
https://www.youtube.com/DebugWithShubham

https://instagram.com/debugwithshubham

https://www.linkedin.com/in/debugwithshubham/

https://topmate.io/debugwithshubham

https://web.telegram.org/k/#@debugwithshubham


# Cognizant GenC 2025

# OOPs

## Interview Question

**Q1. What is OOPs? Why do we use it?**
**Answer:**
OOPs stands for Object-Oriented Programming System.
Isme hum code ko objects ke form me organize karte hain jisse code reusable secure aur easy to manage ho jata.

**Why we use OOPs?**

-Code reuse hota hai

-Maintenance easy hota hai

-Real-world modeling possible hoti hai

-Large applications easy ban jaati hain

**Example:**
Car ek object hai, jisme properties (color, speed) aur functions (drive, brake) hote hain.

**Q2. What is a Class and Object?**
**Answer:**

**Class** → Blueprint / design

**Object** → Real instance of class

**Example:**

**Class**: Car
**Object**: BMW, Audi

Class bina object ke kaam nahi karti, aur object bina class ke exist nahi karta.

**Q3. What are the four pillars of OOPs?**
**Answer:**

OOPs ke 4 pillars hote hain:

Encapsulation
Abstraction
Inheritance
Polymorphism

**Q4. What is Encapsulation?**
**Answer**:

Encapsulaka matlab hai data + methods ko ek unit me bind karna aur direct access se protect karna.

**Example**:
Private variables + public methods

**Why important?**

Data security

Controlled access

**Real life:**
ATM machine  aap balance dekh sakte ho, lekin backend data change nahi kar sakte.

**Q5. What is Abstraction?**
**Answer**:

Abstraction ka matlab hai internal implementation hide karna aur sirf necessary cheeze dikhana.

**Example**:
Car chalate waqt hume engine ka logic nahi pata hota.

**Benefit**:

Complexity kam hoti hai

User ko sirf required cheeze dikhti hain

**Q6. What is Inheritance?**
**Answer**:

Inheritance me ek class dusri class ke properties inherit karti hai.

**Example:**

Parent: Vehicle
Child: Car, Bike

**Benefits**:

Code reuse

Less duplication

## Q7. What is Polymorphism?

**Answer**:

Polymorphism ka matlab hai same function name, different behavior.

**Types**:

Compile-time (Method Overloading)
Runtime (Method Overriding)

**Example**:

draw() → circle
draw()→ rectangle

## Q8. Difference between Method Overloading and Overriding

| Overloading | Overriding |
| --- | --- |
| Same class | Parent–Child class |
| Compile-time | Runtime |
| Different parameters | Same method signature |

**One-liner:**

Overloading = same method, different parameters
Overriding = same method, different implementation

## Q9. What is a Constructor?

**Answer**:

Constructor ek special method hota hai jo object creation ke time automatically call hota hai.

**Types**:

Default constructor

Parameterized constructor

**Purpose**:

Initialize variables

## Q10. Difference between Interface and Abstract Class

| Abstract Class | Interface |
| --- | --- |
| Can have methods with body | Only method declaration |
| Supports constructor | No constructor |
| Multiple inheritance not allowed | Multiple inheritance allowed |

## Q11. What is Encapsulation vs Abstraction?

Encapsulation → data security

Abstraction → hide complexity

## Q12. Can we achieve multiple inheritance in Java?

**Answer**:

No, Java does not support multiple inheritance using classes, but it is achieved using interfaces