МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Комп’ютерне моделювання процесів та систем»

Ефективність та якість архітектурних рішень

**Звіт  
Лабораторна робота №4**

Відмова від спадку. Альтернативні класи з різними інтерфейсами

Виконав ст.гр. ІКМ-М223д

Сушков О.О.

Перевірив викладач

Хорошун А.С.

Харків 2024

1. У вас є базовий клас Animal, який має різні методи, і підклас Dog, який успадковує від базового класу. Ваше завдання - переписати код так, щоб відмовитися від спадкування та замість цього використовувати делегування для виклику методів базового класу..

|  |
| --- |
| class Animal:      def speak(self):          pass      def move(self):          pass  class Dog(Animal):      pass  # Приклад використання класів  dog = Dog()  dog.speak()  dog.move()  shape.calculate\_area() |

Після рефакторингу підклас Dog повинен викликати відповідні методи базового класу Animal через делегування.

Результат:

class Animal:

    def speak(self):

        print("Animal speaks")

    def move(self):

        print("Animal moves")

class Dog:

    def \_\_init\_\_(self):

        self.animal = Animal()

    def speak(self):

        self.animal.speak()

    def move(self):

        self.animal.move()

# Приклад використання класів

dog = Dog()

dog.speak()

dog.move()

Тепер клас Dog містить об’єкт класу Animal, але не успадковує його. Методи класу Dog делегують відповідні методи класу Animal.

1. У вас є два класи Square і Circle, які виконують однакову функцію - обчислення площі фігури, але мають різний інтерфейс. Ваше завдання - переписати код так, щоб обидва класи мали спільний інтерфейс

|  |
| --- |
| class Square:      def area\_square(self, side):          return side \*\* 2  class Circle:      def area\_circle(self, radius):          return 3.14 \* radius \*\* 2  # Приклад використання класів  square = Square()  print("Area of square:", square.area\_square(5))  circle = Circle()  print("Area of circle:", circle.area\_circle(3)) |
|  |

Після рефакторингу класи Square і Circle повинні мати спільний інтерфейс, наприклад, метод area(), який приймає необхідні параметри для обчислення площі фігури.

Реалізація

class Square:

    def area(self, side):

        return side \*\* 2

class Circle:

    def area(self, radius):

        return math.pi \* radius \*\* 2

# Приклад використання класів

square = Square()

print("Area of square:", square.area(5))

circle = Circle()

print("Area of circle:", circle.area(3))

Тепер обидва класи мають метод area(), який приймає необхідні параметри для обчислення площі фігури.