МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП'ЮТЕРНИХ НАУК ТА ІНОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ЗВІТ

для лабораторної роботи №6 з

дисципліни

«Спеціалізовані мови програмування»

Виконав:

студент гр. ІТ-32

Паньків Б. В.

Прийняв:

доц. каф. ІСМ

Щербак С.С.

Львів-2023

Мета. Створення юніт-тестів для додатка-калькулятора на основі класів.

Хід виконання:

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:

```
import unittest

import Calculator as calculator



class TestCalculator(unittest.TestCase):

    def test_add(self):

        self.assertEqual(calculator.add(-7, 2.8), -4.2)

        self.assertEqual(calculator.add(0, -2), -2)

        self.assertEqual(calculator.add(1, 2), 3.0)

        with self.assertRaises(TypeError):

            calculator.add(40, "cat")

            calculator.add("lives", "ice")

            calculator.add("5", "+")
```

```
def test_subtract(self):
    self.assertEqual(calculator.subtract(1, 2.5), -1.5)
    self.assertEqual(calculator.subtract(0.8, -2), 2.8)
    self.assertEqual(calculator.subtract(-9, -7), -2)
    with self.assertRaises(TypeError):
        calculator.subtract("2", "-")
        calculator.subtract("brown", 5)
        calculator.subtract(3, "+")

def test_multiply(self):
    self.assertEqual(calculator.multiply(1.5, 3), 4.5)
    self.assertEqual(calculator.multiply(0, -2), 0)
    self.assertEqual(calculator.multiply(-0.5, -2), 1.0)
    with self.assertRaises(TypeError):
        calculator.multiply(3, "+")
        calculator.multiply("2", "-")
        calculator.multiply("log", 5)

def test_divide(self):
    self.assertEqual(calculator.divide(9, 3), 3)
    self.assertEqual(calculator.divide(0, -2), 0)
    self.assertEqual(calculator.divide(-0.5, 2), -0.25)
    with self.assertRaises(ZeroDivisionError):
        calculator.divide(10, 0)
```

```
        calculator.divide(0, 0)

    with self.assertRaises(TypeError):
        calculator.divide("d", "+")
        calculator.divide(3, "+")
        calculator.divide("94gs", 9)
```

На рис. 1 зображено знімок екрану із середовища розробки.

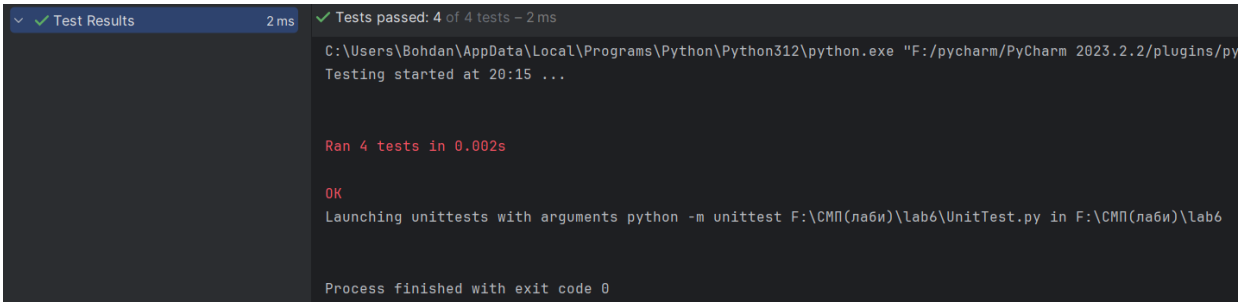

Рис. 1 Виконання програми

Посилання на GitHub-репозиторій із кодом: https://github.com/BOHDAN1329/SMP

Висновки: Я створив тести, які перевіряють правильність основних арифметичних операцій у вашому додатку-калькуляторі. Ці тести допомогли виявити та виправити будь-які проблеми або помилки, які можуть виникнути під час розробки чи обслуговування вашого додатку, забезпечуючи його надійність і точність.