**Object Oriented Design**

A class is the blueprint of an object. The implementation of the class is the object. The members of the class define the behavior of the class. The class is not visible to the world, but the object is.

Pillars of OOP –

Data Abstraction – Representing important and special features without including the background details or explanation about that feature. Data abstraction simplifies database design.

View 1

View 3

View 2

Physical View

(how data is stored)

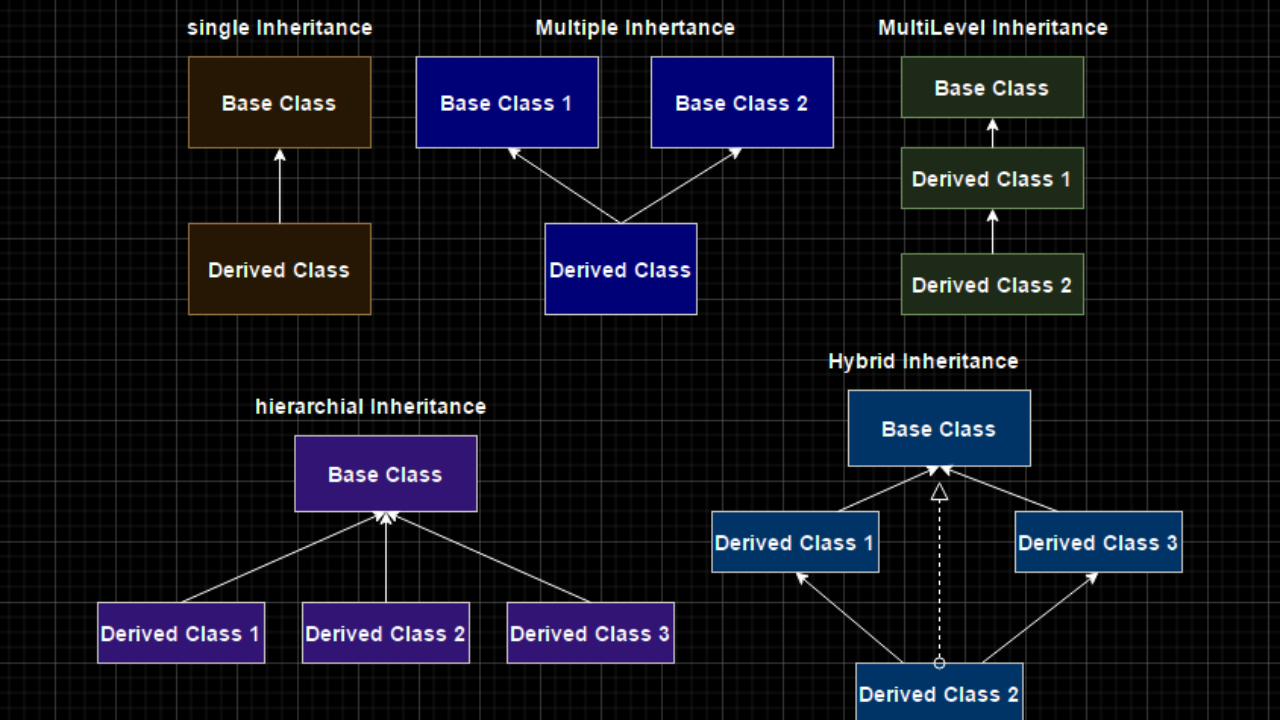
Logical View

(what data is stored)

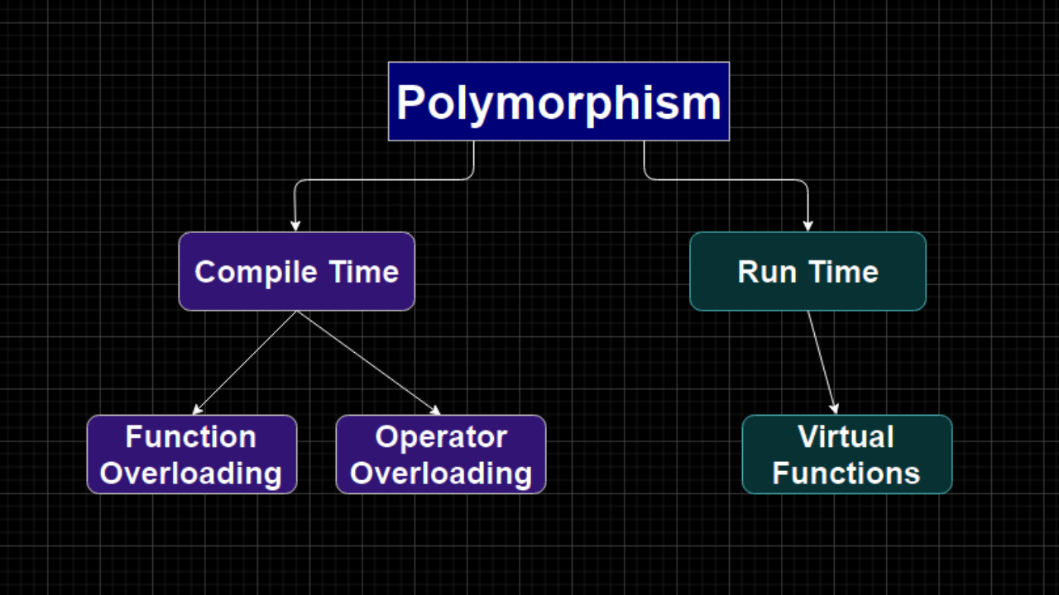
User Level View

Encapsulation – Wrapping data and the methods that work on data within one unit. Example, a stack class. This concept is used to hide the internal state representation of an object from the outside.

Inheritance – Ability of one class to inherit capabilities or properties of another class, aka parent class. This concept allows reuse of code whenever possible and reduce redundancy.



Polymorphism – Ability of data to be processed in more than one form. It allows the performance of the same task in various ways. It consists of method overloading and method overriding.



Approach to solving OOD questions

DISCLAIMER: Do not make assumptions on your own. Always clarify with the interviewer about the requirements.

1. The 6 Ws – Who, What, Where, When, How, Why.
2. Define the core **objects**. For example, for a restaurant, the core objects might be things like Table, Guest, Party, Order, Meal, Employee, Server, Host, Chef etc.
3. Analyze relationships between objects. Which objects are members of which other objects? Do any objects inherit from any others? Are relationships many-to-many or one-to-many?
4. Investigate actions, find the key actions that the objects will perform and how they relate to each other.

Design Patterns

Singleton Class

The Singleton pattern ensures that a class has only one instance and ensures access to the application through the instance. For example, we may want to implement Restaurant such that it has exactly one instance of Restaurant.

Factory Method

The Factory methods offers an interface for creating an instance of a class, with its subclasses deciding which class to instantiate.

Examples

Parking Lot

Key points

* A parking lot can have multiple levels
* Each level has multiple rows of parking spots
* Vehicle can be a car, motorcycle, bus, truck etc.
* Parking lot has a motorcycle spot, compact car spot, and large spots
* A motorcycle can park in any spot
* A car can park in either a single compact spot or a single large spot
* A bus can park in five large spots that are consecutive and within the same row. It cannot park in small spots.