1.What is Object-Oriented Programming (OOP)?

OOP is a programming paradigm based on objects that contain data and functions

2.What is a class in OOPS

A class is a blueprint for creating objects.

3.What is an object in OOP?

An object is an instance of a class

4. What is the Difference between Abstraction and Encapsulation

- Abstraction: Hides *implementation details* and shows only the essential features.
- Encapsulation: Bundles data and methods together

5.What are dunder methods in Python?

Dunder methods are special methods like `__init__`, `__str__`, `__len__` that Python uses internally to perform operations.

6.Explain the concept of inheritance in OOP

Inheritance allows a class child to acquire properties and methods of another class parent.

```
class Animal:
    def speak(self): print("Sound")
class Dog(Animal):
    def speak(self): print("Bark")
```

7.What is polymorphism in OOP?

Polymorphism means the same function or operator can behave differently based on context.

8.How is encapsulation achieved in Python?

By making variables private using `_` or `__` and accessing them via getter/setter methods.

9.What is a constructor in Python?

A constructor is the `__init__` method, called automatically when object is created.

10.What are class and static methods in Python?

- Class method: Works with class variables (`@classmethod`, uses `cls`).
- Static method: Independent function inside class (`@staticmethod`, no `self` or `cls`).

11.What is method overloading in Python?
Python doesn't support true overloading. But it can be mimicked

12.What is method overriding in OOP?
Redefining a parent class method in the child class.

13.What is a property decorator in Python?

14.Why is polymorphism important in OOP?
It allows flexibility, code reuse, and the same interface for different object types.

15.What is an abstract class in Python?
A class with abstract methods (using `abc` module) that cannot be instantiated directly.

16.What are the advantages of OOP?

- Reusability
- Modularity

17. What is the Difference between class variable and instance variable

- Class variable: Shared among all objects.
- Instance variable: Unique to each object.

18.What is multiple inheritance in Python?
A class can inherit from multiple parent classes.

19. Explain the Purpose of " `__str__` 'and '`__repr__` methods in python

- `__str__`: User-friendly string (print output).
- `__repr__`: Debug representation for developers.

20.What is the Significance of `super()` function in python

Used to call methods of the parent class in child class.

21. What is the Significance of `__del__` method in python

Destructor method, called when object is deleted.

22.What is the Difference between @staticmethod and @classmethod

- `@staticmethod`: No `self` or `cls`.
- `@classmethod`: Works with class itself, takes `cls` as first parameter.

23.How does polymorphism work in Python with inheritance?

24.What is method chaining in Python OOP?

Returning `self` from methods allows calling multiple methods in a single line.

25.What is the purpose of the `__call__` method in Python?

Makes an object callable like a function.