Міністерство освіти і науки України

Національний університет «Львівська політехніка»



**ЗВІТ**

про виконання лабораторної роботи № 2

з курсу:

«Спеціалізовані мови програмування»

**Виконав:**

студент гр. ІТ-31

Олександр ГАМАЮНОВ

**Прийняв:**

Сергій ЩЕРБАК

Львів 2023

**Тема:** Основи побудови об’єктно-орієнтованих додатків на Python

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

**Хід 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.

**Main.py:**

from calculator import Calculator

from user import User

#Firstly we need to enter username

user = User(input("Enter your name: "))

if(input('Do you wish to change roundto ? Y N :') == 'y'):

    roundto = int(input('How many number after dot should be visible? (1-9): '))

    user.changeRoundto(roundto)

calculator = Calculator()

while (2+2)!=5:

    calculator.calculations(user)

    if(calculator.repeat() == False):

        break

    calculator.Input()

    #calculator.useResult()

**Calculator.py**

from data import results

from exception import CalculationException, IncorrectOperatorException

from user import User

class Calculator:

     def \_\_init\_\_(self):

          self.Input()

          self.results = results

     def Input(self):

       self.num1 = float(input('Enter a first number: '))

       self.num2 = float(input('Enter a second number: '))

       self.operator = input('Enter a operator(+,-,\*,/,^,√,%) : ')

     def repeat(self):

        answer = input('Do you want to make another calculation? Y N: ')

        if(answer == 'N' or answer == "n"):

            return False

     def calculations(self,u: User):

        match self.operator:

            case('+'):

                result = self.num1+self.num2

                print(round(result,u.roundto))

                self.writeIn(result,u.roundto)

                array = [self.num1,self.num2,self.operator,result]

                u.save(array)

            case('-'):

                result = self.num1 - self.num2

                print(round(result,u.roundto))

                self.writeIn(result,u.roundto)

                array = [self.num1,self.num2,self.operator,result]

                u.save(array)

            case('\*'):

                result = self.num1\*self.num2

                print(round(result,u.roundto))

                self.writeIn(result,u.roundto)

                array = [self.num1,self.num2,self.operator,result]

                u.save(array)

            case('/'):

                try:

                    if(self.num2 == 0):

                        raise CalculationException(self.num2)

                    else:

                        result = self.num1/self.num2

                        print(round(result,u.roundto))

                        self.writeIn(result, u.roundto)

                        array = [self.num1,self.num2,self.operator,result]

                        u.save(array)

                except CalculationException as e:

                    print("Exception occurred:", str(e))

            case('^'):

                result = pow(self.num1,self.num2)

                print(round(result,u.roundto))

                self.writeIn(result,u.roundto)

                array = [self.num1,self.num2,self.operator,result]

                u.save(array)

            case('√'):

                count = 0

                multiplier = self.num2

                while(self.num1 >= self.num2):

                    self.num2 \*= multiplier

                    count+=1

                result = self.num1/self.num2+count

                print(round(result,u.roundto))

                self.writeIn(result,u.roundto)

                array = [self.num1,self.num2,self.operator,result]

                u.save(array)

            case('%'):

                if(self.num2 == 0):

                    print('You can`t divide by 0')

                else:

                    result = self.num1%self.num2

                    print(round(result,u.roundto))

                    self.writeIn(result,u.roundto)

                    array = [self.num1,self.num2,self.operator,result]

                    u.save(array)

            case \_:

                raise IncorrectOperatorException

#Function that save result of calculation into result[] for repetative usage

     def writeIn(self,result,roundto):

        write = input('Do you wish to save the result? Y N: ')

        if(write == 'Y' or write == 'y'):

            self.results.append(round(result, roundto))

#function that return number from result[]

     def useResult(self):

        for i in range (0, len(self.results)):

            print(self.results[i])

            confirm = input('Do you want to use this number as num1? Y N: ')

            if(confirm == 'Y' or confirm == "y"):

                return self.results[i]

**User.py**

from data import defaultroundto

from data import memory

#User class

class User:

    def \_\_init\_\_(self,name):

        self.name = name

        self.roundto = defaultroundto

        self.memory = memory

    def save(self,array):

        self.memory.append(array)

    def changeRoundto(self,roundto):

        self.roundto = roundto

**Exception.py**

class CalculationException(Exception):

    def \_\_init\_\_(self, num2):

        self.num2 = num2

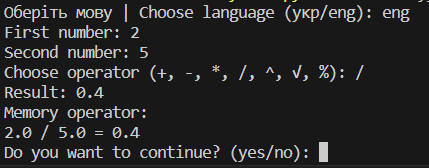
        super().\_\_init\_\_()

class IncorrectOperatorException(Exception):

    def \_\_init\_\_(self, message="Invalid operator"):

        self.message = message

        super().\_\_init\_\_(message)

**

*Рис.1 – Результат виконання програми*

**Висновок:** на даній лабораторній роботі я перетворив консольний калькулятор у об'єктно-орієнтований калькулятор, використовуючи класи в Python. Цей проект допоміг вивчити концепції об'єктно-орієнтованого програмування та організацію, зберігаючи функціональність і інтерфейс користувача калькулятора.