But a method can be overridden with a different return type as long as the new return type extends the original.

For example , method is returning a reference type.

```
Class B extends A {

  A method(int x) {

    //original method

  }

  B method(int x) {

    //overridden method

  }

}
```

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions

Q99. What's the base class of all exception classes?

Ans: In Java, **Java.lang.Throwable** is the super class of all exception classes and all exception classes are derived from this base class.

Q100. What's the order of call of constructors in inheritiance?

Ans: In case of inheritance, when a new object of a derived class is created, first the constructor of the super class is invoked and then the constructor of the derived class is invoked.

AJ's Guide To

Algorithms and Data Structures

90+ Algo and DS Chapters
15+ HR questions