мІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

нАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «лЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНОРМАЦІЙНИХ ТЕХНОЛОГІЙ

|  |
| --- |
|  |



ЗВІТ

Про виконання лабораторної роботи № 9

З дисципліни

«Спеціалізовані мови програмування»

Виконала:

студентка гр. ІТ-31

Катерина ПОПОВА

Прийняв:

Сергій ЩЕРБАК

Львів – 2023

**Тема роботи :** Створення та рефакторінг програмно-інформаційного продукту засобами Python

**Мета 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 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

**Код**

**runner.py**

import unittest

import json

import logging

import logging.config

from consolemenu import ConsoleMenu

from consolemenu import FunctionItem

from sources.lab1.calc import Calculator

from sources.lab2.color\_calc import ColorCalculator

from sources.lab3.source.input\_handler import InputHandler as Lab3InputHandler

from sources.lab3.source.file\_saver import FileSaver as Lab3FileSaver

from sources.lab3.source.output\_handler import OutputHandler as Lab3OutputHandler

from sources.lab3.source.ascii\_art\_generator import ASCIIArtGenerator as Lab3ASCIIArtGenerator

from sources.lab4.source.input\_handler import InputHandler as Lab4InputHandler

from sources.lab4.source.graphic.graphic import Graphic as Lab4Graphic

from sources.lab4.source.file\_saver import FileSaver as Lab4FileSaver

from sources.lab4.source.art\_generator import ArtGenerate as Lab4ArtGenerate

from sources.lab5.source.file\_saver import FileSaver as Lab5FileSaver

from sources.lab5.source.graphic.graphic import Graphic as Lab5Graphic

from sources.lab5.source.art\_generator import ArtGenerate as Lab5ArtGenerate

from sources.lab6.tests\_calculator import TestCalculatorMethods

from sources.lab7.api import API

from sources.lab8.app import VisualizationApplication

class Runner:

"""

Runner class for orchestrating the execution of various labs using a menu system.

"""

def \_\_init\_\_(self) -> None:

"""

Initializes the Runner class with a menu.

"""

self.menu = ConsoleMenu("Runner of labs")

self.menu.append\_item(FunctionItem("Simple Calculator", self.simple\_calc))

self.menu.append\_item(FunctionItem("OOP Calculator", self.oop\_calc))

self.menu.append\_item(FunctionItem("ASCII Art", self.ascii\_art))

self.menu.append\_item(FunctionItem("2D ASCII Art (no libraries)", self.ascii\_art\_2d\_no\_lib))

self.menu.append\_item(FunctionItem("3D ASCII Arts", self.ascii\_art\_3d))

self.menu.append\_item(FunctionItem("Unit tests of calculator", self.unit\_tests))

self.menu.append\_item(FunctionItem("API", self.api))

self.menu.append\_item(FunctionItem("CSV", self.csv\_plot))

def simple\_calc(self):

"""

Run the Simple Calculator lab.

"""

calc = Calculator()

calc.run()

def oop\_calc(self):

"""

Run the OOP Calculator lab.

"""

calculator = ColorCalculator()

calculator.run()

def ascii\_art(self):

"""

Run the ASCII Art lab.

"""

input\_handler = Lab3InputHandler()

output\_handler = Lab3OutputHandler()

file\_saver = Lab3FileSaver()

art\_generator = Lab3ASCIIArtGenerator(input\_handler, output\_handler, file\_saver)

art\_generator.run()

def ascii\_art\_2d\_no\_lib(self):

"""

Run the 2D ASCII Art (no libraries) lab.

"""

art\_generator = Lab4ArtGenerate(Lab4InputHandler(), Lab4Graphic(), Lab4FileSaver())

art\_generator.start()

def ascii\_art\_3d(self):

"""

Run the 3D ASCII Arts lab.

"""

art\_generator = Lab5ArtGenerate(Lab5Graphic(), Lab5FileSaver())

art\_generator.start()

def unit\_tests(self):

"""

Run the unit tests of the calculator.

"""

suite = unittest.TestLoader().loadTestsFromTestCase(TestCalculatorMethods)

test\_runner = unittest.TextTestRunner() # Renamed to test\_runner

test\_runner.run(suite)

input()

def api(self):

"""

Run the API lab.

"""

api = API()

api.run()

def csv\_plot(self):

"""

Run the CSV Plot lab.

"""

path = "c:\\Users\\HP\\Downloads\\iris.csv"

app = VisualizationApplication(path)

app.run()

def show(self):

"""

Show the menu.

"""

self.menu.show()

with open('config.json', 'r', encoding='utf-8') as config\_file:

config = json.load(config\_file)

logging.config.dictConfig(config)

runner = Runner()

runner.show()

**Результат виконання**

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

**Висновок:** Виконавши ці завдання, я створила програмно-інформаційного продукту засобами Python. Цей проект допоміг мені вивчити основний синтаксис Python і концепції, такі як введення користувача, умовні оператори, цикли та обробка помилок.