INTRODUCTION TO JAVA

Overview of Java Programming Language

- **Name :-** Komal sandip Palande
- **Roll no:-** 4236
- **Class:-**TYBCA
- **College Name :-** Bharitya Jain Sanghata Wagholi
- **Subject:-** Java
- **Title:-** INTRODUCTION TO JAVA
- **Subtitle:-**Overview of Java Programming Language
- **Teacher Name:** Mr.Manvatkar Sir

What is Java?

- **A high-level, class-based, object-oriented programming language.**
-
- **Designed to have as few implementation dependencies as possible.**
-
- **Write Once, Run Anywhere (WORA): Compiled Java code can run on all platforms that support Java.**

Key Features of Java

- **Platform Independent**: Java programs can run on different operating systems without modification.
-
- **Object-Oriented**: Encourages reusable, modular code through classes and objects.
-
- **Robust**: Automatic memory management (garbage collection) and strong type checking.
-
- **Multithreading**: Java supports concurrent programming, making efficient use of resources.

Java Architecture

- **Java Compile**r: Converts Java code into bytecode.
-
- **Java Virtual Machine (JVM)**: Executes bytecode on any platform.
-
- **Java Runtime Environment (JRE)**: Contains JVM and libraries needed to run Java programs.
-
- **Java Development Kit (JDK)**: Includes JRE, tools, and libraries for development.

Java Syntax Basics

- **Class Declaration**: public class Example { }
-
- **Main Method**: public static void main(String[] args) { }
-
- **Variables**: int, double, String
-
- **Control Structures**: if-else, for, while

Object-Oriented Programming in Java

- **Classes**: Blueprints for creating objects.
-
- **Objects**: Instances of classes.
-
- **Inheritance**: Enables code reuse by inheriting attributes and methods.
-
- **Polymorphism**: Allows one interface to be used for different types of actions.
-
- **Encapsulation**: Protects the state of objects by restricting access to variables.
-
- **Abstraction**: Hides complex implementation details and exposes only functionality.

Java Libraries and APIs

- **Core Libraries**: java.lang, java.util, java.io
-
- **Common APIs**:
-
- **Collections Framework**: For handling groups of objects.
-
- **Streams API**: For processing sequences of data.
-
- **Networking**: java.net for network communication.
-
- **Concurrency**: java.util.concurrent for multi-threading.

Java Development Tools

- **Integrated Development Environments (IDEs)**: Eclipse, IntelliJ IDEA, NetBeans.
-
- **Build Tools**: Maven, Gradle, Ant.
-
- **Version Control**: Git for source code management.
-
- **Testing Frameworks**: Junit for unit testing.

Applications of Java

- **Web Applications**: Using Java EE (Jakarta EE), Spring.
-
- **Mobile Applications**: Android development.
-
- **Desktop Applications**: JavaFX, Swing.
-
- **Enterprise Applications**: Large-scale, distributed systems using frameworks like Spring or Hibernate.
-
- **Game Development**: Game engines like LibGDX.

Conclusion

- **Summary**: Java’s portability, object-oriented nature, and vast ecosystem make it ideal for a wide range of applications.