

Q1. What is Java?

Ans: Java is a high-level programming language developed by Sun Microsystems. Also, it is a class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. Java app are typically compiled to bytecode that can run on any java virtual machine (JVM). It is general purpose programming language intended to let application developers (WORA).

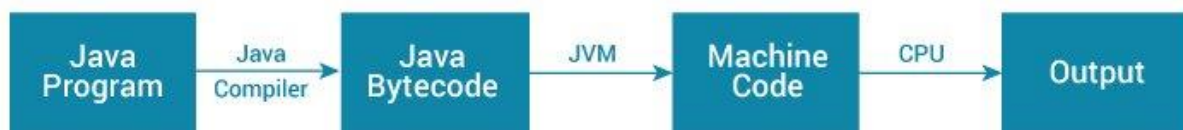
Q2. What is a package in Java? List down various advantages of packages

Ans: A package in java is used to group related classes. we use packages to avoid name conflicts and to write a better maintainable code. Java package is used to categorize by the classes and interfaces.

- It is easy to maintained.
- Java package is provided as access protection.
- It may remove naming collision.
- This package can be providing reusability of code.
- We can create our own package or extend already available package.

Q3. Explain JDK, JRE and JVM?

Ans: JRE (Java Runtime Environment) is the implementation of JVM (java virtual machine) and is defined as a software package that provides java class libraries along JVM and other components to run application written in java programming. Like JDK, JRE is also platform dependent. JVM is platform independent.



Q4. Explain public static void main (String args []) in Java?

Ans: args []: "args []" implies that the value or arguments entered from the Command Line are an array. So overall it implies that public static void main (String args []) is the main method of the class from which the JVM needs to start, and it is public so that it could be accessed from anywhere in the application.

Q5. Why Java is platform independent?


Ans: Java is platform-independent because it does not depend on any type of platform. Hence, Java is platform-independent language. In Java, programs are compiled into byte code and that byte code is platform-independent.

Q6. What are wrapper classes in Java?

Ans: A Wrapper class is a class whose object wraps or contains primitive data types. When we create an object to a wrapper class, it contains a field and, in this field, we can store primitive data types. In other words, we can wrap a primitive value into a wrapper class object.

They convert primitive data types into objects. Objects are needed if we wish to modify the arguments passed into a method (because primitive types are passed by value).

Q7. What are the differences between C++ and Java?



Difference Between C++ and Java

Java	C++
Java doesn't support Pointer concept.	It support pointer concept.
It doesn't support multiple inheritances.	It support multiple inheritance
Java does not include structures or unions.	It have structure and union concept.
Java includes automatic garbage collection.	C++ requires explicit memory management
Java has method overloading, but no operator overloading.	C++ supports both method overloading and operator overloading.
It is platform independent programming language	It is platform dependent programming language.
It is mainly used for design web based application but also use for develop desktop application.	It is used for design only desktop application like OS, Compiler etc.
Java uses compiler and interpreter both	C++ use only Compiler.
Java is high level programming language in java we write code like simple English language.	C++ is more nearer to hardware then Java

Q8. Why pointers are not used in Java?

Ans: Java doesn't have pointers (in the C/C++ sense) because it doesn't need them for general purpose OOP programming. Furthermore, adding pointers to Java would undermine security and robustness and make the language more complex. Memory access via pointer arithmetic: this is fundamentally unsafe. Java has a robust security model and disallows pointer arithmetic for the same reason. ... No pointer support makes Java more **secure** because they point to memory location or used for memory management that loses the security as we use them directly.

Q9. List some features of Java?

Ans: Simple. Java is easy to learn and its syntax is quite simple, clean and easy to understand. ...

2) **Object** Oriented. In java, everything is an **object** which has some data and behaviour. ...

3) Robust.

4) Platform Independent.

5) Secure.

6) Multi-Threading.

7) Architectural Neutral.

8) Portable.

9) Distributed.

10) High Performance.

Q10. Why is Java Architectural Neutral?

Ans: Java's object-oriented model enables programmers to benefit from its large set of existing classes. An attractive feature of Java is that it is architecture neutral. This means that Java code needs to be compiled only once, after which it can run on many CPUs.

Java is architecture neutral because there are no implementation dependent features. Its compiler generates an architecture-neutral object file format, which makes the compiled code to be executable on many processors, with the presence of Java runtime system.

Q11. How Java enabled High Performance?

Ans: Java uses Just-In-Time compiler to enable high performance. Just-In-Time compiler is a program that turns Java bytecode, which is a program that contains instructions that must be interpreted into instructions that can be sent directly to the processor.

Q12. Why Java is considered dynamic?

Ans: Java is considered dynamic because of Bytecode. The source code which is written in one platform that code can be executed in any platform. It loads the class file during runtime only. Hence, anything that happens in runtime is dynamic.

Also, it is designed to adapt to an evolving environment. Java programs can carry an extensive amount of run-time information that can be used to verify and resolve accesses to objects at run-time.

Q13. What is Java Virtual Machine and how it is considered in context of Java's platform independent feature?

Ans: Java Virtual Machine (JVM) is a specification that provides runtime environment in which java bytecode (. class files) can be executed. The JVM is the platform. JVM makes this possible because it is aware of the specific instruction lengths and other particularities of the platform (Operating System).

Q14. List two Java IDE's?

Ans: Eclipse: Eclipse is a Java IDE that is one of the 3 biggest and most popular IDEs in the world. Eclipse is a Java IDE that is one of the 3 biggest and most popular IDEs in the world. It was written mostly in Java but it can also be used to develop applications in other programming languages.

IntelliJ IDEA: is a Java IDE that is one of the 3 biggest and most popular IDEs in the world. It has two versions, namely the free open-source community edition and a paid Ultimate edition.

Q15. Why Java is called as "Platform"?

Ans: Object-oriented: In Java, everything is based on objects. Secure: Java provides a secure platform to develop safe and virus-free applications. Platform Independent: Unlike C and C++, when we compile Java code it is compiled into platform-independent bytecode rather than the platform-specific machine code.

Q16. Which version of java have u learned? Name some of the new features added to it.

Java SE 8 (March 18, 2014)

- ✓ Language-level support for Lambda expressions.
- ✓ Allowed developers to embed JavaScript code within applications.
- ✓ Annotation of Java Types.
- ✓ Provided Date and Time API.
- ✓ Repeating Annotations.
- ✓ Launching of JavaFX applications.
- ✓ Removal of permanent generation.

Java SE 8 is not supported in Windows XP but after JDK 8 update 25, we can install and run it under Windows XP.

Java 8 is set as a default version to download from java.com from October 2014.

Q17.What gives Java its 'write once and run anywhere' nature?

Ans: This is all possible because of JVM. How Java is WORA: . class file), which is interpreted by JVM, so once compiled it generates bytecode file, which can be run anywhere (any machine) which has JVM (Java Virtual Machine) and hence it gets the nature of Write Once and Run Anywhere.

Q18.Difference between path and class path?

Ans: path: it is location of bin files (binary executable files) example- java.exe, javac.exe,
class Path: it is location of your .class file (which is created after compile your java source file)

path and Class path both are operating system level environment variables. Path is used define where the system can find the executables(.exe) files and class path is used to specify the location .class files.

Q19.What is the signature of main function in java?

Ans: The method signature for the main () method contains three modifiers: public indicates that the main () method can be called by any object. static indicates that the main () method is a class method. void indicates that the main () method has no return value.

Q20. Is Java Pure-Object oriented Language?

Ans: A fully object-oriented language needs to have all the 4 oops concepts. In addition to that, all predefined and, user-defined types must be objects and, all the operations should be performed only by calling the methods of a class.

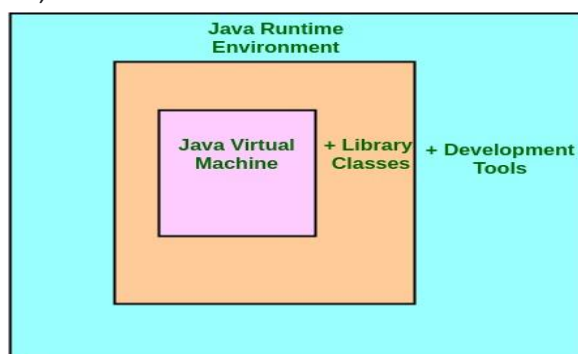
-Though java follows all the four object-oriented concepts,

-Java has predefined primitive data types (which are not objects).

-You can access the members of a static class without creating an object of it.

Q21.What is the difference between JDK and JRE?

Ans: JDK is a software development kit whereas JRE is a software bundle that allows Java program to run, whereas JVM is an environment for executing bytecode. The full form of JDK is Java Development Kit, while the full form of JRE is Java Runtime Environment, while the full form of JVM is Java Virtual



JDK = JRE + Development Tool
JRE = JVM + Library Classes

JVM(java virtual machine)

Q22.What is JVM? What it does?

Ans: JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java bytecode can be executed.

- A specification: where working of Java Virtual Machine is specified. But implementation provider is independent to choose the algorithm. Its implementation has been provided by Oracle and other companies.
- An implementation: Its implementation is known as JRE (Java Runtime Environment).
- Runtime Instance: Whenever you write java command on the command prompt to run the java class, an instance of JVM is created.

The JVM performs following operation:

- Loads code
- Verifies code
- Executes code
- Provides runtime environment

Q23.Why JVM is called as “virtual machine”?

Ans: The Java Virtual Machine, or JVM, is an abstract computer that runs compiled Java programs. The JVM is "virtual" because it is generally implemented in software on top of a "real" hardware platform and operating system. All Java programs are compiled for the JVM. The JVM is mean because it of its ambition.

Q24. What are the main components of JVM? Explain them. Or Explain JVM Architecture.

Ans: JVM (Java Virtual Machine) acts as a run-time engine to run Java applications. JVM is the one that actually calls the **main** method present in a java code. JVM is a part of JRE (Java Runtime Environment)

Components of JVM

- Loading. This component loads the classes. ...
- Linking. Here, the subsystem has a verifier to verify if the bytecode is correct or not. ...
- Initialization. ...
- Method Area. ...
- Heap Area. ...
- Stack Area. ...
- PC Registers. ...
- Native Method Stacks

Q25.What is the difference between Java compiler (javac) and JIT?

Ans: When compiling a java program, the static compiler that is run using the command javac converts the source code to byte code which are in the form of. class files. ... JIT compiles the code when it is needed but not before runtime. Whenever a program is executed this compiled object code is invoked instead of interpreting the entire byte code.

Q26.Is Empty .java file name a valid source file name?

Ans: An empty java file is a perfectly valid source file. java file contains more than one java classes, provided at the most one of them is a public class.

Since, you cannot leave class name empty as well as you can't also change its name to any other since it is public. If you write a file in Java which is already present in the location, it will be overwritten automatically. Unless you are writing to that file with an append flag set to True.

Q27.Is JRE different for different Platforms?

Ans: Whenever we try to run the code, JVM requires some library set and files for code execution and these files are presented in JRE. JRE = JVM + set of libraries. JRE is also platform dependent. That means we have different JRE versions for different platforms.

Q28.Difference between C++ and java in terms of object creation.

Ans: C++ uses only compiler, whereas Java uses compiler and interpreter both. C++ supports both operator overloading & method overloading whereas Java only supports method overloading. C++ supports manual object management with the help of new and delete keywords whereas Java has built-in automatic garbage collection. Java doesn't support pointers.

Q29.Who invokes main () function?

Ans: As we know, the main () method for any Java application as the Java Run time environment calls the main () method first. JVM calls the main method because it is the entry point to any class that has to be loaded in order to execute the class.

Q30.What is .class file known as?

Ans: class file known as bytecode. JVM executes the bytecode and produces Operating System specific machine instructions. These machine instructions are executed directly by CPU. class file is usually produced by a Java compiler from Java programming language source files.

Q31.Can we define more than one public class in a java source code? what is the rule of public class and file name?

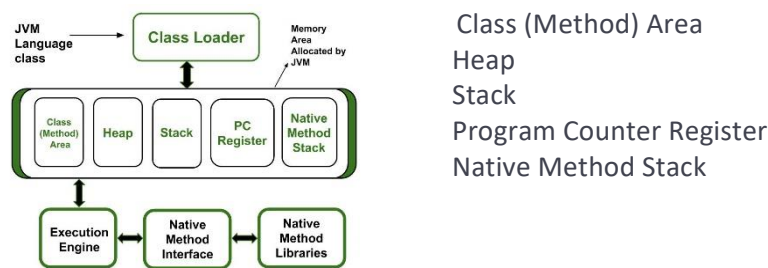
Ans: A Java source file can have only one class declared as public, we cannot put two or more public classes together in a. java file. This is because of the restriction that the file name should be same as the name of the public class with. If you have more than one public classes a single file a compile-time error will be generated.

Q32.What is JIT compiler?

Ans: The Just-In-Time (JIT) compiler is a component of the runtime environment that improves the performance of Java™ applications by compiling bytecodes to native machine code at run time. The JIT compiler helps improve the performance of Java programs by compiling bytecodes into native machine code at run time.

Q33.How many types of memory areas are allocated by JVM?

Ans: The memory in the JVM divided into 5 different parts:



Q34.What is the rule for local member in java.

Ans: Local variables cannot use any of the access level since their scope is only inside the method. Final is the Only NonAccess Modifier that can be applied to a local variable. Local variables are not assigned a default value; hence they need to be initialized.

Q35.What are the various access specifiers in Java?

Ans: Java provides four types of access modifiers or visibility specifiers i.e. default, public, private, and protected. The default modifier does not have any keyword associated with it. When a class or method or variable does not have an access specifier associated with it, we assume it is having default access.

Q37.What is native code?

Ans: Native code is computer programming (code) that is compiled to run with a particular processor. Native code is machine code executed directly by the CPU. This is in contrast to .NET bytecode, which is interpreted by the .NET virtual machine.

Q38.Why there is no sizeof operator in java?

Ans: Because the size of primitive types is explicitly mandated by the Java language. There is no variance between JVM implementations. Moreover, since allocation is done by the new operator depending on its argument there is no need to specify the amount of memory needed. sizeof operator is not available in Java. But, at times you may want to know size of your Java objects.

Q40.U have reference type as a member of class. What is the default value it gets?

Ans: null

The default value of a reference type variable is null when they are not initialized. Null means not referring to any object.

Q40.What kinds of programs u can develop using Java?

Ans: **Java Applications**

- Java Mobile Applications.
- Java Desktop GUI Applications.
- Java Web-based Applications.
- Java Web Servers and Application Servers.
- Java Enterprise Applications.
- Java Scientific Applications.
- Java Gaming Applications.
- Java Big Data Technologies.

Q41.What is the job done by class loader?

Ans: The Java Class Loader is a part of the Java Runtime Environment that dynamically loads Java classes into the Java Virtual Machine. The Java run time system does not need to know about files and file systems because of class loaders. Java classes aren't loaded into memory all at once, but when required by an application.

Q42.Explain the hierarchy of class loaders in java.

Ans: Java run-time has the following built-in class loaders:

Bootstrap class loader: The virtual machine's built-in class loader typically represented as null, and does not have a parent.

Platform class loader: To allow for upgrading/overriding of modules defined to the platform class loader, and where upgraded modules read modules defined to class loaders other than the platform class loader and its ancestors, then the platform class loader may have to delegate to other class loaders, the application class loader for example. *In other words, classes in named modules defined to class loaders other than the platform class loader and its ancestors may be visible to the platform class loader.*

System class loader: It is *also known as application class loader* and is distinct from the platform class loader. The system class loader is typically used to *define classes on the application class path, module path, and JDK-specific tools*. The platform class loader is a parent or an ancestor of the system class loader that all platform classes are visible to it.

Bootstrap class loader → Extension class loader → Application class loader

Q43.What is the role played by Bytecode Verifier?

Ans: The bytecode verifier acts as a sort of gatekeeper: it ensures that code passed to the Java interpreter is in a fit state to be executed and can run without fear of breaking the Java interpreter.

Once the byte code is loaded by the JVM, (with the help of the. class file), the bytecode is checked to see the validity with the help of the verifier.

Q44.What are the memory areas allocated by JVM?

Ans: Heap – Runtime storage allocation for objects (reference types). Stack – Storage for local variables and partial results. A stack contains frames and allocates one for each thread. ... Native method stacks – It contains all the native methods used by the application.

- Class (Method) **Area**→Heap→Stack→Program Counter Register→Native Method Stack.

Q46.When parseInt () method can be used?

Ans: Convert a string to an integer with the parseInt method of the Java Integer class. The parseInt method is to convert the String to an int and throws a Number Format Exception if the string cannot be converted to an int type.

Q47.What is finalized () method?

Ans: Finalize () is the method of Object class. This method is called just before an object is garbage collected. Finalize () method overrides to dispose system resources, perform clean-up activities and minimize memory leaks.

Q48.Difference between C++ pointer and Java reference.

Ans: A reference is a variable that refers to something else and can be used as an alias for that something else Pointers are a particular implementation of the concept of a reference, and the term tends to be used only for languages that give you direct access to the memory address.

So, a pointer is a reference, but a reference is not necessarily a pointer.

Q49.repeated

Q50.repeated