Міністерство освіти і науки України

Національний університет «Львівська політехніка»



**ЗВІТ**

про виконання лабораторної роботи № 1

з курсу:

«Спеціалізовані мови програмування»

**Виконала:**

студентка гр. ІТ-31

Ірина ПЕРХУН

**Прийняв:**

Сергій ЩЕРБАК

Львів 2023

**Тема:** Введення в Python

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

**Хід 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.

*import* math *# To use the math function sqrt*

*# Function to check if a data can be converted to a float. If data can we convert it to float*

def is\_valid\_float(data):

*try*:

      float(data)

*return* True

*except* ValueError: *# Error handling ValueError to check numbers*

*return* False

*# Prompt the user to enter the number*

def get\_number\_from\_user(prompt):

*while* True:

      num\_input = input(prompt)

*if* is\_valid\_float(num\_input):

*return* float(num\_input)

*else*:

          print("Invalid input. Please enter a valid number.")

*# Prompt the user to enter an operator*

def get\_operator\_from\_user():

    valid\_operators = ['+', '-', '\*', '/', '^', '√', '%']

*while* True:

      operator = input("Enter the operator (+, -, \*, /,  ^, √, %): ")

*if* operator in valid\_operators:

*return* operator

*else*:

          print("Invalid operator. Please enter a valid operator from the list (+, -, \*, /, ^, √, %): ")

*# Perform calculations based on the operator which user entered*

def perform\_calculation(number1, number2, operator, decimal\_places):

    result = None

*try*:

*if* operator == '+':

        result = number1 + number2

*elif* operator == '-':

        result = number1 - number2

*elif* operator == '\*':

        result = number1 \* number2

*elif* operator == '/':

*# Check for division by zero*

*if* number2 == 0:

*# Generation of an exception, we create an instance of an exception of type ZeroDivisionError*

*raise* ZeroDivisionError("Error: Division by zero is not allowed")

*else*:

            result = number1 / number2

*elif* operator == '^':  *# exponentiation*

              result = number1 \*\* number2

*elif* operator == '√':  *# sqrt*

              result = math.sqrt(number1)

*elif* operator == '%':  *# Remainder from division*

           result = number1 % number2

*return* result

*# Exception handling*

*except* ZeroDivisionError *as* e: *# The variable e contains information about the exception*

        print(e)

def main():

    memory = None

    calculation\_records = []

*while* True:

       number1 = get\_number\_from\_user("Enter the first number: ")

       number2 = get\_number\_from\_user("Enter the second number: ")

       operator = get\_operator\_from\_user()

       decimal\_places = int(input("Enter the number of decimal places: "))

       result = perform\_calculation(number1, number2, operator, decimal\_places)

*if* result is not None:

            memory = result

*# Store the calculation and expression in history*

            expression = f"{number1:.{decimal\_places}f} {operator} {number2:.{decimal\_places}f} = {result:.{decimal\_places}f}"

            calculation\_records.append({"expression": expression, "result": result})

*# Display the result with the specified decimal places*

            formatted\_result = f"Result: {result:.{decimal\_places}f}"

            print(formatted\_result)

*# Display calculation history*

            history\_str = ", ".join([entry["expression"] *for* entry *in* calculation\_records])

            print(f"Memory operator: {history\_str}")

*else*:

*# If result is None, continue to the next iteration without storing the calculation*

*pass*

*# Ask user if he wants to perform one more calculation*

       another\_calculation = input("Do you want to perform another calculation? (yes/no): ")

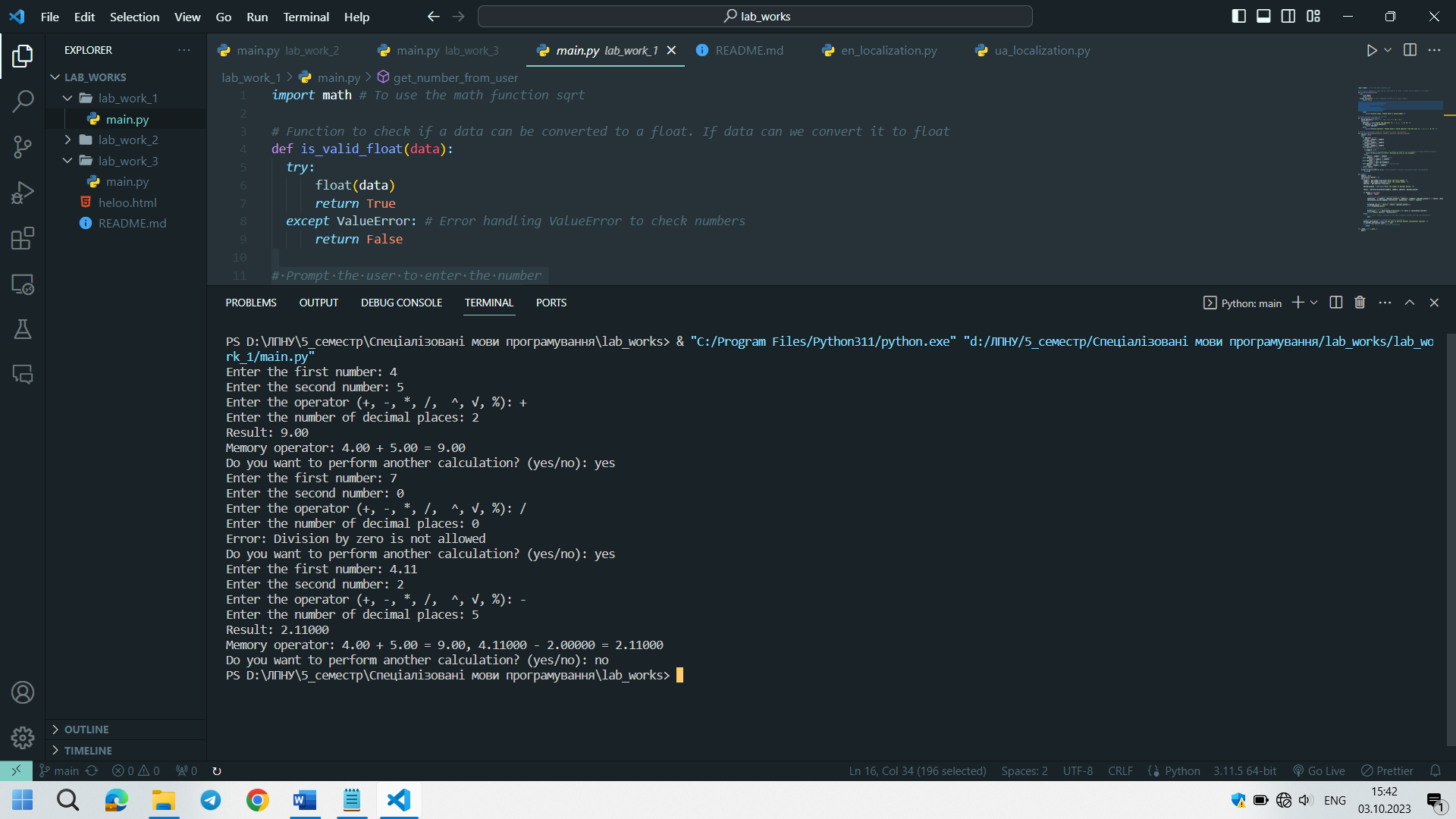
*if* another\_calculation.lower() != 'yes':

*# Exit if user doesn't want one more calculation*

*break*

*if* \_\_name\_\_ == "\_\_main\_\_":

    main()



*Рис.1 – Результат виконання програми*

**Висновок:** на даній лабораторній роботі я створила простий консольний калькулятор на Python, який може виконувати арифметичні операції, обробляти помилки та надавати користувачу зручний інтерфейс. Цей проект допоміг вивчити основний синтаксис Python і концепції, такі як введення користувача, умовні оператори, цикли та обробка помилок.