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:

студент групи РІ-21сп

Дмитрій Сас

Львів – 2024

**Мета виконання лабораторної роботи:** розробка програмно-інформаційного продукту засобами Python.

**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 лабораторної роботи

**Текст програмної реалізації:**

**runner.py:**import os

import logging

import pydoc

# Налаштування логування

logging.basicConfig(

    filename="app.log",

    filemode="a",

    format="%(asctime)s - %(levelname)s - %(message)s",

    level=logging.INFO

)

# Бібліотека класів

class FileManager:

    """Клас для роботи з файлами."""

    def read\_file(self, file\_path: str) -> str:

        """Читає файл."""

        logging.info(f"Читається файл: {file\_path}")

        try:

            with open(file\_path, "r", encoding="utf-8") as file:

                return file.read()

        except Exception as e:

            logging.error(f"Помилка читання файлу: {e}")

            return str(e)

    def write\_file(self, file\_path: str, content: str) -> None:

        """Записує в файл."""

        logging.info(f"Запис у файл: {file\_path}")

        try:

            with open(file\_path, "w", encoding="utf-8") as file:

                file.write(content)

        except Exception as e:

            logging.error(f"Помилка запису у файл: {e}")

class Calculator:

    """Калькулятор базових операцій."""

    @staticmethod

    def add(a: float, b: float) -> float:

        """Додає два числа."""

        result = a + b

        logging.info(f"Додавання {a} + {b} = {result}")

        return result

    @staticmethod

    def subtract(a: float, b: float) -> float:

        """Віднімає два числа."""

        result = a - b

        logging.info(f"Віднімання {a} - {b} = {result}")

        return result

class DataProcessor:

    """Обробка даних."""

    def sort\_data(self, data: list) -> list:

        """Сортує список."""

        logging.info(f"Сортування даних: {data}")

        return sorted(data)

class Logger:

    """Утиліта для роботи з логами."""

    def show\_logs(self):

        """Виводить лог-файл."""

        fm = FileManager()

        logs = fm.read\_file("app.log")

        print(logs)

# Runner (Facade Pattern)

class Runner:

    """Runner для запуску додатків."""

    def \_\_init\_\_(self):

        self.file\_manager = FileManager()

        self.calculator = Calculator()

        self.data\_processor = DataProcessor()

        self.logger = Logger()

    def show\_menu(self):

        """Головне меню."""

        while True:

            print("\n=== Головне меню ===")

            print("1. Читати файл")

            print("2. Записати у файл")

            print("3. Калькулятор")

            print("4. Сортувати дані")

            print("5. Логи")

            print("6. Вихід")

            choice = input("Оберіть опцію: ")

            if choice == "1":

                file\_path = input("Введіть шлях до файлу: ")

                print(self.file\_manager.read\_file(file\_path))

            elif choice == "2":

                file\_path = input("Введіть шлях до файлу: ")

                content = input("Введіть контент: ")

                self.file\_manager.write\_file(file\_path, content)

            elif choice == "3":

                a = float(input("Перше число: "))

                b = float(input("Друге число: "))

                print(f"Результат: {self.calculator.add(a, b)}")

            elif choice == "4":

                data = input("Введіть список чисел через кому: ").split(",")

                data = [int(x) for x in data]

                print(f"Відсортований список: {self.data\_processor.sort\_data(data)}")

            elif choice == "5":

                self.logger.show\_logs()

            elif choice == "6":

                print("До побачення!")

                break

            else:

                print("Невірний вибір, спробуйте знову.")

# Запуск Runner

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

    runner = Runner()

    runner.show\_menu()

**Результати роботи:**

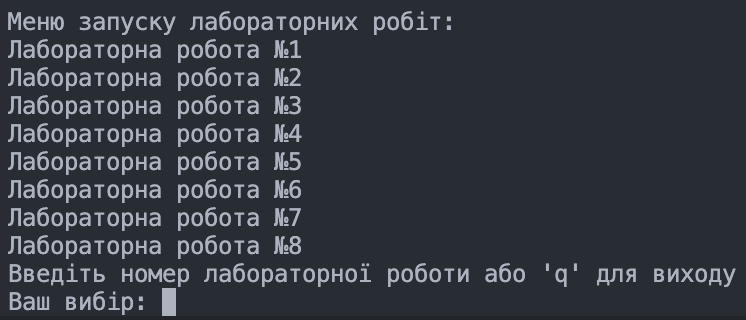
****

Рис. 1. Консольний інтерфейс

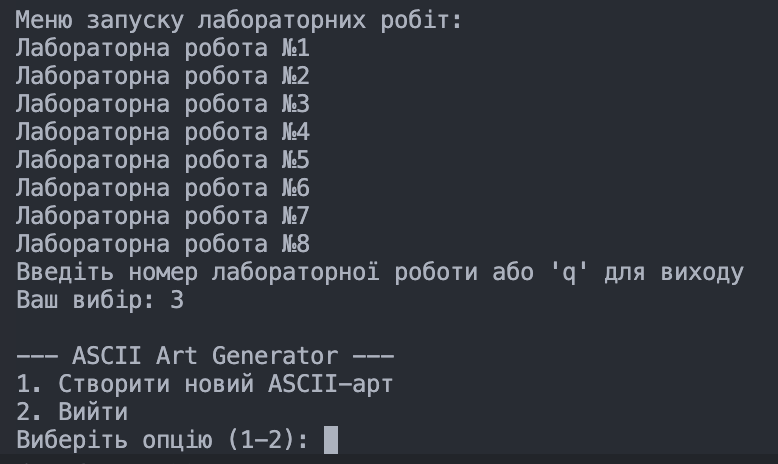
****

Рис. 2. Результат запуску 3 лабораторної

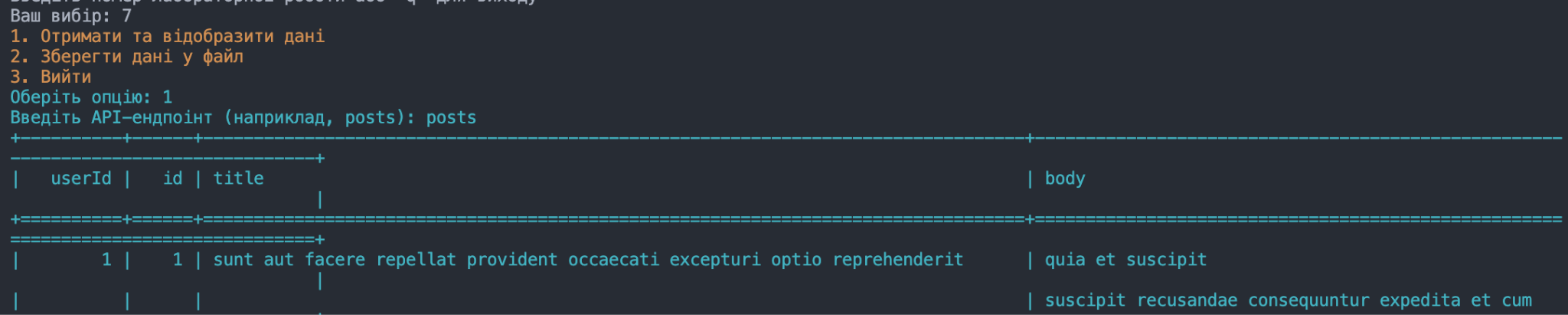
****

Рис. 3. Результат запуску 7 лабораторної

**Висновки:** на цій лабораторній роботі я розробив програмно-інформаційний продукт засобами Python.