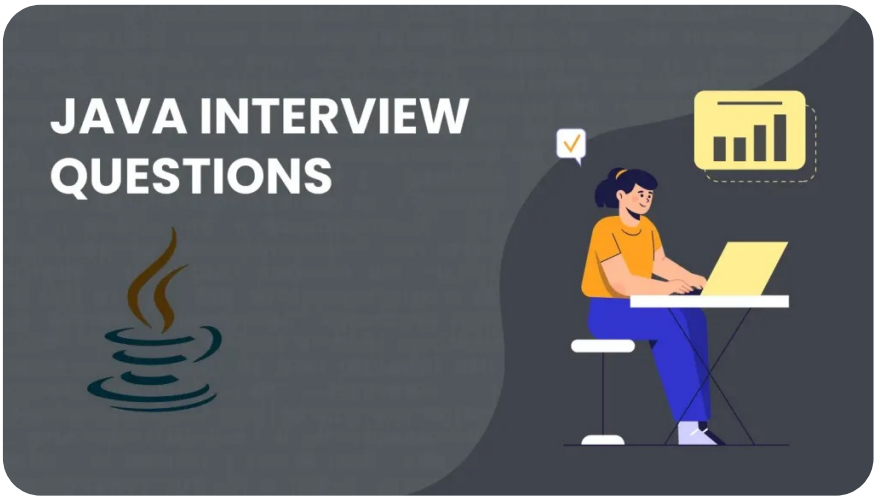

Java Interview Questions and Answers

Published On: April 15, 2023

It's common knowledge that Java is a highly marketable skill that can help you land a high-paying programming job. As a matter of fact, the language is one of the most frequently used in the world due to its general-purpose, class-based, object-oriented design.

The language has so many useful features that it is sought after by both experienced programmers and beginners. Learn more about Java and improve your chances of getting hired by studying these frequently asked **Java interview questions and answers.**

Developers of both software and Android applications will find this collection of **Java interview questions**

Related Courses

- Core Java Course in OMR
- Core Java Online Course
- Advanced Java Course in OMR
- Java Full Stack Course in Chennai

Related Posts

Manual Testing Interview Questions and Answers

Published On: February 21,

These **Java Interview Questions** were crafted to help you prepare for the types of questions that would be asked of you during a job interview centered around the Java programming language. Professional interviewers rarely have a set list of questions they intend to ask you. Rather, they will begin with a broad overview of the topic at hand and then dive further into specifics based on your responses.

Basic Java Interview Questions for Freshers

1. Describe Java.

Java was created by James Gosling and is now maintained and developed by Oracle Corporation; it is a high-level, object-oriented, general-purpose programming language. It is widely recognized as one of the best programming languages available today.

2. Describe the Java Virtual Machine.

The Java Virtual Machine (JVM) is a piece of software that is responsible for interpreting Java byte code and producing the final result. Java's bytecode and the JVM make it possible to run Java applications on a wide variety of platforms.

3. What are Java's features?

Among Java's numerous features are those listed below:

- High Performance: Java can achieve excellent performance with the help of a JIT (Just-In-Time) compiler. The JVM is responsible for running the machine-language code generated by the JIT compiler.
- Multi-threading: In computer terminology, a "thread" refers to a flow of execution. The Java Virtual Machine (JVM) generates a thread known as the main thread. By either extending the thread class or implementing the Runnable interface, multiple

Questions and Answers
Before a software application is released into production, manual...

Amazon Fresher Salary

Published On: February 21, 2024

Amazon Fresher Salary The exceptional chance to shape your professional path by having total control...

Components of Selenium

Published On: February 21, 2024

Components of Selenium Introduction In the field of software testing and automation, selenium is the...

threads can be created in Java.

- OOPs Concepts: Java adheres to a variety of OOPs concepts, including abstraction, encapsulation, inheritance, object-oriented programming, and polymorphism.
- Java Virtual Machine: Java's usage of the Java Virtual Machine (JVM) makes it possible for a single Java program to run across various platforms with no additional tweaks. Learn Java from basic to advanced concepts by enrolling in the best Java training in Chennai.

4. Explain how Java enables high performance.

With Just-in-Time compilation, only the necessary code is compiled and run at runtime. In most cases, this means running the machine code that was generated from the bytecode. It paves the way for optimal efficiency. In Java, the JIT compiler is turned on automatically whenever a method is invoked.

The Java method's bytecode is then compiled into machine code.

The JVM then makes a direct call to the built code without further interpretation.

5. Explain the JIT compiler.

The JIT compiler is a post-execution compiler that converts source code into an optimized form using the CPU's native instruction set. When compared to a static compiler, JIT's optimizations, such as inlining frequently used routines, are superior because of its access to dynamic runtime information.

6. What exactly are Java IDEs?

A Java IDE is a piece of software that streamlines the development and debugging processes for Java applications. Code completion and syntax highlighting

are only two of its many helpful features; it's essentially a collection of different programming tools available through a single interface.

Of the many available Java IDEs, Codenvy, Eclipse, and NetBeans are among the most widely used.

7. The programming language Java is platform-independent. Why?

Since the Java compiler transforms the language into byte code, Java can run on any computer system. Byte code is platform-independent, thus it can be executed on a variety of different computer systems. The primary need is the Java Runtime Environment (JRE), which is a collection of programs used for building Java applications.

8. Describe typecasting.

The act of transferring the value of a variable that belongs to one data type to another variable that belongs to a different data type is typecasting. The boolean data type does not allow for this to happen. Both implicit and explicit kinds exist.

9.What various forms of typecasting are there?

The various forms of typecasting are as follows:

- Implicit: Keeping track of information in the larger data type from the smaller data type. The compiler handles it mechanically for you.
- Explicit: The process of storing the value of a bigger data type in a smaller data type is referred to as explicit storage. This causes a decrease in data quality:
- Truncation: When converting from one data type to another, any excess information is discarded, a process is known as truncation. This snippet of code should help clarify:

float f = 3.14f;

int i = (int) f;

- After the command int i = (int) f is executed, the variable i will only contain 3, without the decimal point.
- Out of Range: Typecasting does not permit assigning a value that is greater than its range; if this were to occur, the data would be lost in these kinds of situations. This snippet of code should help clarify:
- long l = 123456789;
- Because bytes and longs have different ranges, data may be lost when using the expression bytes b = (bytes) l.

Learn Java from the best Java Training in Chennai at SLA and explore these **Java Interview Questions and Answers** to perform explicitly in your interview.

10. Describe Java's access modifiers.

Java has a set of preset keywords called "access modifiers" that can be used to limit who can use a certain class's declared methods, constructors, and data members. Four access modifiers are supported in Java:

- Default
- Private
- Protected
- Public

11. Describe the concept of object-oriented programming.

In OOPs, objects, as opposed to functions, take center stage. It's neither a language or a tool; it's a paradigm that was created to fix the problems of procedural programming.

Java, Python, and Ruby are just a few examples of the

many languages that adhere to OOPs principles. Angular is only one example of a framework that adheres to OOPs principles.

12. Describe the ideas behind OOPs.

The many OOPS Ideas are as follows:

Abstraction

Abstraction is the process of representing key characteristics without having to provide specifics regarding the context. This method is employed whenever a need arises to develop a brand-new data type for a particular program.

Aggregation

Aggregation refers to each individual object continuing to have its own unique lifecycle, despite the fact that ownership is maintained. There is no other item to which a child can belong beside the parent.

Association

Association refers to the link between two objects in which both continue to exist independently of one another. No one has any claim to it.

Class

The term “class” is used to refer to a collection of things that share certain characteristics.

Composition

Composition is a subtype of aggregation that is also known as the death relationship. There is no lifecycle for child objects. When their parent object is removed, they are also removed.

Encapsulation

The term “encapsulation” is used to describe the process of combining data and code into a single unit. In this way, a class’s variables are available exclusively to its parent and are hidden from any child classes.

Inheritance

The process by which one object takes on the characteristics of another item is referred to as inheritance. Inheritance occurs when one object inherits the characteristics of another object. It creates a hierarchical family tree of classes with parents and children. This provides a solid and organic means of organizing and arranging software.

Object

Object is a symbol used to denote an individual instance of a class. Multiple instances of a class are possible. An object stores both its contents and the procedure that will be used to process that data.

Polymorphism

Polymorphism is the capability of a method, object, or variable to take on a number of different manifestations.

Refresh your knowledge about important concepts of Java by reading these **Java Interview Questions for Freshers**.

13. Define Object

An object is a specific instance of a Java class. Behavior and state are two crucial aspects of any Java object. When the JVM encounters a new keyword, it immediately creates a new object.

14. Describe classes in Java.

A collection of objects that share the same kind of data

makes up a class. Classes are defined by the programmer and function similarly to the language's native data types.

This is the class syntax:

Syntax of a class:

classSample{

membervariables

methods()

}

Example of Class:

publicclassShape

{

String Shape name;

voidarea()

{

}

voidvolume ()

{

}

voidnum_sides()

{

}

}

15. Explain “static” methods and variables.

Each class has a variable declaration part and a method declaration section. The terms “instance variable” and “instance method” refer to these concepts. They get this name because whenever a class is instantiated, a new instance of every member of that class is also created.

Declaring a variable or method as static allows it to be shared by all objects and accessible without reference to a specific object. Classes and methods can also access and make use of static members.

Proving yourself in an interview is important to grab your job. To do so, brush up your understanding of Java by going through these **Java Interview Questions and Answers.**

16. Exactly what is meant by “Constructor”?

A function Object() { [native code] } is a special method in a class that shares its name with the class itself. A class’s corresponding function Object() { [native code] } is called whenever a new object is created. The user can manually build a constructor, but one is automatically generated whenever a class is made. It’s the “default function Object() { [native code] }” for a reason. Overloading constructors is possible.

It is important to make a second, parameter-free constructor if the first one requires one.

Intermediate Java Interview Questions And Answers For Freshers and Experienced

17. What’s the difference between local variables and instance variables?

Local variables are variables that can only be accessed within the method or code block in which they were defined. This makes them inaccessible to other parts of the program. However, instance variables are shared

throughout all of a class's methods.

Instance variables, in contrast to local variables, are declared outside of methods but within classes. Instance variables can be null, 0, 0.0, or false even when they are not allocated a value. However, this is not the case with local variables that require a value; in this case, an error will occur if no value is assigned. When a method is invoked, local variables are generated and removed automatically. The new keyword is used to create instance variables.

18. Define Method Overriding

With Java's method overriding, a subclass can provide its own version of a method that was previously given by its superclass. Overriding a method occurs when the subclass method and the superclass method share the following properties:

- The same name
- The same argument
- The same return type

19. Overloading: What Is It?

When multiple distinct procedures or operators share the same representation, this is known as overloading. For instance, the + operator is used to add integers but to join strings together. Similarly, the Add function has multiple uses because it is overloaded.

- Adding two whole numbers
- Joining two strings together

When two methods share the same name but take distinct parameters, we say that they are "overloaded." The return types of the overloaded functions are ambiguous.

20. When using Java, how important is it to use the final keyword? How does it affect a given

variable, procedure, and class?

Only a class, method, or variable can be affected by the final keyword in Java; it is not an access modifier. Depending on the setting, it may have one of several different functions.

- When it comes to classes, if a class is declared to be final, it cannot have any subclasses; in other words, no other classes can extend the final class. This prevents the final class from being extended.
- With a method, the final keyword prevents a subclass from overriding any method that comes with it.
- Using a variable, if you use the final keyword after declaring a variable, that variable's value cannot be changed while the program is running. So it acts like a constant,
- Compare and contrast Array with ArrayList.
- When declaring an array, you are required to state the size of the array. Declaring an array list, on the other hand, does not necessarily require a size because an array list's size might change dynamically. An object's index must be specified when it is inserted into an array. However, an array list has no such limitation. Parameterization can be used to array lists but not to arrays themselves.

Get benefited from these **Java Interview Questions for** Freshers and Experience to deliver your best in your job interview.

21. When comparing String, Stringbuilder, and Stringbuffer, what are the key distinctions?

A continuous string pool is used to hold string variables. Due to the new string reference, the previous value can no longer be removed. If a string already has the value "Old," appending the value "New" will not cause the "Old" value to be removed from storage. It will remain

dormant, but still present.

Stringbuffers use a stack to hold their data. When a string reference is updated, the previous value is no longer used. When compared to StringBuilder, which is likewise a Stringbuffer but is not synchronized, the performance of the synchronized Stringbuffer is slower. Therefore, Stringbuilder has better performance than Stringbuffer.

22. What does String Pool mean?

The term “string pool” is used to describe the group of strings that make up the heap. The string pool is queried each time a new object is produced to see if it duplicates an existing one. A new object is generated in the string pool and the corresponding reference is returned if the variable already contains a reference to the object.

23. What is your understanding of Interfaces?

An interface in Java is a template where the methods are declared but not implemented. It is an option for implementing multiple inheritances in Java. Important details to keep in mind with Java interfaces include:

- Any class that implements the interface is obligated to deliver an implementation for each and every method that is declared in the interface.
- Each and every interface method is completely public abstract void within itself.
- All interface variables are public static final by default.
- Interfaces are implemented rather than extended by classes.

Learning these **Java Interview Questions for Freshers and Experienced**, sound technically strong, and get your job by acing your interview.

24. Differentiate an Abstract class and an

Interface

Here are some of the key distinctions between an Abstract class and an Interface in Java:

- Components: An interface can only have constants, but an abstract class can have instance variables.
- Constructor and Instantiation: In contrast to an abstract class, an interface does not have a constructor, nor can it be instantiated; however, an abstract class can have a default constructor that is called anytime a concrete subclass is instantiated.
- Method Implementation: Each class that implements the interface must offer an implementation for each of the methods that it contains. However, an abstract class inheriting from another class is not required to implement all of the methods of the inheriting class. In the concrete subclass, only the abstract methods should be implemented.
- Type of Methods: Abstract classes have both abstract and non-abstract methods. However, there is only one abstract method of any kind in an Interface.

25. Give an outline of the Abstract class and an Abstract method.

In Java, a class that cannot be instantiated is considered abstract. Subclasses can then utilize this abstract class as a starting point for their own implementations of the abstract methods and for overriding or utilizing the abstract methods defined in the abstract class.

The abstract keyword must be used after the class declaration to make it abstract. Abstract and non-abstract methods can coexist in the same abstract class. An abstract method is a method in Java that just contains the declaration and not the implementation.

The abstract keyword comes after the name of an abstract procedure. The abstract class must have its abstract methods implemented by every concrete subclass that extends it. Nervous about your Job interview? No worries! Read our **Java Interview Questions and Answers** and win your interview.

26. What is the definition of multiple inheritance? Is multiple inheritance supported by Java? Otherwise, how could it be accomplished?

Multiple inheritance occurs when a subclass or child class takes characteristics from more than one base class. In the event where two parent classes share the same set of methods, Java does not support multiple inheritances.

The compiler then has no way of knowing which method of the child class to execute until runtime, when the situation has become unclear.

27. What are Java packages? Describe some benefits.

Java's packages allow you to organize several classes and/or interfaces into logical collections. The objects are classified based on the roles they play. Packagers are like "boxes" in which classes can be stored.

28. The advantages of Packages are as follows:

- It is possible to reuse classes from various other programs.
- It is possible for two packages to contain the same class.
- Classes can be hidden within packages to prevent others from accessing internal programs and classes.
- They additionally divide design from programming.

29. What is the purpose of calling the yield() method?

Thread is the class that contains the yield() method. It returns the current thread to a state where it can run, and it also makes it possible for other threads to run. In other words, it allows threads of equal priority to execute. Yield() does not free any locks because it is a static function.

30. How and when should you use the Runnable interface instead of the thread class?

Only a single class can be extended in Java. That's why we only ever extend the thread class if we have no other classes to extend. The Runnable interface should be used whenever it is necessary for a class to extend a different class than the thread class.

31. Compare the notify() and notifyAll() methods.

The notify() method is used to deliver a signal to a single thread in the waiting pool to notify it to wake up. The notifyAll() method, on the other hand, is used to deliver a signal to all threads in a waiting pool, so waking them up.

32. How will you differentiate a process from a thread?

The following are some of the most basic distinctions between process and threads:

- By definition, a process is an instance of an executable program, while a thread is an integral component of a process.
- A modification that is performed to the parent process does not influence any of the child processes. Altering the main thread can, however, affect how other threads in the same process operate.
- Unlike processes, which must use inter-process communication in order to speak with their siblings, threads inside the same process are able to engage

in direct communication with one another.
- Processes are only able to exercise control over their child processes because the operating system is the one that controls them. Instead, threads are under the programmer's direction and can influence other threads within the same process.
- As opposed to threads, which are dependent, processes are distinct units.
- Processes use their own dedicated memory whereas threads use a shared pool.

33. How many different kinds of exceptions are there? So, how do you deal with them?

Java supports two distinct kinds of exceptions:

- Checked Exceptions: Checked exceptions are classes that extend the Throwable class but do not include Runtime exceptions or Error. These types of exceptions are referred to as checked exceptions. The compiler performs such checks at compilation time. The throws keyword or a try/catch block is required for this kind of exception. A checked exception is ClassNotFoundException.
- Unchecked Exception: The compiler does not perform any checks on such exceptions before they are allowed to be used in the program. Therefore, the compiler need not worry about unchecked exceptions. There are unchecked exceptions such as Arithmetic Exception and ArrayIndexOutOfBounds Exception.

34. Can you explain the distinction between the throw and throws keywords?

- The throw keyword is used to actually throw an exception, while the throws keyword is used to declare exceptions.
- Throw by itself isn't sufficient to propagate checked exceptions but throws make it possible to do so.
- When you use the throws keyword, you follow it

with a class, while when you use throw, you follow it with an instance. The method itself uses the throw keyword, yet the throws keyword is part of the method signature.
- In addition, while you can specify numerous exceptions, you cannot throw more than one at a time.

35. Describe the many Java keywords used to manage exceptions.

Two of Java's most important terms are related to handling exceptions, followed by final, which is optional.

try:

A try block encloses a section of code that may encounter an unexpected condition or error. The catch block is responsible for handling and catching the exception when it is thrown.

Either a catch() or a final() block, or both, must follow the try block.

catch:

The catch block is executed when an exception is thrown from the try block.

final:

This block will be run no matter what the error. It might come after the try block or the catch block.

36. When do we use collections? Who are their members?

Collections are Java's term for a grouped set of objects. Along with date/time facilities, internationalization, legacy collection classes, etc., the various classes and interfaces for collecting can be found in java.util package. Another way of looking at collections is as a

framework for storing objects and modifying their arrangement. Objects can undergo the following transformations when stored in collections: Deletion, Insertion, Manipulation, Searching, and Sorting.

- The various parts of the collections framework are as follows:
- Classes: Array List, Lists, Linked List, and Vector
- Interfaces: Collection, Map, Sorted Map, List, Queue, Set, and Sorted Set
- Maps: HashMap, LinkedHashMap, HashTable, and TreeMap
- Queues: Priority Queue
- Sets: Hash Set, Tree Set. and Linked Hash Set.

Advanced Java Interview Questions For Experienced Professionals

37. What are the key differences between HashMap and HashTable?

- In Java, key/value pairs can be stored in a HashMap, which is a map-based collection class. The notation for this structure is HashMap<Key, Value> or HashMap<K, V>. The individual lists in a HashTable are referred to as “buckets,” while the table itself is an array.
- A HashTable stores values that are distinct and key-dependent. HashMap does not synchronize its methods, while HashTable does so for its key methods.
- While HashTable has thread safety, HashMap does not.
- HashMap employs an iterator, while HashTable makes use of an enumerator, to iterate through values. In contrast to HashMap, which prohibits all null keys and values, HashTable permits zero such entries.
- HashTable’s performance is subpar. HashMap is noticeably quicker in comparison.

38. What are the different kinds of maps, and how do you identify them?

- A key-value pair can be represented in Java via a Map object. Each key can only correspond to one value, and duplicate keys are not allowed. Map uses the equals() method to check if two keys are the same or different. In Java, you can use one of four distinct varieties of Map:
- HashMap is a suitable option when sorting and ordering are not crucial, as it is an unordered and unsorted map. A HashMap does not care about the order of its data and accepts one null key and numerous null values.
- HashTable doesn't accept null values and provides synchronized methods. As a result of supporting thread safety, performance suffers.
- When compared to HashMap, LinkedHashMap is slower, but it iterates more quickly and preserves insertion order.
- TreeMap is a sorted Map that allows you to build your own sort order using its constructor

39. Define Priority Queues

Priority queues are abstract data types similar to ordinary queues, with the added feature that each item in the queue is assigned a priority.

In a priority queue, the item with the highest priority gets processed before the item with the lowest priority. Elements in a priority queue can be sorted in two ways: by the comparator and by natural precedence. A priority queue's element order reflects its relative importance.

40. Define a Set. Describe the different Java Collections types.

Sets in Java are collections of distinct objects. The equals() method is used to check if two objects are equivalent. Java Collections provide several different

kinds of sets, including

- Hash Set: An unsorted set that combines objects by their hash codes. When the order of the items in the collection doesn't matter.
- The Linked Hash: Set keeps a doubly-linked list of all the items, making it an ordered variant of the hash set. Applied when strict enumeration order is required. The order in which elements are inserted into the Set is the same as the insertion order.
- Tree Set: belongs to Java's two sorted collections and employs the structure of the Read-Black tree and ascertains that its components will be present in increasing order.

Job interviews are no more frightening. SLA's **Java Interview Questions for Experienced** cover from basic to advanced concepts to help you land your dream job.

41. What sorted and ordered collections?

An ordered collection stores its values in a predetermined order that does not depend on the values themselves. Eg: List. In the case of a sorted list, for instance, the elements are arranged according to their values. Example: SortedSet

42. Explain Serialization and deserialization

- When working with Java objects, serialization is used to transform them into a byte stream.
- In deserialization, Java objects are unpacked from a byte stream, the inverse process of serialization.
- To serialize a Java object, the object must be written to an ObjectOutputStream, while to deserialize it, the object must be retrieved from an ObjectInputStream.

43. Discuss the Java statement public static void main(String args[]).

In Java, the main() method, or public static void main(String args[]), is the first line of code that is executed.

- Public: When a class or method is marked as public, it indicates that anyone with the appropriate permissions can access it. The main() method is declared public so that it can be called from any class.
- Static: When you see the keyword “static,” you know that the corresponding variable or method is a class method. To avoid having to create an instance of the class just to use the method main(), we make it static. Since the JVM always calls main() before creating any objects, and only static methods can be properly invoked via the class, the compiler will give an error if main() is not made static.
- Void: The method’s return type is void. The non-returning procedure is characterized by the keyword void.
- main: The JVM will only run code that has this specific method’s signature when it is first started.
- Args[: It] is the parameter supplied to the main method, and it is a string.

44. Define the terms “boxing,” “unboxing,” “autoboxing,” and “auto unboxing”

- The act of enclosing a primitive value within an object is known as boxing.
- “Unboxing” means “obtaining the object’s primitive value.”
- Autoboxing is the practice of giving an integer object a numeric value directly.
- Auto unboxing is the process of automatically transferring the primitive value into the integer object.

publicclassBoxUnbox

```
{

publicstaticvoidmain(String args[])

{

int i = 5;

Integer ii = new Integer(i); /*Boxing*/

Integer jj = i; /*Unboxing*/

int j = jj.intValue(); /*Unboxing*/

int k = jj; /*AutoUnboxing*/

}

}
```

45. Will the program continue to function if you change the public static void to a static public void?

Since the sequence of the specifiers is irrelevant, the program would compile and operate without any problems.

Upskill yourself for a high-paying job by enrolling in Java Training in Chennai at SLA.

46. Programming Java Interview Questions

You will be evaluated not just on how well you understand the basics of Java programming, but also on how well you can write code in Java. Here are some of the most often-asked Java Interview questions, all appropriate for beginners in the field.

47. Could you write the coding for string reversal without involving an inbuilt function?

```
publicclassReversal
```

```
{

publicstaticvoidmain(String args[])

{

String input = “Java Interview”;

System.out.println(“Given String -> ” + “Java Interview”);

char charArray[] = input.toCharArray();

System.out.println(“Reversed String -> “);

for(int i = charArray.length-1;i>=0; i–)

{

System.out.print(charArray[i]);

}

System.out.println();

}

}
```

48. Write down the program to delete the duplicates in the array

```
import java.util.ArrayList;

import java.util.LinkedHashSet;

import java.util.List;

import java.util.Set;

classRemoveDuplicates

{

publicstaticvoidmain(String args[])
```

```
{

/*create ArrayList with duplicate elements*/

ArrayList duplicate = new ArrayList();

duplicate.add(5);

duplicate.add(7);

duplicate.add(1);

duplicate.add(4);

duplicate.add(1);

duplicate.add(7);

System.out.println(“Given array: “+ duplicate);

Set <Integer> withoutDuplicates = new
LinkedHashSet<Integer>(duplicate)

duplicate.clear();

duplicate.addAll(withoutDuplicates);

System.out.println(“Array without duplicates: “+
duplicate);

}

}
```

49. Pen down a program for reversing a number

```
import java.util.Scanner;

publicclassNumberReversal

{

publicstaticvoidmain(String args[])
```

```
{

System.out.println(“Please enter the number to be
reversed”);

Scanner sc = new Scanner (System.in);

int number = sc.nextInt();

int reverse = reverse(number);

System.out.println(“Reverse of number: ” + number + ” is
” + reverse(number));

}

publicstaticintreverse(int number){

int reverse = 0;

int remainder = 0;

do{

remainder = number%10;

reverse = reverse*10 + remainder;

number = number/10;

}while(number > 0);

return reverse;

}

}
```

50. What is the program to execute binary search

```
import java.util.Scanner;

import java.util.Arrays;
```

```
publicclassBinary {

publicstaticvoidmain(String[] args) {

System.out.println(“Enter total number of elements : “);

Scanner s = new Scanner (System.in);

int length = s.nextInt();

int[] input = newint[length];

System.out.printf(“Enter %d integers”, length);

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

input[i] = s.nextInt();

}

/* binary search requires the input array to be sorted so
we must sort the array first*/

Arrays.sort(input);

System.out.print(“the sorted array is: “);

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

{

System.out.println(input[i] + ” ,”);

}

System.out.println(“Please enter number to be searched
in sorted array”);

int key = s.nextInt();

int index = BSearch(input, key);

if (index == -1) {
```

```
System.out.printf(“Sorry, %d is not found in array %n”,
key);

} else {

System.out.printf(“%d is found in array at index %d %n”,
key,

index);

}

}

publicstaticintBSearch(int[] input, int number) {

int low = 0;

int high = input.length – 1;

while (high >= low) {

int middle = (low + high) / 2;

if (input[middle] == number) {

return middle;

} elseif (input[middle] < number) {

low = middle + 1;

} elseif (input[middle] > number) {

high = middle – 1;

}

}

return-1;
```

```
}

}
```

51. Verify whether the number is Prime

```
import java.util.Scanner;

publicclassPrime

{

publicstaticvoidmain(String args[])

{

System.out.println(“Enter the number to check: “);

Scanner sc = new Scanner(System.in);

int num = sc.nextInt();

boolean isPrime = false;

if(num!=0)

{

isPrime = checkPrime(num);

}else

{

System.out.println(“Enter valid number”);

}

if(isPrime == false)

{

System.out.println(” NOT PRIME!!”);
```

```
}

else

{

System.out.println(“PRIME!!”);

}

}

publicstatic boolean checkPrime(int number)

{

int sqrt = (int) Math.sqrt(number) + 1;

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

{

if(number % i== 0)

{

returnfalse;

}

}

returntrue;

}

}
```

52. Programming for Fibonacci Series

```
import java.util.Scanner;

publicclassFibo
```

```
{

publicstaticvoidmain(String args[])

{

System.out.println(“Enter the number upto which
Fibonacci series should be printed “);

Scanner sc = new Scanner(System.in);

int num = sc.nextInt();

System.out.println(“Fibonacci Series upto %d is” + num);

for(int i=1; i<=num; i++)

{

System.out.print(fib(i) + ” “);

}

}

publicstaticintfib(int n)

{

if(n ==1 || n==2)

{

return1;

}

return fib(n-1) + fib(n-2);

}

}
```

53. Verify whether the number is Palindrome

```
import java.util.Scanner;

publicclassPalinDrome

{

publicstaticvoidmain(String args[])

{

System.out.println(“Enter the string to check”);

Scanner sc = new Scanner(System.in);

String str = sc.nextLine();

boolean isPalindrome;

isPalindrome = checkPalindrome(str);

if(str.equals(” “))

{

System.out.println(“Enter valid string”);

}

else

{

if(isPalindrome)

{

System.out.println(“PALINDROME!!”);

}

else

{
```

```
System.out.println(“NOT A PALINDROME!!”);

}

}

}

publicstatic boolean checkPalindrome(String input)

{

int str_length = input.length();

int i=0, j= str_length-1;

while(i<j)

{

if(input.charAt(i) != input.charAt(j))

returnfalse;

i++;

j–;

}

returntrue;

}

}
```

54. Program for Swapping of two numbers

```
import java.util.Scanner;

publicclassSwap

{
```

```
publicstaticvoidmain(String args[])

{

Scanner s = new Scanner(System.in);

System.out.println(“Enter a number: “);

int a = s.nextInt();

System.out.println(“Enter second number: “);

int b = s.nextInt();

System.out.println(“Value of a and b before swapping: ”
+ “a = ” +a + ” b = ” + b);

swap(a,b);

}

publicstaticvoidswap(int a , int b)

{

int swap_variable;

swap_variable = a;

a = b;

b = swap_variable;

System.out.println(“Value of a and b after swapping: ” +
“a = ” +a + ” b = ” + b);

}

}
```

55. Verify whether the number is Armstrong number

```
import java.util.Scanner;

publicclassArmstrong

{

publicstaticvoidmain(String args[])

{

Scanner s = new Scanner(System.in);

System.out.println(“Enter a number: “);

int number = s.nextInt();

int a=number, sum = 0, num=0;

while(a%10 !=0)

{

num = a%10;

sum = sum + (num*num*num);

a = a/10;

}

if(sum == number)

{

System.out.println(“Armstrong Number!”);

}

else

{

System.out.println(“Not an Armstrong Number!”);
```

```
}

}

}
```

56. To Sum Up

You will find that going over these fundamental Java Interview Questions and advanced Java Programming Interview Questions is an effective method to prepare for the interview. To better prepare for the Java interview training, you should consider looking at Java Training in Chennai at SLA. Good Luck with your interview.

Meta Description: Master SLA's comprehensive Java interview questions and answers devised by industry experts to ace your interview. Join SLA for the **best Java Training in Chennai**.

EASY WAY TO IT JOB

SLA Institute

KK Nagar [Corporate Office]

No.10, PT Rajan Salai, K.K. Nagar, Chennai – 600

078.

Landmark: Karnataka Bank Building

Phone: +91 86818 84318

Email: enquiry@softlogicsys.in

Map: Google Maps Link

OMR

No. E1-A10, RTS Food Street
92, Rajiv Gandhi Salai (OMR),
Navalur, Chennai - 600 130.

Landmark: Adj. to AGS Cinemas

Phone: +91 89256 88858

Email: info@softlogicsys.in

Map: Google Maps Link

Trending Courses

- Tableau
- DotNet Development
- Software Testing
- Angularjs Programming
- MEAN Stack

Social Media Links