МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра інформаційних систем та мереж

**Звіт**

До лабораторної роботи №1

**Введення в Python**

Виконав:

ст. гр. ІТ-21сп

**Микола ПЛЕБАНСЬКИЙ**

Прийняв:

доц. каф. ІСМ

**Сергій ЩЕРБАК**

Львів ‒ 2023

**Лабораторна робота №1**

**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, +, -, \*, /).

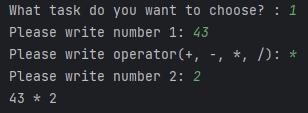
Код:

number1 = int(input(variables.number1))  
operator = input(variables.operator)  
number2 = check\_operator.check\_sqrt(operator)

if selected\_task == '1':  
 task1.output(number1, operator, number2)

def output(number1, operator, number2):  
 return print(number1, operator, number2)

Результат:



**Рис.1** Результат виконання завдання 1

Завдання 2: Перевірка оператора

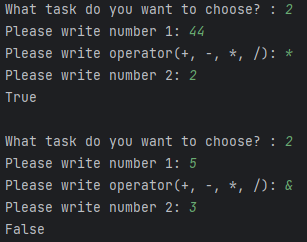
Перевірте чи введений оператор є дійсним (тобто одним із +, -, \*, /). Якщо ні, відобразіть повідомлення про помилку і попросіть користувача ввести дійсний оператор.

Код:

elif selected\_task == '2':  
 done\_task2 = task2.check\_operator(operator)  
 print(done\_task2)  
 if not task2.check\_operator(operator):  
 print(variables.error\_operator)  
 continue

def check\_operator(operator):  
 return operator in ['+', '-', '\*', '/']

Результат:



**Рис.2** Результат виконання завдання 2

Завдання 3: Обчислення

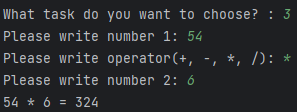
Виконайте обчислення на основі введення користувача (наприклад, додавання, віднімання, множення, ділення) і відобразіть результат.

Код:

**e**lif selected\_task == '3':  
 if not task2.check\_operator(operator):  
 print(variables.error\_operator)  
 continue  
 task3.calculation(number1, operator, number2)

def calculation(number1, operator, number2):  
 if operator == "+":  
 result = number1 + number2  
 elif operator == '-':  
 result = number1 - number2  
 elif operator == '\*':  
 result = number1 \* number2  
 elif operator == '/':  
 result = number1 / number2  
  
 return print(f"{number1} {operator} {number2} = {result}")

Результат:



**Рис.3** Результат виконання завдання 3

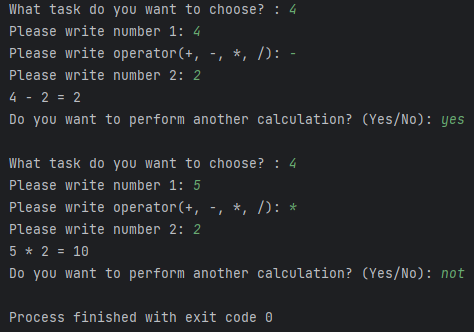
Завдання 4: Повторення обчислень

Запитайте користувача, чи він хоче виконати ще одне обчислення. Якщо так, дозвольте йому ввести нові числа і оператор. Якщо ні, вийдіть з програми.

Код:

elif selected\_task == '4':  
 if not task2.check\_operator(operator):  
 print(variables.error\_operator)  
 continue  
 task3.calculation(number1, operator, number2)  
 repeat = input(variables.choose\_exit)  
 if repeat.lower() != "yes":  
 break

Результат:



**Рис.4** Результат виконання завдання 4

Завдання 5: Обробка помилок

Реалізуйте обробку помилок для обробки ділення на нуль або інших потенційних помилок. Відобразіть відповідне повідомлення про помилку, якщо виникає помилка.

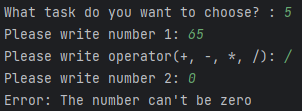
Код:

elif selected\_task == '5':  
 if not task2.check\_operator(operator):  
 print(variables.error\_operator)  
 continue  
 task5.check\_division(number1, operator, number2)  
 repeat = input(variables.choose\_exit)  
 if repeat.lower() != "yes":  
 break

def check\_division(number1, operator, number2):  
 if operator == '/':  
 if check\_exeption.check\_division\_zero(number1, number2):  
 return True  
 else:  
 task3.calculation(number1, operator, number2)  
 else:  
 task3.calculation(number1, operator, number2)

def check\_division\_zero(number1, number2):  
 try:  
 number1 / number2  
 except ZeroDivisionError:  
 print(variables.zero)  
 return ZeroDivisionError

Результат:



**Рис.5** Результат виконання завдання 5

Завдання 6: Десяткові числа

Змініть калькулятор так, щоб він обробляв десяткові числа (плаваючу кому) для більш точних обчислень.

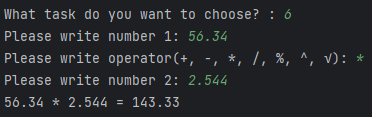
Код:

number1 = float(input(variables.number1))  
operator = input(variables.operator\_additional)  
number2 = check\_operator.check\_sqrt\_float(operator)  
  
if selected\_task == '6':  
 if not task2.check\_operator(operator):  
 print(variables.error\_operator)  
 continue  
 task6.check\_division\_float(number1, operator, number2)

def check\_sqrt\_float(operator):  
 if operator != "√":  
 return float(input(variables.number2))

def calculation\_float(number1, operator, number2):  
 if operator in ['+', '-', '\*']:  
 if operator == "+":  
 result = number1 + number2  
 return result  
 elif operator == '-':  
 result = number1 - number2  
 return result  
 elif operator == '\*':  
 result = number1 \* number2  
 return result  
 elif operator == '/':  
 result = number1 / number2  
 return result  
  
  
def check\_division\_float(number1, operator, number2):  
 if operator == '/':  
 if check\_exeption.check\_division\_zero(number1, number2):  
 return True  
 else:  
 result = calculation\_float(number1, operator, number2)  
 return print(f"{number1} {operator} {number2} = {result:.2f}")  
 else:  
 result = calculation\_float(number1, operator, number2)  
 return print(f"{number1} {operator} {number2} = {result:.2f}")

Результат:



**Рис.6** Результат виконання завдання 6

Завдання 7: Додаткові операції

Додайте підтримку додаткових операцій, таких як піднесення до степеня (^), квадратний корінь (√) і залишок від ділення (%).

Код:

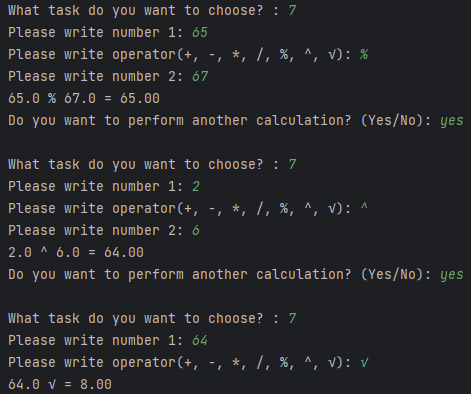
elif selected\_task == '7':  
 if not task2.check\_additional\_operator(operator):  
 print(variables.error\_operator)  
 continue  
 if task2.check\_operator(operator):  
 task6.check\_division\_float(number1, operator, number2)  
 else:  
 task7.check\_root\_float(number1, operator, number2)

def check\_additional\_operator(operator):  
 return operator in ['+', '-', '\*', '/', '%', '^', '√']

def check\_root\_float(number1, operator, number2):  
 if operator == '√':  
 if check\_exeption.check\_sqrt\_less\_zero(number1):  
 return True  
 else:  
 result = calculation\_additional\_float(number1, operator, number2)  
 return print(f"{number1} {operator} = {result:.2f}")  
 else:  
 result = calculation\_additional\_float(number1, operator, number2)  
 return print(f"{number1} {operator} {number2} = {result:.2f}")

def check\_sqrt\_less\_zero(number1):  
 try:  
 math.sqrt(number1)  
 except ValueError:  
 print(variables.sqrt)  
 return ValueError

Результат:



**Рис.7** Результат виконання завдання 7

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.

Код:

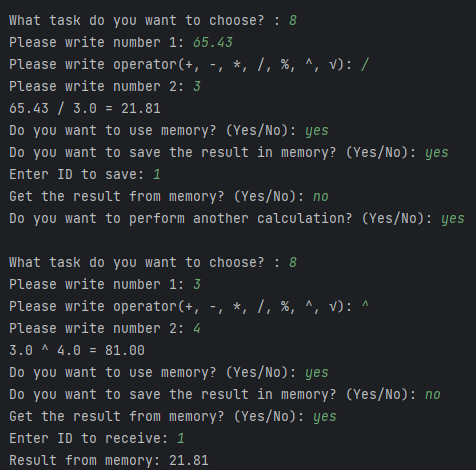
elif selected\_task == '8':  
 if not task2.check\_additional\_operator(operator):  
 print(variables.error\_operator)  
 continue  
 if not functions.check\_option.check\_zero\_float(number1, operator, number2):  
 result = functions.check\_option.check\_option(number1, operator, number2)  
 task8.working\_memory(result)

def check\_zero\_float(number1, operator, number2):  
 if task2.check\_operator(operator):  
 return task6.check\_division\_float(number1, operator, number2)  
 else:  
 return task7.check\_root\_float(number1, operator, number2)

def check\_option(number1, operator, number2):  
 if task2.check\_operator(operator):  
 return task6.calculation\_float(number1, operator, number2)  
 else:  
 return task7.calculation\_additional\_float(number1, operator, number2)

def store\_result(identifier, result):  
 variables.memory[identifier] = result  
  
  
def retrieve\_result(identifier):  
 return variables.memory.get(identifier, None)  
  
  
def working\_memory(result):  
 memory\_text = input(variables.memory\_text)  
 if memory\_text.lower() == "yes":  
 save = input(variables.save)  
 if save.lower() == "yes":  
 identifier\_save = input(variables.identifier\_save)  
 store\_result(identifier\_save, result)  
 retrieve = input(variables.retrieve)  
 if retrieve.lower() == "yes":  
 identifier\_retrieve = input(variables.identifier\_retrieve)  
 saved\_result = retrieve\_result(identifier\_retrieve)  
 if saved\_result is not None:  
 print(f"Result from memory: {saved\_result:.2f}")  
 else:  
 print(f"Saved result with ID '{identifier\_retrieve}' not found.")

Результат:



**Рис.8** Результат виконання завдання 8

Завдання 9: Історія обчислень

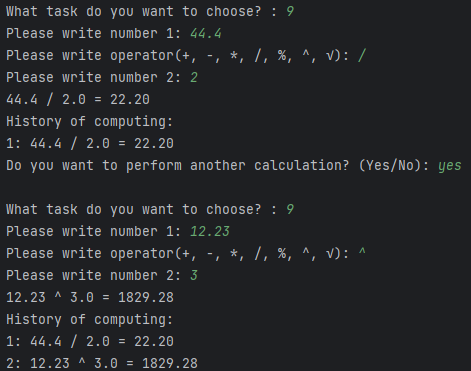
Створіть журнал, який зберігає історію попередніх обчислень, включаючи вираз і результат. Дозвольте користувачам переглядати історію своїх обчислень.

Код:

elif selected\_task == '9':  
 if not task2.check\_additional\_operator(operator):  
 print(variables.error\_operator)  
 continue  
 if not functions.check\_option.check\_zero\_float(number1, operator, number2):  
 result = functions.check\_option.check\_option(number1, operator, number2)  
 variables.history.append((number1, operator, number2, result))  
 task9.history()

def history():  
 print("History of computing:")  
 for i, calculation in enumerate(variables.history):  
 n1, op, n2, res = calculation  
 if op != '√':  
 print(f"{i + 1}: {n1} {op} {n2} = {res:.2f}")  
 else:  
 print(f"{i + 1}: {n1} {op} = {res:.2f}")

Результат:



**Рис.9** Результат виконання завдання 9

Завдання 10: Налаштування користувача

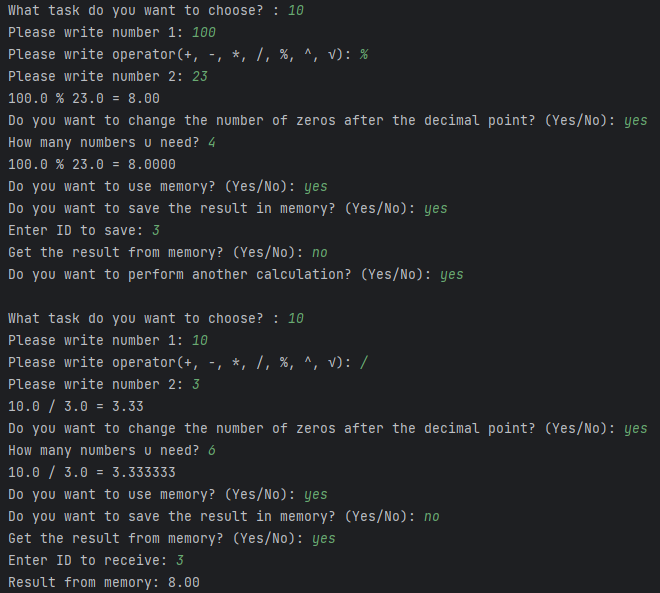
Надайте користувачам можливість налаштувати поведінку калькулятора, таку як зміну кількості десяткових розрядів, які відображаються, або налаштування функцій пам'яті.

Код:

elif selected\_task == '10':  
 if not task2.check\_additional\_operator(operator):  
 print(variables.error\_operator)  
 continue  
 if not functions.check\_option.check\_zero\_float(number1, operator, number2):  
 change\_float = input(variables.change\_float)  
 if change\_float.lower() == "yes":  
 numbers = int(input(variables.numbers))  
 result = task10.check\_root\_floating(number1, operator, number2, numbers)  
 task8.working\_memory(result)

def check\_root\_floating(number1, operator, number2, numbers):  
 if operator == '√':  
 if check\_exeption.check\_sqrt\_less\_zero(number1):  
 return True  
 else:  
 result = check\_option.check\_option(number1, operator, number2)  
 print(f"{number1} {operator} = {result:.{numbers}f}")  
 return result  
 else:  
 result = check\_option.check\_option(number1, operator, number2)  
 print(f"{number1} {operator} {number2} = {result:.{numbers}f}")  
 return result

Результат:



**Рис.10** Результат виконання завдання 10

**Висновок:** Виконавши ці завдання, я створив 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.