МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
Інститут комп'ютерних наук та інформаційних технологій
Кафедра інформаційних систем та мереж

Лабораторна робота №2
з дисципліни: «Спеціалізовані мови програмування»
на тему: «Основи побудови об’єктно-орієнтованих додатків на Python »

Виконала:
студентка групи ІТ-32
Моляща Ю.А.
Прийняв:
Щербак С.С.

Львів - 2023

Лабораторна робота №2

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

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

Завдання на лабораторну роботу.

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.

Хід роботи.

Код програми:

main.py

```
# Include the parent directory in the system's import path
import sys
import os

current_dir = os.path.dirname(os.path.abspath(__file__))
parent_dir = os.path.abspath(os.path.join(current_dir, '..'))
sys.path.append(parent_dir)

from lab2.calculator import Calculator

def main():
    calcultor = Calculator()
    calcultor.calculate()

if __name__ == "__main__":
    main()
```

calculator.py

```
"""Calculator class for the calculator program."""

# Include the parent directory in the system's import path
import sys
import os

current_dir = os.path.dirname(os.path.abspath(__file__))
parent_dir = os.path.abspath(os.path.join(current_dir, '..'))
sys.path.append(parent_dir)
```

```
# Importing functions from lab1
from lab1.history_handling import save_result
from lab1.console_calculator import calculate_option, settings_option

HISTORY_FILE = 'source/lab2/history.txt'

class Calculator():
    def __init__(self):
        self.number_of_calculations = None

    def calculate(self):
        while True:
            print("\nOptions:")
            print("1. Perform calculation")
            print("2. Settings")
            print("3. Quit")
            choice = input("Enter your choice (1/2/3): ")
            if choice == '1':
                result = self.calculate_option()
                self.save_result(result, HISTORY_FILE,
self.number_of_calculations)
            elif choice == '2':
                self.settings_option(HISTORY_FILE)
            elif choice == '3':
                print("Exiting the calculator. Goodbye!")
                break
            else:
                print("Invalid choice. Please select a valid option (1/2/3).")

    def calculate_option(self):
        return calculate_option()

    def settings_option(self, HISTORY_FILE):
        settings_option(HISTORY_FILE)
```

```
    def save_result(self, result, HISTORY_FILE, number_of_calculations):
        save_result(result, HISTORY_FILE, number_of_calculations)
```

history.txt

```
3.0 + 4.0 = 7.0
4.0 - 23.0 = -19.0
3.0 + 4.0 = 7.0
6.0 + 3.0 = 9.0
1.0 + 1.0 = 2.0000000000
```

Висновок: під час виконання лабораторної роботи було розроблено консольной калькулятор в об’єктно орієнтованому стилі з використанням класів.