Assignment 3.
Interview questions.

Q1) Explain the components of the JDK.

ans:- Components of the JDK are as follows:

(1) Java Runtime Environment (JRE)
JRE is the core component of the JDK that allows Java program to run. It acts as an abstract machine for the program to run.

(2) Java Compiler (Javac)
The Compiler translates the program into machine readable bytecode (Java files to .class)

(3) Java Debugger (Jdb):
A utility that allows you to debug your Java program in the Command line interface.

(4) Java Archiver (Jar)
The jar helps the archives to manage the jar files in the package library.

5) Libraries:
The JDK includes all the Libraries needed to run programs in Java Standard edition

(6) Applet Viewer:
A Command line argument that allows developers to test their Java applets by running them in browser.

7) Java Native Interface (JNI)
The native programming interface for Java which ensures that code is portable across all platforms.

(8) Open JDK : offical
The open source implementation of the JDK which includes a class library a virtual machine and Java Compiler

(9) Javadoc :-
The Javadoc Components of the JDK generate the nessasary documentation for any components that are added in the Source Code

(10) Java launcher (Java) : Launches java applications by executing bytecode on JVM

(11) Java Doc
generates documentation from Java Source Code Comments.

(12) Java API :
sets of class libraries that provides the core functionality of Java

Q2] Differentiate between JDK, JVM & JRE ?

ans:

(1) JVM (Java Virtual Machine)

- provides Runtime environment in which Java bytecode can be executed.

Tasks of JVM - load Code
- verifies Code
- Execute Code
- provide Runtime environment.

- JVM

Jvm is an abstract machine. It is called a virtual machine because it doesn't physically exist. It's a specification that provides a runtime environment.

- JRE - JVM is platform dependent because there are different JVM implimentation for each platform.
- the Java bytecode that JVM runs is platform independent allowing the same Java program to run on different platforms as long as the appropriate JVM is available.

- JRE

JRE → JVM + Set of Libraries

- JRE is the implementation of JVM.
- To run any Java Code JRE is minimum required.
- JRE Contains Set of Libraries that JVM uses at runtime.
- JRE physically exists.

- It is platform dependent.

(8) JDK:

JDK → JRE + Development Tool

- The JDK is a platform-dependent development tools kit that includes ~~thread~~ JRE, the Java Compiler & various development tools It is essential to any one who wants to write & build Java application

Components of JDK: JRE, Java Compiler, Java Debugger, Java archiver Java ~~doc~~ doc

~~(9)~~

(9) Explain the memory management System of JVM

→ JVM divides memory in several regions

- Heap: stores objects & arrays. It's shared among all of threads
- Stack: stores local variables & method Call information. Each thread has ~~one~~ own stack
- Method ~~Case~~ area: Stores Class level data like Static, variables, Constant pool, method data & the Code for method.
- Program Counter Register: Stores the address of the Currently executing instruction.
- Native method: ~~Counter Register~~ Stack: Holds ~~address~~ native method information when Java interface with other languages

garbage Collection automatically reclaims memory by removing objects no longer in use from the heap.

(Q3) What is the role of the JVM in Java & & How does the JVM execute Java code?

→ JVM

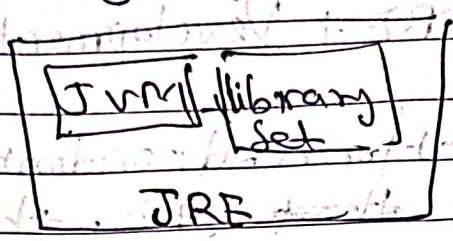
Java Virtual machine

Just provide environment

Source code

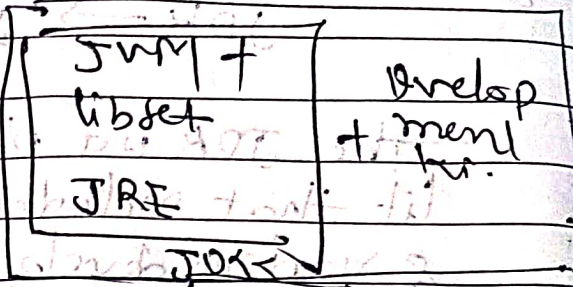
Java Compiler

Java bytecode → JVM → output.



Java Runtime Environment

Code To run Cod



Java kit development

To run cod & for development of Programing Coding

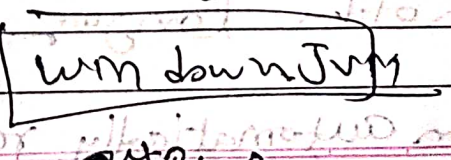
Java execute Java bytecode

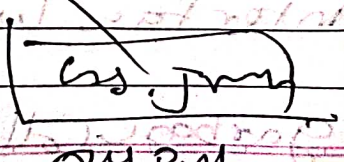
- Role of JVM is to provide runtim environment to execute the code
- JVM executes - Example

Hello World.java & Compilation

After ↓ compilation

Hellow class, .class file ka bytecode





Q5) what are JIT Compiler & its role in the JVM? what is bytecode & why it is important for Java?

→ JIT [@jit stands for just-in-Time compiler]

JIT is Java is an integral part of JVM. It improves the performance of JVM

-Role of JVM:

At first Java source Code are compiled into bytecode. After that JVM converses the byte Code into the native code or that's how JIT Compiler helps JVM to increase the performance of JVM

At Compile time

Source code → Compiler → bytecode → JVM (JIT) → Native machine code

- Bytecode

JVM Convert source code into intermediate Code Called bytecod.

Bytecodes is important for Java because Java platform independent language.

JVM

Source code → Compiler → bytecod → Binary Code

- Java bytecod Can be interpreted on any system that provide JVM

Q 7] How does the Java achive platform indepen. but through the JVM?

→ Java virtual machine provide runtime environment in which bytecod can be executed. Java program is written in one or open at any system can run any other operating system. that's why Java achive platform indepondence through JVM.

(Q) what is the significance of the loader in Java & what is the process of garbage Collection in Java?

→ Class loader :- It runtime loads classes dynamically memory & Concepts them to necessary libraries.

Garbage Collection: An authontical memory management function that helps heps Stop memory leter leater by recovering memory utilize by things that can no longer refrence or Scope.

(Q) What are the significance of the loader in java what is process of garbage collection in java?

→ Class loader :- At runtime loads class dynamically memory & connects them to necessary libraries

garbage Collection :- An automatic memory management function that helps stop memory leaks by recovering memory utilised by things that are no longer referenced or in scope

(Q9) what are the four access modifiers in java how do they from each other?

→ four access modifiers in java.

① public :- Acess from any where

② protected : Acessible, within the same package & subclasses.

③ default : (package-private) : Acessible only within the same package

④ Private :- Acessible only within the same class

(Q10) what are diff. betwen public, protected & default acess modifiers?

→

- public : visible everywhere
- Protected : the access level of protected modifier within the package & outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.

Default :- the access level of a default modifier is only within the package. It cannot be accessed from the outside package

(Q11) Can you override a method with different access modifier in a subclass? for example. Can a protected ~~nested in method~~ in subclass? Explain

→

A method with more restrictive access modifier cannot be overridden. A private method in a subclass for instance cannot override a protected method in a superclass there is just one possible access level. more or the same

(Q12) What is the diff between protected & default (package-private) access?

→

- Protected - Available to subclasses & other classes in the same package.

- Default (package-private) : visible only within the same package. not to subclass outside the package

(Q13) I> It is possible to make a class private in Java? If yes where can it be done & what are the limitations?

→

No it is not possible to make a class private at top level since nested (inner) class can be at top level. Since nested (inner) class can be private, restricting

access to within the outer class.

(14) Can a top-level class in java to declared as protected ~~private~~ private & ~~why~~ why or why not?

→ A top level cannot be declared as ~~pro~~ protected or private. It can only be public or package private (default). This restrictions ensures it ensures that the class is accessable as needed access the application.

(15) What happens if you declare a variable or as ~~pu~~ in class & ~~why~~ to access it from another class within the same package?

→ Even within the same package, a private variable or method cannot be accessed directly from another visibility is limited to remember & the declaring class alone with private access.

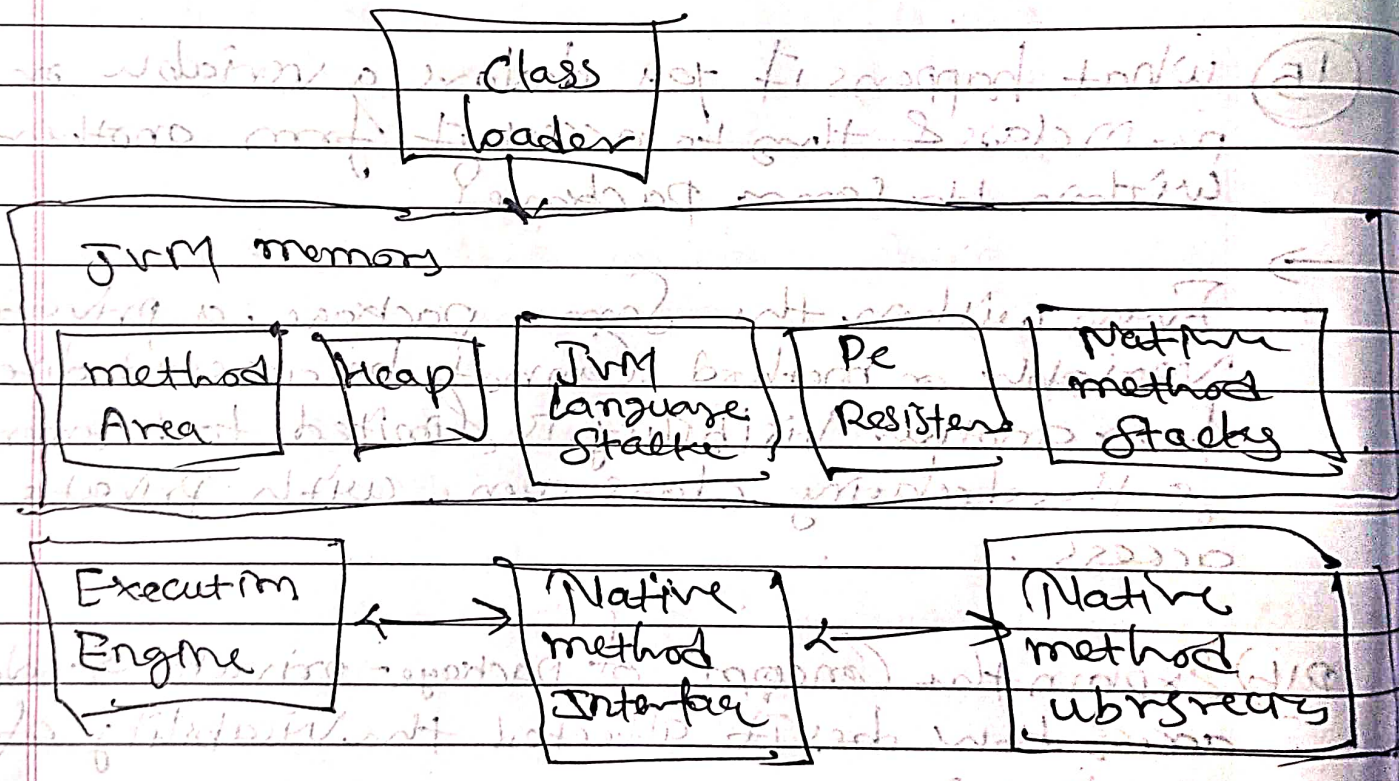
(16) Explain the concept of package-private or default access how does it affected the visability of data members?

→ package-private (default) access ~~a class, method~~ or you can only access a class, method or variable that is part of the same package if you don't specifically an explicit access modifier this is help full for organizing related classes into groups & managing their visability without exposing them to the general public

[26] JVM architecture:-

→ JVM (Java Virtual machine) run Java application as a run main method present in java code.

when we Compile a java file class Contains bytes code with the same name present in java file are generated by the java Compiler. this file goes into various steps when we run it these steps together describes JVM



- Class loader ... Subsystem
  - loading, linking, Initialization)

JVM architecture includes

(1) Class loader : loads class file into memory
(2) memory area : Divides memory into ~~area~~ regions regions like heap, stacks, method area etc.
(3) Execution Engine : ~~Execution~~ Executes the bytecode includes the interpreter, JIT Compiler & garbage ~~Collector~~ Collector.
(4) Native Interface :- ~~Interf~~ interacts with native libraries.
(5) Native method libraries : libraries written in other languages. eg. or C++)