URBAN EDGE

Chapter - 8 : Introduction to OOPs

Object Oriented programming tries to map code instructions with real world making the code short and easier to understand

What is Object Oriented Programming

Solving a problem by creating objects is one of the most popular approaches in programming. This is called Object Oriented Programming.

What is DRY?

DRY stands for - Do not repeat yourself
↳ Focuses on code reusability

Class

A class is a blueprint for creating objects.

JEE Application form ⇒ Filled by an Student ⇒ Application for that Student

Class ⇒ Object Instantiation ⇒ Object

Contains info to create a valid object.

Object

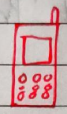
An Object is an instantiation of a class. When a class is defined, a template (info) is defined. Memory is allocated only after object instantiation.

How to model a problem in OOPs
We identify the following:

Noun → Class → Employee
Adjective → Attributes → name, age, salary
Verb → Methods → getSalary(), increment()

OOPs Terminology

1. Abstraction → Hiding internal details [show only essential info!]



⇒ use this phone without bothering about how it was made

2. Encapsulation → The act of putting various components together (in a capsule).

⇒ Laptop is a single entity with wifi + speaker + storage in a single box!

In Java, encapsulation simply means that the sensitive data can be hidden from the users

3. Inheritance → The act of deriving new things from existing things.

Rickshaw ⇒ E-Rickshaw
Phone ⇒ Smart Phone

Implements DRY!

4. Polymorphism → One entity many forms

Smartphone → Phone Smartphone → Calculator

URBAN EDGE

Writing a Custom Class

We can write a custom class as follows:

```
public class Employee {
    int id;          → Attribute 1
    String name;     → Attribute 2
}
```

Any real world object = Properties + Behaviour
Object in OOPs = Attributes + Methods.

A class with Methods

We can add methods to our class Employee as follows:

```
public class Employee {
    public int id;
    public String name;

    public int getSalary() {
        // code
    }
    public void getDetails() {
        // code
    }
};
```