МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж



Лабораторна робота №2

з дисципліни Спеціалізовані мови програмування

на тему

Основи побудови об’єктно-орієнтованих додатків на Python

Виконав:

студент групи РІ-21сп

Андрій ПІГУЛЯК

Львів – 2024

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

**Хід роботи**

1. Створити клас Calculator як основу для програми калькулятора.
2. Реалізувати метод init для ініціалізації атрибутів.
3. Додати метод для введення двох чисел і оператора.
4. Реалізувати перевірку введеного оператора (+, -, \*, /).
5. Створити метод для виконання обчислень (додавання, віднімання тощо).
6. Додати обробку помилок (наприклад, ділення на нуль).
7. Додати функцію повторення обчислень за бажанням користувача.
8. Додати підтримку десяткових чисел.
9. Розширити калькулятор для операцій: степінь (^), корінь (√), залишок (%).
10. Покращити інтерфейс: чіткі запити, повідомлення, форматований вивід.
11. **Опис і структура програми:**

Програма демонструє практичне застосування об'єктно-орієнтованого програмування таких як інкапсуляція та розділення обов’язків, також організована у вигляді модульного проєкту з використанням класів.

Основні компоненти:

1. **Класи:**
   * Calculator: виконує арифметичні операції, записує історію в файл.
   * Memory: дозволяє зберігати, додавати, віднімати значення та очищувати пам'ять.
2. **Функції:**
   * GetOperator: обробляє введення оператора з перевіркою коректності.
   * GetNumber: обробляє введення чисел, включаючи конвертацію їх у потрібний формат.
3. **Файл main.py:**   
   Забезпечує роботу з користувачем через текстове меню.
4. **Функціонал:**

* Арифметичні операції: додавання, віднімання, множення, ділення, модуль, степінь, квадратний корінь.
* Обробка помилок: перевірка типів аргументів, ділення на нуль, недопустимі оператори, корінь із від’ємного числа.
* Операції з пам’яттю: додавання, віднімання, очищення та читання значення з пам’яті, історія обчислень: запис результатів у файл History.txt.

1. **Реалізація коду:** 
   1. Перевірки типів аргументів

def SquareRoot(self, number):  
 if not isinstance(number, (int, float)):  
 raise ValueError("Argument must be a number")  
 if number < 0:  
 raise ValueError("Square root of a negative number is undefined")  
 result = math.sqrt(number)  
 self.Save(f"Square root of {number} = {result}")  
 return result

* 1. Запис результатів у файл

def Save(self, value):  
 with open('History.txt', 'a') as file:  
 file.write(f"{value}\n")

Цей фрагмент демонструє обробку помилки з відєʼмним коренем а також метод запису здійснених операцій у файл, для подальшого використання з роботою памʼяті.

**4.** **Приклад роботи:**

**Приклад 1:** Помилка, корінь з відʼємного числа

A screenshot of a computer

Description automatically generated

Рис. 1 Результати помилки на відʼємний корінь

**Приклад 2:** Результат записаної памʼяті в файл

A screenshot of a computer

Description automatically generated

Рис. 2 Результат зчитування історії обчислень з файлу

**Приклад 3: Результат з памʼяттю після додавання і віднімання**

A screenshot of a phone

Description automatically generated

Рис. 3 Результат після роботи з памʼяттю

**Висновок:** У результаті виконання роботи було розроблено програму-калькулятор із розширеним функціоналом, яка дозволяє виконувати основні математичні операції, обробляти помилки та працювати з історією обчислень і пам’яттю. Завдяки модульній структурі програма легко підтримується і розширюється.