**✅ Basic Theory Questions**

1. **What is Object-Oriented Programming (OOP)?**

OOP stands for **Object-Oriented Programming**. It's a programming paradigm based on the concept of **"objects"**, which are instances of **classes**.

In OOP, we try to model real-world entities using code by representing them as objects that have:

* **State** (through variables or attributes)
* **Behavior** (through methods or functions)

1. **What are the four pillars of OOP?**

**1. Encapsulation**

It means **hiding the internal details** of an object and only exposing what is necessary through public methods.  
We use **private variables** and **public getters/setters** to achieve this.

✅ *Example:* A class Student has private marks and provides getMarks() to access it.

**2. Inheritance**

It allows one class (**child/subclass**) to inherit properties and behaviors from another class (**parent/superclass**).  
This promotes **code reusability** and a clear hierarchy.

✅ *Example:* class Car extends Vehicle

**3. Polymorphism**

It means **many forms**. It allows methods to behave differently based on the object or input.

* **Compile-time polymorphism** → Method overloading
* **Runtime polymorphism** → Method overriding

✅ *Example:* draw() method behaves differently in Circle and Rectangle classes.

**4. Abstraction**

It means showing only the **essential details** and hiding the rest.  
We achieve abstraction using **abstract classes** or **interfaces**.

✅ *Example:* We use List interface but don’t worry about whether it’s an ArrayList or LinkedList.

1. **What is a class and an object in Java?**
2. **What is the difference between a class and an object**
3. **What is the need for OOP?**
4. **What is a constructor? Types of constructors?**
5. **What is constructor overloading?**

**Constructor overloading** means having **multiple constructors** in the same class with **different parameter lists**.

1. **What is the difference between method overloading and method overriding?**
2. **What is this keyword?**
3. **What is the use of super keyword?**

**✅ Intermediate Theory Questions**

1. What is inheritance in Java? Types of inheritance supported?
2. What is polymorphism? Static vs dynamic?
3. What is method overriding? What are the rules?
4. What is abstraction? How is it achieved in Java?
5. What is an interface? How is it different from abstract class?
6. What is encapsulation? How is it implemented?
7. What is upcasting and downcasting in Java?
8. Can you override a static method? Why/why not?
9. What is the difference between == and .equals()?
10. What is final class, final method, final variable?

**✅ Advanced Theory Questions**

1. What is the difference between composition and inheritance?
2. What is the difference between abstract class and interface in Java 8+?
3. What is multiple inheritance? Does Java support it?
4. Why do we use interfaces instead of abstract classes?
5. Explain the SOLID principles.
6. What is cohesion and coupling in OOP?
7. What are default and static methods in interfaces (Java 8+)?
8. What are anonymous classes in Java?
9. What is the diamond problem in OOP?
10. How does Java handle memory management for objects?

**✅ Coding Questions (Easy to Hard)**

1. Create a class Student with name, age, and marks. Print student details.
2. Write a Java program to demonstrate constructor overloading.
3. Create a base class Vehicle and subclass Car to demonstrate inheritance.
4. Write a program to show method overriding using Animal and Dog classes.
5. Implement an abstract class Shape with area() method and two subclasses.
6. Create an interface Flyable and implement it in two different classes.
7. Write a program to demonstrate runtime polymorphism.
8. Demonstrate encapsulation using private variables and public setters/getters.
9. Implement upcasting and downcasting between Person and Employee.
10. Create a simple Java program following all 4 pillars of OOPS.