Basic Level OOPs Questions in Python

1. What is Object-Oriented Programming (OOPs)?
2. What are classes and objects in Python? Explain with an example.
3. What is the difference between procedural programming and OOPs?
4. Define __init__ method in Python. Why is it used?
5. What is the difference between instance variables and class variables?
6. How do you create an object of a class in Python?
7. Explain the concept of encapsulation with an example.
8. What are access specifiers in Python (public, protected, private)?
9. Explain the use of self in Python classes.
10. Differentiate between a function and a method in Python.

- **Intermediate Level OOPs Questions**

11. What is inheritance in Python? Explain types of inheritance with examples.
12. What is method overriding? Provide an example.
13. Explain the difference between method overloading and method overriding in Python.
14. What is multiple inheritance? How does Python handle conflicts (MRO - Method Resolution Order)?
15. Explain polymorphism with an example.
16. What is an abstract class in Python? How do you create one?
17. What are static methods and class methods in Python? Give examples.
18. How does Python support operator overloading? Write an example.
19. What is the difference between composition and inheritance?
20. Why is super() used in Python? Explain with an example.

- **Advanced Level OOPs Questions**

21. What is metaclass in Python? Why is it used?

22. Explain duck typing in Python OOPs.
23. How is encapsulation implemented in Python (since it does not strictly enforce private variables)?
24. What is the diamond problem in multiple inheritance? How does Python solve it?
25. Explain __str__ and __repr__ methods with examples.
26. What are descriptors in Python OOPs?
27. How does Python implement interfaces without keywords like interface (unlike Java)?
28. Explain the difference between shallow copy and deep copy in Python objects.
29. What are data classes in Python 3.7+? How are they useful?
30. How is memory management done in Python for objects?

- **Coding-Based OOPs Questions**

31. Write a Python class to represent a Bank Account with deposit, withdraw, and balance check methods.
32. Write a program to demonstrate single inheritance (e.g., Person → Student).
33. Write a program to show multiple inheritance and how Python resolves ambiguity using MRO.
34. Create a Shape base class and Circle, Rectangle subclasses implementing area() method.
35. Write a program to overload the + operator for adding two custom objects (e.g., Complex numbers).
36. Create a class Employee that keeps track of the number of employees using a class variable.
37. Implement an abstract class Vehicle with abstract method start(), and concrete classes Car and Bike.
38. Demonstrate encapsulation by creating a class with private variables and public getter/setter methods.
39. Write a program that uses super() to call parent class methods in a child class.
40. Create a Student class where equality (==) compares students based on their roll numbers.