МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Лабораторна робота №6

з дисципліни

СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ

на тему

Розробка та Unit тестування Python додатку

Виконала:

ст. гр. РІ-31

Оксана ЛЕСЮК

Прийняв:

Сергій Сергійович ЩЕРБАК

Львів-2024

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

**Завдання:**

План 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.

Виконавши ці завдання, у вас буде 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 вашого додатку, забезпечуючи його надійність і точність

**Хід роботи**

**Test\_calculator.py:**

class UnitTestCalculator(unittest.TestCase):

def test\_addition(self):

self.assertEqual(calculateExpression(5, 3, '+'), 8)

self.assertEqual(calculateExpression(-5, 3, '+'), -2)

self.assertEqual(calculateExpression(-5, -3, '+'), -8)

def test\_subtraction(self):

self.assertEqual(calculateExpression(5, 3, '-'), 2)

self.assertEqual(calculateExpression(3, 5, '-'), -2)

self.assertEqual(calculateExpression(-5, -3, '-'), -2)

def test\_multiplication(self):

self.assertEqual(calculateExpression(5, 3, '\*'), 15)

self.assertEqual(calculateExpression(5, 0, '\*'), 0)

self.assertEqual(calculateExpression(-5, 3, '\*'), -15)

self.assertEqual(calculateExpression(-5, -3, '\*'), 15)

def test\_division(self):

self.assertEqual(calculateExpression(6, 3, '/'), 2)

self.assertEqual(calculateExpression(-6, 3, '/'), -2)

self.assertIsNone(calculateExpression(6, 0, '/'), "You cannot divide by zero!")

def test\_error\_handling(self):

self.assertIsNone(calculateExpression(6, 0, '/'), "You cannot divide by zero!")

self.assertIsNone(calculateExpression(-9, None, '√'), "Number cannot be lower than zero!")

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

unittest.main()

**Посилання на гіт-репозиторій:**

https://github.com/lesiukoksana/SMP-labs

**Висновок:**

У ході роботи було розроблено та реалізовано юніт-тести для основних арифметичних операцій калькулятора, включаючи додавання, віднімання, множення, ділення та обробку помилок. Для кожної операції було забезпечено перевірку різних сценаріїв, таких як робота з додатними, від'ємними числами, а також крайові випадки (наприклад, ділення на нуль).