practice coding questions based on Object-Oriented Programming (OOP) principles in Python:

Question 1: Class Inheritance

Define a base class called Animal with a method speak().

Create two subclasses, Dog and Cat, that inherit from Animal.

Override the speak() method in each subclass to print a different sound.

```
class Animal:
        def speak(self):
                print("Generic animal sound")
class Dog(Animal):
        def speak(self):
                print("Woof!")
class Cat(Animal):
        def speak(self):
                print("Meow!")
# Test the classes
dog = Dog()
cat = Cat()
dog.speak() # Output should be "Woof!"
cat.speak() # Output should be "Meow!"
```

GitHub Repo Link- https://github.com/amirkhan1092/Batch2023-24

Question 2: Encapsulation and Getter/Setter

Create a class called Circle with a private attribute radius.

Implement getter and setter methods to access and modify the radius.

Include a method calculate_area() that returns the area of the circle.

```
import math

class Circle:
        def __init__(self, radius):
                self.__radius = radius # Private attribute
        def get_radius(self):
                return self.__radius
        def set_radius(self, radius):
                if radius > 0:
                        self.__radius = radius
        def calculate_area(self):
                return math.pi * self.__radius**2



# Test the class
circle = Circle(5)
print("Radius:", circle.get_radius())
print("Area:", circle.calculate_area())
```

GitHub Repo Link- https://github.com/amirkhan1092/Batch2023-24

Question 3: Polymorphism

Create a base class called Shape with an abstract method area().

Create two subclasses, Circle and Square, that inherit from Shape.

Implement the area() method in each subclass.

```
from abc import ABC, abstractmethod

class Shape(ABC):
        @abstractmethod
        def area(self):
                pass
class Circle(Shape):
        def __init__(self, radius):
                self.radius = radius
        def area(self):
                return math.pi * self.radius**2
class Square(Shape):
        def __init__(self, side_length):
                self.side_length = side_length
        def area(self):
                return self.side_length**2


# Test the classes
circle = Circle(5)
square = Square(4)
print("Circle Area:", circle.area())
print("Square Area:", square.area())
```

GitHub Repo Link- https://github.com/amirkhan1092/Batch2023-24

These questions cover aspects of class inheritance, encapsulation, and polymorphism in Python. Feel free to use them for practice and modify them as needed!

GitHub Repo Link- https://github.com/amirkhan1092/Batch2023-24