Міністерство освіти і науки України

Національний університет «Львівська політехніка»



Лабораторна робота №6

з дисципліни “Спеціалізовані мови програмування”

**Виконала:**

студентка групи ІТ-31

Анастасія КАМІНОВСЬКА

**Прийняв:**доцент кафедри ІСМ

Сергій ЩЕРБАК

Львів 2023

**Тема:** Розробка та Unit тестування Python додатку

**Мета роботи:** 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.

import unittest

from calculator import Calculator

class TestCalculator(unittest.TestCase):

    def setUp(self):

        self.calc = Calculator('en\_US')

    def test\_addition(self):

        result = self.calc.OPERATORS['+'](10, 5)

        self.assertEqual(result, 15)

    def test\_subtraction(self):

        result = self.calc.OPERATORS['-'](10, 5)

        self.assertEqual(result, 5)

    def test\_multiplication(self):

        result = self.calc.OPERATORS['\*'](10, 5)

        self.assertEqual(result, 50)

    def test\_division(self):

        result = self.calc.OPERATORS['/'](10, 5)

        self.assertEqual(result, 2)

        result = self.calc.OPERATORS['/'](10, 0)

        self.assertEqual(result, "Error: division on zero")

    def test\_sqrt(self):

        result = self.calc.OPERATORS['√'](9, 0)

        self.assertEqual(result, 3)

    def test\_power(self):

        result = self.calc.OPERATORS['^'](2, 3)

        self.assertEqual(result, 8)

    def test\_modulo(self):

        result = self.calc.OPERATORS['%'](10, 3)

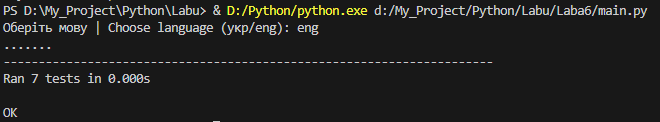
        self.assertEqual(result, 1)

        result = self.calc.OPERATORS['%'](10, 0)

        self.assertEqual(result, "Error: division on zero")

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

    unittest.main()



*Рис.1 – Результат виконання програми*

**Висновок:** на даній лабораторній роботі було створено набір юніт-тестів, які перевіряють правильність основних арифметичних операцій у вашому додатку-калькуляторі. Ці тести допоможуть виявити та виправити будь-які проблеми або помилки, які можуть виникнути під час розробки чи обслуговування мого додатку, забезпечуючи його надійність і точність.