Que 1) Explain the components of the JDK

→

i) Javac : This component is responsible for compiling java source code files into bytecode files (.class) which can be executed by the java virtual machine (JVM).

ii) Java Runtime Environment: (JRE)

JRE consist of the JVM & running libraries required to run java application it provides the environment necessary for executing java bytecode on a specific platform.

iii) JVM: (Java virtual machine)

JVM is abstract computing machine that enables java bytecode to be executed on different hardware platform.

iv) JDK Tools: The JDK includes various command tools and utilities for java development, such as

* javadoc : doc for java source code comments
* jar : Creating & Managing java Archive files

v): Java API Libraries: includes libraries known as Java API. libraries provided classes & method.

Que 2 Differentiate between JDK, JVM & JRE

1) JDK :

i) JDK is comprehensive development kit used by java developers for creating java application

ii) it includes tools for compiling, debugging & running

iii) jdk contain java compiler, JRE, various development tools & libraries.

iv) jdk use to write, compile & package java application

2) JVM :

i) enable java bytecode to be executed on diff^n hardware platform.

ii) it interprets the bytecode & translates it into machine code specific to the underlying hardware

iii) JVM provides a runtime environment for executing java application by loading class & handling exception

iv) it included in the JRE for running java applicatio & jdk for compiling the test java code.

3) JRE :

i) Required to Run java application

ii) consist of JVM & set of libraries & othe components necessary for executing java bytecode

iii) not contain tools or compiler.

iv) for run java program need to install JRE on their system

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

→ ① Execution of Bytecodes :

JVM executes java bytecode, which is compiled from the java source code.

② Memory Management :

JVM manage memory allocation & deallocation for java obj it handles task such as garbage collection.

③ Platform independence :

JVM provides platform independence by abstracting the underlying hardware & operating system.

④ Exception handing :

JVM includes mechanisms for handling exceptions that occure during program execution it ensures that exceptions are properly caught & handled according to the java exception handling mechanism.

~~Que 4~~ How does the JVM execute java code

→ Loading :

The JVM loads bytecode from compiled, class file into memory.

execution : The JVM interprets bytecode instruction or my use just-in-time (JIT) compilation to translate bytecode into native machine code for improve performances

20 Sat

Q.4 Explain Memory management System of the JVM.

The memory management system of the JVM is responsible for allocating & deallocating memory for java object during the execution of java program. It includes several components & process to efficiently manages memory usage.

i) Heap Memory:

I) heap is the primary memory are used by the JVM for storing obj created by java application.

II) it is divided into two main region : the young generation & the old generation.

ii) Obj Allocation :

- when ~~en~~ new obj is create using the 'new' keyword in java, memory is allocated from the heap.

iii) Garbage Collection :

Garbage collection is the process of reclaiming memory occupied by unreachable obj & making it available for new allocation.

iv) Monitoring :

JVM memory management can be tuned & monitored using various command line options & monitoring tools.

Oct 2018

Ques What is JIT compiler & it role in JV
What is the bytecode & why it importan

→ 1) JIT compiler:

- JIT compiler is dynamic compiler that convert java bytecode into native machine code that can be executed directly by the CPU.

2) Role of JVM:

- JIT compiler is to the ~~compiler~~ performance java application by translating frequently execut bytecode into efficient native machine code.

1) Bytecode:

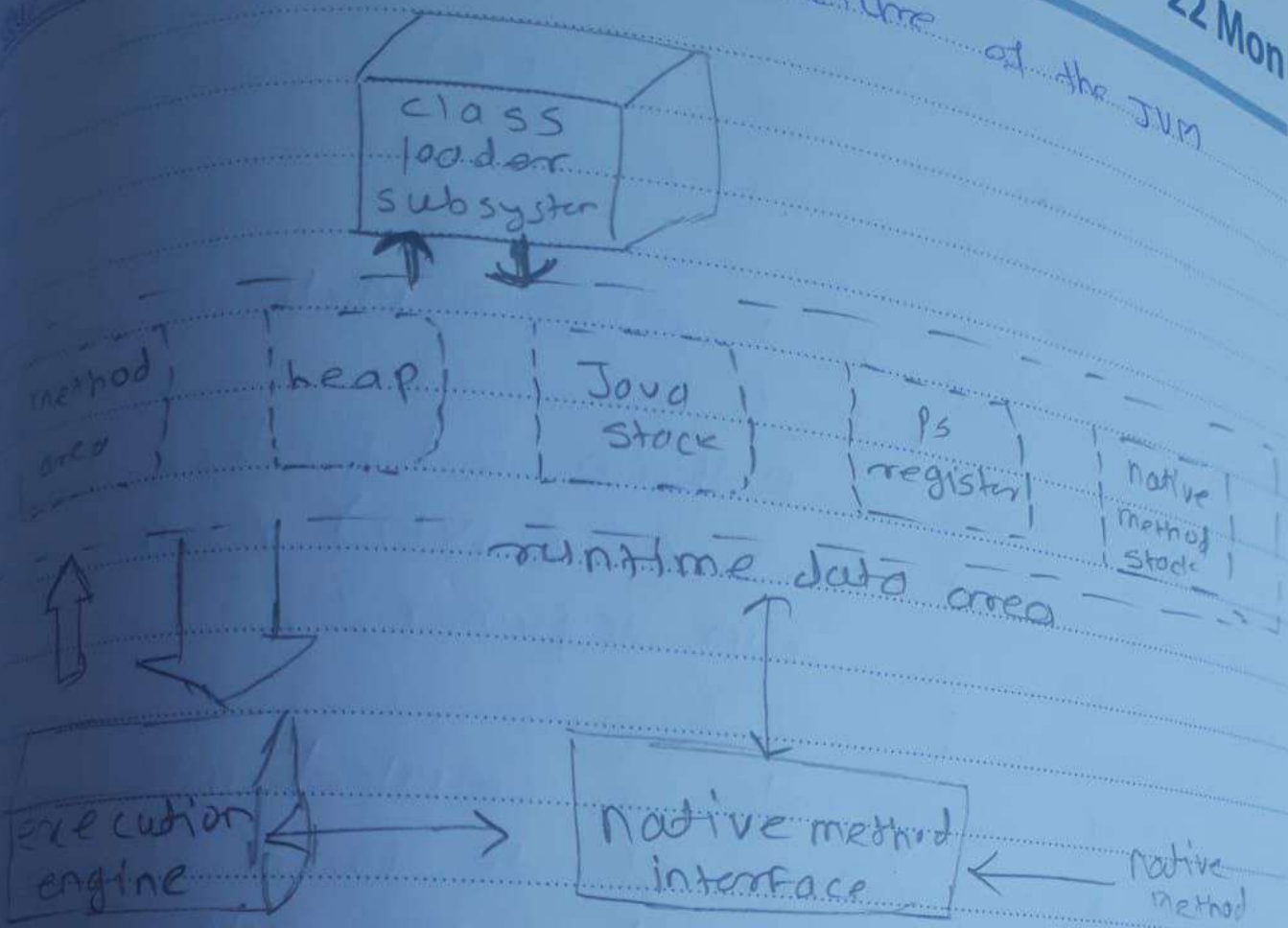
- Bytecode is an intermediate representation of Java source code that is generated by the java compiler during compilation process
- its platform-independent binary format that can be executed by any JVM, regardless of the underlying hardware & O.S
- Bytecode store in .class file.

2) importance for Java:

- Bytecodes plays a crucial role in achieving java platform Independene
- Bytecode also enable java's "write one, run anywhe

22 Mon

Describe the architecture of the JVM



* Components of Java Virtual Machine
  * class loader subsystem
    - Bootstrap class loader
    - Extension class loader
    - System class Loader
    - Custom classLoader

* Runtime data areas
  - Method Area
  - Heap
  - Java stacks
  - PC Registers
  - Native method stacks

Oct

* Execution engine
  - interpreter
  - Just in Time (JIT) compiler
  - Garbage collector

Que 7 : How does java achieve platform independence through the JVM ?

→ Java achieves platform independence through the java virtual machine (JVM) of the concept of "write once, run anywhere".

i) Compilation :

Java source code is compiled into bytecode by the java compiler. Bytecode is a of instructions designed to be executed by the JVM.

ii) Bytecode Execution :

The JVM is an interpreter of runtime environment for java bytecode. It interprets the bytecode & translates it into machine code specific to the underlying hardware of O.S. This translation process occurs dynamically at runtime, allowing the same bytecodes to be executed on any device or platform that has a compatible JVM.

24 Wed

iii) Standard java libraries:

Java provides a comprehensive standard library that abstracts way many platform-specific tasks, such as file I/O, networking, & GUI programs. These libraries ensure that java application can access platform-specific resource in a platform independent manner

iv) Runtime environment:

The JVM provides a runtime environment that manage memory allocation, garbage collection security & other runtime aspects of java application. This environment shields the application from the underlying platform, ensuring consistent behaviour across diff^n platform.