Course Contents

- **Unit 1: Programming in Java (8 Hrs.)**

1.1. Java Architecture, Java Buzzwords, Path and ClassPath variables, Sample Java Program, Compiling and Running Java Programs.

1.2. Arrays, for each loop, Class and Object, Overloading, Access Privileges, Interface, Inner Class, Final and Static Modifiers, Packages, Inheritance, Overriding.

1.3. Handling Exceptions: Try, Catch, Finally, Throws, and Throw keywords, Creating Exception Class

1.4. Concurrency: Introduction, Thread States, Writing Multithreaded Programs, Thread Properties, Thread Synchronization, Thread Priorities

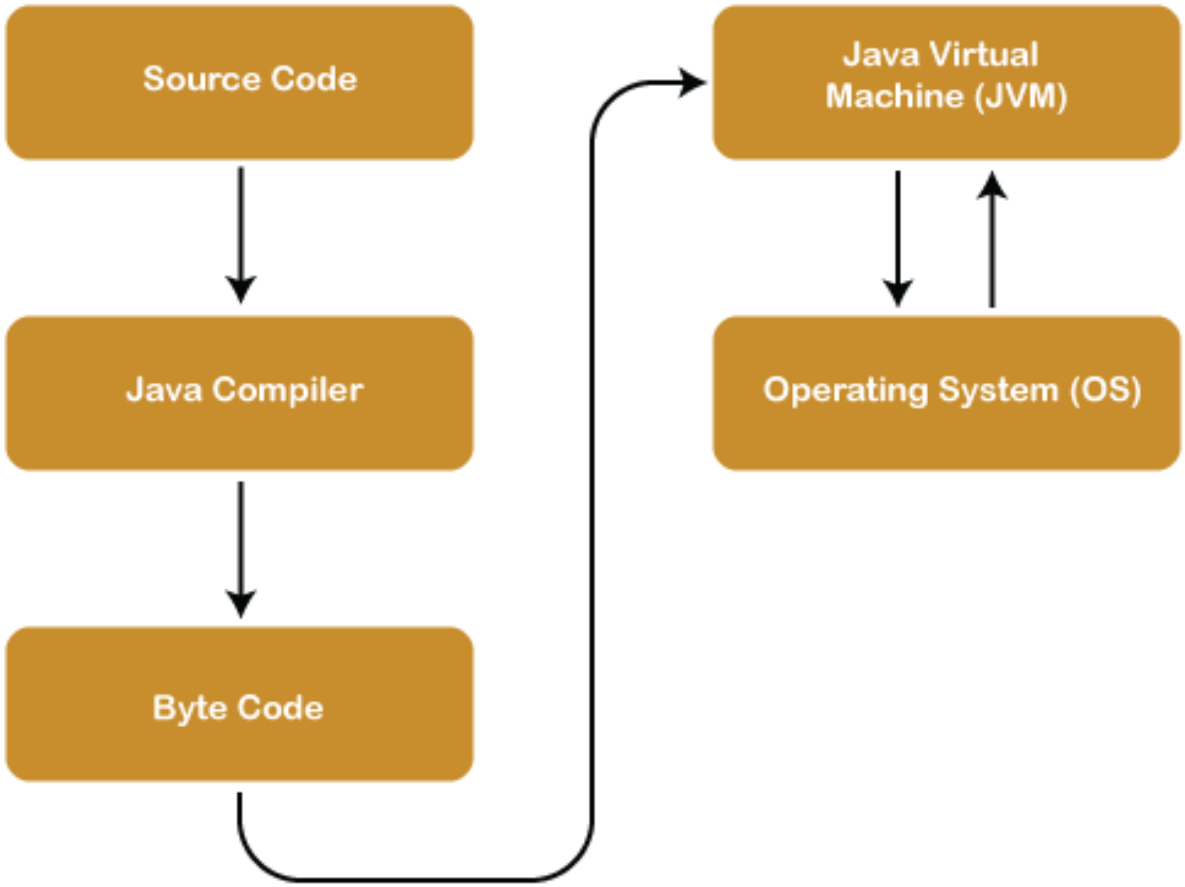
1.5. Working with Files: Byte Stream Classes, Character Stream Classes, Random Access File, Reading and Writing Objects.

Java Architecture

- **Java Architecture** is a collection of components, i.e., **JVM, JRE,** and **JDK**. **It** integrates the process of interpretation and compilation. It defines all the processes involved in creating a Java program. **Java Architecture** explains each and every step of how a program is compiled and executed.

Java Architecture can be explained by using the following steps:

- There is a process of compilation and interpretation in Java.
- Java compiler converts the Java code into byte code.
- After that, the JVM converts the byte code into machine code.
- The machine code is then executed by the machine.


Source Code
Java Compiler
Byte Code
Java Virtual Machine (JVM)
Operating System (OS)

Components of Java Architecture

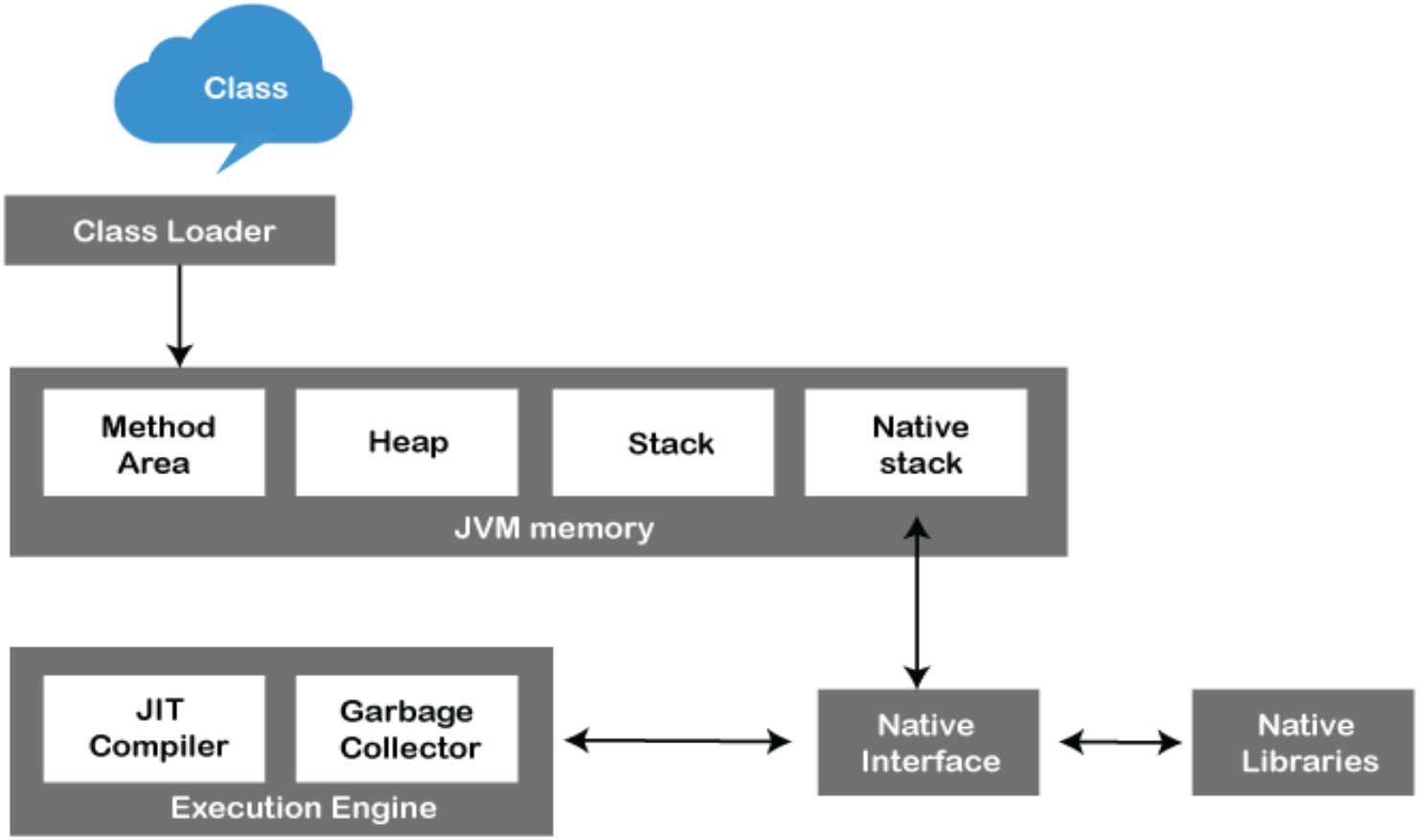
- The Java architecture includes the three main components:
- Java Virtual Machine (JVM)
- Java Runtime Environment (JRE)
- Java Development Kit (JDK)

Java Virtual Machine

- The main feature of Java is **WORA**. WORA stands for **Write Once Run Anywhere**.
- The feature states that we can write our code once and use it anywhere or on any operating system.
- Our Java program can run any of the platforms only because of the Java Virtual Machine.
- It is a Java platform component that gives us an environment to execute java programs.
- JVM's main task is to convert byte code into machine code.
- JVM, first of all, loads the code into memory and verifies it.
- After that, it executes the code and provides a runtime environment.

JVM Architecture

- JVM is an abstract machine that provides the environment in which Java bytecode is executed.



- **Class Loader:** Class Loader is a subsystem used to load class files. Class Loader first loads the Java code whenever we run it.
- **Class Method Area:** In the memory, there is an area where the class data is stored during the code's execution. Class method area holds the information of static variables, static methods, static blocks, and instance methods.
- **Heap:** The heap area is a part of the JVM memory and is created when the JVM starts up. Its size cannot be static because it increase or decrease during the application runs.
- **Stack:** It is also referred to as thread stack. It is created for a single execution thread. The thread uses this area to store the elements like the partial result, local variable, data used for calling method and returns etc.
- **Native Stack:** It contains the information of all the native methods used in our application.

- **Execution Engine:** It is the central part of the JVM. Its main task is to execute the byte code and execute the Java classes. The execution engine has three main components used for executing Java classes.
  - **Interpreter:** It converts the byte code into native code and executes. It sequentially executes the code. The interpreter interprets continuously and even the same method multiple times. This reduces the performance of the system, and to solve this, the JIT compiler is introduced.
  - **JIT Compiler:** JIT compiler is introduced to remove the drawback of the interpreter. It increases the speed of execution and improves performance.
  - **Garbage Collector:** The garbage collector is used to manage the memory, and it is a program written in Java. It works in two phases, i.e., **Mark** and **Sweep**. Mark is an area where the garbage collector identifies the used and unused chunks of memory. The Sweep removes the identified object from the **Mark**
- **Java Native Interface**

  Java Native Interface works as a mediator between Java method calls and native libraries.

Java Runtime Environment

- It provides an environment in which Java programs are executed. JRE takes our Java code, integrates it with the required libraries, and then starts the JVM to execute it.

Java Development Kit

- It is a software development environment used in the development of Java applications and applets.
- Java Development Kit holds JRE, a compiler, an interpreter or loader, and several development tools in it

Java Buzzzwords

- Java is the most popular object-oriented programming language. Java has many advanced features, a list of key features is known as Java Buzz Words.
- The java team has listed the following terms as java buzz words.
  - **Simple**
  - **Secure**
  - **Portable**
  - **Object-oriented**
  - **Robust**
  - **Architecture-neutral (or) Platform Independent**
  - **Multi-threaded**
  - **Interpreted**
  - **High performance**
  - **Distributed**
  - **Dynamic**

Simple

- Java programming language is very simple and easy to learn, understand, and code.
- Most of the syntaxes in java follow basic programming language C and object-oriented programming concepts are similar to C++.
- In a java programming language, many complicated features like pointers, operator overloading, structures, unions, etc. have been removed.
- One of the most useful features is the garbage collector it makes java more simple.

Secure

- Java is said to be more secure programming language because it does not have pointers concept, java provides a feature "applet" which can be embedded into a web application.
- The applet in java does not allow access to other parts of the computer, which keeps away from harmful programs like viruses and unauthorized access.

Portable

- Portability is one of the core features of java which enables the java programs to run on any computer or operating system.
- For example, an applet developed using java runs on a wide variety of CPUs, operating systems, and browsers connected to the Internet.

Object-oriented

- Java is said to be a pure object-oriented programming language.
- In java, everything is an object.
- It supports all the features of the object-oriented programming paradigm.

Robust

- Java is more robust because
  - the java code can be executed on a variety of environments,
  - java has a strong memory management mechanism (garbage collector),
  - java is a strictly typed language, it has a strong set of exception handling mechanism, and many more.

Architecture-neutral (or) Platform Independent

- Java has invented to archive "write once; run anywhere, any time, forever".
- The java provides JVM (Java Virtual Machine) to to archive architectural-neutral or platform-independent.
- The JVM allows the java program created using one operating system can be executed on any other operating system.

Multi-threaded

- Java supports multi-threading programming, which allows us to write programs that do multiple operations simultaneously.

Interpreted

- Java enables the creation of cross-platform programs by compiling into an intermediate representation called Java bytecode.
- The byte code is interpreted to any machine code so that it runs on the native machine.

High performance

- Java provides high performance with the help of features like JVM, interpretation, and its simplicity.

Distributed

- Java programming language supports TCP/IP protocols which enable the java to support the distributed environment of the Internet.
- Java also supports Remote Method Invocation (RMI), this feature enables a program to invoke methods across a network.

Dynamic

- Java is said to be dynamic because the java byte code may be dynamically updated on a running system and it has a dynamic memory allocation and deallocation (objects and garbage collector).

Path and classpath

Path

- PATH is an environment variable that is used to find and locate binary files like “java” and “javac” and to locate needed executables from the command line or Terminal window.
- To set PATH in the window OS.

PATH=C:\Program Files\Java\JDK1.5.10\bin

Classpath

- Classpath is an environment variable that is used by the application ClassLoader or system to locate and load the compiled Java bytecodes stored in the .class file. To set CLASSPATH.
- / To set CLASSPATH in window OS. CLASSPATH=location of class file

Path	Classpath
An environment variable is used by the operating system to find the executable files.	An environment variable is used by the Java compiler to find the path of classes.
PATH setting up an environment for the operating system. Operating System will look in this PATH for executables.	Classpath setting up the environment for Java. Java will use to find compiled classes
Refers to the operating system	Refers to the Developing Environment
In path variable, we must place .\bin folder path	In classpath, we must place .\lib\jar file or directory path in which .java file is available.
PATH is used by CMD prompt to find binary files.	CLASSPATH is used by the compiler and JVM to find library files.

Sample Java Program

```
public class AddNumbers {
   public int add(int x, int y)
   {
      return x+y;
   }
   public static void main(String args[])
   {
      AddNumbers a=new AddNumbers();
      int res=a.add(3,4);
      System.out.println("hello this is CSIT"+res);
   }
}
```