Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра інформаційних систем та мереж

ЗВІТ

про виконання лабораторної роботи № 6

**“ Розробка та Unit тестування Python додатку ”**

з дисципліни **“** **Спеціалізовані мови програмування”**

Виконав:

студент групи ІТ-32

ОЛЕНЮК О. М.

Прийняв:

ЩЕРБАК С. С.

**Львів – 2023**

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

**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.

**Код програми:**

сalculatorTesting.py:

import sys  
import unittest  
from importlib.machinery import SourceFileLoader  
sys.path.append(r'C:\Users\marki\PycharmProjects\pyt\_lab\lab\1')  
calculator = SourceFileLoader('calculator', r'C:\Users\marki\PycharmProjects\pyt\_lab\lab\1\calculator.py').load\_module()  
  
  
class CalculatorTesting(unittest.TestCase):  
  
 def test\_add(self):  
 self.assertEqual(calculator.add(3, 1), 4)  
 self.assertEqual(calculator.add(-10, 4.4), -5.6)  
 self.assertEqual(calculator.add(-9.3, 0), -9.3)  
  
 def test\_subtract(self):  
 self.assertEqual(calculator.subtract(12, 30), -18)  
 self.assertEqual(calculator.subtract(-11, 11), -22)  
 self.assertEqual(calculator.subtract(-8, -8), 0)  
  
 def test\_multiply(self):  
 self.assertEqual(calculator.multiply(22, 2), 44)  
 self.assertEqual(calculator.multiply(-2, -2), 4)  
 self.assertEqual(calculator.multiply(1, -1), -1)  
 self.assertEqual(calculator.multiply(0, 5), 0)  
  
 def test\_divide(self):  
 self.assertEqual(calculator.divide(12,4), 3)  
 self.assertEqual(calculator.divide(-5, 1), -5)  
  
 def test\_error\_handling(self):  
 with self.assertRaises(TypeError):  
 calculator.add("hello", "yara")  
 with self.assertRaises(TypeError):  
 calculator.subtract(1, "-")  
 with self.assertRaises(TypeError):  
 calculator.multiply("hi", "12")  
 with self.assertRaises(ArithmeticError):  
 calculator.divide(11, 0)  
 with self.assertRaises(ZeroDivisionError):  
 calculator.divide(1, 0)  
 with self.assertRaises(TypeError):  
 calculator.divide(3, "hello")  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

**Результат програми:**

На рис.1 зображений результат виконання роботи у програмі.

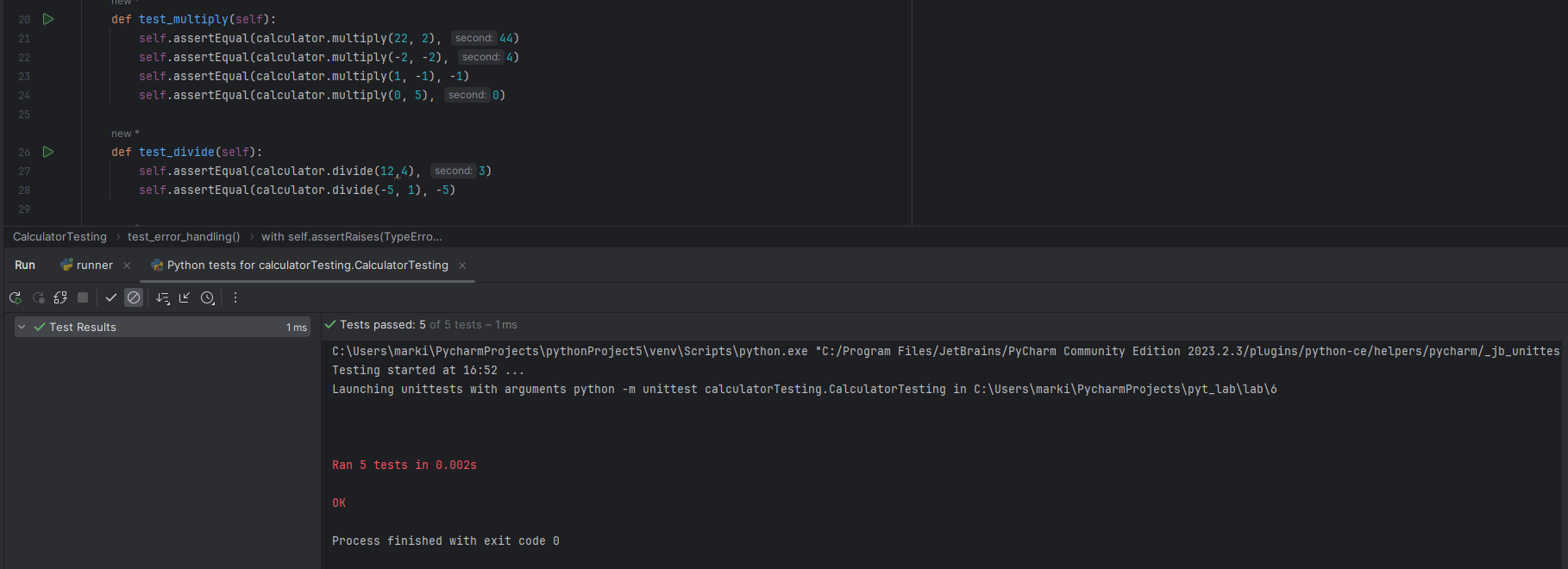


Рис.1

**Посилання на репозиторій**: <https://github.com/Oleksandr2004Oleniuk/SMP.git>

**Висновок**. у цій лабораторній роботі я створив набір юніт-тестів, які перевіряють правильність основних арифметичних операцій у додатку-калькуляторі. Ці тести допомогли виявити та виправити будь-які проблеми або помилки, які можуть виникнути під час розробки чи обслуговування додатку, забезпечуючи його надійність і точність.