# **МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА” ІНСТИТУТ КОМП'ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ** **Кафедра ІСМ**

# **Звіт**

# **до лабораторної роботи №6**

# **З дисципліни “Спеціальні мови програмування”**

**Виконав:  
ст.гр.ІТ-31  
Шельвах Максим  
  
Прийняв:  
Щербак С.С.**

# **Львів — 2023**

**Тема роботи:** Розробка та Unit тестування Python додатку.

**Мета роботи:** 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.

Хід роботи

**Реалізований код:**

interface.py

from calculator.operations.operations import Addition, Subtraction, Multiplication, Division, Exponentiation, SquareRoot, Modulus

from calculator.exceptions.exceptions import exception

class Calculator:

def \_\_init\_\_(self):

self.result = 0.0

def getUserChoice(self):

return input('Operation: [ +, -, \*, /, ^, s, % ]: ')

def getInputNumber(self, prompt):

while True:

try:

num = float(input(prompt))

return num

except ValueError:

print('Please enter a valid number.')

def getInputData(self):

choice = self.getUserChoice()

num1 = self.getInputNumber('Enter the first number: ')

num2 = None

if choice not in ('√'):

num2 = self.getInputNumber('Enter the second number: ')

return choice, num1, num2

def performOperation(self, choice, num1, num2):

operation = None

if choice == '+':

operation = Addition(num1, num2)

elif choice == '-':

operation = Subtraction(num1, num2)

elif choice == '\*':

operation = Multiplication(num1, num2)

elif choice == '/':

operation = Division(num1, num2)

elif choice == '^':

operation = Exponentiation(num1, num2)

elif choice == 's':

operation = SquareRoot(num1, None)

elif choice == '%':

operation = Modulus(num1, num2)

if operation:

try:

self.result = operation.execute()

except Exception as e:

self.result = exception(e)

else:

self.result = exception(ValueError(

'Invalid choice. Please try again.'))

@staticmethod

def askToContinue():

result = input('Do you want to continue? (y/n) ')

return result == 'y'

def userInterface(self):

while True:

try:

choice, num1, num2 = self.getInputData()

self.performOperation(choice, num1, num2)

print(f'Result: {self.result}')

except Exception as e:

print(f'Something went wrong: {e}. Please try again.')

to\_continue = self.askToContinue()

if not to\_continue:

print('Good bye')

break

main.py

import unittest

from calculator.interface import Calculator

from calculator.tests.test import CalculatorTest

def main():

test\_calculator = unittest.TestLoader().loadTestsFromTestCase(CalculatorTest)

test\_loader = unittest.TextTestRunner()

test\_loader.run(test\_calculator)

input()

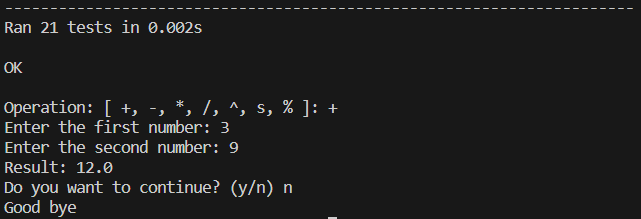
if '\_\_main\_\_' == \_\_name\_\_:

main()

calc = Calculator()

calc.userInterface()

**Результат виконання програми:**



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