Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Лабораторна роботи №1

з дисципліни «Спеціалізовані мови програмування»

на тему

«Введення в Python»

Виконав:

Ложовський В. А.

Перевірив:

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.  
  
**Реалізація:**

**Файл AppSettings.py:**  
import lab1.constants.constants as const

class AppSettings:

    def \_\_init\_\_(self):

        self.decimal\_places = const.DECIMALS

        self.console\_color = "white"

    def change\_decimals(self, decimals):

        self.decimal\_places = decimals

    def change\_color(self, color):

        self.console\_color = color

**Папка functions, файл calcFunctions.py:**import math

def add(a, b):

    return a + b

def subtract(a, b):

    return a - b

def multiply(a, b):

    return a \* b

def divide(a, b):

    try:

        return a / b

    except ZeroDivisionError:

        return "Error: Division by zero"

def power(a, b):

    return a \*\* b

def sqrt(a):

    return math.sqrt(a)

def modulus(a, b):

    return a % b

**Папка interface, файл userInterface.py:**

def get\_user\_input(prompt):

return input(prompt)

def display\_output(message):

print(message)

**Папка functions, файл Calculator.py:**

import math

class Calculator:

    def \_\_init\_\_(self):

        self.num1 = 0.0

        self.num2 = 0.0

        self.operator = ''

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

    def get\_user\_input(self):

        expression = input("Enter expression (e.g. '5 + 3' or '√9'): ").strip()

        return self.parse\_expression(expression)

    def parse\_expression(self, expression):

        if '√' in expression:

            self.operator = '√'

            self.num1 = float(expression[1:])

            self.num2 = None

        else:

            for operator in self.valid\_operators:

                if operator in expression:

                    self.num1, self.num2 = map(float, expression.split(operator))

                    self.operator = operator

                    break

        if self.operator not in self.valid\_operators:

            raise ValueError("Invalid operator!")

    def check\_operator(self):

        if self.operator not in self.valid\_operators:

            raise ValueError(f"Invalid operator: {self.operator}")

    def calculate(self):

        try:

            if self.operator == '+':

                return self.num1 + self.num2

            elif self.operator == '-':

                return self.num1 - self.num2

            elif self.operator == '\*':

                return self.num1 \* self.num2

            elif self.operator == '/':

                if self.num2 == 0:

                    raise ZeroDivisionError("Cannot divide by zero!")

                return self.num1 / self.num2

            elif self.operator == '^':

                return self.num1 \*\* self.num2

            elif self.operator == '√':

                if self.num1 < 0:

                    raise ValueError("Cannot take square root of negative number!")

                return math.sqrt(self.num1)

            elif self.operator == '%':

                return self.num1 % self.num2

        except (ValueError, ZeroDivisionError) as e:

            print(f"Error: {e}")

            return None

    def display\_result(self, result):

        if result is not None:

            print(f"Result: {result:.2f}")

        else:

            print("Error occurred during calculation.")

    def run(self):

        while True:

            try:

                self.get\_user\_input()

                result = self.calculate()

                self.display\_result(result)

            except ValueError as e:

                print(e)

            choice = input("Do you want to perform another calculation? (y/n): ").lower()

            if choice != 'y':

                break

**Файл main.py:**

from lab1.functions.calcFunctions import add, subtract, multiply, divide, power, sqrt, modulus

from lab1.init import memory, app\_settings

import lab1.constants.constants as const

history = []

def logToFile(expression, result):

    with open("lab1/logs/calculator\_history.txt", "a") as logFile:

        logFile.write(f"{expression} = {result}\n")

def displayHistory():

    if history:

        print("\n--- Calculation History ---")

        for record in history:

            print(record)

    else:

        print("No history available.")

def parse\_expression(expression):

    try:

        expression = expression.replace(' ', '')

        if '√' in expression:

            num = float(expression[1:])

            return num, None, '√'

        for operator in const.VALID\_OPERATORS:

            if operator in expression:

                num1, num2 = expression.split(operator)

                return float(num1), float(num2), operator

    except ValueError:

        raise ValueError("Invalid expression format")

def main():

    while True:

        print("\nTo view calculation history, type 'history'")

        userInput = input("Enter an expression (e.g., '5 + 3' or 'exit' to quit): ")

        if userInput.lower() == "history":

            displayHistory()

            continue

        elif userInput.lower() == "exit":

            break

        try:

            num1, num2, operator = parse\_expression(userInput)

            if operator not in const.VALID\_OPERATORS:

                print("Invalid operator!")

                continue

            if operator == '+':

                result = add(num1, num2)

            elif operator == '-':

                result = subtract(num1, num2)

            elif operator == '\*':

                result = multiply(num1, num2)

            elif operator == '/':

                result = divide(num1, num2)

            elif operator == '^':

                result = power(num1, num2)

            elif operator == '√':

                result = sqrt(num1)

            elif operator == '%':

                result = modulus(num1, num2)

            expression = f"{num1} {operator} {num2}" if operator != '√' else f"{operator}({num1})"

            result = f"{expression} = {round(result, app\_settings.decimal\_places)}"

            history.append(result)

            logToFile(expression, result)

            print(result)

        except ValueError:

            print("Error: Invalid input.")

        choice = input("Do you want to perform another calculation? (y/n): ")

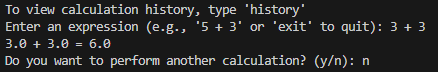
        if choice.lower() != 'y':

            break

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

    main()

**Результат виконання:**

****

**Висновок:** на цій лабораторній роботі я створив простий консольний калькулятор на Python, який може виконувати арифметичні операції, обробляти помилки та надавати користувачу зручний інтерфейс.