# OOPs_Part-2

## Assignment Questions

PW SKILLS

# Theory Assignment:

1. **What is the purpose of an abstract class in Python, and how is it different from a regular class?**

2. **Explain the concept of method overriding in the context of object-oriented programming.**

3. **How does the `@abstractmethod` decorator in Python affect the behavior of a class?**

4. **Describe the concept of polymorphism in object-oriented programming.**

5. **What is the difference between an "Is-A" and a "Has-A" relationship in object-oriented programming?**

6. **Explain the purpose of the `__init__()` method in a Python class, and how does it differ in its implementation between aggregation and composition relationships?**

7. **What is the concept of composition in object-oriented programming, and how does it differ from aggregation?**

8. **Explain the benefits of using abstract classes in object-oriented design.**

9. **How does method overriding contribute to the principle of polymorphism in object-oriented programming?**

10. **Discuss the use cases and benefits of composition over inheritance in object-oriented design.**

## Programming Assignment:

1. **Implement an abstract class Shape with an abstract method calculate_area(). Create subclasses such as Rectangle and Circle that inherit from Shape and override the calculate_area() method to calculate the area of their respective shapes.**

2. **Create a base class Vehicle with a method drive(). Define subclasses like Car and Motorcycle that inherit from Vehicle and override the drive() method to provide their own implementation of vehicle driving.**

3. **Define a class Calculator with overloaded methods add() to perform addition with different numbers of arguments. Test the calculator by adding numbers using different numbers of arguments.**

4. **Implement a class Department and another class Employee. The Department class should contain a list of Employee objects as an attribute. Demonstrate aggregation by adding employees to different departments.**

5. **Design classes representing a House and its various components like Bedroom, LivingRoom, and Kitchen. Implement composition by creating a House object that contains instances of its components.**

6. **Create a base class Shape with a method area() to calculate the area of different shapes. Implement subclasses like Rectangle and Circle that inherit from Shape and override the area() method with appropriate formulas.**

7. **Define a class Math with static methods for common mathematical operations like addition, subtraction, multiplication, and division. Also, implement a class method to compute the factorial of a number.**