Setup of Java:

* Java - 1995 by sun micro systems - lead by Jamses gosling, later took by Oracle.
* WORE - Write Once Run Anywhere
* We need an editor to write something, normal editors like notepad,notepad++, wordpad where we can only write the code but can not run,compile,debug the code.
* We need an IDE(Integrated Development Env) to perform all above actions.
* We need a compiler to compile and it is JDK(Java Dev Kit)
* Download Oracle JDK 17 version of .exe file
* Open command prompt and type Java -- version for Java Installed, Javac – version for compiler status
* If Java path is not set up then Javac – version gives an error. Go to edit system variables and add the path of the bin folder (normally underc/program files/java-jdk-17/bin) because our compiler will be found in the bin.

Jshell:

* We have a shell for java as well, which is jshell where we can directly write,run and compile the small snippets of code.
* Here we can directly use the inbuilt methods/functions, it compile and run them automatically.
* print() is an inbuilt method in Java to print something.
* print(6); → error ca
* System.out.print(6) → 6
* But this is not suitable for Enterprise applications where huge code is present.

Core Java Theory:

* Java is said to be(PIL) Platform independent Language, which means Java has JVM(Java Virtual Machine) which is placed on top of the OS, and it is responsible to Run Java by ignoring which type of OS the Machine has.
* Java is PIL but JVM is not. As JVM is tightly coupled with OS but the thing which makes Java PIL is the Byte code.
* JVM Can not understand the normal/java language. Which knows byteCode only. So on any OS once we have JVM on top of it and by providing the bytecode to JVM from anywhere(independent of OS), it runs successfully.
* Bytecode is independent of the OS and it runs on any JVM by ignoring the OS on which our JVM installed.
* User→Hello.java→compiler(JavaC-complie)→Bytecode(Hello.class)→JVM(Run)
* JVM can run only one file from a folder but what happens if we have 10 files of code for a project with 100 line each and all are interlinked to form an application.
* Here we need to tell the what is that one file needs to be run which can run all other files internally. Here comes the file with main method which is treated as main file.
* So JVM check for the file which has main method signature and it starts the execution from there only.
* javac <filename> → to compile
* java <classname> → to run

Note: in between JVM and OS, JRE(Java Runtime Env) is there, JVM needs an environment and some libraries to run the bytecode which JRE Provides. So JDK has JRE,JVM and JRE has JVM.