Java interview questions

**1. Why is Java a platform independent language?**

Java language was developed so that it does not depend on any hardware or software because the compiler compiles the code and then converts it to platform-independent byte code which can be run on multiple systems.

The only condition to run that byte code is for the machine to have a runtime environment (JRE) installed in it.

**2. Why is Java not a pure object oriented language?**

Java supports primitive data types - byte, boolean, char, short, int, float, long, and double and hence it is not a pure [object oriented language](https://www.interviewbit.com/oops-interview-questions/).

**3. Difference between Heap and Stack Memory in java. And how java utilizes this.**

Stack memory is the portion of memory that was assigned to every individual program. And it was fixed. On the other hand, Heap memory is the portion that was not allocated to the java program but it will be available for use by the java program when it is required, mostly during the runtime of the program.

**Java Utilizes this memory as -**

When we write a java program then all the variables, methods, etc are stored in the stack memory.

And when we create any object in the java program then that object was created in the heap memory. And it was referenced from the stack memory.

**4. How is Java different from C++?**

C++ is only a  compiled language, whereas Java is compiled as well as an interpreted language.

Java programs are machine-independent whereas a c++ program can run only in the machine in which it is compiled.

C++ allows users to use pointers in the program. Whereas java doesn’t allow it. Java internally uses pointers.

C++ supports the concept of Multiple inheritances whereas Java doesn't support this. And it is due to avoiding the complexity of name ambiguity that causes the diamond problem.

**5. Pointers are used in C/ C++. Why does Java not make use of pointers?**

Pointers are quite complicated and unsafe to use by beginner programmers. Java focuses on code simplicity, and the usage of pointers can make it challenging. Pointer utilization can also cause potential errors. Moreover, security is also compromised if pointers are used because the users can directly access memory with the help of pointers.

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

**Instance variables** are those variables that are accessible by all the methods in the class. They are declared outside the methods and inside the class.

**Local variables** are those variables present within a block, function, or constructor and can be accessed only inside them. The utilization of the variable is restricted to the block scope. Whenever a local variable is declared inside a method, the other class methods don’t have any knowledge about the local variable.

**7. What are the default values assigned to variables and instances in java?**

There are no default values assigned to the variables in java. We need to initialize the value before using it. Otherwise, it will throw a compilation error of (**Variable might not be initialized**).

But for instance, if we create the object, then the default value will be initialized by the default constructor depending on the data type.

If it is a reference, then it will be assigned to null.

If it is numeric, then it will assign to 0.

If it is a boolean, then it will be assigned to false. Etc.

**8. What do you mean by data encapsulation?**

Data Encapsulation is an Object-Oriented Programming concept of hiding the data attributes and their behaviours in a single unit.

**9. Tell us something about JIT compiler.**

A Just-In-Time (JIT) compiler is a component of the runtime environment that improves the performance of applications by compiling code at runtime, rather than ahead-of-time (AOT).

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

The ‘**==**’ operator is used to compare primitives and reference variables.

The **equals()** method is intended for logical comparison of objects, meaning it compares the values or states of the objects.

**11. Briefly explain the concept of constructor overloading.**

Constructor overloading is the process of creating multiple constructors in the class consisting of the same name with a difference in the constructor parameters.

**12. Define Copy constructor in java.**

Copy Constructor is the constructor used when we want to initialize the value to the new object from the old object of the same class.

**13. Can the main method be Overloaded?**

Yes, the **main** method in Java can be overloaded. However, only the standard **public static void main(String[] args)** method will be used as the entry point of the program. Eg :

class Main {

public static void main(String args[]) { System.out.println(" Main Method"); }

public static void main(int[] args){ System.out.println("Overloaded Integer array Main Method"); }

public static void main(char[] args){ System.out.println("Overloaded Character array Method"); }

public static void main(double[] args){ System.out.println("Overloaded Double array Method"); }

}

**14. Comment on method overloading and overriding.**

**Method overloading** is a feature in Java that allows a class to have more than one method with the same name, as long as their parameter lists are different.

**Method overriding** is a feature in Java that allows a subclass to provide a specific implementation of a method that is already defined in its superclass.

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

Yes, in Java, a single **try** block can have multiple **catch** blocks to handle different types of exceptions that might be thrown within the **try** block only the first **catch** block satisfying the **catch** condition is executed.

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

**final variable:** When a variable is declared as final in Java, the value can’t be modified once it has been assigned.If any value has not been assigned to that variable, then it can be assigned only by the constructor of the class.

**final method:** A method declared as final cannot be overridden by its children's classes.

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

**Final :** If any restriction is required for classes, variables, or methods, the final keyword comes in handy.

**Finally :** It is the block present in a program where all the codes written inside it get executed irrespective of handling of exceptions.

**Finalize:** Prior to the garbage collection of an object, the finalize method is called so that the clean-up activity is implemented.

**18. Is it possible that the ‘finally’ block will not be executed? If yes then list the case.**

Yes. It is possible that the ‘finally’ block will not be executed. The cases are :-

1. Suppose we use **System.exit()** in the above statement.

2. If there are fatal errors like Stack overflow, Memory access error, etc.

**19. When can you use super keyword?**

**1. Accessing Superclass Methods :** To call a method from the superclass that has been overridden in the subclass.

**2. Accessing Superclass Constructors :** To invoke a constructor of the superclass from within a subclass constructor.

**3. Accessing Superclass Fields :** To access a field of the superclass that has been hidden by a field in the subclass.

**20. Can the static methods be overloaded?**

Yes! There can be two or more static methods in a class with the same name but differing input parameters!

**21. Why is the main method static in Java?**

The main method is always static because static members are those methods that belong to the classes, not to an individual object. So if the main method will not be static then for every object, It is available. And that is not acceptable by JVM. JVM calls the main method based on the class name itself. Not by creating the object.

**22. Can the static methods be overridden?**

No! Declaration of static methods having the same signature can be done in the subclass but run time polymorphism can not take place in such cases.

**23. Difference between static methods, static variables, and static classes in java.**

**Static Methods and Static variables** are those methods and variables that belong to the class of the java program, not to the object of the class.

For example - We have used mathematical functions in the java program like - **max(), min(), sqrt(), pow(),** etc. And if we notice that, then we will find that we call it directly with the class name. Like - **Math.max(), Math.min(),** etc. So that is a static method.

**Static classes**- A class in the java program cannot be static except if it is the inner class. If it is an inner static class, then it exactly works like other static members of the class.

**24. What is the main objective of garbage collection?**

The main objective of this process is to free up the memory space **(Heap Memory)** occupied by the unnecessary and unreachable objects during the Java program execution by deleting those unreachable objects.

**25. What are shallow copy and deep copy in java?**

**Shallow copy** **-** The shallow copy only creates a new reference and points to the same object.

**Deep Copy** **-** In a deep copy, we create a new object and copy the old object value to the new object.

**26. What is a singleton class in Java?**

Singleton classes are those classes, whose objects are created only once. And with only that object the class members can be accessed.

**27. What is String pool?**

The string pool in Java is a special memory region in the Java heap where string literals are stored. It optimizes memory usage by reusing immutable string objects. When a string literal is created, the JVM checks the pool; if the string already exists, it returns a reference to the existing string instead of creating a new one.

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

**Storage area:** In string, the String pool serves as the storage area. For StringBuilder and StringBuffer, heap memory is the storage area.

**Mutability:** A String is immutable, whereas both the StringBuilder and StringBuffer are mutable.

**Efficiency:** It is quite slow to work with a String. However, StringBuilder is the fastest in performing operations. The speed of a StringBuffer is more than a String and less than a StringBuilder.

**Thread-safe:** In the case of a threaded environment, StringBuilder and StringBuffer are used whereas a String is not used. However, StringBuilder is suitable for an environment with a single thread, and a StringBuffer is suitable for multiple threads.

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

**Availability of methods:** Only abstract methods are available in interfaces, whereas non-abstract methods can be present along with abstract methods in abstract classes.

**Variable types:** Static and final variables can only be declared in the case of interfaces, whereas abstract classes can also have non-static and non-final variables.

**Inheritance:** Multiple inheritances are facilitated by interfaces, whereas abstract classes do not promote multiple inheritances.

**Data member accessibility:** By default, the class data members of interfaces are of the public- type. Conversely, the class members for an abstract class can be protected or private also.

**30. What is a Comparator in java?**

Comparator is the interface in java that contains the compare method. And by overloading the compare method, we can define that on what basis we need to compare the values.

**31. What are the differences between JVM, JRE and JDK in Java?**

**JDK-** For making java programs, we need some tools that are provided by JDK (Java Development Kit). JDK is the package that contains various tools, Compiler, Java Runtime Environment, etc.

**JRE -** To execute the java program we need an environment. (Java Runtime Environment) JRE contains a library of Java classes + JVM. **What are JAVA Classes?** It contains some predefined methods that help Java programs to use that feature, build and execute. **For example -** there is a system class in java that contains the print-stream method, and with the help of this, we can print something on the console.

**JVM -** (Java Virtual Machine) JVM is a part of JRE that executes the Java program at the end. Actually, it is part of JRE, but it is a software that converts bytecode into machine-executable code to execute on hardware.

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

**Thread Safety:**

**HashMap:** Not thread-safe. Concurrent modifications are not safe.

**Hashtable:** Thread-safe. All methods are synchronized.

**Null Keys and Values:**

**HashMap:** Allows one null key and multiple null values.

**Hashtable:** Does not allow null keys or null values.

**Performance:**

**HashMap:** Generally faster due to the lack of synchronization overhead.

**Hashtable:** Slower due to synchronization on every operation.

**33. What is the difference between the program and the process?**

A program can be defined as a line of code written in order to accomplish a particular task. Whereas the process can be defined as the programs which are under execution.

**34. What is the difference between the ‘throw’ and ‘throws’ keyword in java?**

The **‘throw’** keyword is used to manually throw the exception to the calling method. And the **‘throws’** keyword is used in the function definition to inform the calling method that this method throws the exception. So if you are calling, then you have to handle the exception.

**35. Java works as “pass by value” or “pass by reference” phenomenon?**

Java always works as a **“pass by value”**. There is nothing called a “pass by reference” in Java. However, when the object is passed in any method, the address of the value is passed due to the nature of object handling in Java. When an object is passed, a copy of the reference is created by Java and that is passed to the method.

**36. What is the ‘IS-A ‘ relationship in OOPs java?**

**‘IS-A’** relationship is another name for inheritance. When we inherit the base class from the derived class, then it forms a relationship between the classes. So that relationship is termed an **‘IS-A’** Relationship.

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

If the static modifier is not included in the main method signature in Java, the program will not run and will produce a **runtime error**. The main method must be static because the Java Virtual Machine (JVM) needs to call this method without creating an instance of the class.

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

The program can't compile as the compiler says that the method has been already defined inside the class.

**39. Will the finally block get executed when the return statement is written at the end of try block and catch block as shown below?**

**Finally block** will be executed irrespective of the exception or not. The only case where finally block is not executed is when it encounters **‘System.exit()’** method anywhere in try/catch block.

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

It is because the 0 index array avoids the extra arithmetic operation to calculate the memory address.

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

In the linked list, we only need to adjust the references when we want to delete the element from either end or the front of the linked list. But in the array, indexes are used. So to manage proper indexing, we need to adjust the values from the array So this adjustment of value is costlier than the adjustment of references.