CDAC Mumbai PG-DAC AUGUST 24
Assignment No- 3

Note: Write down this Interview questions & answers in your notebook .take a screenshorts ,make word file & upload on Github.

1) Explain the components of the JDK.

2) Differentiate between JDK, JVM, and JRE.

3) What is the role of the JVM in Java? & How does the JVM execute Java code?

4) Explain the memory management system of the JVM.

5) What are the JIT compiler and its role in the JVM? What is the bytecode and why is it important for Java?

6) Describe the architecture of the JVM.

7) How does Java achieve platform independence through the JVM?

8) What is the significance of the class loader in Java? What is the process of garbage collection in Java.?

9)What are the four access modifiers in Java, and how do they differ from each other?

10) What is the difference between public, protected, and default access modifiers?

11) Can you override a method with a different access modifier in a subclass? For example, can a protected method in a superclass be overridden with a private method in a subclass? Explain.

12) What is the difference between protected and default (package-private) access?

13) Is it possible to make a class private in Java? If yes, where can it be done, and what are the limitations?

14) Can a top-level class in Java be declared as protected or private? Why or why not?

15) What happens if you declare a variable or method as private in a class and try to access it from another class within the same package?

16) Explain the concept of "package-private" or "default" access. How does it affect the visibility of class members?

① Explain the components of the JDK.

Ans. To develop Java Software, a developer needs JDK on their machine.

Java Development Kit typically includes -

~~i) Java Development Tools~~

1) Java Compiler (javac)

- The Java compiler (javac) is a key component of JDK that transforms Java source code (.java files) into bytecode (.class files).
- The generated bytecode can be executed on any platform with JVM, ensuring WORA philosophy of Java.

2) Java Virtual Machine (JVM)

- JVM is the rutime engine that executes Java bytecode. It provides an abstraction layer between the Java application and the underlying OS.

3) Java Runtime Environment (JRE)

- JRE is a subset of JDK that includes the JVM and essential class libraries.

4) Java API Libraries

- It is a vast collection of pre-built classes and methods that simplify common programming tasks. These libraries cover areas such as input/output, networking, db connectivity, GUI and more.

classmate
Date ________
Page ________

5) Java Debugger (jdb)
- It is a powerful tool for debugging Java application
- It allows developers to set breakpoints, inspect variables, and Step through the code to identify and fix issues during developement.

② Differentiate between JDK, JVM and JRE.

Ans. JDK - A Development Kit that includes the JRE, compiler, and tools needed to develop Java applications.
JVM - Java Virtual machine is an engine that runs Java bytecode, making Java platform-independent.
JRE - A Java Runtime Environment contains the JVM and core libraries needed to run Java applications, but without development tools.

③ What is the role of the JVM in Java? How does the JVM execute Java code?

Ans. The JVM executes Java bytecode, making Java platform-independent by translating bytecode into machine-specific instructions.

① Class Loading - The JVM loads the compiled .class files (bytecode).
② Bytecode Verification - It checks the bytecode for security and correctness.

③ Execution : The JVM interprets or uses Just In Time (JIT) compilation to convert bytecode into native machine code, which is then executed by the host machine.

④ Explain the memory management system of the JVM.

Ans. The JVM's memory management system is responsible for allocating, managing, and reclaiming memory used by Java applications. It consists of several key components -

1) Heap memory - It is the runtime data area in which objects are allocated.
2) Stack - Java stack stores frames. It holds local variables and partial results, and plays a part in method invocation and return.
Each thread has a private JVM stack, created at the same time as thread. A new frame is created each time a method is invoked, A frame is destroyed when its method invocation completes.
3) Class (method) Area -
It stores per-class structures such as the runtime constant pool, field and method data, the code for methods.

4) Program Counter Register - It contains the address of the Java virtual Machine instruction currently being executed.
5) Native Method Stack - It contains all the native methods used in the application.

(5) What are the JIT compiler and its role in JVM? What is the bytecode, and why is it important for Java?

Ans. JIT compiler -
Just In Time (JIT) compiler is part of the JVM that improves the performance of Java applications by compiling bytecode into native machine code at runtime, allowing the JVM to execute the program faster.
Bytecode -
Bytecode is an intermediate code generated by the Java compiler. It's platform independent, allowing Java programs to run on any system with a JVM, which interprets the bytecode and translates it to machine code.

(6) Describe the architecture of the JVM.

Ans. JVM architecture consists of:
1) Class Loader - Load class files into memory.
2) Bytecode verifier - Ensures bytecode is safe and doesn't violate access rights.
3) Interpreter - Executes bytecode line by line.
4) JIT Compiler - Compiles bytecode to native code for performance improvement.

5) Garbage Collector - manage memory by automatically reclaiming unused objects.

(7) How does Java achieve platform independence through the JVM?

Ans. Java achieves platform independence by compiling source code into bytecode, which can be executed on any machine with a JVM. The JVM interprets or compiles the bytecode into native machine code, making the program runnable on different platform without modification.

(8) What is the significance of the class loader in Java? What is the process of garbage collection in Java?

Ans. Class loader - The class loader is responsible for dynamically loading classes into the JVM at runtime.

Garbage Collection - It is an automatic memory management process that identifies and removes objects that are no longer in use, freeing up memory and preventing memory leaks.

classmate
Date
Page

(9) what are the four access modifiers in Java, and how do they differ from each other?

Ans. Public - The class, method or variable is accessible from any other class.

Protected - Accessible within the same package and by subclass.

Default - (package Private) Accessible only within the same package.

Private - Accessible only within the class where it is defin

(10) What is the difference between public, protected and default access modifiers?

Ans. Same as Q.9.

(11) Can you override a method with a different access modifier in a subclass? for example, can a protected method in a superclass be overridden with a private method in a subclass?

Ans. NO. we cannot override a method with a more restrictive access modifier. for instance, a protected method cannot be overridden with a private method, as it would reduce the visibility of the method in the subclass.

(12) what is the difference between protected and default access?

Ans. Same as Q.9.

classmate
Date
Page

(13) It is possible to make a class private in Java?
If yes, where can it be done?

Ans. yes, an inner class (a class within another class) can be made private. However, top-level classes cannot be declared as Private, meaning they must have public or package-private access.

(14) Can a top-level class in Java be declared as protected or private? why or why not?

Ans. NO, a top-level class in Java cannot be declared as protected or private. Top-level classes can only be public or package-private because they need to be accessible by the JVM and other classes in the package.

(15) What happens if you declare a variable or method as private in a class and try to access it from another class within the same package?
Ans. If ~~you~~ we declare a variable or method as private, it is not accessible from any other class, even within the same package. Attempting to access it from another class will result in a compilation error.

(16) Explain the concept of 'package-private' access?
Ans. when no access modifier is specified, the class, the method or Variable is accessible only within its own package. It is not accessible from classes in other packages, restricting its visibility to the package level.