Міністерство освіти і науки України

Національний університет «Львівська політехніка»



**ЗВІТ**

про виконання лабораторної роботи № 6

з курсу:

«Спеціалізовані мови програмування»

**Виконала:**

студентка гр. ІТ-31

Ірина ПЕРХУН

**Прийняв:**

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.*

"""

Module Docstring: A brief description of the module.

"""

import unittest

from classes.lab2.base\_calculator import Calculator

from classes.lab2.scientific\_calculator import ScientificCalculator

from classes.lab2.en\_localization import EnglishLocalization as localization

class TestCalculatorAddition(unittest.TestCase):

    """

    Class Docstring: A brief description of the test class.

    """

    def setUp(self):

        """

        Set up method for initializing test fixtures.

        """

        self.calculator = Calculator(None)

        self.scientific\_calculator = ScientificCalculator(None)

    def test\_addition\_positive\_numbers(self):

        """

        Check addition of positive numbers.

        """

        result = self.calculator.perform\_calculation(2.0, 3.0, '+', 0.0)

        self.assertEqual(result, 5.0)

    def test\_addition\_negative\_numbers(self):

        """

        Check addition of negative numbers.

        """

        result = self.calculator.perform\_calculation(-2.0, -3.0, '+', 0.0)

        self.assertEqual(result, -5.0)

    def test\_addition\_mixed\_numbers(self):

        """

        Check addition of positive and negative numbers.

        """

        result = self.calculator.perform\_calculation(2.0, -3.0, '+', 0)

        self.assertEqual(result, -1.0)

    # Subtraction

    def test\_subtraction\_positive\_numbers(self):

        """

        Check subtraction of positive numbers.

        """

        result = self.calculator.perform\_calculation(5.0, 3.0, '-', 0.0)

        self.assertEqual(result, 2.0)

    def test\_subtraction\_negative\_numbers(self):

        """

        Check subtraction of negative numbers.

        """

        result = self.calculator.perform\_calculation(-5.0, -3.0, '-', 0.0)  # -5 - -3 = -5 + 3 = -2

        self.assertEqual(result, -2.0)

    def test\_subtraction\_mixed\_numbers(self):

        """

        Check subtraction of positive and negative numbers.

        """

        result = self.calculator.perform\_calculation(5.0, -3.0, '-', 0.0)  # 5 - -3 = 5 + 3 = 8

        self.assertEqual(result, 8.0)

    # Multiplication

    def test\_multiplication\_positive\_numbers(self):

        """

        Check multiplication of positive numbers.

        """

        result = self.calculator.perform\_calculation(2.0, 3.0, '\*', 0.0)

        self.assertEqual(result, 6.0)

    def test\_multiplication\_negative\_numbers(self):

        """

        Check multiplication of negative numbers.

        """

        result = self.calculator.perform\_calculation(-2.0, -3.0, '\*', 0.0)

        self.assertEqual(result, 6.0)

    def test\_multiplication\_mixed\_numbers(self):

        """

        Check multiplication of positive and negative numbers.

        """

        result = self.calculator.perform\_calculation(2.0, -3.0, '\*', 0)

        self.assertEqual(result, -6.0)

    def test\_multiplication\_with\_zero(self):

        """

        Check multiplication by zero.

        """

        result = self.calculator.perform\_calculation(5.0, 0.0, '\*', 0.0)

        self.assertEqual(result, 0)

    # Division

    def test\_division\_positive\_numbers(self):

        """

        Check division of positive numbers.

        """

        result = self.calculator.perform\_calculation(6.0, 3.0, '/', 0.0)

        self.assertEqual(result, 2.0)

    def test\_division\_negative\_numbers(self):

        """

        Check division of negative numbers.

        """

        result = self.calculator.perform\_calculation(-6.0, -3.0, '/', 0.0)

        self.assertEqual(result, 2.0)

    def test\_division\_mixed\_numbers(self):

        """

        Check division of positive by negative number.

        """

        result = self.calculator.perform\_calculation(6.0, -3.0, '/', 0.0)

        self.assertEqual(result, -2.0)

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

        """

        Check division by zero.

        """

        self.calculator.localization = localization

        result = self.calculator.perform\_calculation(5.0, 0.0, '/', 0.0)

        self.assertIsNone(result)

    def test\_square\_root\_negative\_number\_error(self):

        """

        Check handling error of taking square root of a negative number.

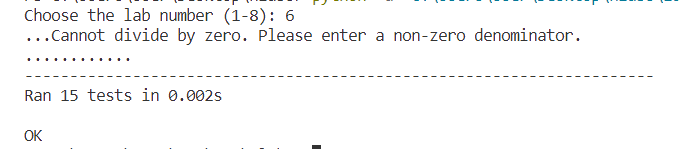
        """

        with self.assertRaises(ValueError):

            self.scientific\_calculator.calculate\_square\_root(-5)

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

    unittest.main()



*Рис. 1 – Результат проходження тестів*

**Висновок:** на даній лабораторній роботі я створила юніт-тести для додатка-калькулятора на основі класів.