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

Виконав:

студент групи РІ-21сп

Богдан Д.А.

Прийняв:

Шербак С.С.

Львів-2024

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

**Завдання**

**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**

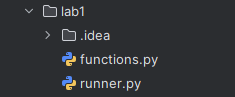
Створіть журнал, який зберігає історію попередніх обчислень, включаючи вираз і результат. Дозвольте користувачам переглядати історію своїх обчислень.

**Завдання 10: Налаштування користувача**

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

**Виконання роботи**

**Структура файлів:**



**Вміст файлу functions.py:**

import math  
  
from Shared.constants.GlobalVariables import history  
from Shared.decimal\_settings.settings import DecimalSettings  
from Shared.validators.math\_utils import validate\_operator  
decimal\_settings = DecimalSettings()  
def set\_memory\_value(result):  
 global memory\_value  
 memory\_value = result  
  
def get\_memory\_value():  
 return memory\_value  
  
def calculate(num1, num2, operator):  
 try:  
 if operator == '+':  
 return num1 + num2  
 elif operator == '-':  
 return num1 - num2  
 elif operator == '\*':  
 return num1 \* num2  
 elif operator == '/':  
 if num2 == 0:  
 raise ZeroDivisionError("Ділення на нуль не дозволено")  
 return num1 / num2  
 elif operator == '^':  
 return num1 \*\* num2  
 elif operator == '%':  
 if num2 == 0:  
 raise ZeroDivisionError("Ділення на нуль не дозволено")  
 return num1 % num2  
 elif operator == '√':  
 if num1 < 0:  
 raise ValueError("Квадратний корінь з від'ємного числа не дозволений")  
 return math.sqrt(num1)  
 raise ZeroDivisionError  
 except ZeroDivisionError as e:  
 return e  
 except ValueError as e:  
 return e  
  
def get\_input():  
 while True:  
 num1\_input = input("Введіть перше число або 'm' для використання пам'яті: ").lower()  
 if num1\_input == 'm' and get\_memory\_value() is not None:  
 num1 = get\_memory\_value()  
 print(f"Використано число з пам'яті: {num1}")  
 break  
 elif num1\_input == 'm' and get\_memory\_value() is None:  
 print("Помилка: пам'ять порожня.")  
 else:  
 try:  
 num1 = float(num1\_input)  
 break  
 except ValueError:  
 print("Помилка: введіть дійсне число або 'm'.")  
  
 operator = input("Введіть оператор (+, -, \*, /, ^, %, √): ")  
 while not validate\_operator(operator):  
 print("Неправильний оператор! Спробуйте ще раз.")  
 operator = input("Введіть оператор (+, -, \*, /, ^, %, √): ")  
  
 num2 = None  
 if operator != '√':  
 while True:  
 num2\_input = input("Введіть друге число або 'm' для використання пам'яті: ").lower()  
 if num2\_input == 'm' and get\_memory\_value() is not None:  
 num2 = get\_memory\_value()  
 print(f"Використано число з пам'яті: {num2}")  
 break  
 elif num2\_input == 'm' and get\_memory\_value() is None:  
 print("Помилка: пам'ять порожня.")  
 else:  
 try:  
 num2 = float(num2\_input)  
 break  
 except ValueError:  
 print("Помилка: введіть дійсне число або 'm'.")  
 return num1, num2, operator  
  
def memory\_store(result):  
 return result  
  
def ask\_for\_another\_calculation():  
 while True:  
 again = input("Бажаєте виконати ще одне обчислення? (y/n): ").lower()  
 if again == 'y':  
 return True  
 elif again == 'n':  
 return False  
 else:  
 print("Неправильний вибір, спробуйте знову.")  
def perform\_calculation():  
 while True:  
 num1, num2, operator = get\_input()  
 result = calculate(num1, num2 if operator != '√' else None, operator)  
 expression = f"{num1} {operator} {num2}" if operator != '√' else f"√{num1}"  
  
 if isinstance(result, (int, float)):  
 result = round(result, decimal\_settings.get\_decimal\_precision())  
 history.append(f"{expression} = {result}")  
 print(f"Результат: {result}")  
 if isinstance(result, (int, float)):  
 memory\_choice = input("Зберегти результат у пам'яті? (y/n): ").lower()  
 if memory\_choice == 'y':  
 set\_memory\_value(result)  
 print(f"Результат {result} збережено в пам'ять.")  
 else:  
 print("Помилка не може бути збережена в пам'ять.")  
  
 if not ask\_for\_another\_calculation():  
 break  
  
def validate\_operator(operator):  
 return operator in ['+', '-', '\*', '/', '^', '%', '√']  
  
def show\_memory():  
 if get\_memory\_value() is not None:  
 print(f"Збережене в пам'яті число: {get\_memory\_value()}")  
 else:  
 print("Пам'ять порожня.")  
  
def show\_history():  
 if history:  
 print("Історія обчислень:")  
 for record in history:  
 print(record)  
 else:  
 print("Історія порожня.")  
  
def exit\_calculator():  
 print("Дякуємо за використання калькулятора!")  
 return True  
  
def main():  
 while True:  
 print("\n--- Меню ---")  
 print("1. Виконати нове обчислення")  
 print("2. Переглянути історію обчислень")  
 print("3. Змінити кількість знаків після коми")  
 print("4. Очистити число збережене в пам'яті")  
 print("5. Показати збережене число в пам'яті")  
 print("6. Вийти\n")  
 choice = input("Оберіть дію (1, 2, 3, 4, 5, 6): ")  
  
 match choice:  
 case '1':  
 perform\_calculation()  
 case '2':  
 show\_history()  
 case '3':  
 decimal\_settings.set\_decimal\_precision()  
 print(f"Кількість знаків після коми змінено на {decimal\_settings.get\_decimal\_precision()}.")  
 case '4':  
 set\_memory\_value(0)  
 print("Збережене значення очищено")  
 case '5':  
 show\_memory()  
 case '6':  
 if exit\_calculator():  
 break  
 case \_:  
 print("Невірний вибір, спробуйте знову.")

Результатом виконання є простий консольний калькулятор з можливістю зберігання та використання значення з пам’яті та історією.

На рисунку 1 зображені обчислення за допомогою калькулятора.

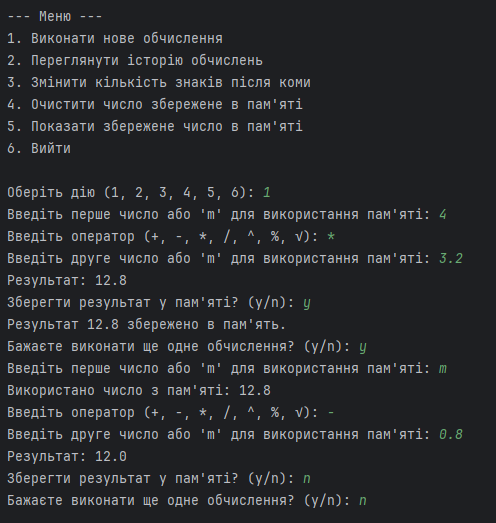
****

Рисунок 1 – Обчислення в консольному калькуляторі

На рисунку 2 зображено виклик історії обчислень калькулятора.

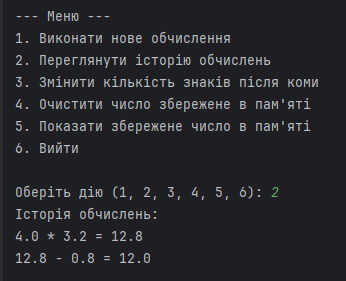
****

Рисунок 2 – Перегляд історії калькулятора

**Висновок:** У ході виконання лабораторної роботи я створив створив простий консольний калькулятор на Python, який може виконувати арифметичні операції, оперувати з числом в пам’яті, обробляти помилки та надавати користувачу зручний інтерфейс.