1) Explain the components of JDK.

=>

JDK => Java development kit.

It's a development enviornment used to develop applications.

JRE, JVM are main components of JDK.

JRE => Java Runtime Enviornment.

JRE contains essential components needed to run Java application

ex. java => file which contains all java API.

JVM => Java virtual m/c.

A JVM executes Java bytecode enabling Java program to runs on any device or on O.S.

2) Differentiate bet° JDK, JVM & JRE.

=>

* JVM => JVM is heart of the java programming language when we execute a java program.

JVM is responsible for converting byte code to m/c-specific code.

JRE.

JRE is implementation of JVM
JRE provides a platform to execute java program.
It contains JVM, java binaries & other classes to execute any program successfully.

JVM.

It provides platform to run program. & provides platform independance.

3). What is the role of JVM in java? How does the jvm executes java code.

=>

JVM is integral part of JDK. present .java file. It acts as an interpreter for java bytecode, allowing it to run on any device with a jvm installed. The JVM translates the bytecode & converts it to m/c code. for execution which is executed by O.S.

Steps =>

* Compile the java source code into bytecode.

* verify the bytecode & load the java program through class, loader into jvm memory.

* Create a class object for each each class & load to method area.

* Initialize bytecode instruction & data into memory.

* find the main method & create a stack frame.

* Initialize the value inside the program counter as the memory address of the main method.

4) Explain the memory management system of jvm.

=> JVM allocates & deallocates the memory in the main memory to object.

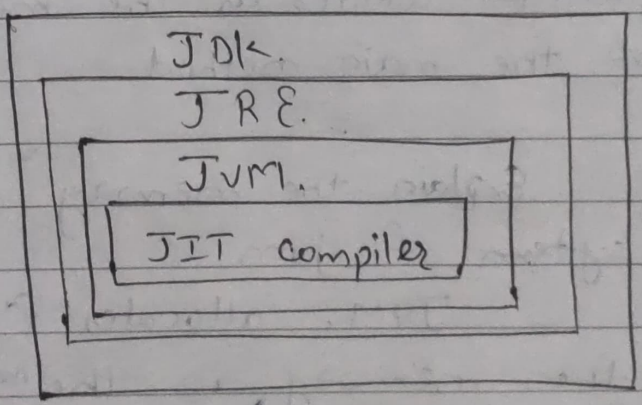
It uses a daemon process / programatic tool called Garbage collector which automatically free up the memory from main memory to save ~~the~~ / avoid the memory leak.

Byte code collects object that are no longer referenced by the application.

This process ensures that JVM has optimal memory usage.

5). What is JIT compiler? It's role in JVM. Bytecode in Java & it's importance for java.

→ JIT is a long running computer intensive program that provides the best performance environment. It optimizes the performance of java application at compile or run time.



The JIT compilation includes two approaches ahead of time & interpretation to translate java code to m/c code. (m/c language).

Byte code.

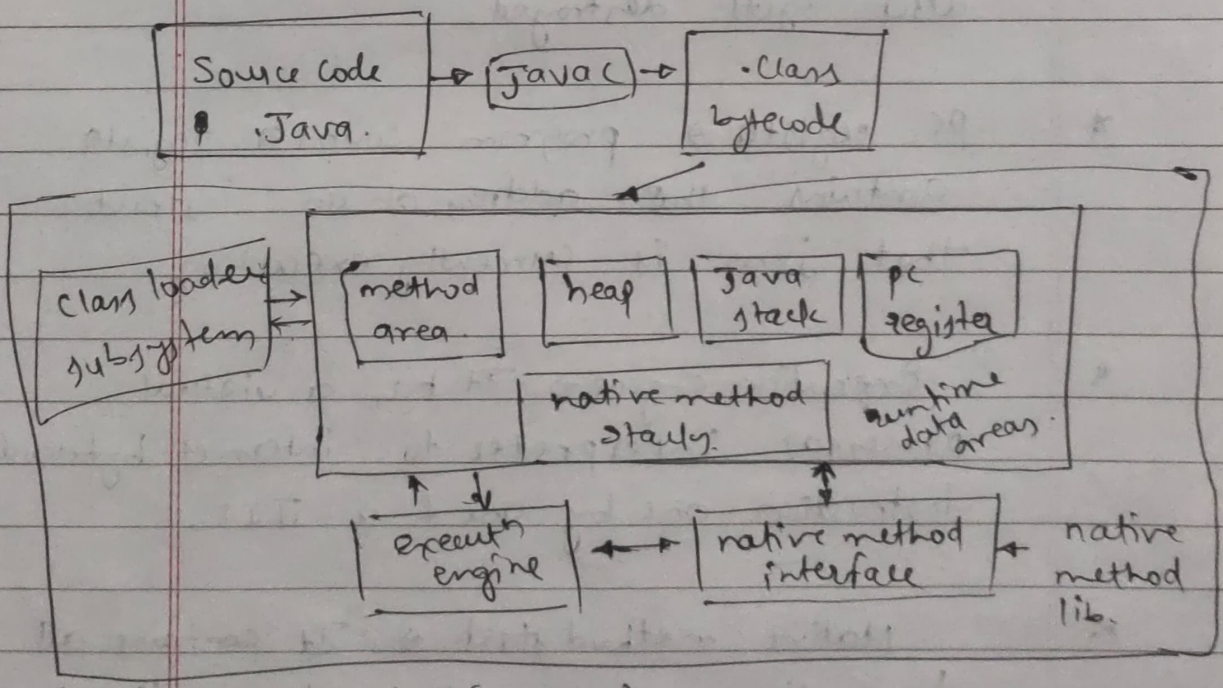
Java bytecode is instruction of jvm., crucial for executing program written in java lang. & other jvm-compatible language.

A single instruction is called bytecode.

These are the codes which executes in memory.

6) Describle the architecture of JVM.

=>



(JVM).

* Class loader ⇒ loads class file in JVM.

* Class area ⇒ Storage area for a class element, structure like fields method etc

* Heap ⇒ Runtime storage allocat^n for object.

* Stack ⇒ storage for local variable & partial results. A stack contains frames & allocates one for each thread. Once a thread gets completed, this frame also gets destroyed.

* PC register ⇒ program counter register contains the address of an instruction that jvm is currently executing.

* Execution Engine ⇒ It has a virtual processor interpreter to interpret bytecode instruction one by one & a JIT.

* Native method stack ⇒ It contains all the native method used by the application

7) How does java achive platform independance through the JVM.?

⇒

Java achieves platform independance through the JVM., which serves as an abstract^n layer b/w. the compiled Java code & the underlying hardware & operating system.

Java achieves platform independance by compiling source code to m/c code (bytecode) which is executed by JVM & dynamically translated into m/c code.

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

⇒. Class loader loads java class to jvm at runtime. JVM does not need to know about underlying files or filesystem in order to run Java program.

Garbage collect^n in java is automated process of deleting code that's no longer needed.