МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра інформаційних систем та мереж

Лабораторна робота №6

з курсу

СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ

Виконала студентка

групи ІТ-21сп

**Нечайко П.В.**

Прийняв

**Щербак С.С.**

Львів - 2023

**Мета:** створення 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 виконаних завдань представлено нижче.

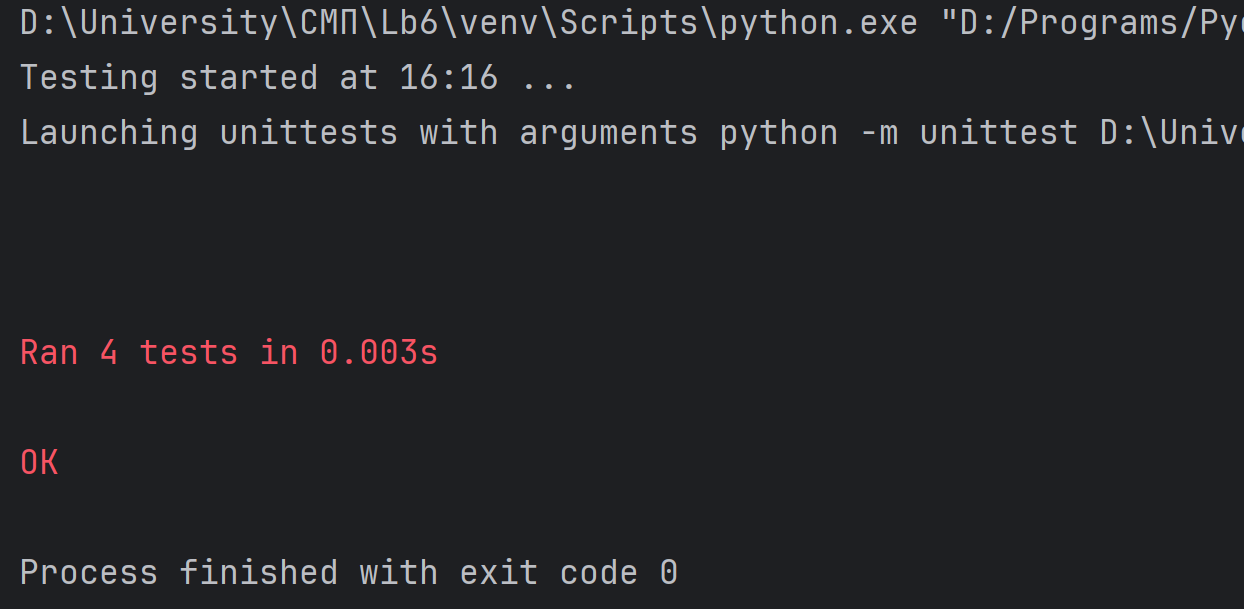
Клас lab2:

import math  
class Calculator:  
 def \_\_init\_\_(self):  
 self.result = None  
  
 def user\_input(self):  
 number1 = float(input("Enter the first number (the √ is taken from this number): "))  
 number2 = float(input("Enter the second number: "))  
 operator = input("Enter the operator (+, -, \*, /, ^, √, %): ")  
 return number1, operator, number2  
  
 def validate\_operator(self, operator):  
 valid\_operators = ['+', '-', '\*', '/', '^', '√', '%']  
 if operator not in valid\_operators:  
 print("Error: Invalid operator entered.")  
 return False  
 return True  
  
 def calculate(self, number1, operator, number2):  
 if operator == '+':  
 self.result = number1 + number2  
 elif operator == '-':  
 self.result = number1 - number2  
 elif operator == '\*':  
 self.result = number1 \* number2  
 elif operator == '/':  
 if number2 == 0:  
 print("Error: Division by zero is not possible.")  
 return  
 self.result = number1 / number2  
 elif operator == '^':  
 self.result = number1 \*\* number2  
 elif operator == '√':  
 self.result = math.sqrt(number1)  
 elif operator == '%':  
 self.result = number1 % number2  
 return self.result  
  
 def run\_calculator(self):  
 while True:  
 number1, operator, number2 = self.user\_input()  
 if not self.validate\_operator(operator):  
 continue  
 self.calculate(number1, operator, number2)  
 print(f"Result: {self.result:.2f}")  
 choice = input("Do you want to perform another calculation? (y-yes; another symbol-no): ").lower()  
 if choice != 'y':  
 break

Клас test:

import unittest  
from lab2 import Calculator  
  
class TestCalculator(unittest.TestCase):  
 def test\_addition(self):  
 calc = Calculator()  
 result = calc.calculate(5, '+', 3)  
 self.assertEqual(result, 8)  
  
 def test\_subtraction(self):  
 calc = Calculator()  
 result = calc.calculate('5', '-', 3)  
 self.assertEqual(result, 2)  
  
 def test\_multiplication(self):  
 calc = Calculator()  
 result = calc.calculate(5, '\*', 3)  
 self.assertEqual(result, 15)  
  
 def test\_division(self):  
 calc = Calculator()  
 result = calc.calculate(10, '/', 2)  
 self.assertEqual(result, 5)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 unittest.main()

На рисунку 1 зображено результат виконання програми.



*Рис.1 Робота unit-тестів*

**Висновок:** під час виконання лабораторної роботи я навчилася виявляти та виправляти будь-які проблеми або помилки, які можуть виникнути під час розробки чи обслуговування додатку, забезпечуючи його надійність і точність.