ЗВІТ

про виконання лабораторної роботи № 1

«Введення в Python»

з дисципліни

«Спеціалізовані мови програмування»

студентки групи РІ-32

Мацкули Ангеліни Іванівни

прийняв доц. кафедри ІСМ

Щербак С.С.

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.

Текст main програми:

```
from itertools import repeat

from input import get_user_input
from operator_check import validate_operator
from calculate import calculate
from recalculate import ask_for_repeat
from errors import safe_calculate
from memory import memory_clear, memory_store, memory_recall, memory_add
from history import add_to_history, show_history
from settings import set_decimal_precision

# Початкові налаштування
precision = 2  # Кількість десяткових знаків за замовчуванням
memory = 0  # Початкове значення пам'яті

def handle_memory_input(value):
    """Перевірка чи введене значення є командою 'MR', якщо так — повертає значення з пам'яті."""
   if value.lower() == 'mr':
       return memory_recall()
   else:
       return float(value)

def main():
   global memory, precision

   while True:
       # Завдання 1: Введення користувача з можливістю використання пам'яті
        num1_input = input("Введіть перше число або 'MR' для числа з пам'яті: ")
       num1 = handle_memory_input(num1_input)

        num2_input = input("Введіть друге число або 'MR' для числа з пам'яті: ")
       num2 = handle_memory_input(num2_input)

       operator = input("Введіть оператор (+, -, *, /, ^, %, √): ")

       # Завдання 2: Перевірка оператора
       if not validate_operator(operator):
```

```
            continue

        # Завдання 5: Обробка помилок та обчислення
        result = safe_calculate(num1, num2, operator)

        # Перевірка результату та виведення
        if isinstance(result, float):
            rounded_result = round(result, precision)
            print(f"Результат: {rounded_result}")

            # Завдання 9: Додавання обчислення в історію
            add_to_history(f"{num1} {operator} {num2}", rounded_result)

        else:
            print(result)

        # Завдання 4: Запит на повторення обчислень
        if not ask_for_repeat():
            break

        # Запит на налаштування
           user_choice = input("Налаштувати калькулятор? (пам'ять, історія,
розряди) або 'продовжити': ").lower()

        if user_choice == 'пам\'ять':
             memory_action = input("Виберіть дію з пам'яттю (MC, MR, MS, M+):
").upper()
            if memory_action == 'MC':
                memory_clear()
                print("Пам'ять очищена.")
            elif memory_action == 'MR':
                print(f"Число з пам'яті: {memory_recall()}")
            elif memory_action == 'MS':
                memory_store(rounded_result)
                print(f"Значення {rounded_result} збережене в пам'ять.")
            elif memory_action == 'M+':
                memory_add(rounded_result)
                   print(f"Додано {rounded_result} до пам'яті, нове значення:
{memory_recall()}.")
        elif user_choice == 'історія':
            show_history()
        elif user_choice == 'розряди':
            precision = set_decimal_precision()
                     print(f"Кількість десяткових розрядів встановлена на
{precision}.")


if __name__ == "__main__":
   main()
```

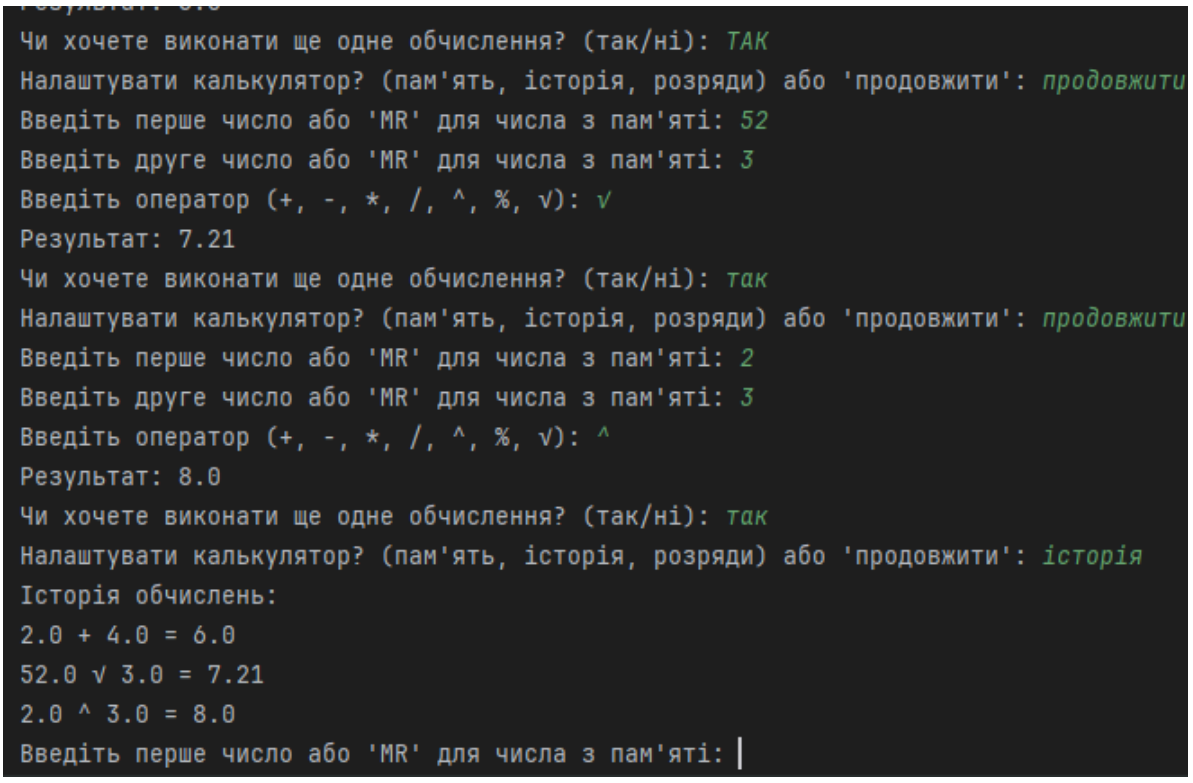

рис.1

Посилання на GitHub репозиторій: https://github.com/aangll/calc_nulp

Висновки: Виконавши ці завдання, було створено простий консольний калькулятор на Python, який може виконувати арифметичні операції, обробляти помилки та надавати користувачу зручний інтерфейс. Цей проект допоміг нам вивчити основний синтаксис Python і концепції, такі як введення користувача, умовні оператори, цикли та обробка помилок.