Page No :
Date : / /

# JDK Assignment

① Explain the components of the JDK

→ ① Java compiler (javac) :- The Java compiler translates Java source code (.java files) into bytecode (.class files) that can be executed by Java Virtual machine.

② Java Virtual Machine (JVM) :→ JVM provides runtime environment for Java to execute. It interprets bytecode & provide services such as memory mgmt, garbage collection

③ Java Runtime Environment (JRE) :- The JRE includes the JVM and essential libraries required to run Java application. It provides runtime environment to execute Java bytecode.

④ Java Development Tools :→ It includes various development tools
ie javac :- Java compiler
java - Java application
Javadoc - Java
Jar :- Java archive tool
& jdb :- Java debugger.

⑤ Java libraries (API) - It includes a vast collections of standard Java libraries known as the Java API. It provides components and classes for performing task such as input/output operations.

(Q2) Differentiate between JDK, JVM and JRE

→

| JDK | JRE | JVM |
|---|---|---|
| ① JDK is a software development kit that develops applications in Java. JDK also consist of various development tools (Java Debugger, JavaDoc etc) | ① Java Runtime Environment is an implementation of JVM. It is a type of software package that provides class libraries of Java, JVM for running application. | ① JVM is a platform independent abstract machine that has three notions in the form of specification |
| ② JDK primarily used for executing code. | ② JRE has responsibility for creating an environment for execution of code | ② JVM performs execution of the implementation |
| ③ JDK is platform ~~dependent~~ dependent. It means that for every different platform, you required a diff JDK | ③ JRE is also platform dependent. It means that for every diff plfrm, you require different JRE | ③ JVM is a plfrm independent It means it doesn't required a different for every different platform |

(3) What is the role of JVM in Java & how does JVM execute Java code

→ (1) JVM role is to compiled Java into java byte code, which is translated into a specialized platform by interrupted Java interpreter. This Java interpreter is called Java virtual machine.

(2) The role of JVM is an abstract machine designed to be implemented on the top of existing processors. It hides underlying operating system from Java applications

(3) Also to permit Java program to run on operating system and any device & to manage and enhance program memory.

JVM Execution:

(1) Java source code is compiled into bytecode & then it verify the bytecode & load the Java program through class loader into JVM memory.

(4) Explain the memory management system of the JVM

→ Memory management system of Java virtual machine (JVM) is responsible for allocating and managing memory for Java applications.

(1) Memory Area :→ Heap :- JVM Heap is primary memory area used for storing objects & their instance variable.

Stack :- Each thread in Java application has its own stack which stores local variables parameters, etc.

Method Area :- The method area stores class metadata, static variables, constant pool data & method code.

Native Heap :→ Mem allocated by native libraries ex JVM (Java Native Interface) code resides in native heap

③ Garbage collection :→ It scan the heap to identify and reclaim memory occupied by objects that are no longer reachable or in use by the application

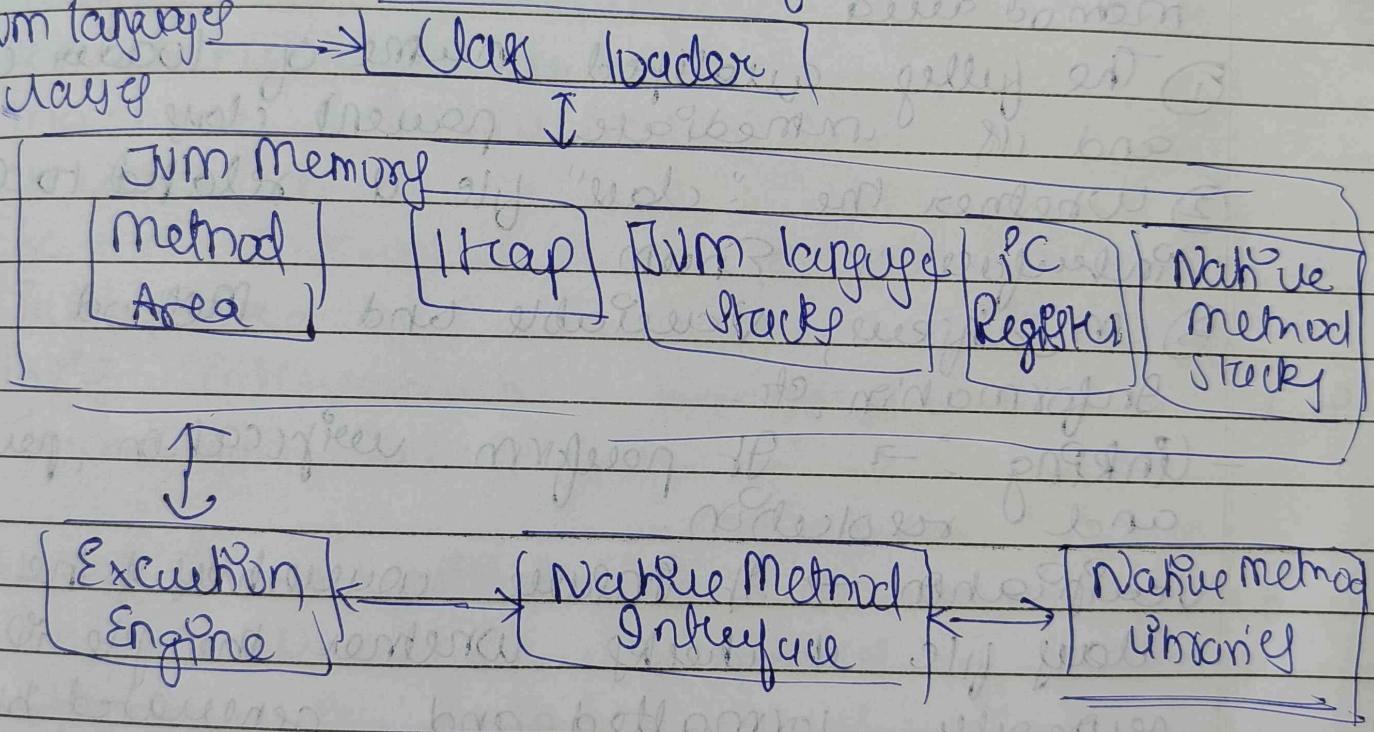
Q4 What are the JIT compiler and its role in the JVM? What is the bytecode and why is it important for Java?

→ Just in Time compiler is an essential part of the JRE (Java Runtime Environment) that is responsible for performance optimization of java based applications during runtime.

Role :- It is interpreter that convert bytecode into machine level native language.

① Bytecode → It is instruction set of the Java virtual machine (JVM), ~~and~~ it is crucial for executing programs written in the Java building language and other JVM compitable languages. Each bytecode operation in the JVM is represented by a single code byte, hence the name bytecode making it a compact form of instruction.

Q6 Describe the architecture of the JVM




→ JVM acts as runtime engine to run Java applications. JVM is the one that actually calls main method present in Java code. JVM is a part of JRE (Java Runtime Environment).
Java applications are called WORA (Write Once Run Anywhere).

class loader Subsystem

It is mainly responsible for three activities
① loading
② Linking
③ Initialization

loading: The class loader reads the .class file generate the binary data and save it in method area. For each class file JVM stores following information in method area.

① The fully qualified name of loader class and its immediate parent class
② Whether the ".class" file is related to class or Interface or Enum
③ Modifiers, Variable and Method information, etc.

- Linking :→ It perform verification, preparation and resolution.

Verification → It ensure correctness of the class file it checks whether this file is properly formatted and generated by valid compiler or not.

Preparation :→ JVM allocates memory for class static variable and initializing memory to default values.

Resolution :→ It is process of replacing symbolic reference from type with direct reference. It is done by searching into method area to locate referenced entity

Page No :
Date : / /

Q.7) How does Java achieve platform Independent through the JVM?

→ Java is platform Independent because it is compiled to a bytecode that can be run on any device that has a JVM. This means that you can write a Java program on one platform and then run it on a different platform. So here, a compiler is a program that translate source code for another program from a programming language into executable code. This executable code may be a sequence of machine instructions that can be executed by CPU directly or it may be an intermediate representation that is interpreted by a virtual machine. This intermediate representation in Java is Java Byte code.

Q8) What is significance of the class loader in Java? What is the process of garbage collection in Java

→ Class loaders are responsible for loading Java class dynamically to JVM. They are also part of JRE. Therefore the JVM doesn't need to know about the underlying files or file system in order to run Java programs. Garbage collection in Java is the process by which Java programs perform automatic memory management

Java program compile to bytecode that can be run on a JVM.
Java garbage collection is an automatic process. Automatic garbage collection is process of looking at heap memory identifying which objects are in use and which are not and deleting unused objects. An in use object or a reference object means that some part of your program still maintains a pointer to that object. So the memory used by an unreferenced object can be reclaimed. The garbage collection implementation lives in the JVM.