Python Object-Oriented Programming (OOPs) – Detailed Notes

🐍 Python Object-Oriented Programming (OOPs) – Detailed Notes

1. Introduction to OOP

Object-Oriented Programming (OOP) organizes code around objects.

Objects represent real-world entities and contain data (attributes) and behavior (methods).

Benefits:

- Reusability

- Encapsulation

- Abstraction

- Modularity

- Maintainability

2. Basic OOP Terminology

Class – Blueprint for objects

Object – Instance of class

Method – Function inside class

Attribute – Variable inside class

3. Class and Object Example

class Car:

def \_\_init\_\_(self, brand, model):

self.brand = brand

self.model = model

def show\_details(self):

print(f"Car Brand: {self.brand}, Model: {self.model}")

car1 = Car("Tesla", "Model S")

car1.show\_details()

4. Class vs Instance Variables

class Employee:

company\_name = "Infosys"

def \_\_init\_\_(self, name, salary):

self.name = name

self.salary = salary

5. Types of Methods

- Instance Method

- Class Method (@classmethod)

- Static Method (@staticmethod)

6. Encapsulation

Private variables are prefixed with \_\_

Example: self.\_\_balance

7. Inheritance

Allows child class to acquire parent class properties.

Types:

- Single

- Multilevel

- Hierarchical

- Multiple

8. Polymorphism

Same method behaves differently based on object type.

Example: method overriding.

9. Abstraction

Hiding complex details using abstract classes (abc module).

10. Constructor & Destructor

\_\_init\_\_() – Constructor

\_\_del\_\_() – Destructor

11. Real-world Example – BankAccount

class Account:

def \_\_init\_\_(self, acc\_no, name, balance=0):

self.acc\_no = acc\_no

self.name = name

self.\_\_balance = balance

def deposit(self, amt):

self.\_\_balance += amt

12. Key Concepts Summary

Class – Blueprint

Object – Instance

Encapsulation – Data hiding

Inheritance – Code reuse

Polymorphism – Multiple behavior

Abstraction – Hide complexity

13. Interview Questions

1. What are four pillars of OOP?

2. What is the difference between @staticmethod and @classmethod?

3. What is self?

4. What is MRO?

5. Explain Encapsulation with example.