**Week 1 Lecture 1: Introduction to IDE, Introduction to Java and its Components**

### Introduction to Integrated Development Environment (IDE)

* **Definition**: An Integrated Development Environment (IDE) is a software application that provides comprehensive facilities to programmers for software development.
* **Importance in Java Programming**:
  + Code editing with syntax highlighting and auto-completion.
  + Integrated compiler and debugger.
  + Streamlined project management and build automation.
  + Example IDEs: Eclipse, IntelliJ IDEA, NetBeans.

### Introduction to Java

* **Purpose**: Java is a high-level, object-oriented programming language used for building secure, portable, and scalable applications.
* **Design Philosophy**:
  + Write Once, Run Anywhere (WORA)
  + Object-Oriented Programming (OOP)
  + Robust and Secure
* **Significance in Modern Software Development**:
  + Used in enterprise applications, web development, mobile applications, and IoT.

### Main Components of Java

* **Java Virtual Machine (JVM)**: Converts bytecode into machine-specific code.
* **Java Runtime Environment (JRE)**: Provides libraries and JVM for running Java applications.
* **Java Development Kit (JDK)**: Includes JRE and development tools like compiler and debugger.

**Week 1 Lecture 2: Brief History of Java, Features of Java, Compilation Process, Types of Java Applications, Java Development Kit, Java Editions, and Java Development Tools**

### Brief History of Java

* Developed by Sun Microsystems in 1995, now maintained by Oracle.
* Major releases: Java 1.0 (1996) to Java 17 (2021+)

### Features/Characteristics of Java

* **Platform Independence**: Runs on any OS with JVM.
* **Object-Oriented**: Follows principles like inheritance, encapsulation.
* **Secure**: Provides security features like bytecode verification.
* **Multi-threading**: Supports concurrent execution.

### Java Compilation Process

* **Steps**:
  1. Write source code (.java file)
  2. Compile using javac to generate bytecode (.class file)
  3. Execute using java command (JVM interprets bytecode)

### Types of Java Applications

* **Standalone Applications**: Desktop software (e.g., calculators, media players)
* **Web Applications**: Servlets, JSP-based applications
* **Mobile Applications**: Android apps (Java/Kotlin)

### Java Development Kit (JDK)

* Includes compiler (javac), debugger, libraries, and JVM.
* Required for developing and running Java applications.

### Java Editions

* **Java SE** (Standard Edition): Core Java programming.
* **Java EE** (Enterprise Edition): Web and enterprise applications.
* **Java ME** (Micro Edition): Mobile and embedded devices.

### Java Development Tools

* **Eclipse**: Open-source IDE with powerful plugins.
* **IntelliJ IDEA**: Smart IDE for Java developers.
* **NetBeans**: Beginner-friendly IDE with GUI builder.

**Week 2 Lecture 3: Difference between JRE, JDK, JVM, and JIT, Java Execution Flow**

### Difference between JRE, JDK, JVM, and JIT

| **Component** | **Description** |
| --- | --- |
| JVM | Abstract machine that executes Java bytecode |
| JRE | Runtime environment with JVM and libraries |
| JDK | Full development kit with compiler, debugger, and JRE |
| JIT Compiler | Converts bytecode into native machine code at runtime |

### Java Execution Flow

1. **Source Code**: Written in .java file.
2. **Compilation**: Translated into .class file (bytecode).
3. **Class Loading**: Bytecode loaded into JVM.
4. **Bytecode Verification**: Ensures secure execution.
5. **Interpretation/Compilation**: JIT compiles bytecode to native code.
6. **Execution**: Runs on target machine.

**Week 2 Lecture 4: Compiling and Execution of Java Program through Command Prompt, Path and CLASSPATH, Anatomy of Java Program**

### Compiling and Executing Java Program via Command Prompt

* **Steps**:
  1. Write Java code in a text editor.
  2. Save file as Program.java.
  3. Open Command Prompt and navigate to file location.
  4. Compile: javac Program.java
  5. Execute: java Program

### Path and CLASSPATH in Java

* **Path**: System variable pointing to JDK binaries (javac, java).
  + Set via: set PATH=C:\Program Files\Java\jdk\bin;%PATH%
* **CLASSPATH**: Specifies location of Java class files.
  + Example: set CLASSPATH=.;C:\MyJavaPrograms;

### Anatomy of a Java Program

public class HelloWorld {

public static void main(String[] args) {

System.out.println("Hello, World!");

}

}

* **Class Declaration**: public class HelloWorld.
* **Main Method**: public static void main(String[] args).
* **Statement Execution**: System.out.println("Hello, World!"); prints output.