Object-Oriented Programming in Java

1. Introduction to OOP:

OOP helps organize software design using real-world concepts like objects and classes.

2. Principles of OOP:

- Encapsulation: Hiding internal state using access modifiers.
- Inheritance: Code reuse from parent classes.
- Polymorphism: Multiple forms (method overloading/overriding).
- Abstraction: Hiding complexity using interfaces and abstract classes.

3. Class and Object:

Example:

```
class Car {
    String color;
    void drive() {
        System.out.println("Driving");
    }
}
```

```
Car myCar = new Car();
```

4. Constructors and Methods:

Constructors initialize objects.

Methods define behavior.

Example:

```
public Car(String model) {
    this.model = model;
}
```

5. Access Modifiers:

- public

- private
- protected
- default

6. Practical Use:

Design systems like Bank Accounts, Library Management, etc., using OOP.

Summary:

OOP makes Java scalable and maintainable. Mastering these concepts is key to writing clean and efficient Java code.