**Day 1: Introduction to Java**

* History and Features of Java
* JDK, JRE, and JVM
* Setting up the development environment (Installation and setup)
* Writing your first Java program
* Java Syntax Basics

**Day 2: Data Types and Variables**

* Primitive and Non-Primitive Data Types
* Variables and Constants
* Type Casting and Type Conversion
* Unicode and Escape Sequences
* Java Naming Conventions

**Day 3: Operators in Java**

* Arithmetic, Relational, Logical, and Bitwise Operators
* Assignment and Compound Operators
* Increment/Decrement Operators
* Operator Precedence and Associativity
* Ternary Operator

**Day 4: Control Statements**

* If-else, Switch-case
* Loops (For, While, Do-While)
* Break, Continue, and Return Statements
* Nested Loops

**Day 5: Arrays**

* Declaration and Initialization
* Single-dimensional Arrays
* Multi-dimensional Arrays
* Enhanced for Loop
* Array Utility Methods (Arrays class)

**Day 6: Strings**

* String Class and Methods
* StringBuffer and StringBuilder
* Immutability of Strings
* String Operations and Comparisons
* StringTokenizer Class

**Day 7: Object-Oriented Programming (OOP) Basics**

* Class and Objects
* Constructors (Default, Parameterized)
* Instance Variables and Methods
* The this keyword
* Method Overloading

**Day 8: Inheritance**

* Basics of Inheritance
* Types of Inheritance (Single, Multilevel, Hierarchical)
* Super Keyword
* Method Overriding
* Runtime Polymorphism

**Day 9: Encapsulation and Abstraction**

* Access Modifiers (Public, Private, Protected, Default)
* Getters and Setters
* Abstract Classes and Methods
* Interfaces
* Differences between Abstract Class and Interface

**Day 10: Packages and Access Modifiers**

* Creating and Using Packages
* Importing Classes and Packages
* Static Imports
* Access Levels (Private, Default, Protected, Public)

**Day 11: Exception Handling**

* Types of Exceptions (Checked and Unchecked)
* Try, Catch, Finally
* Throw and Throws
* Custom Exceptions
* Multi-catch Blocks

**Day 12: Multithreading**

* Thread Life Cycle
* Creating Threads (Extending Thread Class, Implementing Runnable Interface)
* Synchronization
* Inter-thread Communication
* Thread Priority and Daemon Threads

**Day 13: Collections Framework (Part 1)**

* Introduction to Collections
* List Interface (ArrayList, LinkedList)
* Set Interface (HashSet, LinkedHashSet, TreeSet)
* Iterator and ListIterator

**Day 14: Collections Framework (Part 2)**

* Map Interface (HashMap, LinkedHashMap, TreeMap)
* Queue Interface (PriorityQueue, Deque)
* Comparator and Comparable Interfaces
* Collections Utility Methods

**Day 15: File Handling**

* Reading and Writing Files
* FileReader and FileWriter
* BufferedReader and BufferedWriter
* Serialization and Deserialization
* File Class and Methods

**Day 16: Java 8 Features**

* Lambda Expressions
* Functional Interfaces
* Stream API
* Default and Static Methods in Interfaces
* Optional Class

**Day 17: JDBC (Java Database Connectivity)**

* JDBC Architecture
* Setting Up MySQL with Java
* CRUD Operations using JDBC
* Statement, PreparedStatement, CallableStatement
* ResultSet and ResultSetMetaData

**Day 18: Advanced Topics**

* Generics
* Annotations
* Enum Types
* Reflection API
* Regular Expressions

**Day 19: Design Patterns in Java**

* Singleton Pattern
* Factory Pattern
* Observer Pattern
* Builder Pattern
* MVC Design Pattern

**Day 20: Revision and Practice**

* Review all topics covered
* Solve Java-based coding problems
* Work on a small project (e.g., Library Management System, Student Management System)
* Clarify any remaining doubts or revisit tough topics

**1. What is Java?**

* **Definition**:  
  Java is a high-level, object-oriented, and secure programming language.  
  It is also a platform, meaning it provides an environment (JRE) for programs to run.
* **Origin**:
  + Developed by Sun Microsystems in 1995.
  + James Gosling, known as the "Father of Java," originally named it **Oak**.
  + Later renamed to **Java**, inspired by Java coffee beans.
* **Platform Explanation**:
  + A platform is a hardware/software environment where programs run.
  + Java is platform-independent, thanks to the JVM (Java Virtual Machine).

**2. Writing Your First Java Program**

**Example: A Simple Java Program**

class Simple {

public static void main(String[] args) {

System.out.println("Hello Java");

}

}

**Explanation**:

* + class Simple: Defines a class named Simple.
  + public static void main(String[] args): Entry point of the program.
  + System.out.println("Hello Java");: Prints "Hello Java" to the console.

**3. History of Java**

* **1991**: Started as a project by the Green Team at Sun Microsystems.
* **Initial Purpose**: Designed for digital devices like TVs and set-top boxes.
* **Evolution**: From "Greentalk" to "Oak," and finally named "Java" in 1995.
* **Current Use**:
  + Internet programming.
  + Mobile applications.
  + Enterprise systems and more.

**4. Features of Java**

**Key Features:**

1. **Simple**:
   * Syntax based on C++.
   * No explicit pointers or operator overloading.
   * Automatic garbage collection.
2. **Object-Oriented**:
   * Based on OOP principles: Class, Object, Inheritance, Polymorphism, Abstraction, Encapsulation.
3. **Platform Independent**:
   * Write Once, Run Anywhere (WORA).
   * Java code is compiled into bytecode, which runs on any platform with JVM.
4. **Secure**:
   * No explicit pointers, ensuring memory safety.
   * Bytecode verification and runtime security checks.

**5. JDK, JRE, and JVM**

* **JDK (Java Development Kit)**:
  + Contains tools like compiler, debugger, and libraries to develop Java applications.
* **JRE (Java Runtime Environment)**:
  + Provides libraries and JVM to run Java programs.
* **JVM (Java Virtual Machine)**:
  + Converts bytecode into machine-specific instructions.
  + Enables platform independence.

**6. Types of Java Applications**

1. **Standalone Applications**:
   * Desktop software like Media Player, Antivirus.
   * Built using AWT and Swing libraries.
2. **Web Applications**:
   * Server-side programs generating dynamic web pages.
   * Technologies: Servlets, JSP, Spring, Hibernate.
3. **Enterprise Applications**:
   * Distributed systems like banking apps.
   * Features: High security, scalability.
4. **Mobile Applications**:
   * Built for devices using Android SDK or Java ME.

**7. Java Editions**

1. **Java SE (Standard Edition)**: Core programming APIs (OOP, Collections, Multithreading).
2. **Java EE (Enterprise Edition)**: Tools for web and enterprise apps (JSP, Servlets).
3. **Java ME (Micro Edition)**: Focused on mobile and embedded devices.
4. **JavaFX**: Builds rich internet applications with a graphical user interface.

**8. Setting Up the Development Environment**

**Steps:**

1. Download the JDK from Oracle's official website.
2. Install and set the environment variable JAVA\_HOME.
3. Verify installation using the command: java -version
4. Use an IDE like Eclipse, IntelliJ IDEA, or Visual Studio Code for development.