# JAVA INTRODUCTION

## B1. Features of Java (or Java Buzzwords)?

Java is known for its platform independence and versatility. The key features or buzzwords of Java include:
**Simple:** Java was designed to be easy to use and learn.
**Object-Oriented:** Java is based on the object-oriented programming paradigm.
**Distributed:** It enables the development of distributed applications.
**Multithreaded:** Java supports multiple threads of execution, allowing concurrent processing.
**Robust:** It emphasizes compile-time checking and runtime checking to ensure robustness.
**Secure:** Java provides a secure execution environment with features like the sandboxing of applets.
**Architecture-neutral:** Java is designed to be architecture-neutral and portable.
**Dynamic:** Java supports dynamic memory allocation and automatic garbage collection.

## B2. Difference between JDK, JRE, and JVM?

**JDK (Java Development Kit):** It is a software development kit that includes the necessary tools for Java development, such as the compiler and debugger.
**JRE (Java Runtime Environment):** It provides the runtime environment for executing Java applications but does not include development tools.
**JVM (Java Virtual Machine):** It is an abstract machine that provides an execution environment for Java bytecode. JVM is included in both JDK and JRE.

## B3. Java is Platform Independent?

Java achieves platform independence through the "Write Once, Run Anywhere" principle. Java source code is compiled into bytecode, which is platform-neutral and can be executed on any device with a Java Virtual Machine (JVM).

## B4. Three Flavors of Java?

**Java SE (Standard Edition):** For building desktop applications and applets.
**Java EE (Enterprise Edition):** For building enterprise-level applications.
**Java ME (Micro Edition):** For building applications on small devices like mobile phones.

### B5. Types of Memory Areas Allocated by JVM?

**Heap Memory:** For objects and JRE classes.
**Stack Memory:** For local variables and method call information.
**Method Area:** For class-level structures (like static fields).

### B6. Latest Version of Java?

The latest version of Java is Java 21 or JDK 21 released on September, 19th 2023.

### B7. Write Once, Run Anywhere (WORA)?

It is a principle in Java that emphasizes the portability of Java programs. Once a Java program is written and compiled into bytecode, it can run on any device with a Java Virtual Machine (JVM), regardless of the underlying hardware and operating system.

### B8. Is Java a Pure/Fully Object-Oriented Language?

Java is not considered a fully object-oriented language because it supports primitive data types (int, float, etc.) that are not objects. However, everything in Java is treated as an object except for these primitive data types.

### B9. Bytecode?

Bytecode is an intermediate code generated by the Java compiler. It is platform-independent and can be executed by any Java Virtual Machine (JVM). Java source code is compiled into bytecode before it is executed.

### B10. Heap Space in Java?

Heap space is the memory area in Java where objects are allocated. It is used for dynamic memory allocation during the runtime of a program. The garbage collector manages the heap space by reclaiming memory occupied by objects that are no longer referenced.

### B11. Difference between EAR, JAR, and WAR files in J2EE?

**EAR (Enterprise Archive):** It is used for packaging J2EE applications, including JAR files, WAR files, and other resources.
**JAR (Java Archive):** It is used for aggregating and compressing Java class files, associated metadata, and resources into a single file.
**WAR (Web Archive):** It is used for packaging web applications, including servlets, JSP pages, HTML files, and other resources.

## B12. Memory Leak in Java?

A memory leak in Java occurs when objects are no longer needed by the program but are not released by the garbage collector. This leads to the gradual consumption of memory, potentially causing performance issues and eventual program failure.

## B13. How Garbage Collection Works in Java?

Garbage collection in Java involves identifying and reclaiming memory occupied by objects that are no longer reachable or referenced by the program. The Java Virtual Machine's garbage collector automatically manages memory by deallocating unreferenced objects.

### B14. Does Java Garbage Collector Clean Both Heap and Stack Memory?

No, the Java garbage collector primarily cleans the heap memory, which is used for dynamic memory allocation. Stack memory, which holds local variables and method call information, is managed differently. The memory allocated on the stack is automatically released when a method completes its execution.

## B15. Why Garbage Collection is Required in Java?

Garbage collection is required in Java to automatically reclaim memory occupied by objects that are no longer needed, preventing memory leaks and ensuring efficient memory utilization. It simplifies memory management for developers by automating the process of deallocating memory for unreferenced objects.