МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра інформаційних систем та мереж



Лабораторна робота №1

з дисципліни «Спеціалізовані мови програмування»

на тему «Введення в Python»

Виконала студентка

групи РІ-32

Титиш О.І.

Прийняв:

Щербак С.С

Львів – 2024

**Мета:** створення консольної програми-калькулятора за допомогою основних синтаксичних конструкцій Python, з іншим завданням на заміну тестуванню та валідації.

**План роботи**

Завдання 1: Введення користувача

Створіть Python-програму, яка приймає введення користувача для двох чисел і оператора (наприклад, +, -, \*, /).

Завдання 2: Перевірка оператора

Перевірте чи введений оператор є дійсним (тобто одним із +, -, \*, /). Якщо ні, відобразіть повідомлення про помилку і попросіть користувача ввести дійсний оператор.

Завдання 3: Обчислення

Виконайте обчислення на основі введення користувача (наприклад, додавання, віднімання, множення, ділення) і відобразіть результат.

Завдання 4: Повторення обчислень

Запитайте користувача, чи він хоче виконати ще одне обчислення. Якщо так, дозвольте йому ввести нові числа і оператор. Якщо ні, вийдіть з програми.

Завдання 5: Обробка помилок

Реалізуйте обробку помилок для обробки ділення на нуль або інших потенційних помилок. Відобразіть відповідне повідомлення про помилку, якщо виникає помилка.

Завдання 6: Десяткові числа

Змініть калькулятор так, щоб він обробляв десяткові числа (плаваючу кому) для більш точних обчислень.

Завдання 7: Додаткові операції

Додайте підтримку додаткових операцій, таких як піднесення до степеня (^), квадратний корінь (√) і залишок від ділення (%).

Завдання 8: Функція пам'яті

Реалізуйте функцію пам'яті, яка дозволяє користувачам зберігати і відновлювати результати. Додайте можливості для зберігання та отримання значень з пам'яті.

Завдання 9: Історія обчислень

Створіть журнал, який зберігає історію попередніх обчислень, включаючи вираз і результат. Дозвольте користувачам переглядати історію своїх обчислень.

Завдання 10: Налаштування користувача

Надайте користувачам можливість налаштувати поведінку калькулятора, таку як зміну кількості десяткових розрядів, які відображаються, або налаштування функцій пам'яті.

**Код програми**

operators.py

*# Список дозволених операторів*OPERATORS = ['+', '-', '\*', '/', '^', '%', '√']

logger.py

import os  
  
def log\_operation(expression):  
 log\_file = os.path.join(os.path.dirname(\_\_file\_\_), 'log.txt')  
 with open(log\_file, 'a') as file:  
 file.write(expression + "\n")  
  
def log\_history(expression, result):  
 history\_file = os.path.join(os.path.dirname(\_\_file\_\_), 'history.txt')  
 with open(history\_file, 'a') as file:  
 file.write(f"{expression} = {result}\n")  
  
def show\_history():  
 history\_file = os.path.join(os.path.dirname(\_\_file\_\_), 'history.txt')  
 with open(history\_file, 'r') as file:  
 history = file.read()  
 print(history)

AppSettings.py

*# Налаштування користувача*decimal\_places = 2 *# Кількість десяткових знаків за замовчуванням*

functions.py

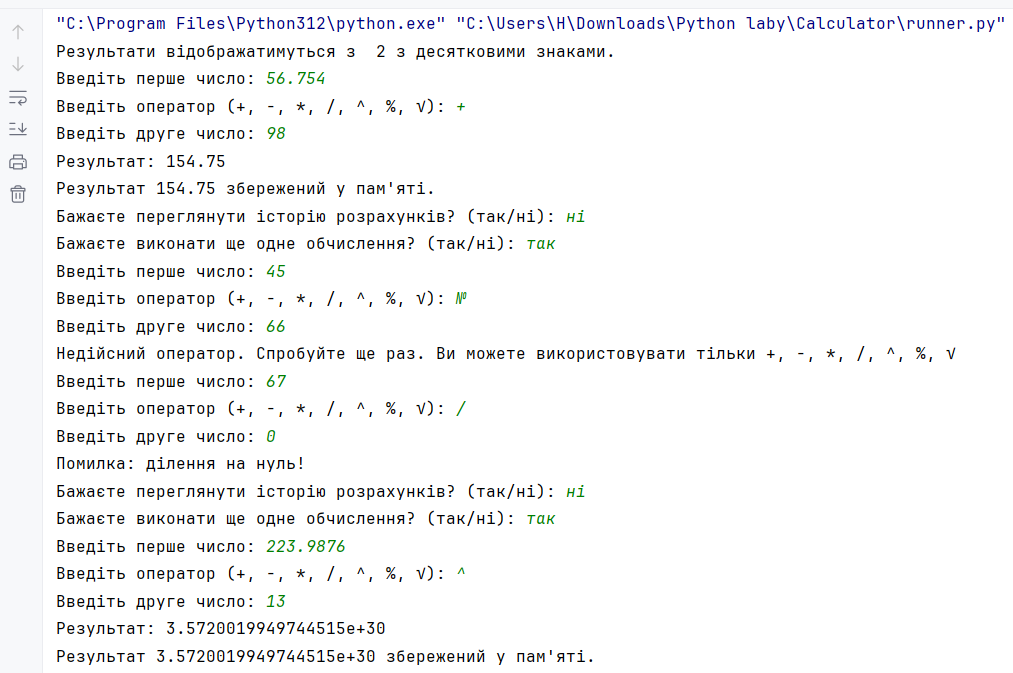
from const\_GlobalVariables.operators\_LAB1 import OPERATORS  
import math  
  
*# Основна функція обчислень*def calculate(num1, operator, num2=None):  
 match operator:  
 case '+':  
 return num1 + num2  
 case '-':  
 return num1 - num2  
 case '\*':  
 return num1 \* num2  
 case '/':  
 match num2:  
 case 0:  
 raise ZeroDivisionError("Помилка: ділення на нуль!")  
 case \_:  
 return num1 / num2  
 case '^':  
 return num1 \*\* num2  
 case '%':  
 return num1 % num2  
 case '√':  
 match num1:  
 case n if n < 0:  
 raise ValueError("Помилка: не можна знайти квадратний корінь з від'ємного числа!")  
 case \_:  
 return math.sqrt(num1)  
 case \_:  
 raise ValueError("Неправильний оператор")  
  
*# Перевірка правильності оператора*def is\_valid\_operator(operator):  
 return operator in OPERATORS

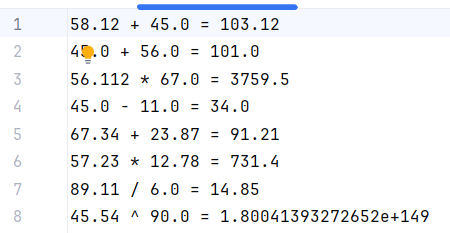
runner.py

from functions\_LAB1 import calculate, is\_valid\_operator  
from Shared.AppSettings\_LAB1 import decimal\_places  
from Shared.logs.logger\_LAB1 import log\_operation, log\_history, show\_history

def get\_input():  
 try:  
 num1 = float(input("Введіть перше число: "))  
 operator = input("Введіть оператор (+, -, \*, /, ^, %, √): ")  
 num2 = None  
 if operator != '√':  
 num2 = float(input("Введіть друге число: "))  
 return num1, operator, num2  
 except ValueError:  
 print("Неправильний ввід. Спробуйте знову.")  
 return get\_input()  
  
def store\_in\_memory(result):  
 global memory  
 memory = result  
 print(f"Результат {result} збережений у пам'яті.")  
  
def recall\_memory():  
 if memory is not None:  
 print(f"Збережене значення: {memory}")  
 return memory  
 else:  
 print("Пам'ять порожня.")  
 return None  
  
def ask\_to\_continue():  
 return input("Бажаєте виконати ще одне обчислення? (так/ні): ").lower() == 'так'  
  
def is\_valid\_operator(operator):  
 return operator in ['+', '-', '\*', '/', '^', '%', '√']  
  
def calculator():  
 print(f"Результати відображатимуться з {decimal\_places} з десятковими знаками.")  
 while True:  
 num1, operator, num2 = get\_input()  
  
 match is\_valid\_operator(operator):  
 case False:  
 print("Недійсний оператор. Спробуйте ще раз. Ви можете використовувати тільки +, -, \*, /, ^, %, √ ")  
 continue  
 case True:  
 try:  
 result = calculate(num1, operator, num2)  
 result = round(result, decimal\_places)  
 print(f"Результат: {result}")  
 store\_in\_memory(result)  
  
 expression = f"{num1} {operator} {num2 if operator != '√' else ''}"  
 log\_operation(f"{expression} = {result}")  
 log\_history(expression, result)  
  
 except (ZeroDivisionError, ValueError) as e:  
 print(e)  
  
 match input("Бажаєте переглянути історію розрахунків? (так/ні): ").lower():  
 case 'так':  
 show\_history()  
 case \_:  
 pass  
  
 match ask\_to\_continue():  
 case False:  
 break  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 calc = calculator()

**Результат виконання програми**

****

****

*Рисунок 1, 2. Результати виконання*

Посилання на GitHub: <https://github.com/olyyaaa>

**Висновок:** На цій лабораторній роботі було створено простий консольний калькулятор на Python, який може виконувати арифметичні операції, обробляти помилки та надавати користувачу зручний інтерфейс. Проект допоміг вивчити основний синтаксис Python і концепції, такі як введення користувача, умовні оператори, цикли та обробка помилок.