МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Лабораторна робота №9

з дисципліни

СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ

на тему

Створення та рефакторинг програмно-інформаційного продукту засобами Python

Виконав:
ст. гр. PI-21сп
Рак В.П.
Прийняв:
Щербак С.С.

Львів-2024

Мета лабораторної роботи: Розробка програмно-інформаційного продукту засобами Python

Завдання

Завдання 1. Створити скрипт запуску лабораторних робіт 1-8 (Runner) з єдиним меню для управління додатками використовуючи патерн FACADE https://refactoring.guru/uk/design-patterns/facade.

Завдання 2. Зробити рефакторинг додатків, які були зроблені в лб 1-8, для підтримки можливості запуску через Runner.

Завдання 3. Зробити рефакторинг додатків, які були зроблені в лб 1-8, використовуючи багаторівневу архітектуру додатків (див. приклад нижче) та всі принципи об'єктно-орієнтованого підходу.

Завдання 4. Створити бібліотеку класів, які повторно використовуються у всіх лабораторних роботах та зробити рефакторинг додатків для підтримки цієї бібліотеки. Таких класів в бібліотеці має буде як найменш 5.

Завдання 5. Додати логування функцій в класи бібліотеки програмного продукту використовуючи https://docs.python.org/uk/3/howto/logging.html.

Завдання 6. Додати коментарі до програмного коду та сформувати документацію програмного продукту засобами pydoc. Документація має бути представлена у вигляді сторінок тексту на консолі, подана у веб-браузері та збережена у файлах HTML.

Завдання 7. Документація та код програмного продукту має бути розміщено в GIT repo.

Завдання 8. Проведіть статичний аналіз коду продукту засобами PYLINT https://pylint.readthedocs.io/en/stable/ та виправте помилки, які були ідентифіковані. Первинний репорт з помилками додайте до звіту лабораторної роботи.

Завдання 9. Підготуйте звіт до лабораторної роботи.

Виконання роботи

Текст програмної реалізації:

runner.py:

```
import importlib
import sys
import os
import unittest
import coverage

# Додавання шляхів для лабораторних робіт
sys.path.append(os.path.join(os.getcwd(), 'lab3'))
sys.path.append(os.path.join(os.getcwd(), 'lab4'))
sys.path.append(os.path.join(os.getcwd(), 'lab5'))  # Додано для lab5
sys.path.append(os.path.join(os.getcwd(), 'lab6'))  # Додано для lab6
sys.path.append(os.path.join(os.getcwd(), 'lab7'))  # Додано для lab7
sys.path.append(os.path.join(os.getcwd(), 'lab8'))  # Вже додає шлях до lab8

class Lab1Command:
    """
    Клас для виконання лабораторної роботи 1.
    Цей клас відповідає за завантаження модуля lab1.main та виклик його функції run_lab1.
    """
    def execute(self):
        """
        Виконує лабораторну роботу 1, завантажуючи модуль lab1.main
        і викликаючи функцію run_lab1.
        """
        lab1_module = importlib.import_module("lab1.main")
        lab1_module.run_lab1()  # Викликаємо функцію run_lab1 для лабораторної роботи 1

class Lab2Command:
    """
    Клас для виконання лабораторної роботи 2.
```

```
    Цей клас відповідає за завантаження модуля lab2.main та виклик його функції run_lab2.
    """
    def execute(self):
        """
        Виконує лабораторну роботу 2, завантажуючи модуль lab2.main
        і викликаючи функцію run_lab2.
        """
        lab2_module = importlib.import_module("lab2.main")
        lab2_module.run_lab2()  # Викликаємо функцію run_lab2 для лабораторної роботи 2


class Lab3Command:
    """
    Клас для виконання лабораторної роботи 3.
    Цей клас відповідає за завантаження модуля lab3.main та виклик його функції run_lab3.
    """
    def execute(self):
        """
        Виконує лабораторну роботу 3, завантажуючи модуль lab3.main
        і викликаючи функцію run_lab3.
        """
        lab3_module = importlib.import_module("lab3.main")
        lab3_module.run_lab3()  # Викликаємо функцію run_lab3 для лабораторної роботи 3


class Lab4Command:
    """
    Клас для виконання лабораторної роботи 4.
    Цей клас відповідає за завантаження модуля lab4.main та виклик його функції main.
    """
    def execute(self):
        """
        Виконує лабораторну роботу 4, завантажуючи модуль lab4.main
        і викликаючи функцію main.
        """
```

```
        lab4_module = importlib.import_module("lab4.main")
        lab4_module.main()  # Викликаємо основну функцію для лабораторної роботи 4


class Lab5Command:
    """
    Клас для виконання лабораторної роботи 5.
    Цей клас відповідає за завантаження модуля lab5.main та виклик його функції main.
    """
    def execute(self):
        """
        Виконує лабораторну роботу 5, завантажуючи модуль lab5.main
        і викликаючи функцію main.
        """
        lab5_module = importlib.import_module("lab5.main")  # Імпортуємо main.py для lab5
        lab5_module.main()  # Викликаємо основну функцію для lab5


class Lab6Command:
    """
    Клас для виконання лабораторної роботи 6.
    Цей клас відповідає за завантаження модуля lab6.main та виклик функції run.
    """
    def execute(self):
        """
        Виконує лабораторну роботу 6, запускаючи калькулятор із lab6.
        """
        lab6_module = importlib.import_module("lab6.main")
        lab6_module.Calculator().run()  # Запускаємо калькулятор із lab6


class Lab6TestCommand:
    """
    Клас для виконання тестів лабораторної роботи 6.
    Цей клас запускає тести для калькулятора з lab6.
    """
```

```
    def execute(self):
        """
        Запускає тести для лабораторної роботи 6, використовуючи покриття та юніт-тести.
        """
        # Запуск покриття (coverage) для тестів
        cov = coverage.Coverage(source=["lab6"])
        cov.start()

        # Запуск юніт-тестів
        test_loader = unittest.defaultTestLoader
        test_suite = test_loader.loadTestsFromName("test_calculator")
        test_runner = unittest.TextTestRunner()
        test_runner.run(test_suite)

        # Остановка покриття та виведення результатів
        cov.stop()
        cov.save()
        cov.report()  # Вивести звіт про покриття

class Lab7Command:
    """
    Клас для виконання лабораторної роботи 7.
    Цей клас відповідає за завантаження модуля lab7.main та виклик його функції main.
    """
    def execute(self):
        """
        Виконує лабораторну роботу 7, завантажуючи модуль lab7.main
        і викликаючи функцію main.
        """
        lab7_module = importlib.import_module("lab7.main")
        lab7_module.main()  # Викликаємо основну функцію для лабораторної роботи 7

class Lab8Command:
```

```
    """
    Клас для виконання лабораторної роботи 8.
    Цей клас відповідає за завантаження модуля lab8.main та виклик його функції main.
    """
    def execute(self):
        """
        Виконує лабораторну роботу 8, завантажуючи модуль lab8.main
        і викликаючи функцію main.
        """
        lab8_module = importlib.import_module("lab8.main")  # Завантажуємо main.py з lab8
        lab8_module.main()  # Викликаємо основну функцію для лабораторної роботи 8

# Реєструємо команди
commands = {
    "1": Lab1Command(),
    "2": Lab2Command(),
    "3": Lab3Command(),
    "4": Lab4Command(),
    "5": Lab5Command(),  # Додаємо команду для 5 лабораторної роботи
    "6": Lab6Command(),
    "6test": Lab6TestCommand(),
    "7": Lab7Command(),
    "8": Lab8Command(),  # Додаємо команду для 8 лабораторної роботи
}

def main():
    """
    Головна функція, яка виконує запуск лабораторних робіт через меню.
    Користувач може вибрати лабораторну роботу, яка буде виконана.
    """
    while True:  # Це забезпечить повторний вибір лабораторної роботи після виконання
        print("Оберіть лабораторну роботу:")
        for key, command in commands.items():
```

```
        print(f"{key}: Лабораторна робота {key}")


    choice = input("Ваш вибір: ")
    if choice in commands:
        commands[choice].execute()


        # Після виконання роботи запитуємо, чи хоче користувач обрати іншу лабораторну роботу
        another = input("Бажаєте вибрати іншу лабораторну роботу? (так/ні): ")
        if another.lower() != "так":
            print("До побачення!")
            break  # Виходимо з циклу, якщо користувач не хоче продовжити
    else:
        print("Невірний вибір.")


if __name__ == "__main__":
    main()
```

Результат роботи програми:

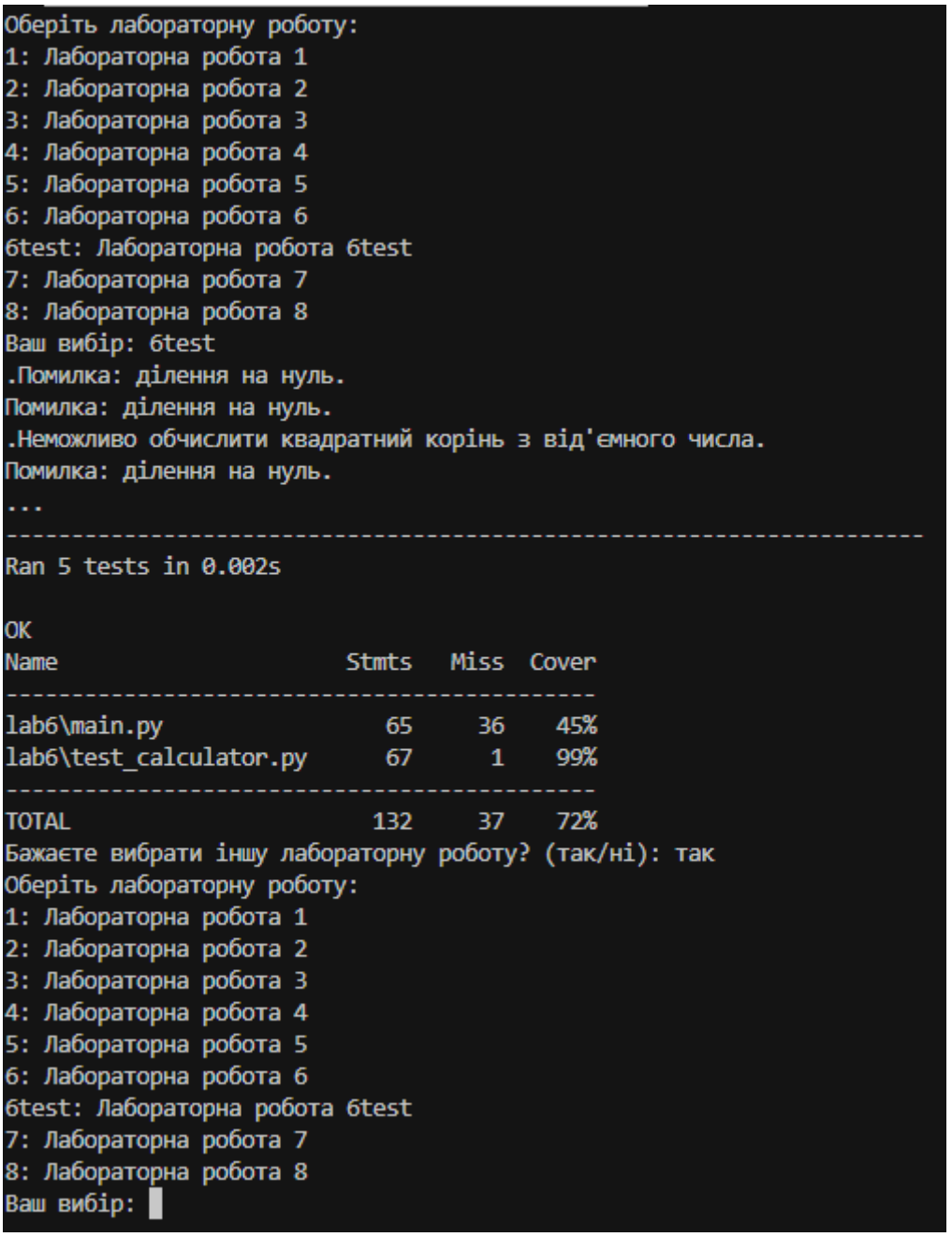

Рис. 1 – Результат виконання програми

Висновок: У ході виконання лабораторної роботи я створив скрипт запуску лабораторних робіт 1-8 використовуючи паттерн Facade.