МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж



Лабораторна робота №6

з дисципліни Спеціалізовані мови програмування

на тему

Розробка та Unit тестування Python додатку

Виконав:

студент групи РІ-21сп

Данило ДЯЧОК

Львів – 2024

**Мета виконання лабораторної роботи:** Створення юніт-тестів для додатка-калькулятора на основі класів

**План роботи**

**Завдання 1: Тестування Додавання**

Напишіть юніт-тест, щоб перевірити, що операція додавання в вашому додатку-калькуляторі працює правильно. Надайте тестові випадки як для позитивних, так і для негативних чисел.

**Завдання 2: Тестування Віднімання**

Створіть юніт-тести для переконання, що операція віднімання працює правильно. Тестуйте різні сценарії, включаючи випадки з від'ємними результатами.

**Завдання 3: Тестування Множення**

Напишіть юніт-тести, щоб перевірити правильність операції множення в вашому калькуляторі. Включіть випадки з нулем, позитивними та від'ємними числами.

**Завдання 4: Тестування Ділення**

Розробіть юніт-тести для підтвердження точності операції ділення. Тести повинні охоплювати ситуації, пов'язані з діленням на нуль та різними числовими значеннями.

**Завдання 5: Тестування Обробки Помилок**

Створіть юніт-тести, щоб перевірити, як ваш додаток-калькулятор обробляє помилки. Включіть тести для ділення на нуль та інших потенційних сценаріїв помилок. Переконайтеся, що додаток відображає відповідні повідомлення про помилки.

**Текст програмної реалізації:**

**test\_calculator.py:**

import unittest

from bll.operations import Calculator

class TestCalculator(unittest.TestCase):

def setUp(self):

self.calc = Calculator()

def test\_addition(self):

self.assertEqual(self.calc.perform\_calculation(2, 3, '+'), 5)

def test\_subtraction(self):

self.assertEqual(self.calc.perform\_calculation(5, 3, '-'), 2)

def test\_multiplication(self):

self.assertEqual(self.calc.perform\_calculation(2, 3, '\*'), 6)

def test\_division(self):

self.assertEqual(self.calc.perform\_calculation(6, 3, '/'), 2)

def test\_division\_by\_zero(self):

with self.assertRaises(ValueError):

self.calc.perform\_calculation(6, 0, '/')

def test\_square\_root(self):

self.assertEqual(self.calc.perform\_calculation(9, None, '√'), 3)

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

unittest.main()

**Результати тестування:**

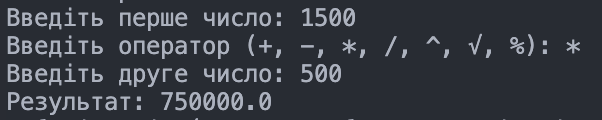


Рис. 1. Результати множення

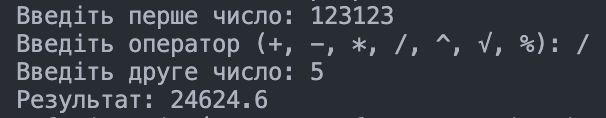


Рис. 2. Результати ділення

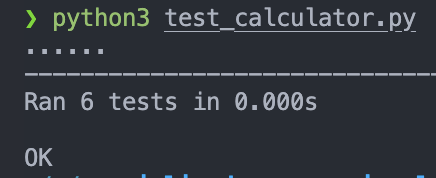


Рис. 3. Результат запуску тестів

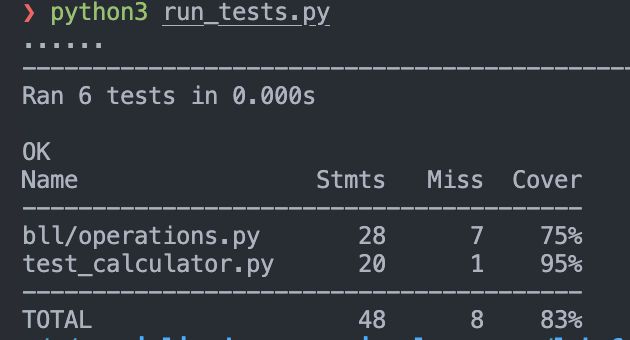


Рис. 4. Покриття тестами застосунок

**Висновки:** на цій лабораторній роботі я імплементував юніт-тести для додатка-калькулятора на основі класів.