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

Виконала:

ст. гр. ІТ-32

Ольга ЧИГИРИК

Прийняв

доцент каф. ІСМ:

Сергій ЩЕРБАК

|  |  |
| --- | --- |
| **Балів** | **Дата** |
|  |  |

Львів-2023

**Мета:** 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.

**Main.py**

#from console1 import choose\_task  
from Scripts.all\_python\_labs.shared\_lib.console\_ui import ConsoleUI  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 console\_ui = ConsoleUI()  
 console\_ui.choose\_task()  
 #choose\_task()

**Console.py**

from functions1 import get\_input, is\_valid\_operator, Calculator, safe\_calculate  
from memory\_manager1 import MemoryManager  
from history\_manager1 import HistoryManager  
  
def choose\_task():  
 *"""Дозволяє користувачеві вибрати завдання для виконання."""* calculator = Calculator()  
 memory\_manager = MemoryManager()  
 history\_manager = HistoryManager()  
  
 while True:  
 print("\nВиберіть завдання для виконання:")  
 print("1: Введення користувача і обчислення")  
 print("2: Перевірка історії обчислень")  
 print("3: Отримання значення з пам'яті")  
 print("4: Вихід з програми")  
  
 choice = input("Ваш вибір: ")  
  
 if choice == "1":  
 num1, num2, operator = get\_input()  
 while not is\_valid\_operator(operator):  
 print("Недійсний оператор. Спробуйте ще раз.")  
 operator = input("Введіть оператор (+, -, \*, /, ^, √, %): ")  
 result = safe\_calculate(num1, num2, operator)  
 print(f"Результат: {result}")  
 history\_manager.add(f"{num1} {operator} {num2}", result)  
 memory\_manager.store(result)  
  
 elif choice == "2":  
 history\_manager.view()  
  
 elif choice == "3":  
 value = memory\_manager.retrieve()  
 print(f"Значення у пам'яті: {value}")  
  
 elif choice == "4":  
 print("Вихід з програми.")  
 break  
  
 else:  
 print("Невідомий вибір. Спробуйте ще раз.")

**Functions.py**

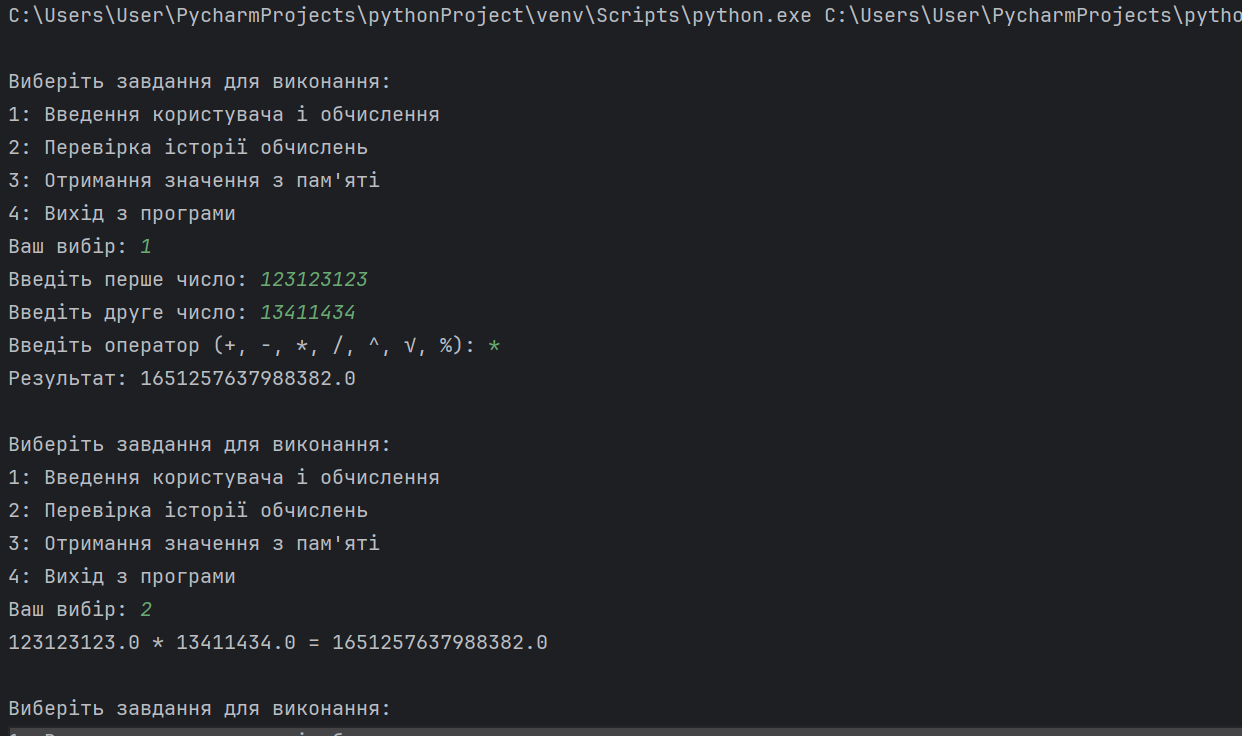
*"""  
This module provides a simple calculator application which allows users to perform basic arithmetic operations.  
It includes functions for user input, operator validation, and safe calculation, along with a Calculator class for performing calculations.  
"""*# Завдання 1: Введення користувача  
def get\_input():  
 *"""  
 Obtain input from the user.  
  
 Asks the user to input two numbers and an operator.  
  
 Returns:  
 tuple: A tuple containing two floats (num1, num2) and a string (operator).  
 """* """Отримання введення від користувача."""  
 num1 = float(input("Введіть перше число: "))  
 num2 = float(input("Введіть друге число: "))  
 operator = input("Введіть оператор (+, -, \*, /, ^, √, %): ")  
 return num1, num2, operator  
  
# Завдання 2: Перевірка оператора  
def is\_valid\_operator(operator):  
 *"""  
 Check if the given operator is valid.  
  
 Parameters:  
 operator (str): The operator to check.  
  
 Returns:  
 bool: True if the operator is valid, False otherwise.  
 """* """Перевірка чи оператор є дійсним."""  
 valid\_operators = ['+', '-', '\*', '/', '^', '√', '%']  
 return operator in valid\_operators  
  
# Завдання 3: Обчислення - включено до класу Калькулятора  
class Calculator:  
 *"""  
 The Calculator class provides methods to perform basic arithmetic calculations.  
  
 The calculations are performed based on the operator provided by the user. Supported operators include addition (+),  
 subtraction (-), multiplication (\*), division (/), exponentiation (^), root (√), and modulo (%).  
 """* @staticmethod  
 def calculate(num1, num2, operator):  
 *"""  
 Perform calculations based on user input.  
  
 Parameters:  
 num1 (float): The first number.  
 num2 (float): The second number.  
 operator (str): The arithmetic operator.  
  
 Returns:  
 float: The result of the arithmetic operation.  
 """* """Виконання обчислень на основі введення користувача."""  
 if operator == '+':  
 return num1 + num2  
 elif operator == '-':  
 return num1 - num2  
 elif operator == '\*':  
 return num1 \* num2  
 elif operator == '/':  
 return num1 / num2  
 elif operator == '^':  
 return num1 \*\* num2  
 elif operator == '√':  
 return num1 \*\* (1 / num2)  
 elif operator == '%':  
 return num1 % num2  
  
# Завдання 5: Обробка помилок  
def safe\_calculate(num1, num2, operator):  
 *"""  
 Safely perform calculations with error handling.  
  
 Parameters:  
 num1 (float): The first number.  
 num2 (float): The second number.  
 operator (str): The arithmetic operator.  
  
 Returns:  
 float or str: The result of the arithmetic operation, or an error message if an error occurs.  
 """* """Безпечне виконання обчислень з обробкою помилок."""  
 try:  
 return Calculator.calculate(num1, num2, operator)  
 except ZeroDivisionError:  
 return "Помилка: ділення на нуль!"  
 except Exception as e:  
 return f"Помилка: {e}"  
  
# Завдання 10: Налаштування користувача  
settings = {  
 "decimal\_places": 2  
}  
  
def update\_settings(key, value):  
 *"""  
 Update the application settings.  
  
 Parameters:  
 key (str): The setting to update.  
 value: The new value for the setting.  
 """* """Оновлення налаштувань."""  
 settings[key] = value  
  
def get\_setting(key):  
 *"""  
 Get the value of a specific setting.  
  
 Parameters:  
 key (str): The setting to retrieve.  
  
 Returns:  
 The value of the setting, if it exists.  
 """* """Отримання значення налаштувань."""  
 return settings.get(key)

**Memory\_manager.py**

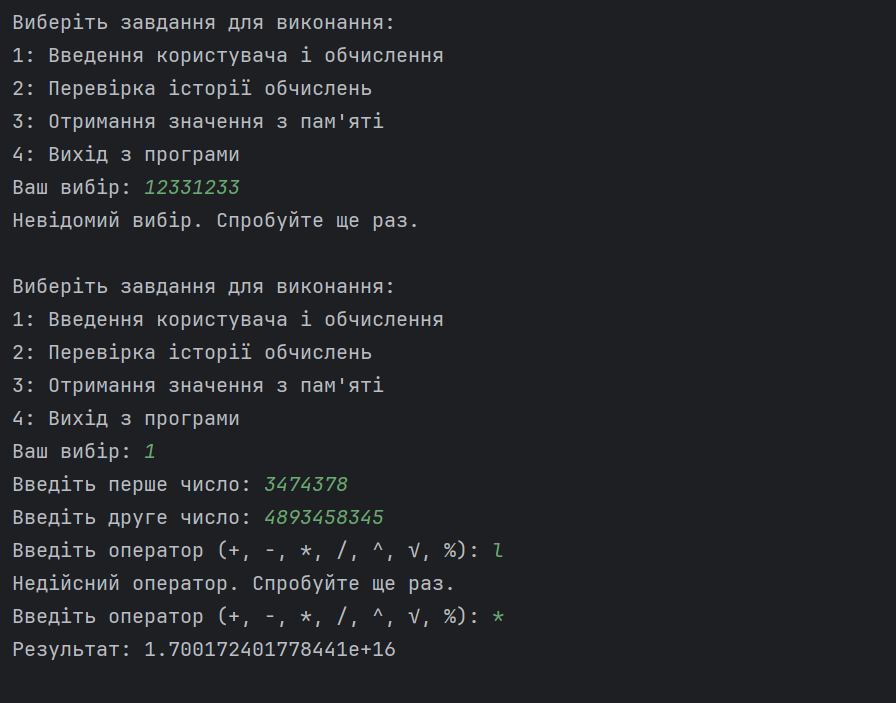
class MemoryManager:  
 def \_\_init\_\_(self):  
 self.memory = None  
  
 def store(self, value):  
 *"""Store a value in memory."""* self.memory = value  
  
 def retrieve(self):  
 *"""Retrieve the stored value from memory."""* if self.memory is not None:  
 return self.memory  
 else:  
 return "В пам'яті немає значень"

**History\_manager.py**

class HistoryManager:  
 def \_\_init\_\_(self):  
 self.history = []  
  
 def add(self, expression, result):  
 *"""Add a calculation and its result to the history."""* self.history.append((expression, result))  
  
 def view(self):  
 *"""Display the calculation history."""* if not self.history:  
 print("Історії не знайдено.")  
 else:  
 for expr, res in self.history:  
 print(f"{expr} = {res}")



*Рис. 1 Результат виконання програми*



*Рис. 2 Результат виконання програми*

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