Practice Questions for

DAY-5:

Single Inheritance

46. Question: Create a base class Animal with a method sound() that prints "Some sound". Derive a class Dog from Animal and override the sound() method to print "Bark". Instantiate an object of Dog and call the sound() method.

Multiple Inheritance

47. Question: Create two base classes Swimming with a method swim() that prints "Swimming" and Flying with a method fly() that prints "Flying". Derive a class Duck from both Swimming and Flying. Instantiate an object of Duck and call both the swim() and fly() methods.

Multilevel Inheritance

48. Question: Create a base class LivingBeing with a method live() that prints "Living". Derive a class Animal from LivingBeing and add a method breathe() that prints "Breathing". Derive a class Human from Animal and add a method speak() that prints "Speaking". Instantiate an object of Human and call all three methods (live(), breathe(), and speak()).

Hierarchical Inheritance

49. Question: Create a base class Shape with a method area() that returns 0. Derive two classes Circle and Rectangle from Shape. Override the area() method in both derived classes to calculate and return the area of a circle and a rectangle, respectively. Instantiate objects of both Circle and Rectangle and call their area() methods.

Hybrid Inheritance

50. Question: Create a base class Vehicle with a method drive() that prints "Driving". Create two classes Car and Truck that inherit from Vehicle and add specific methods carry_passengers() for Car and carry_cargo() for Truck. Create another class PickupTruck that inherits from both Car and Truck. Instantiate an object of PickupTruck and call the methods drive(), carry_passengers(), and carry_cargo().

Abstract Classes

51. Question: Create an abstract base class Employee with an abstract method calculate_salary(). Derive two classes PermanentEmployee and ContractEmployee from Employee. Implement the calculate_salary() method in both derived classes to print respective salary calculations. Instantiate objects of both PermanentEmployee and ContractEmployee and call their calculate_salary() methods.

More on Inheritance

52. Question: Create a class Person with a method greet() that prints "Hello". Create another class Student that inherits from Person and overrides the greet() method to print "Hello, I am a student". Create a third class Teacher that also inherits from Person and overrides the greet() method to print "Hello, I am a teacher". Instantiate objects of Student and Teacher and call their greet() methods.

53. Question: Create an abstract class Appliance with an abstract method turn_on(). Create two classes WashingMachine and Refrigerator that inherit from Appliance. Implement the turn_on() method in both derived classes to print "Washing machine is now on" and "Refrigerator is now on", respectively. Instantiate objects of both WashingMachine and Refrigerator and call their turn_on() methods.

54. Question: Create a class Device with a method operate() that prints "Operating device". Create two classes Phone and Tablet that inherit from Device and add specific methods make_call() for Phone and browse() for Tablet. Create another class Smartphone that inherits from both Phone and Tablet. Instantiate an object of Smartphone and call the methods operate(), make_call(), and browse().

55. Question: Create a base class BankAccount with a method deposit() that increases the balance by a given amount. Derive a class SavingsAccount from BankAccount and add a method add_interest() that increases the balance by a fixed interest rate. Instantiate an object of SavingsAccount, call the deposit() method, and then call the add_interest() method.

56. Question: Create an abstract class Shape with an abstract method perimeter(). Derive two classes Square and Triangle from Shape. Implement the perimeter() method in both derived

classes to calculate and return the perimeter of a square and a triangle, respectively. Instantiate objects of both Square and Triangle and call their perimeter() methods.

57. Question: Create a class User with a method login() that prints "User logged in". Create another class Admin that inherits from User and overrides the login() method to print "Admin logged in". Create a third class Guest that inherits from User and overrides the login() method to print "Guest logged in". Instantiate objects of Admin and Guest and call their login() methods.