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

Данило Вергун

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 sys

import os

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

sys.path.append(os.path.join(os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_3"))

sys.path.append(os.path.join(os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_4"))

sys.path.append(os.path.join(os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_5"))

sys.path.append(os.path.join(os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_7"))

sys.path.append(os.path.join(os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_8"))

lab\_3\_venv\_path = os.path.join(

os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_3", "venv", "lib", "python3.13", "site-packages"

)

sys.path.append(lab\_3\_venv\_path)

lab\_5\_venv\_path = os.path.join(

os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_5", "venv", "lib", "python3.13", "site-packages"

)

sys.path.append(lab\_5\_venv\_path)

lab\_7\_venv\_path = os.path.join(

os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_7", "venv", "lib", "python3.13", "site-packages"

)

sys.path.append(lab\_7\_venv\_path)

lab\_8\_venv\_path = os.path.join(

os.path.dirname(os.path.abspath(\_\_file\_\_)), "lab\_8", "venv", "lib", "python3.13", "site-packages"

)

sys.path.append(lab\_8\_venv\_path)

from lab\_1.main import Command as Lab1Command

from lab\_2.main import Command as Lab2Command

from lab\_3.main import Command as Lab3Command

from lab\_4.main import Command as Lab4Command

from lab\_5.main import Command as Lab5Command

from lab\_6.main import Command as Lab6Command

from lab\_7.main import Command as Lab7Command

from lab\_8.main import Command as Lab8Command

class Runner:

def \_\_init\_\_(self):

self.commands = {}

self.\_initialize\_commands()

def \_initialize\_commands(self):

try:

self.commands["1"] = Lab1Command()

self.commands["2"] = Lab2Command()

self.commands["3"] = Lab3Command()

self.commands["4"] = Lab4Command()

self.commands["5"] = Lab5Command()

self.commands["6"] = Lab6Command()

self.commands["7"] = Lab7Command()

self.commands["8"] = Lab8Command()

except ImportError as e:

print(f"Помилка при завантаженні команд: {e}")

def display\_menu(self):

print("\nМеню запуску лабораторних робіт:")

if not self.commands:

print("Немає доступних лабораторних робіт")

return

for number in sorted(self.commands.keys()):

print(f"Лабораторна робота №{number}")

print("Введіть номер лабораторної роботи або 'q' для виходу")

def start(self):

print("Ласкаво просимо до системи виконання лабораторних робіт!")

while True:

try:

self.display\_menu()

choice = input("Ваш вибір: ").strip().lower()

if choice == 'q':

print("Дякуємо за використання програми!")

break

if not choice:

print("Будь ласка, зробіть вибір")

continue

command = self.commands.get(choice)

if command:

command.execute()

else:

print("Невірний номер лабораторної роботи")

except KeyboardInterrupt:

print("\nПрограму завершено користувачем")

break

except Exception as e:

print(f"Виникла помилка: {e}")

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

runner = Runner()

runner.start()

**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.