ЗВІТ
про виконання лабораторної роботи № 6
«Розробка та Unit тестування Python додатку »
з дисципліни
«Спеціалізовані мови програмування»
ст. групи PI-31
Танечник Марічки

Мета: Створення юніт-тестів для додатка-калькулятора на основі класів
Умова завдання:

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 програми:

```
import unittest
from classes.Calculator import Calculator
from classes.Operation import Addition, Subtraction, Multiplication, Division, SquareRoot

class TestCalculator(unittest.TestCase):
    def setUp(self):
        """Set up the Calculator instance before each test"""
        self.calc = Calculator()

    def test_addition(self):
        """Test addition operation"""
        result = self.calc.perform_operation('+', 5, 3)
        self.assertEqual(result, 8)
```

```
    def test_subtraction(self):
        """Test subtraction operation"""
        result = self.calc.perform_operation('-', 10, 4)
        self.assertEqual(result, 6)

    def test_multiplication(self):
        """Test multiplication operation"""
        result = self.calc.perform_operation('*', 3, 7)
        self.assertEqual(result, 21)

    def test_division(self):
        """Test division operation"""
        result = self.calc.perform_operation('/', 20, 4)
        self.assertEqual(result, 5)

    def test_division_by_zero(self):
        """Test division by zero raises error"""
        with self.assertRaises(ZeroDivisionError):
            self.calc.perform_operation('/', 10, 0)

    def test_square_root(self):
        """Test square root operation"""
        result = self.calc.perform_operation('sqrt', 16)
        self.assertEqual(result, 4)

    def test_invalid_operator(self):
        """Test invalid operator raises ValueError"""
        with self.assertRaises(ValueError):
            self.calc.perform_operation('%', 5, 3)


if __name__ == "__main__":
    unittest.main()
```

Висновки: виконавши ці завдання,я створила набір юніт-тестів, які перевіряють правильність основних арифметичних операцій у моєму додатку-калькуляторі.