1) How JAVA works internally ?

Java works on the principle of Write Once, Run Anywhere.
When we compile a Java program, the .java source file is first compiled by the Java Compiler (javac) into bytecode (.class file).
This bytecode is platform-independent and runs on the Java Virtual Machine (JVM).

The JVM then uses the Class Loader to load classes, the Bytecode Verifier to check security and validity, and the JIT (Just-In-Time) Compiler to convert bytecode into native machine code at runtime for faster execution.

What is Compiler ?

A compiler is a program that converts high-level source code into bytecode or machine code.
In Java, the javac compiler translates .java files into .class files (bytecode), which are later executed by the JVM.

What is Interpreter ?

An interpreter executes the program line by line instead of compiling it all at once.
In Java, the JVM acts as the interpreter — it reads and executes the bytecode produced by the compiler.
This makes Java platform-independent, as the same bytecode can run on any system with a JVM.

What is JDK & JRE?

JDK stands for Java Development Kit — it contains tools to develop, compile, and run Java applications.
It includes the JRE, compiler (javac), and other development utilities.

JRE stands for Java Runtime Environment — it provides the libraries and JVM needed to run Java applications, but it doesn't include development tools like the compiler.

What is JVM ?

JVM is the engine that runs Java programs.
It takes the compiled bytecode and converts it into machine code that the OS can execute.
It provides platform independence, memory management, and garbage collection.