

OOP in Python — Index by 4 Pillars

1. 🧱 Encapsulation

Wrapping data and methods into a single unit (class).

- Class & Object — class, object
 - Instance Variables & Methods — self
 - Class Variables & Methods — @classmethod, cls
 - Static Methods — @staticmethod
 - Access Modifiers — _protected, __private
 - Getters & Setters — @property, @<var>.setter
 - Name Mangling — _ClassName__var
 - Constructor & Destructor — __init__(), __del__()
 - Encapsulation with Data Hiding
 - Encapsulation with Properties
-

2. 🕵️ Abstraction

Hiding unnecessary details and showing only the essential features.

- Abstract Classes — abc module, ABC, @abstractmethod
 - Abstract Base Classes (ABC)
 - Interfaces in Python (via ABC)
 - Method Overriding for abstraction
 - Abstraction using Properties
 - Encapsulation vs Abstraction
-

3. 🚀 Inheritance

Acquiring properties and behavior of a parent class into a child class.

- Single Inheritance — class Child(Parent)
- Multiple Inheritance

- Multilevel Inheritance
 - Hierarchical Inheritance
 - Hybrid Inheritance
 - `super()` keyword usage
 - Method Resolution Order (MRO) — `ClassName.__mro__`, `super()`
 - `issubclass()`, `isinstance()`
 - Constructor chaining with `super()`
 - Diamond Problem handling in Python (MRO)
-

4. 🌟 Polymorphism

One interface, many implementations.

- Function Polymorphism — built-in functions
 - Operator Overloading — `__add__`, `__sub__`, etc.
 - Method Overloading (through default args)
 - Method Overriding
 - Duck Typing concept
 - Dynamic Typing
 - Polymorphism with Inheritance
 - Abstract Methods + Polymorphism
 - Special Methods (Dunder methods)
-

🧠 Additional Advanced OOP Concepts

(Not a separate pillar but important for interviews & projects)

- Magic / Dunder Methods — `__str__`, `__repr__`, `__eq__`, etc.
- Composition vs Inheritance
- Mixins
- Metaclasses — `type`, `__new__`, `__init__`
- Custom Exceptions (OOP style)

- Operator Overloading Deep Dive
- Singleton Pattern (OOP Design Pattern)
- Class Decorators
- Data Classes — @dataclass
- Enum Class — Enum
- __slots__ for memory optimization