Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Звіт про виконання лабораторної роботи №2

з дисципліни «Спеціалізовані мови програмування»

на тему «Основи побудови об’єктно-орієнтованих додатків на Python»

Виконала:  
студентка групи РІ-31  
Заплетнюк Ю.І.

Прийняв:  
Щербак С.С.

**Мета роботи:** розробка консольного калькулятора в об’єктно орієнтованому стилі з використанням класів

**План роботи**

Завдання 1: Створення класу Calculator

Створіть клас Calculator, який буде служити основою для додатка калькулятора.

Завдання 2: Ініціалізація калькулятора

Реалізуйте метод \_\_init\_\_ у класі Calculator для ініціалізації необхідних атрибутів або змінних.

Завдання 3: Введення користувача

Перемістіть функціональність введення користувача в метод у межах класу Calculator. Метод повинен приймати введення для двох чисел і оператора.

Завдання 4: Перевірка оператора

Реалізуйте метод у класі Calculator, щоб перевірити, чи введений оператор є дійсним (тобто одним із +, -, \*, /). Відобразіть повідомлення про помилку, якщо він не є дійсним.

Завдання 5: Обчислення

Створіть метод у класі Calculator, який виконує обчислення на основі введення користувача (наприклад, додавання, віднімання, множення, ділення).

Завдання 6: Обробка помилок

Реалізуйте обробку помилок у межах класу Calculator для обробки ділення на нуль або інших потенційних помилок. Відобразіть відповідні повідомлення про помилку.

Завдання 7: Повторення обчислень

Додайте метод до класу Calculator, щоб запитати користувача, чи він хоче виконати ще одне обчислення. Якщо так, дозвольте йому ввести нові числа і оператор. Якщо ні, вийдіть з програми.

Завдання 8: Десяткові числа

Модифікуйте клас Calculator для обробки десяткових чисел (плаваюча кома) для більш точних обчислень.

Завдання 9: Додаткові операції

Розширте клас Calculator, щоб підтримувати додаткові операції, такі як піднесення до степеня (^), квадратний корінь (√) та залишок від ділення (%).

Завдання 10: Інтерфейс, зрозумілий для користувача

Покращте інтерфейс користувача у межах класу Calculator, надавши чіткі запити, повідомлення та форматування виводу для зручності читання.

**Код програмної реалізації**

calculator.py

import sys

import os

sys.path.append(os.path.join(os.path.dirname(os.path.abspath(\_\_file\_\_)), '../..'))

import lab1.functions.functions as cf # cf - calculator functions

class Calculator:

def \_\_init\_\_(self):

self.memory = 0

self.history = []

def get\_number(self, prompt):

return cf.get\_number(prompt)

def get\_operator(self):

return cf.get\_operator()

def calculate(self, num1, num2, operator):

return cf.calculate(num1, num2, operator)

def store\_in\_memory(self, result):

cf.store\_in\_memory(result)

self.memory = result

def add\_to\_history(self, expression, result):

cf.add\_to\_history(expression, result)

def view\_history(self):

cf.view\_history()

def main(self):

while True:

use\_memory = input("Use value from memory? (yes/no): ").strip().lower() == "yes"

if use\_memory:

num1 = self.memory

else:

num1 = self.get\_number("Enter the first number: ")

operator = self.get\_operator()

if operator == '√':

result = self.calculate(num1, None, operator)

expression = f"√{num1}"

else:

num2 = self.get\_number("Enter the second number: ")

result = self.calculate(num1, num2, operator)

expression = f"{num1} {operator} {num2}"

if result is not None:

print(f"Result: {result}")

self.add\_to\_history(expression, result)

if input("Save the result in memory? (yes/no): ").strip().lower() == "yes":

self.store\_in\_memory(result)

if input("View calculation history? (yes/no): ").strip().lower() == "yes":

self.view\_history()

if input("Do another calculation? (yes/no): ").strip().lower() != "yes":

break

**runner.py**

import sys

import os

sys.path.append(os.path.join(os.path.dirname(os.path.abspath(\_\_file\_\_)), '..'))

from classes.calculator import Calculator

calc = Calculator()

calc.main()

На рис. 1 показано результат виконання завдання.

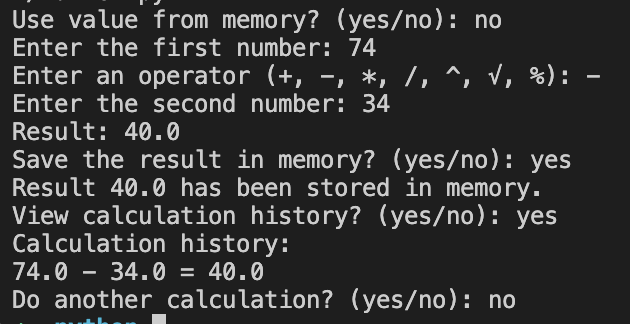


Рис. 1. Результат виконання завдання

**Висновки:** виконавши завдання лабораторної роботи, було перетворено консольний калькулятор у об'єктно-орієнтований калькулятор, використовуючи класи в Python. Цей проект допоміг мені вивчити концепції ООП та організацію, зберігаючи функціональність і інтерфейс користувача калькулятора.