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:

ст. гр. РІ-21сп

Костецький І.С.

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:**

A screenshot of a computer

Description automatically generated

**Вміст файлу functions.py:**

import sys

import os

sys.path.append(os.path.abspath(os.path.dirname(file)))

import math

from variables import decimals, memory, history\_file

history\_file = os.path.join(os.path.dirname(file), history\_file)

def set\_decimals(value):

global decimals

decimals = value

def get\_decimals():

return decimals

def set\_memory(value):

global memory

memory = value

def get\_memory():

return memory

def check\_operator(operator):

return operator in ['+', '-', '\*', '/', '^', '√', '%']

def get\_operator():

while True:

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

if check\_operator(operator):

return operator

else:

print("Невірний оператор. Будь ласка, введіть правильний оператор.")

def save\_to\_history(entry, history\_file):

with open(history\_file, "a") as file:

file.write(entry + "\n")

def display\_history(history\_file):

try:

with open(history\_file, "r") as file:

history = file.readlines()

if not history:

print("Історія обчислень пуста.")

else:

print("Історія обчислень:")

for entry in history:

print(entry.strip())

except FileNotFoundError as e:

raise FileNotFoundError("Історія обчислень відсутня.")

def change\_settings():

new\_decimals = input(f"Поточна кількість десяткових знаків: {get\_decimals()}. Введіть нову кількість: ")

if new\_decimals.isalpha():

raise TypeError("Помилка! Введіть ціле (додатнє) число для кількості десяткових знаків.")

elif int(new\_decimals) < 0:

raise ValueError("Помилка! Введіть ціле (додатнє) число для кількості десяткових знаків.")

set\_decimals(int(new\_decimals))

print(f"Кількість десяткових знаків змінено на {new\_decimals}.")

def store\_memory\_value():

try:

custom\_value = float(input("Введіть число для збереження в пам'ять: "))

set\_memory(custom\_value)

print(f"Число {custom\_value} збережене в пам'ять (M).")

except ValueError as e:

raise ValueError("Помилка! Введіть дійсне число.")

def get\_input():

num1\_input = input("Введіть перше число (або 'm' для використання збереженого значення): ")

if num1\_input.lower() == 'm':

num1 = get\_memory()

elif num1\_input.isalpha():

raise TypeError("Необхідно ввести число")

elif num1\_input.isnumeric():

num1 = float(num1\_input)

else:

raise ValueError("Необхідно ввести число")

operator = get\_operator()

if operator != '√':

while True:

num2\_input = input("Введіть друге число (або 'm' для використання збереженого значення): ")

if num2\_input.lower() == 'm':

num2 = get\_memory()

elif num1\_input.isalpha():

raise TypeError("Необхідно ввести число")

elif num2\_input.isnumeric():

num2 = float(num2\_input)

else:

raise ValueError("Необхідно ввести число")

break

else:

num2 = None

return num1, operator, num2

def calculate(num1, operator, num2):

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 неможливе")

return num1 / num2

elif operator == "^":

return num1 \*\* num2

elif operator == "√":

if num1 < 0:

raise ValueError("Квадратний корінь з від'ємного числа неможливий")

return math.sqrt(num1)

elif operator == "%":

if num2 == 0:

raise ZeroDivisionError("Неможливо отримати залишок з нуля")

return num1 % num2

else:

raise ValueError("Недійсний оператор!")

def calculator(history\_file):

num1, operator, num2 = get\_input()

result = calculate(num1, operator, num2)

result = round(result, get\_decimals())

print(f"Результат: {result}")

save\_to\_history(f"{num1} {operator} {num2 if num2 is not None else ''} = {result}", history\_file)

save\_to\_memory = input("Бажаєте зберегти результат у пам'ять? (Y/N): ").lower()

if save\_to\_memory == 'y':

set\_memory(result)

print(f"Результат {result} збережений у пам'ять (M).")

try\_again = input("Бажаєте повторити обчислення? (Y/N): ").lower()

if try\_again == 'y':

calculator(history\_file)

def main():

while True:

print("\nМеню:")

print("1. Провести обчислення")

print("2. Вивести історію обчислень")

print("3. Записати число у памʼять")

print("4. Вивести число з памʼяті")

print("5. Поміняти налаштування (кількість десяткових знаків)")

print("6. Вийти з калькулятора")

choice = input("Оберіть опцію (1-6): ")

if choice == '1':

try:

calculator(history\_file)

except ValueError as e:

print(e)

except TypeError as e:

print(e)

except ZeroDivisionError as e:

print(e)

elif choice == '2':

try:

display\_history(history\_file)

except FileNotFoundError as e:

print(e)

elif choice == '3':

try:

store\_memory\_value()

except ValueError as e:

print(e)

elif choice == '4':

print(f"число збережене у пам'яті: {get\_memory()}")

elif choice == '5':

try:

change\_settings()

except ValueError as e:

print(e)

except TypeError as e:

print(e)

elif choice == '6':

print("Вихід з калькулятора.")

break

else:

print("Невірний вибір. Спробуйте ще раз.")

Результат виконання: а саме виконання обчислення на рисунку 1 та перегляд історії обчислень на рисунку 2.

A computer screen shot of a black screen

Description automatically generated

Рисунок 1 – Обчислення

A screen shot of a computer

Description automatically generated

Рисунок 2 – Перегляд історії калькулятора

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