Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ІСМ



Звіт

про виконання лабораторної роботи № 6

«Розробка та Unit тестування Python додатку»

з дисципліни

«Спеціалізовані мови програмування»

Виконав:

Студент групи ІТ-32,

Садовенко А. Ю.

Прийняв:

Щербак С.С

Львів 2023

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

**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.

**Код:**

**test.py:**

import unittest

import smp\_lab1.Calculator as calculator

class TestCalculator(unittest.TestCase):

def test\_basic\_addition(self):

self.assertAlmostEqual(calculator.add(3.5, 2), 5.5)

self.assertAlmostEqual(calculator.add(-4, -6), -10)

self.assertAlmostEqual(calculator.add(0, 0), 0)

def test\_addition\_with\_errors(self):

with self.assertRaises(TypeError):

calculator.add("hello", 2)

calculator.add(2.5, "hello")

def test\_basic\_subtraction(self):

self.assertAlmostEqual(calculator.subtract(5, 3), 2)

self.assertAlmostEqual(calculator.subtract(-7, -2), -5)

self.assertAlmostEqual(calculator.subtract(8, 10), -2)

def test\_subtraction\_with\_errors(self):

with self.assertRaises(TypeError):

calculator.subtract("hello", 4)

calculator.subtract(3.6, "hello")

def test\_basic\_multiplication(self):

self.assertAlmostEqual(calculator.multiply(6, 7), 42)

self.assertAlmostEqual(calculator.multiply(-3, 4), -12)

self.assertAlmostEqual(calculator.multiply(0, 5), 0)

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

with self.assertRaises(TypeError):

calculator.multiply("hello", 8)

calculator.multiply(3.14, "hello")

def test\_basic\_division(self):

self.assertAlmostEqual(calculator.divide(15, 3), 5)

self.assertAlmostEqual(calculator.divide(-10, 5), -2)

self.assertAlmostEqual(calculator.divide(5, 2), 2.5)

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

with self.assertRaises(ZeroDivisionError):

calculator.divide(5, 0)

calculator.divide(-1, 0)

def test\_division\_with\_errors(self):

with self.assertRaises(TypeError):

calculator.divide("hello", 3)

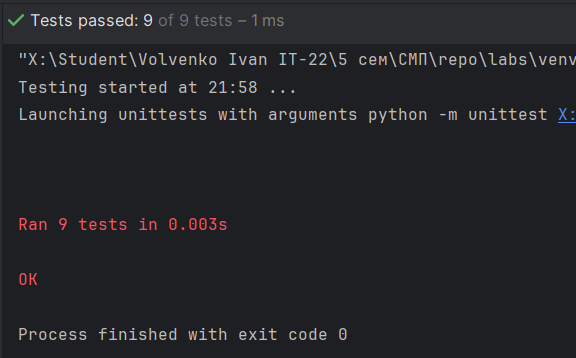
calculator.divide(7, "hello")

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

unittest.main()

**Виконання програми**

На рис. 1 зображенно виконання програми:



*Рис. 1 Виконання програми*

**Посилання на GitHub репозиторій:** <https://github.com/ArtemSadovenko/smp/tree/master>

**Висновки:** Виконавши ці завдання, я зробив 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 мого додатку, забезпечуючи його надійність і точність.