Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Лабораторна роботи №2

з дисципліни «Спеціалізовані мови програмування»

на тему

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

Виконав:

Костюк В. О.

Перевірив:

Щербак С. С.

Львів 2024

**Мета:** 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.

**Реалізація:**

**Папка constants, файл GlobalVariables.py:**  
DEFAULT\_DECIMAL\_PLACES = 2

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

import math

from lab2.functions.memory import Memory

from lab2.functions.history import History

from lab2.functions.settings import AppSettings

class Calculator:

    def \_\_init\_\_(self):

        self.memory = Memory()

        self.history = History()

        self.settings = AppSettings()

    def addition(self, x, y):

        return x + y

    def subtraction(self, x, y):

        return x - y

    def multiplication(self, x, y):

        return x \* y

    def division(self, x, y):

        if y == 0:

            return "Помилка: ділення на нуль"

        return x / y

    def power(self, x, y):

        return x \*\* y

    def square\_root(self, x):

        if x < 0:

            return "Помилка: квадратний корінь з від'ємного числа"

        return math.sqrt(x)

    def modulus(self, x, y):

        return x % y

    def valid\_operator(self, operator):

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

    def run\_calculator(self):

     while True:

        try:

            num1 = float(input("Введіть перше число: "))

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

            while not self.valid\_operator(operator):

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

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

            if operator == '√':

                result = self.square\_root(num1)

                expression = f"√{num1}"

            else:

                num2 = float(input("Введіть друге число: "))

                result = self.perform\_operation(num1, num2, operator)

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

        except ValueError:

            print("Невірне значення. Будь ласка, введіть число.")

            continue

        print(f"Результат: {result:.{self.settings.decimal\_places}f}")

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

        if self.settings.memory\_enabled:

            self.memory.save\_to\_memory(result)

        next\_step = input("Хочете отримати значення з пам'яті (м), переглянути історію (іст) або завершити (ні)? ").strip().lower()

        if next\_step == 'м':

            self.memory.recall\_from\_memory()

        elif next\_step == 'іст':

            self.history.show\_history()

        elif next\_step == 'ні':

            pass  # Продовжити до запитання про наступне обчислення

        continue\_calculating = input("Бажаєте виконати ще одне обчислення? (так/ні): ").strip().lower()

        if continue\_calculating != 'так':

            break

    def perform\_operation(self, num1, num2, operator):

        if operator == '+':

            return self.addition(num1, num2)

        elif operator == '-':

            return self.subtraction(num1, num2)

        elif operator == '\*':

            return self.multiplication(num1, num2)

        elif operator == '/':

            return self.division(num1, num2)

        elif operator == '^':

            return self.power(num1, num2)

        elif operator == '%':

            return self.modulus(num1, num2)

**Папка functions, файл history.py:**class History:

    def \_\_init\_\_(self):

        self.history = []

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

        self.history.append(f"{expression} = {result:.{2}f}")

    def show\_history(self):

        if self.history:

            print("Історія обчислень:")

            for entry in self.history:

                print(entry)

        else:

            print("Історія порожня.")

**Папка functions, файл memory.py:**

class Memory:

    def \_\_init\_\_(self):

        self.value = None

    def save\_to\_memory(self, result):

        self.value = result

        print(f"Результат {result} збережено в пам'ять.")

    def recall\_from\_memory(self):

        if self.value is not None:

            print(f"Отримане значення з пам'яті: {self.value}")

            return self.value

        else:

            print("Пам'ять порожня.")

            return None

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

class AppSettings:

    def \_\_init\_\_(self):

        self.decimal\_places = 2

        self.memory\_enabled = False  # За замовчуванням вимкнена

        self.configure\_settings()

    def configure\_settings(self):

        while True:

            try:

                self.decimal\_places = int(input("Введіть кількість десяткових розрядів (0-10): "))

                if 0 <= self.decimal\_places <= 10:

                    break

                else:

                    print("Будь ласка, введіть число в межах від 0 до 10.")

            except ValueError:

                print("Невірне значення. Спробуйте ще раз.")

        while True:

            memory\_option = input("Включити функцію пам'яті? (так/ні): ").strip().lower()

            if memory\_option in ['так', 'ні']:

                self.memory\_enabled = memory\_option == 'так'

                if self.memory\_enabled:

                    print("Функція пам'яті увімкнена.")

                else:

                    print("Функція пам'яті вимкнена.")

                break

            else:

                print("Будь ласка, введіть 'так' або 'ні'.")

**Папка interfaces, файл ICalculator.py:**

class ICalculator:

    def addition(self, x, y):

        pass

    def subtraction(self, x, y):

        pass

    def multiplication(self, x, y):

        pass

    def division(self, x, y):

        pass

    def power(self, x, y):

        pass

    def square\_root(self, x):

        pass

    def modulus(self, x, y):

        pass

**Файл main.py:**

from lab2.functions.calculator import Calculator

def main ():

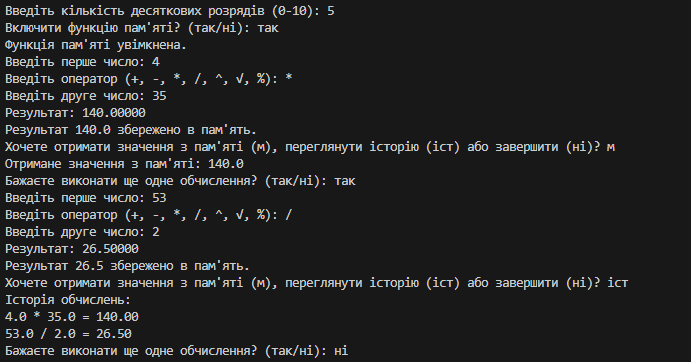
        calculator = Calculator()

        calculator.run\_calculator()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Результат виконання:**



**Висновок:** на цій лабораторній роботі я перетворив консольний калькулятор у об'єктно-орієнтований калькулятор, використовуючи класи в Python.