Технології програмування

Лабораторна робота №4

**Зворотний польський запис та обчислення математичного виразу**

**Мета роботи**: Використовуючи теоретичне підґрунтя про зворотний польський запис розробити програму на вхід якої подається математичний вираз, що має довільний набір операндів, операторів та дужок, на виході програма обчислює результат математичного виразу.

**Завдання до лабораторної роботи**

Використовуючи теоретичне відомості розробити програму яка на вхід отримує математичний вираз з довільною кількістю операндів, операторів та дужок. В першу чергу сформувати послідовність символів у ЗПН. На другому етапі виконання лабораторної роботи вирахувати результат послідовності, що була сформована, використовуючи алгоритм запису математичного виразу у ЗПН.

Текст програми разом зі звітом розмістити в директорії lab\_04. Директорію lab\_04 розмістити в директорії, що використовується для виконання практичних завдань по кожній лекції та має назву **TP-KB-22[1 or 2]-Name-Surname**.

Хід роботи

1. Функція перетворення виразу у ЗПН



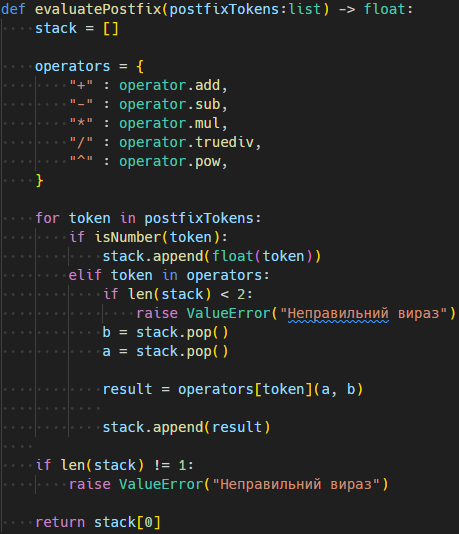
Функція повертає ЗПН у вигляді списку

з 2+2\*2

в

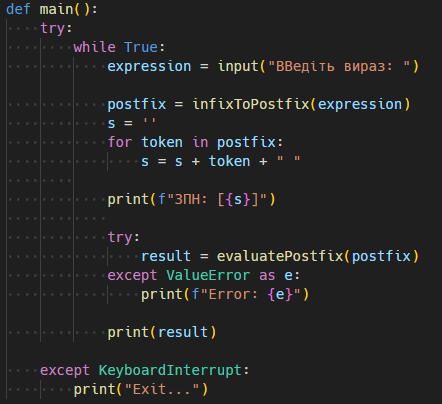


2. Функція обчислення ЗПН



функція повертає результат в вигляді числа з типу "float" або помилку в разі неможливості обрахувати результат.

3. Основна функція програми



реалізує обробку помилок та безкінечний цикл запитів. Завершення програвми відбувається на Ctrl + C

Код програми:

import re

import operator

def isNumber(token) -> bool:

    try:

        float(token)

        return True

    except ValueError:

        return False

def getPrecedence(operator: str) -> int:

    precedence = {

        '+': 1,

        '-': 1,

        '\*': 2,

        '/': 2,

        '^': 3

    }

    return precedence.get(operator, 0)

def infixToPostfix(expression:str) -> list:

    tokens = re.findall(r'\d+\.?\d\*|[+\-\*/^()]', expression.replace(' ', ''))

    output = []

    oprStack = []

    for token in tokens:

        if isNumber(token):

            output.append(token)

        elif token == "(":

            oprStack.append(token)

        elif token == ")":

            while oprStack and oprStack[-1] != "(":

                output.append(oprStack.pop())

            oprStack.pop()

        elif token in ["+", "-", "\*", "/", "^"]:

            while oprStack and getPrecedence(token) <= getPrecedence(oprStack[-1]):

                output.append(oprStack.pop())

            oprStack.append(token)

    while oprStack:

        output.append(oprStack.pop())

    return output

def evaluatePostfix(postfixTokens:list) -> float:

    stack = []

    operators = {

        "+" : operator.add,

        "-" : operator.sub,

        "\*" : operator.mul,

        "/" : operator.truediv,

        "^" : operator.pow,

    }

    for token in postfixTokens:

        if isNumber(token):

            stack.append(float(token))

        elif token in operators:

            if len(stack) < 2:

                raise ValueError("Неправильний вираз")

            b = stack.pop()

            a = stack.pop()

            result = operators[token](a, b)

            stack.append(result)

    if len(stack) != 1:

        raise ValueError("Неправильний вираз")

    return stack[0]

def main():

    try:

        while True:

            expression = input("ВВедіть вираз: ")

            postfix = infixToPostfix(expression)

            s = ''

            for token in postfix:

                s = s + token + " "

            print(f"ЗПН: [{s}]")

            try:

                result = evaluatePostfix(postfix)

            except ValueError as e:

                print(f"Error: {e}")

            print(result)

    except KeyboardInterrupt:

        print("Exit...")

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

    main()

Висновок

На даній лабораторній роботі я використовуючи теоретичне підґрунтя розробив програму що обчислює математичні вирази через ЗПН.