МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет електроніки і комп’ютерних технологій

Кафедра системного проєктування

**Звіт**

про виконання лабораторної роботи № 5

«Інфіксна, префіксна та постфіксна форми запису виразів»

**Виконав:**

студент групи Феп-12

Богонос Ю. Р.

**Викладач:**

Вельгош С. Р.

**Львів 2021**

**Тема:** Інфіксна, префіксна та постфіксна форми запису виразів.

**Мета:** Навчитись реалізовувати запис математичних формул у інфіксній/префіксній/префіксній формах та обчислювати їх.

**Інструменти:** Ноутбук, Сode::Blocks v10.05.

**Дата виконання:** **20.04.2021р.**

**Теоретичні відомості:**

Математичні вирази (арифметичні, логічні тощо) переважно записують у вигляді, коли оператор знаходиться між двома операндами (наприклад, a b + – у випадку бінарних операцій). Така форма запису називається інфіксною і вона є найбільш природньою для людини. Однак таке представлення виразу не є однозначним. Порядок виконання операцій в ньому визначається їх пріоритетом (наприклад, введення в степінь ( ^ ) виконується першим, далі виконують множення і ділення ( × , / ), потім додавання та віднімання ( + , – )). Якщо ж потрібно змінити пріоритет виконання операцій, то використовують дужки. Існують інші форми запису математичних виразів, які інтерпретуються однозначно і не використовують дужок. Кажуть, що вираз є записаний у префіксній (польській1 ) формі, якщо в ньому знак операції безпосередньо передує операндам, на які він діє. Такий вираз слід читати справа наліво. Вираз є записаний у постфіксній (зворотній польській) формі, якщо в ньому знак операції знаходиться безпосередньо після операндів, на які він діє. Такий вираз слід читати зліва направо. Приклад. ( ) ( ) ^ a b c d x + × + – інфіксна форма запису; × + + ab cdx ^ – відповідна префіксна форма запису; ab cd x + + ×^ – відповідна постфіксна форма запису. Розглянемо алгоритми переводу математичних виразів з інфіксної у префіксну та постфіксну форми запису та навпаки. Ці алгоритми передбачають використання структури даних типу стек (див. лабораторну роботу № 4). 1 Цю форму запропонував у 1924 році польський логік та математик Ян Лукасевич, українець за походженням. 2 Алгоритм переводу виразу з інфіксної у постфіксну форму запису. 1. Записати вхідний вираз у інфіксній формі у повністю „одужкованому” вигляді. Це означає, що кожен оператор і операнди на які він діє беруться в дужки. 2. Зчитати посимвольно вхідний вираз зліва направо. 2.1.якщо зчитаний символ є операндом, то його записують у вихідний вираз. 2.2.якщо зчитаний символ є знаком операції, то його заносять у стек. 2.3.якщо зчитаний символ є знаком закритої дужки ")" (це означає, що ми досягли кінця операції), то видобуваємо зі стеку елемент (знак операції) і записуємо його у вихідний вираз. Алгоритм переводу виразу з інфіксної у префіксну форму запису. Алгоритм переводу виразу з інфіксної у префіксну форму запису є аналогічним попередньому з незначними модифікаціями. Зауваження. Отриманий в результаті цього алгоритму вихідний вираз для подання його у префіксній формі потрібно інвертувати (переписати символи у оберненій послідовності). Для інвертування слід використати структуру даних типу стек. Алгоритм переводу виразу з постфіксної у інфіксну форму запису. 1. Зчитуємо посимвольно вхідний вираз зліва направо. 1.1.якщо зчитаний символ є операндом, то його записують у стек; 1.2.якщо зчитаний символ є знаком операції, то: 1.1.1. видобути зі стеку два останні елементи; 1.1.2. помістити знак операції між видобутими зі стеку операндами (при цьому слід врахувати, що операції ділення і віднімання не комутують) та взяти отриманий вираз в дужки; 1.1.3. помістити вираз, отриманий в попередньому пункті назад у стек. 3 2. Записати вміст стеку у вихідний вираз. Алгоритм переводу виразу з префіксної у інфіксну форму запису. Алгоритм переводу виразу з інфіксної у префіксну форму запису є аналогічним попередньому з незначними модифікаціями. Зауваження. Два останні алгоритми можна використовувати для обчислення значень арифметичних виразів у постфіксній та префіксній формах, відповідно (передбачається, що операнди є числами), якщо пункт 1.1.2 замінити на обчислення числового результату відповідної операції. Зауваження. Описані алгоритми не передбачають використання унарних операцій, а також перетворення виразів в неповністю „одужкованому” вигляді. Префіксну, постфіксну та інфіксну форми запису математичних виразів можна отримати, якщо представити ці вирази у формі бінарних дерев та здійснити обхід цих дерев у прямому, зворотньому чи внутрішньому порядку, відповідно (див. лабораторну роботу № 7). Внутрішнім вершинам таких дерев відповідають знаки операцій, а листкам – операнди. Приклад. Для виразу ( ) ( ) ^ a b c d x + × + (відповідно: × + + ab cdx ^ та ab cd x + + ×^ ) дерево матиме вигляд: 4 .

**Хід роботи.**

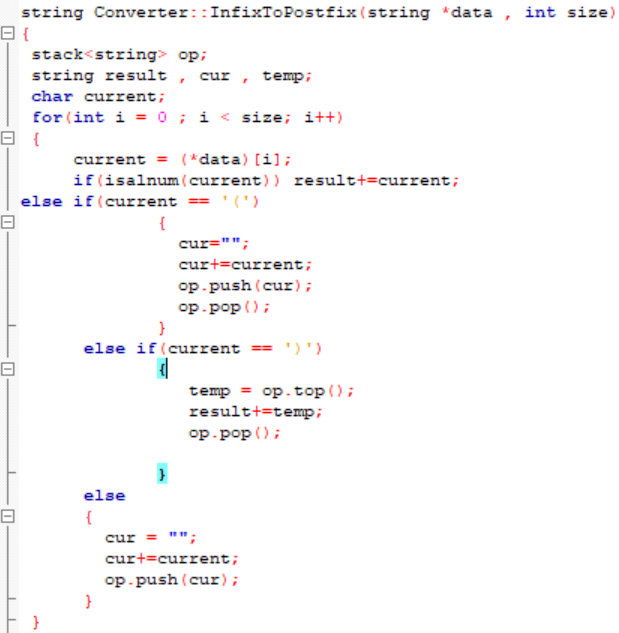
**Частина I**

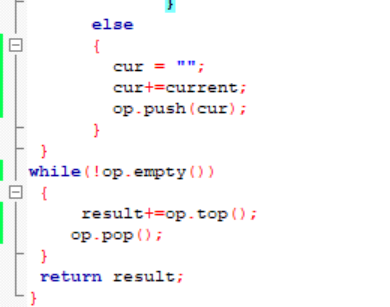
1. Створити нову бібліотеку Converter (файли Converter.h, Converter.cpp). До цієї бібліотеки підключити раніше створену бібліотеку List (див. лабораторну роботу № 4) для роботи з лінійними списками (файли List.h, List.cpp). Модифікувати бібліотеку List для роботи з даними символьного/стрічкового типу (команда typedef у файлі List.h).

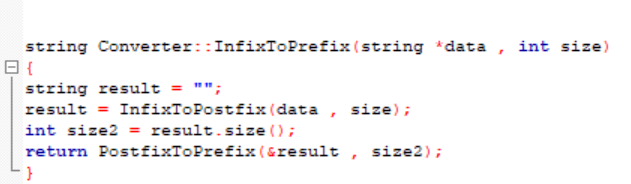
2. У бібліотеці Converter, згідно описаних вище алгоритмів реалізувати функції для переводу з інфіксної у постфіксну та префіксну форми запису. (Функції InfixToPostfix(…), InfixToPrefix(…)). Функції повинні приймати як аргумент вхідну стрічку виразу у інфіксній формі і видавати як значення вихідну стрічку виразу у постфіксній/префіксній формі.

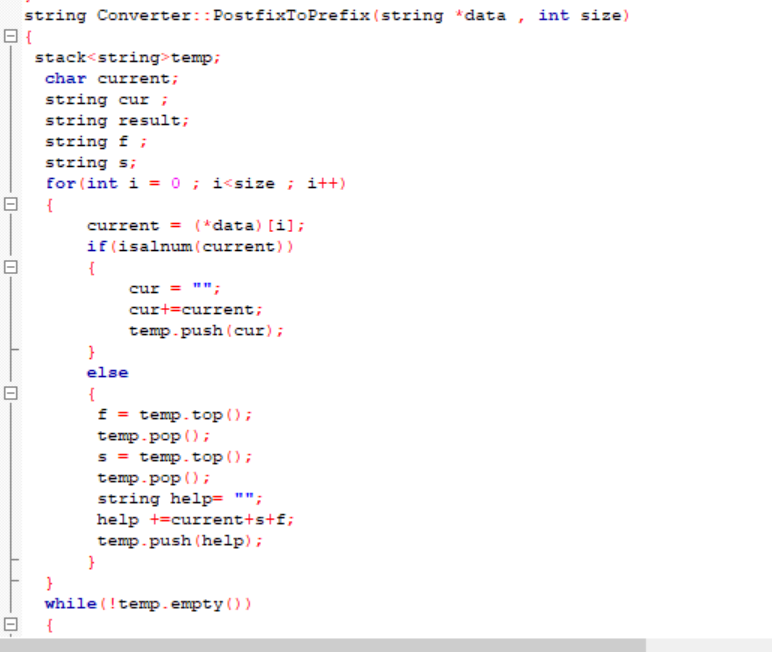
3. Створити новий проект Lab\_8 до якого підключити бібліотеку Converter. У функції main() проекту реалізувати меню для вибору способу перетворення виразів. x ^ + b × + a c d 5 4. Отримати від викладача завдання (вираз у інфіксній формі запису), привести його (вручну) у повністю „одужкований” вигляд, перевести заданий вираз у постфіксну та префіксну форми запису. Продемонструвати результат викладачеві.

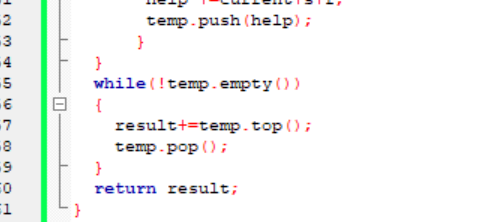
**Результат роботи:**

****

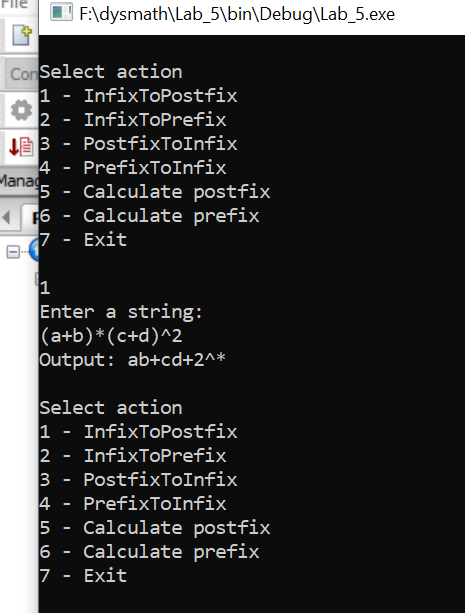
****

****

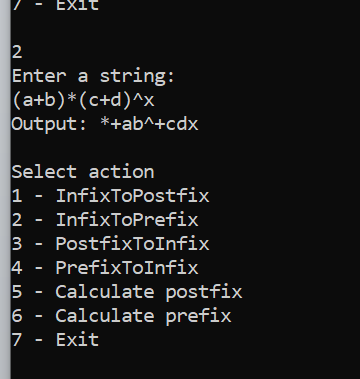
****

****

**Результати консолі:**

****

**(InfixToPostfix)**

****

**(PostfixToInfix)**

**Частина II**

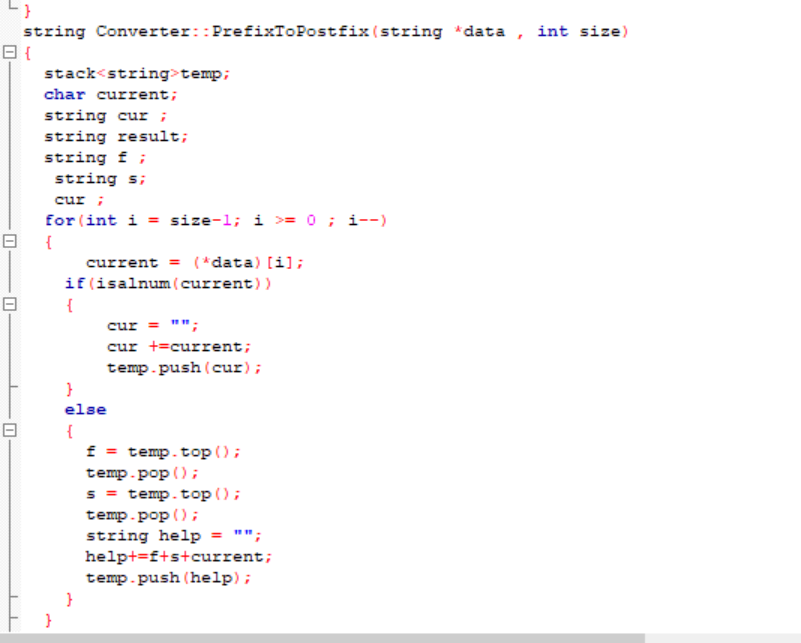
1. У бібліотеці Converter, згідно описаних вище алгоритмів реалізувати функції для переводу з постфіксної та префіксної у інфіксну форму запису. (Функції PostfixToInfix(…), PrefixToInfix(…)). Функції повинні приймати як аргумент вхідну стрічку виразу у постфіксній/префіксній формі і видавати як значення вихідну стрічку виразу у інфіксній формі.

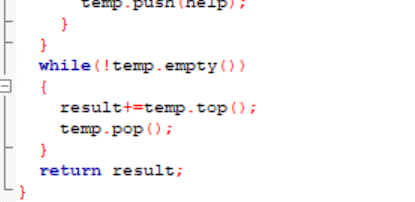
2. Створити аналогічні функції для обчислення числового значення виразу в постфіксній/префіксній формі. Функції повинні приймати як аргумент вхідну стрічку виразу у постфіксній/префіксній формі з операндами у вигляді чисел і видавати числове значення.

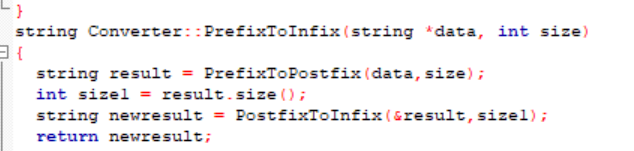
3. У функції main() проекту Lab\_8 доповнити створене у першій частині роботи меню можливістю перетворення виразів з постфіксної та префіксної у інфіксну форму запису та обчислення числових значень виразів.

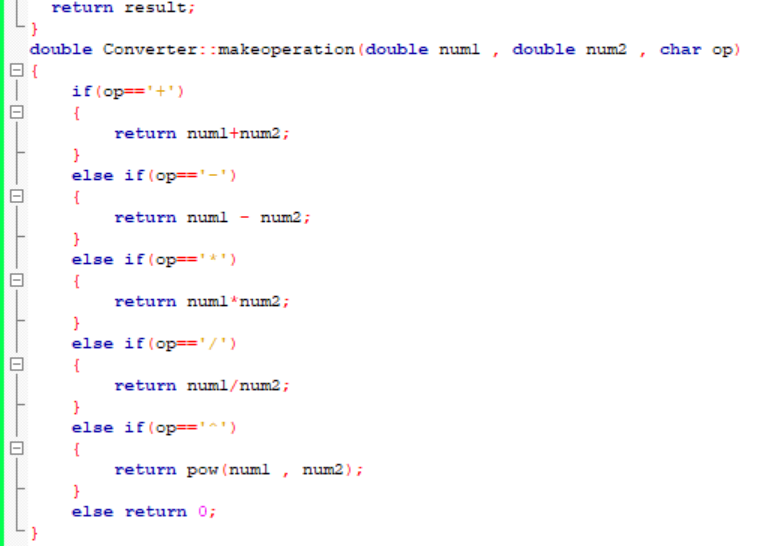
4. Отримати від викладача завдання (вирази у постфіксній та префіксній формах запису). Перевести задані вирази у інфіксну форму запису. Продемонструвати результат викладачеві.

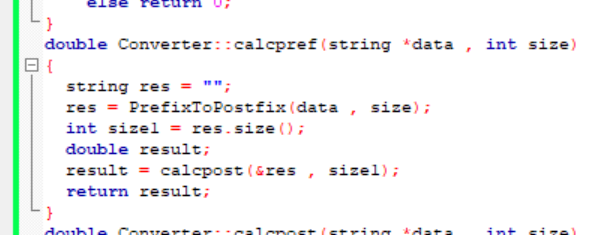
5. Отримати від викладача завдання (вирази у постфіксній та префіксній формах запису у яких операндами служать числа). Обчислити числові значення цих виразів. Продемонструвати результат викладачеві.

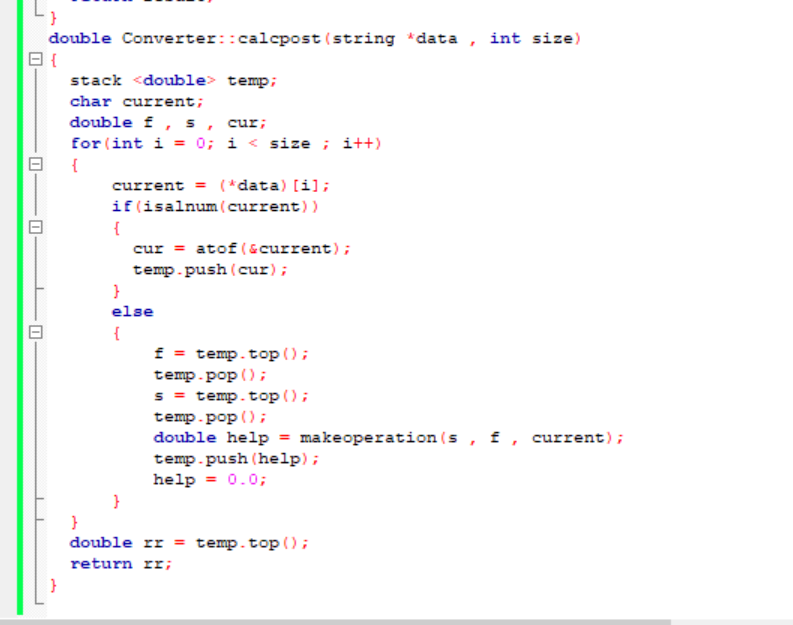
****

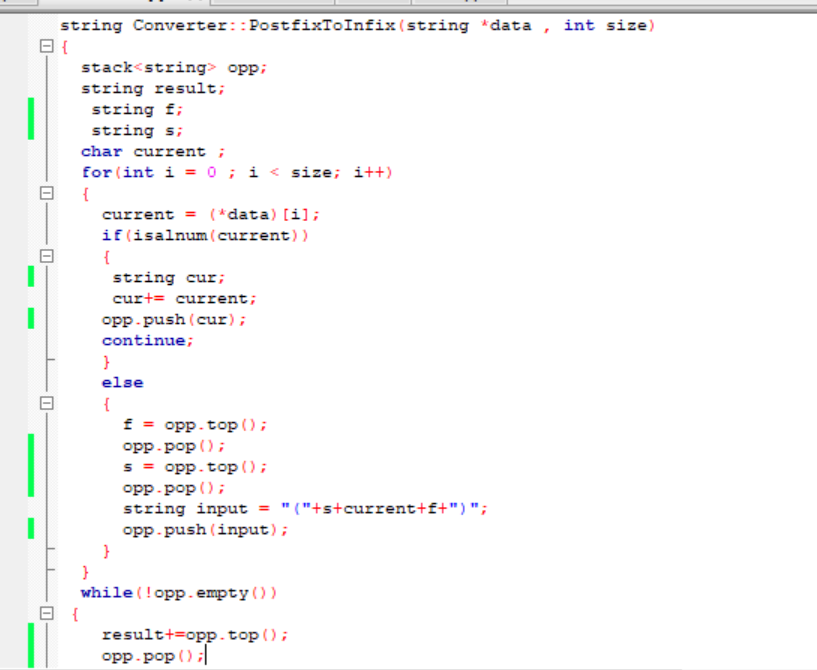
****

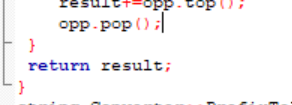
****

****

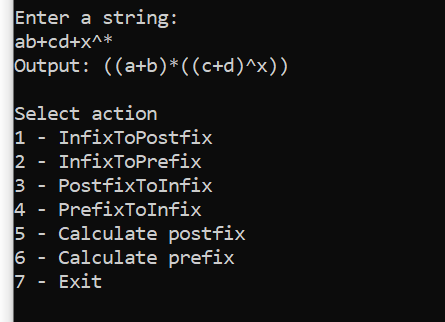
****

****

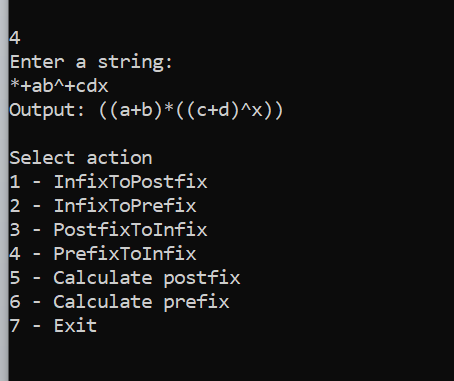
****

****

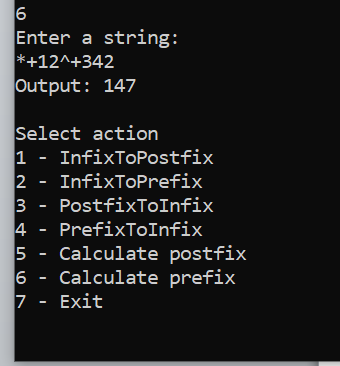
**Вивід консолі:**

****

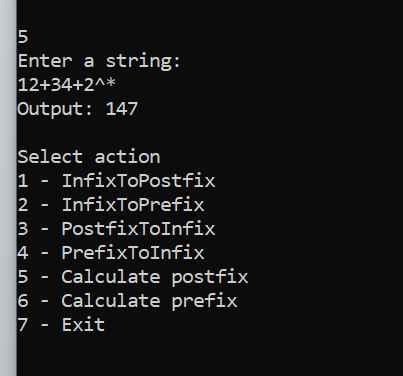
**(PostfixToInfix)**

****

**(PrefixToInfix)**

****

**(Calculate prefix)**

****

**(Calculate postfix)**

**Висновок:** Навчився реалізовувати запис математичних формул у інфіксній/префіксній/префіксній формах та обчислювати їх.