By OnlineInterviewQuestions.com

Core Java Interview Questions for Beginners

What is Java?

Java is a platform independent, case sensitive language which is used to create secure and robust applications. It was developed by James Gosling in 1991. Apart from having a C like notation, it was much easier, uniform and simple than C/C++. The first version of Java was released by Sun Microsystems in 1995. It works on a "Write Once Run Anywhere" platform. Java is open source software. The J2 versions of Java were renamed as Java SE, Java EE, and Java ME. Java is an object-oriented programming language. Java is a combination of the features of C and C++. It has inherited its features from C and object-oriented programming feature from C++.

Q1. Explain the significance of class loaders in Bootstrap?

In order to load the Java Classes into a virtual environment, Classloader are used. A class loader will load key classes such as java.lang.object and several other codes into the memory. Usually, these classes are loaded only if a demand occurs. Since Java Runtime Environment includes loaders, they need not know about file and file systems. Also, the loaders are responsible for converting the named class into an equivalent binary form.

Q2. What is the difference between JDK, JRE, and JVM?

It is important to understand the difference between JDK, JRE, and JVM in Java.

JVM (Java Virtual Machine):

Java virtual machine is actually an abstract machine which provides a runtime environment in which the java bytecode gets executed. A JVM performs some main tasks such as- loading, verifying and executing the code. It also provides a runtime environment.

JRE (Java Runtime Environment):

The Java runtime environment is used to provide a runtime environment. It is implemented by JVM. It consists of certain libraries which are used by JVM at runtime.

JDK (Java Development Kit):

The Java development kit consists of JRE and all the development tools which are necessary for compiling a program.

Q3. What are the various access specifiers in Java?

Access specifiers in java are the keywords which define the access scope of the function. It can be used before a class or a method. There are basically 4 types of access specifiers in java: –

- **Public: –** class, methods, and fields are accessible from anywhere.
- **Protected: –** methods and fields are accessed from the same class to which they belong. Even from the subclass and class from the same package. Any outside class cannot access the methods and fields.
- **Default: –** only from the same package, the methods and fields be accessed and not from the outside package.
- **Private: –** the methods and fields can be only accessed from the same class to which they belong.

Q4. What do you understand by looping in java? Explain the different types of loops.

If we want to execute a statement or a block of statements repeatedly in java, then loops are used. And such a process is known as **looping**. There are basically 3 types of looping. They are: –

- **"For loop"**: – for executing a statement or a set of statements for a given number of times, for loops are used. It is used when the programmer knows the number of times a statement is to be executed.
- **"While loop": –** in this, a particular statement or set of statements are executed until a particular condition is fulfilled. In this, before the execution of statements, the condition is checked. It is known as an entry controlled loop.
- **"Do While loop": –** there is only one difference between the while and do-while loop and that is- in do-while loop, the condition is checked after the execution of the block of statements. It is known as an exit controlled loop.

Q5. Outline the major features of Java.

The major features of Java are listed below: –

- **Object-oriented: –** Java language is based on object-oriented programming. In this, the class and the methods describe the state and behavior of an object. The programming is done by relating a problem with the real world object.
- **Portable: –** the conversion of java program into Java bytecodes take place. This helps in running the program in any environment or platform without any dependency.
- **Platform independent: –** java ensures “write once run anywhere” as its programs can be run on multiple platforms like- Windows, Mac, Sun Solaris, Linux, etc.
- **Robust: –** java’s strong memory due to no pointer allocations, make it robust. The memory leaks are prohibited with the help of automatic garbage collection in java.
- **Interpreted: –** the java code is converted into bytecodes by the java compiler which is then interpreted and executed by the Java interpreter or a just-in-time compiler (JIT).

Q6. Compare java and python.

Java and Python, both the languages hold an important place in today's IT industry. But in some factors, one is better than the other. Such factors are:

- Java is easy to use, whereas Python is very good in this case.
- The speed of coding in Java is average, whereas in Python it is excellent.
- In Java, the data types are statically typed, whereas in python the data types are dynamically typed.
- Java is average in the case of data sciences and machine learning applications, whereas data science and machine learning applications are majorly based on python. Python is very good in this case.

Q7. What do you mean by an object in java?

An **object** is a basic entity in an object-oriented programming language that represents the real-life entities. Many objects are created by a java program that interacts with the invoking methods. The state of an object is represented by its attributes and reflects its properties. The behavior of an object is represented by the methods of the object and reflects the relationship of an object with the other objects. The identity of an object is a unique name provided to an object which helps one object to interact with other objects.

Q8. What do you understand by classes in java?

In Java, a **class** is a user-defined data type from which objects are created. It can also be called as a blueprint or prototype. A class is a collection of various methods and variables which represent a set of properties that are common to all the objects of one type. A class includes components such as: –

- **Modifiers:** it states that a class can be public or can have a default access.
- **Class name:** the name of a class should begin with a letter and should be the same as the file name.
- **Body:** the class body is surrounded by braces- {}.

Q9. Define inheritance with reference to Java.

Inheritance in Java is an important concept in any object-oriented programming language. Using inheritance, one class is allowed to inherit the properties and features of another class. The properties and features include fields and methods. Inheritance takes place with the help of the keyword **"extends"**. There is a superclass whose features are inherited. It is also known as the base class or parent class. There is a **subclass** that inherits the properties of the base class. It is also known as a derived class, extended class or the child class. Inheritance also supports the concept of "reusability".

Q10. Differentiate between StringBuffer and StringBuilder in java.

Following are the differences between **StringBuffer** and **StringBuilder** in java: –

- StringBuffer methods are synchronized, whereas StringBuilder methods are non-synchronized.
- In StringBuffer, a storage area is a heap which can be modified easily, whereas, in StringBuilder, storage is heap which can be modified easily.
- StringBuffer is threading safe, whereas StringBuilder is not.
- The performance of StringBuffer is very slow, whereas the performance of StringBuilder is fast.

Q11. Differentiate between array list and vector in java.

Following are some of the differences between array list and vectors in JAVA: –

- An array list is not synchronized, whereas a vector is synchronized.
- An array list works fast, whereas a vector does not.
- There is an increase of 50% in the array size whenever any element is inserted inside the array list, whereas vector defaults to doubling the size of its array.
- The increment size is not defined in an array list, whereas in vector it is defined.
- Only iterators can be used in an array list for traversing an array, whereas vectors can use both enumeration and iterators for traversing an array.

Q12. Define an abstract class with reference to Java.

An abstract class is a class in java which contains the abstract keyword in its declaration. It can have both abstract and non-abstract methods. The abstract class can have the public, private, protected, constant and default variables. This class needs to be extended and we need to implement the methods. This class can't be instantiated. A class can be declared as an abstract class if it has at least one abstract method. An abstract class has at least one virtual function. Keyword "abstract" is used to make a class abstract.

Q13. Explain constructors and types of constructors in java.

A **constructor in Java** is basically a block of code which initializes the objects which are newly created. Although it resembles an instance method in java it is not since it does not return any value or have a return type. People usually refer constructors as special types of methods in java. Constructors and methods are two different things. In java, or in any other object-oriented programming language, a constructor has the same name as that of the class. There are basically 3 types of constructors in java. They are: –

- Default constructors
- Parameterized constructors, and
- Non-parameterized constructors.

Q14. Name and explain the types of ways which are used to pass arguments in any function in java.

In Java, there are basically two ways by which arguments can be passed in any function. They are: –

- **Pass by value**- in this, a copy of the value of the arguments is passed in the function. If any changes are made to the parameters of the sub-routine, then there is no effect on the arguments which are used to call it.
- **Pass by reference**- In this, the copy of the address of the parameters are passed in the function and if any changes are made to the parameters, then it will have an effect on the arguments used to call the sub-routine.

Q15. Explain super keyword in java.

The **super** keyword in java is basically related to the parent class. A super keyword can be used for several reasons such as-

- It can be used to call the superclass or the parent class constructor.
- It can also be used to access the method of the superclass which has been hidden by a subclass.
- A super keyword can also be used to call the constructor of the parent class.

Q16. What do you understand by overloading and overriding in java?

When in a program there is more than one method with the same name in a single class but the arguments used in them are different, then such thing is referred to as method overloading.

Overriding concept in java means when there are two methods with the same signature, one is in the parent class and the other one is in the child class. The override annotation can be used in the child class overridden method.

Q17. What mechanism does Java use for memory management?

Java use garbage collection which is performed automatically in Java to free the memory space from unused objects automatically. It is used to identify and dispose those objects that are no longer needed for the application. In other languages like C and C++ you must perform garbage collection by yourself to manage the memory, it means Java provides good memory management.

Q18. Why we do exception handling in java and how many types of exceptions are there?

We do expectation handling to handle the errors or problems that arise due to the syntax error or logical error. If these errors are not handled properly then it disturbs the flow of the application and performs abnormal behavior like stops the application.so, these errors need to be handled carefully for smooth and error-free flow.

There are 2 types of exceptions.

1. Checked Exception
2. Unchecked Exception

Q19. When will we prefer to use Set and List in Java and why?

When we have to store a collection of objects which is unordered we prefer the Set interface. The other main reason for using Set is we can ignore duplicate entities and keep our data unique. For example, if we have a Set containing different entities "John", "Smith", "Smith", "John", "Alisha". It will display "John" and "Smith" only one time in output by ignoring the duplicated entities. On the other hand, List is and ordered the collection of objects. So, when we have to keep our entities in ordered form while not caring about the duplication we will prefer List.

Q20. What is the purpose of final keyword and when to use it?

The purpose of the final keyword is to make things unchanged in application lifeline. We can make a class, method or variable unchanged by writing "final" keyword before the method, class, and variable. When the final is written before the variable, it remains unchanged (you can’t change it). A function which use final keyword cannot be overridden. On the other hand, if we make a class final we cannot extend it. When we have to make these unchanged conditions, we prefer to use the final keyword.

Q21. What is early binding and late binding in Java?

The process of the connecting function call with function body is called binding. When this binding is done at compile time, it is called **early binding**. Early binding is static binding it is done when the type of an object is determined at compile time. On the hand, late binding is done at runtime. Late binding is dynamic binding because it is done at runtime by the compiler itself.

Q22. Why we use multi threading instead of multiprocessing?

The purpose of using multithreading is to make multiple lightweight processes running simultaneously because threads have a plus point, they are lightweight process. Multiple processes not only reserve the memory space but also make the increase the work complication as well. To eliminate the memory space issue and complications we prefer multithreading because they used a shared memory area which saves memory and performs better memory management.

Q23. How Thread Scheduler schedule the task?

Thread scheduler maintains threading states by using primitive and time slicing scheduling. It schedules the tasks by setting priority and time. Only one thread can run at a time in a single process by thread scheduler. When the highest priority is set to a process, the first process at highest priority gets starts before coming into dead or waiting state, after that the next process in the waiting state comes and make it task done and so on. This is how the thread scheduler schedules the task.

Q24. How dead lock situation occurs in java and how you can identify it?

Deadlock is a situation that occurs when the same resource is required by two processes. This situation occurs in multithreading when one thread holds a resource which is required by the second thread and second thread holds a resource which is required by the first thread. To identify the deadlock, we can use the jConsole/VisualVM, it will show exactly which threads are getting locked and on which object.

Q25. Justify your answer that you can't define a method inside another method in java, if you can then how?

It is not possible in java to write the definition of a function inside another function. It is like we are creating confusion for the compiler in making a decision for which function definition part to terminate and which is not. It is not recommended and not professional. For this purpose, we can use Lambda expression.

Note: Lambda expression is introduced in Java 8 which can be created without belonging to any class.

Q26. Why object class is super class for every class in java?

After creating the object **JVM** internally calls the toString method to generate the indirect address of the object. To string method is present inside object class, which is defined as non-static. So that object class makes it as a superclass for every class in java.

Q27. Explain what are final variable in Java?

In Java programming, a final variable is a variable that can only be assigned a value once. Once a final variable has been assigned a value, it cannot be reassigned or modified. This means that the value of a final variable is constant and cannot be changed throughout the lifetime of the program.

There are two types of final variables in Java:

- Final instance variables
- Final static variables

Final variables can be used in a number of ways, such as in defining constants, creating immutable objects, or creating variables that are only set once during the lifetime of a program

Examples:

```
final int x=10; //final variable
final static double PI=3.14; //final static variable
```

Q28. Why do we create public static method in Java?

In Java language, a **public static method** is a method that can be called without creating an instance of the class in which it is defined. Because these methods are not associated with any specific object, they can be invoked directly using the class name, without the need to create an instance of the class.

There are many reasons available, why we create public static methods in Java:

Utility methods: These are methods that perform some specific task and can be reused across multiple classes. Public static methods are useful for creating utility methods because they can be called from anywhere in the program without the need to create an instance of the class.

Factory methods: These are methods that are used to create and return objects of a particular class. Public static methods are useful for creating factory methods because they can be called directly using the class name.

Main method: The main method of a Java program is a public static method. This method is the entry point of a program and is executed when the program is run.

Performance: Accessing a static method is faster than accessing a non-static method because there is no need to create an instance of the class before calling the method.

Q29. What is Spliterator in Java?

Like **Iterator** and **ListIterator**, **Spliterator** is also a Java Iterator, which is used to iterate elements one-by-one from a List implemented object. **Java Spliterator** is one of the four iterators among Enumeration, Iterator, ListIterator, and Spliterator. Java Spliterator is an interface in Java Collection API. Spliterator is introduced in Java 8 release in java. util package and it supports Parallel Programming functionality.

Q30. What is fail fast iterator in Java?

Iterators in java provide us the fluency to traverse over the collection objects. Iterators restored by the collection in nature are **fail-fast** or **fail-safe**. **Fail-Fast iterators** promptly throw ConcurrentModificationException if a collection is adjusted while iterating over it.

Q31. What is semaphore in Java?

In Java, **semaphore** manages access to a shared resource by the use of a counter and if the counter is bigger than zero, then access is granted else if it is zero, then access is refused. What the limiter is counting are permits that allow access to the shared resource. Thus, to obtain the resource, a thread must be granted a permit from the semaphore. Technically, to use a semaphore, the thread that needs access to the shared resource attempts to acquire a permit.

Q32. What is default access modifier in Java?

If a class has no modifier (the default, also known as package-private), it is visible only within its own package.

Q33. What is byte code in Java?

Bytecode is a computer program code that has passed through compilation from source code to machine code. The code produced after it is compiled is easily recognized by the microprocessor. There are different types of byte code such as Java byte code and these byte codes make use of distinct syntax. A virtual machine like the JVM in full, Java Virtual Machine has the capability of executing a byte code. It is on this machine that a java byte code can run on.

Q34. Why string is immutable in java?

The **string is Immutable in Java language** for certain reasons. One of the reasons for doing so is due to the fact that every string object is cached in the String pool. There is a risk associated because of multi-sharing of string literals among many clients as an action of a particular client affect other clients. To reduce this risk of compromising string class and to optimize performance, a string has to be immuted in java. Because of the importance and popularity of strings just like the HashMap, it is vital for the strings to be immutable as a mutable string can give two distinct hash codes.

Q35. What is the need of the JAVA?

The **need of Java** is that it enforces an object-oriented programming model and can be used to create complete applications that can run on a single computer or be distributed across servers and clients in a network thus it can easily build mobile applications or run on desktop applications that use different operating systems and servers.

Q36. How to convert integer to String in Java?

There are several ways to convert an integer to a string in Java:

- Using the + operator: int i = 42; String s = "" + i;
- Using String.format(): int i = 42; String s = String.format("%d", i);
- sing String.valueOf() method: int i = 42; String s = String.valueOf(i);
- Using the StringBuilder or StringBuffer class: int i = 42; String s = new StringBuilder().append(i).toString();
- Using the Integer wrapper class: Integer i = new Integer(42); String s = i.toString();

You can use any of the above methods depending on your requirement.

Q37. What are Java generics?

Java Generics is a feature introduced in Java 5 that allows for type-safe collections and other types. They allow you to write a single class or method that can work with multiple types of data, rather than having to write a separate version of the class or method for each type. This can lead to more reusable and maintainable code.

Here is an example of a simple generic class in Java:

```
public class Box {
    private T t;
    public void add(T t) {
        this.t = t;
    }
    public T get() {
        return t;
    }
}
```

In this example, the class "Box" is parameterized with a type "T". The type "T" is used as a placeholder for an actual type that will be specified when an object of the "Box" class is created. The class can be used to create boxes that can hold any type of object, such as "Box" or "Box".

Q38. What is the difference between a Java int and a Java Integer?

In Java, 'int' is a primitive data type that represents a 32-bit signed integer. It is a basic data type that is used to store numerical values.

While 'Integer' is a wrapper class for the 'int' primitive data type. It is an object that contains a single field of type 'int'. The 'Integer' class provides additional methods for working with integers, such as converting an 'int' to a 'String' or comparing two 'Integer' objects.

Here are some key differences between 'int' and 'Integer':

- int is a primitive data type, whereas Integer is a reference data type.
- int variables are stored in the stack memory, whereas Integer objects are stored in the heap memory.
- int variables are directly used in arithmetic and comparison operations, whereas Integer objects need to be

unwrapped using the intValue() method before they can be used in arithmetic or comparison operations.
- int variables cannot be null but Integer class objects can be null.
- int variables are more efficient to work with than Integer objects because they require less memory and have fewer method calls.

Q39. What happens when a class is immutable?

An **immutable class** is a class whose state cannot be modified after it is created. Once an object of an immutable class is created, it cannot be modified in any way.

Here are some popular characteristics of immutable classes:

- All fields of the class are final and therefore cannot be modified after the object is created.
- The class has no setter methods, so the values of the fields cannot be changed once they are set in the constructor.
- The class should not provide any methods that can modify its state.
- If a class has mutable fields (like collections) it should return a new instance of the collection with the new state, instead of modifying the existing one.
- The class should be thread-safe since it can be shared among multiple threads without the need for explicit synchronization.

Q40. Is Java a pure object-oriented programming language?

Java is considered to be an object-oriented programming (OOP) language, but it is not a pure OOP language.

It supports many of the key concepts of OOP such as encapsulation, inheritance, and polymorphism. It uses classes and objects to model real-world entities, and it allows for the creation of complex, reusable software by breaking it down into smaller, manageable units.

Q41. What is wrapper class in Java?

In Java, a **wrapper class** is a class that wraps (or "encapsulates") a primitive data type so that it can be used as an object. Java provides a wrapper class for each of the eight primitive data types:

- Byte
- Short
- Integer
- Long
- Float
- Double
- Character
- Boolean

For example, the wrapper class for the int primitive data type is Integer, the wrapper class for the char primitive data type is Character, and so on.

Q42. What is difference between Iterator and Enumeration?

Both Iterator and Enumeration are used in Java to iterate over a collection of elements, but they have some important differences.

- Iterator was introduced in JDK 1.2, while Enumeration was available from the initial version of Java.
- An iterator is a more modern and powerful way of traversing a collection, it has more functionality than Enumeration.
- Iterator allows the caller to remove elements from the underlying collection during the iteration with well-defined semantics, Enumeration is read-only and does not provide this functionality.
- Iterator has a remove() method to remove the current element from the collection, Enumeration does not have this method.
- Iterator is used in the latest collection classes like ArrayList, LinkedList,