Міністерство освіти і науки України

Національний університет «Львівська політехніка»

******

Звіт про виконання лабораторної роботи №6  
з дисципліни  
“Спеціалізовані мови програмування”  
на тему  
“Розробка та Unit тестування Python додатку ”

Виконала:  
студентка групи РІ-31,  
Пятницька Вікторія

Прийняв:

Щербак С.С.

**Мета.** Cтворення юніт-тестів для додатка-калькулятора на основі класів.

**Хід 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.

**Код програми:**

***test\_operations.py****:*

import unittest

from shared.operations import addition, subtraction, multiplication, division

class TestMathOperations(unittest.TestCase):

def test\_addition(self):

self.assertEqual(addition(5, 3), 8, "5 + 3 should equal 8")

self.assertEqual(addition(-2, 5), 3, "-2 + 5 should equal 3")

self.assertEqual(addition(0, 0), 0, "0 + 0 should equal 0")

self.assertEqual(addition(1.5, 2.5), 4.0, "1.5 + 2.5 should equal 4.0")

def test\_subtraction(self):

self.assertEqual(subtraction(5, 3), 2, "5 - 3 should equal 2")

self.assertEqual(subtraction(-2, 5), -7, "-2 - 5 should equal -7")

self.assertEqual(subtraction(0, 0), 0, "0 - 0 should equal 0")

self.assertEqual(subtraction(1.5, 0.5), 1.0, "1.5 - 0.5 should equal 1.0")

def test\_multiplication(self):

self.assertEqual(multiplication(5, 3), 15, "5 \* 3 should equal 15")

self.assertEqual(multiplication(-2, 5), -10, "-2 \* 5 should equal -10")

self.assertEqual(multiplication(0, 100), 0, "0 \* 100 should equal 0")

self.assertEqual(multiplication(1.5, 2.0), 3.0, "1.5 \* 2.0 should equal 3.0")

def test\_division(self):

self.assertEqual(division(6, 3), 2, "6 / 3 should equal 2")

self.assertEqual(division(-9, 3), -3, "-9 / 3 should equal -3")

self.assertAlmostEqual(division(1, 3), 0.3333333333333333, places=7,

msg="1 / 3 should approximately equal 0.3333333")

with self.assertRaises(ValueError) as context:

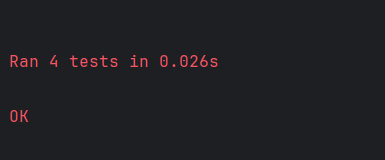
division(5, 0)

self.assertEqual(str(context.exception), "Division by zero is not allowed.")

# Виконання тестів

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**

*Рис.1.Результат виконання програми*

**Висновки:** внаслідок виконання лабораторної роботи я створила юніт-тести для додатка-калькулятора на основі класів.