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»

Виконав студент

групи РІ-32:

Фігура Т.Я

Прийняв:

Щербак С.С

Львів - 2024

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

**Хід виконання 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.

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

"""Test the addition of two positive numbers."""

result = self.calc.add(10, 5)

self.assertEqual(result, 15)

Завдання 2: Тестування Віднімання

Створіть юніт-тести для переконання, що операція віднімання працює правильно. Тестуйте різні сценарії, включаючи випадки з від'ємними результатами.

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

"""Test subtraction of two positive numbers where the result is positive."""

result = self.calc.subtract(15, 5)

self.assertEqual(result, 10)

Завдання 3: Тестування Множення

Напишіть юніт-тести, щоб перевірити правильність операції множення в вашому калькуляторі. Включіть випадки з нулем, позитивними та від'ємними числами.

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

"""Test multiplication of two positive numbers."""

result = self.calc.multiply(10, 5)

self.assertEqual(result, 50)

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.

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

"""Test division of two positive numbers."""

result = self.calc.divide(10, 2)

self.assertEqual(result, 5)

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.

def divide(self, a, b):

if b == 0:

return "Error: Division by zero is undefined."

return a / b

**Посилання на GitHub:** [***https://github.com/FihuraTaras/SPL\_Python\_Project.git***](https://github.com/FihuraTaras/SPL_Python_Project.git)

**Висновок:** Виконавши ці завдання, в мене появився набір юніт-тестів, які перевіряють правильність основних арифметичних операцій у моєму додатку-калькуляторі. Ці тести допоможуть виявити та виправити будь-які проблеми або помилки, які можуть виникнути під час розробки чи обслуговування додатку, забезпечуючи його надійність і точність