Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Лабораторна роботи №9

з дисципліни «Спеціалізовані мови програмування»

на тему

«Створення та рефакторінг програмно-інформаційного продукту засобами Python»

Виконав:

Ложовський В. А.

Перевірив:

Щербак С. С.

Львів 2024

**Мета:** розробка програмно-інформаційного продукту засобами.

**План роботи**

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 рефакторінг 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 рефакторінг 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 лабораторной роботи

**Реалізація:**

**Файл runner.py:**

import sys

import logging

import os

import subprocess

sys.path.append(os.path.dirname(os.path.abspath(\_\_file\_\_)))

logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

from lab1.main import main as lab1\_main

from lab2.main import main as lab2\_main

from lab3.main import main as lab3\_main

from lab4.main import main as lab4\_main

from lab5.main import main as lab5\_main

from lab6.main import main as lab6\_main

from lab7.main import main as lab7\_main

from lab8.main import main as lab8\_main

class LabRunnerFacade:

    def \_\_init\_\_(self):

        logging.info("Ініціалізація лабораторних робіт...")

        self.labs = {

            1: lab1\_main,

            2: lab2\_main,

            3: lab3\_main,

            4: lab4\_main,

            5: lab5\_main,

            6: lab6\_main,

            7: lab7\_main,

            8: lab8\_main,

        }

        logging.info("Ініціалізація завершена.")

    def run\_lab(self, lab\_number):

        if lab\_number in self.labs:

            print(f"\nЗапуск лабораторної роботи {lab\_number}...\n")

            try:

                self.labs[lab\_number]()

            except Exception as e:

                print(f"Помилка під час виконання лабораторної роботи {lab\_number}: {e}")

        else:

            print(f"Лабораторна робота {lab\_number} не знайдена.")

    def run\_tests(self, lab\_number):

        test\_files = {

            1: "lab6/tests/AdditionTests.py",

            2: "lab6/tests/DividionTests.py",

            3: "lab6/tests/ErrorHandlingTests.py",

            4: "lab7/tests/test\_api\_client.py",

        }

        test\_file = test\_files.get(lab\_number)

        if not test\_file:

            print(f"Для лабораторної роботи №{lab\_number} тести не знайдено.")

            return

        if os.path.exists(test\_file):

            print(f"Запускаємо тести для лабораторної роботи №{lab\_number}...")

            subprocess.run(["python", "-m", "unittest", test\_file])

        else:

            print(f"Файл з тестами не знайдено: {test\_file}.")

def display\_menu():

    print("=== Меню запуску лабораторних робіт ===")

    print("1. lab1")

    print("2. lab2")

    print("3. lab3")

    print("4. lab4")

    print("5. lab5")

    print("6. lab6")

    print("7. lab7")

    print("8. lab8")

    print("9. unittests")

    print("0. Exit")

def display\_unittest():

    print("\n=== Меню запуску тестів ===")

    print("1. lab6\_AdditionTests")

    print("2. lab6\_DividionTests")

    print("3. lab6\_ErrorHandlingTests")

    print("4. lab7\_test\_api\_client")

    print("0. Back")

def main():

    """Головна функція Runner."""

    runner = LabRunnerFacade()

    while True:

        display\_menu()

        try:

            choice = int(input("Input: "))

            if choice == 0:

                sys.exit()

                runner.run\_lab(choice)

            elif choice == 9:

                while(True):

                    display\_unittest()

                    try:

                        choice = int(input("Input: "))

                        if choice == 0:

                            break

                        elif 1 <= choice <= 4:

                            runner.run\_tests(choice)

                        else:

                            print("Некоректний вибір.")

                    except ValueError:

                        print("Помилка: введіть коректне число.")

                    except Exception as e:

                        print(f"Сталася помилка: {e}")

            elif 1 <= choice <= 8:

                runner.run\_lab(choice)

            else:

                print("Некоректний вибір.")

        except ValueError:

            print("Помилка: введіть коректне число.")

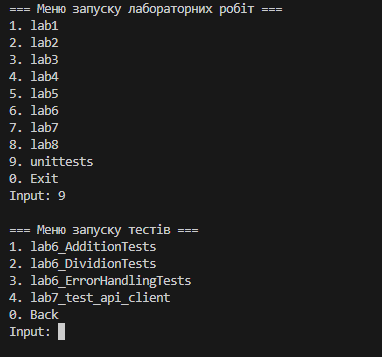
        except Exception as e:

            print(f"Сталася помилка: {e}")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Результат виконання:**

****

**Висновок:** на цій лабораторній роботі я об’єднав і доопрацював всі лабораторні роботи в один великий проект, який працює через runner.py