✅ OOP Part 1: Intro & Concepts – Class, Object, Constructor, Keywords

🔸 Introduction & Notes

- Teaser & Overview
- Importance of OOP in Java
- Notes on learning approach

🔸 OOP Basics

- What is OOP?
- Key Concepts of OOP
- Why OOP is important

🔸 Java Classes & Objects

- Introduction to Java Classes
- Example of a Java Class
- What is an Object in Java?
- Difference between Class and Object
- Properties of an Object

🔸 Working with Objects

- Object Declaration and Initialization
- Accessing Instance Variables
- Creating Objects in Different Ways
- Dynamic Memory Allocation in Java
- Manipulating Object State

🔸 Java Constructors

- What is a Constructor?
- Default Constructor
- Creating Custom Constructors

- Use of `this` Keyword in Constructor
- Constructor Overloading
- Calling One Constructor from Another (Constructor Chaining)

🔸 Keywords & Concepts

- Why `new` is not used with Primitive Types
- How `new` Keyword Allocates Memory
- Wrapper Classes (int → Integer, etc.)
- `final` Keyword Usage
- Java Garbage Collection

✅ OOP Part 2: Packages, Static, Singleton Class

🔸 Java Packages

- Real-life Example of Packages
- What are Java Packages?
- Types: Built-in and User-defined
- Using `import` Statement

🔸 Static Elements

- Static Variables, Methods, and Blocks
- Difference Between Static and Non-static Members
- Calling Non-static inside Static
- Calling Static inside Non-static
- Using `this` inside Static Context
- Static Initialization

🔸 Inner Classes & Singleton

- Introduction to Inner Classes

- How Java Internally Handles Statements
- Singleton Class: What & Why

✅ OOP Part 3: Principles – Inheritance, Polymorphism, Encapsulation, Abstraction

🔸 Core OOP Principles

- Overview of the 4 OOP Pillars

🔸 Inheritance

- What is Inheritance?
- Example: Box Class
- Explanation of Reuse
- Access Modifiers like `private`
- `super` Keyword
- Types of Inheritance
  - Single
  - Multiple (via interfaces)
  - Hierarchical
  - Hybrid

🔸 Polymorphism

- Introduction to Polymorphism
- Example: Shapes
- Static Polymorphism (Method Overloading)
- Dynamic Polymorphism (Method Overriding)
- How Overriding Works Internally
- Method Resolution by JVM
- Use of `final` in Overriding

- Can Static Methods be Overridden?

🔸 Encapsulation & Abstraction

- What is Encapsulation?
- What is Abstraction?
- Difference Between Them
- Real-life Example
- Data Hiding

✅ OOP Part 4: Access Control, Java Packages, Object Class

🔸 Access Modifiers

- `private`, `public`, `protected`, and Default
- Rules and Usage Scenarios
- When to Use Which Modifier
- Key Notes on `protected`

🔸 Java In-built Packages

- Overview of Java Built-in Packages
- Key Packages:
    - `lang`
    - `io`
    - `util`
    - `applet`
    - `awt`
    - `net`

🔸 Object Class & Its Methods

- What is the Object Class
- Methods:
    - `hashCode()`
    - `equals()`
    - `instanceof`
    - `getClass()`

✅ OOP Part 5: Abstract Classes, Interfaces, Annotations

🔸 Abstract Classes

- Need for Abstraction
- Defining Abstract Classes and Methods
- Abstract Constructors
- Can Abstract Classes Have Objects?
- Static Methods in Abstract Classes
- `final` with Abstract Classes
- Multiple Inheritance with Abstract Classes

🔸 Interfaces

- What are Interfaces?
- Interface Example
- Interface as a Data Type
- Multiple Classes Implementing Same Interface
- Extending Interfaces
- Static Methods in Interfaces
- Nested Interfaces

🔸 Annotations

- Introduction to Java Annotations
- Use in Modern Java
- Special Notes on Static Interface Methods

✅ OOP Part 6: Generics, Custom ArrayList, Lambda, Exception Handling, Cloning

🔸 Custom ArrayList

- Building Your Own ArrayList
- Limitations of Non-generic List

🔸 Generics

- Using Generics in Java
- Java Wildcards
- Generic Comparison

🔸 Lambda Expressions

- Introduction to Lambda Functions
- Syntax and Use Cases

🔸 Exception Handling

- Exception Handling Mechanism
- Keywords: `try`, `catch`, `finally`, `throw`, `throws`
- Creating Custom Exceptions

🔸 Object Cloning

- What is Cloning?
- Shallow vs Deep Copy
- Real-life Application of Cloning

✅ OOP Part 7: Collections, Vector, Enums

🔸 Collection Framework

- What is Collection Framework?
- Need for Collection Framework
- Core Interfaces and Hierarchy

🔸 Vector Class

- What is Vector?
- Vector Synchronization
- Vector Example Code

🔸 Enums

- Enum Definition and Usage
- Enum Inheritance
- Best Practices with Enums
