**Java Basics**

**1️Structure of a Java Program**

**Java programs are built using classes and methods. The entry point is always the main method.**

**public class HelloWorld {**

**public static void main(String[] args) {**

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

**}**

**}**

* **Every file should match the name of its public class**
* **Code is organized into reusable blocks (methods)**

**2️JDK vs JRE vs JVM**

| **Component** | **Description** |
| --- | --- |
| **JDK (Java Development Kit)** | **Tools for developing Java apps (includes compiler javac)** |
| **JRE (Java Runtime Environment)** | **Environment to run Java apps (includes JVM + libraries)** |
| **JVM (Java Virtual Machine)** | **Executes bytecode and enables platform independence** |

**3️Scanner Methods: next() vs nextLine()**

| **Method** | **Behavior** |
| --- | --- |
| **next()** | **Reads input until a space; good for single words** |
| **nextLine()** | **Reads the full line including spaces** |

**Use nextLine() when reading full sentences or multi-word input.**

**4️Arithmetic Operations Program**

**import java.util.Scanner;**

**public class ArithmeticDemo {**

**public static void main(String[] args) {**

**Scanner sc = new Scanner(System.in);**

**int a = sc.nextInt();**

**int b = sc.nextInt();**

**System.out.println("Addition: " + (a + b));**

**System.out.println("Subtraction: " + (a - b));**

**System.out.println("Multiplication: " + (a \* b));**

**System.out.println("Division: " + (a / b));**

**}**

**}**

**5️ Leap Year Check**

**public class LeapYear {**

**public static void main(String[] args) {**

**int year = 2024;**

**if ((year % 400 == 0) || (year % 4 == 0 && year % 100 != 0)) {**

**System.out.println(year + " is a Leap Year");**

**} else {**

**System.out.println(year + " is not a Leap Year");**

**}**

**}**

**}**

**6️ Student Result Calculation**

**public class StudentResult {**

**public static void main(String[] args) {**

**int marks1 = 80, marks2 = 75, marks3 = 90;**

**int total = marks1 + marks2 + marks3;**

**double percentage = total / 3.0;**

**System.out.println("Total: " + total);**

**System.out.println("Percentage: " + percentage);**

**}**

**}**

**7️ Odd or Even Checker**

**public class OddEven {**

**public static void main(String[] args) {**

**int num = 10;**

**if (num % 2 == 0) {**

**System.out.println(num + " is Even");**

**} else {**

**System.out.println(num + " is Odd");**

**}**

**}**

**}**

**8️ Exception Handling in Java**

**Java uses try-catch blocks to handle runtime errors gracefully.**

**try {**

**int result = 10 / 0;**

**} catch (ArithmeticException e) {**

**System.out.println("Cannot divide by zero");**

**}**

**Types of Exceptions:**

* **Checked: Must be handled (e.g., IOException)**
* **Unchecked: Runtime errors (e.g., NullPointerException)**
* **Errors: Serious issues (e.g., OutOfMemoryError)**

**9️ Object-Oriented Programming (OOP) Concepts**

| **Concept** | **Description** |
| --- | --- |
| **Encapsulation** | **Bundling data and methods in a class** |
| **Inheritance** | **Reusing properties from a parent class** |
| **Polymorphism** | **Multiple forms (method overloading/overriding)** |
| **Abstraction** | **Hiding internal details, exposing essentials** |

**10 Java Collections Framework**

**Java Collections help manage groups of objects efficiently.**

| **Collection** | **Description** |
| --- | --- |
| **List** | **Ordered, allows duplicates (e.g., ArrayList)** |
| **Stack** | **LIFO structure for last-in-first-out operations** |
| **Queue** | **FIFO structure for task scheduling** |
| **Vector** | **Synchronized version of ArrayList (legacy)** |