МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет електроніки і комп’ютерних технологій

**Звіт**

Про виконання лабораторної роботи № 5

**«Інфіксна, префіксна та постфіксна форми запису виразів.»**

**Виконав:**

Студент групи ФЕП-14с

Грицюк Максим Олегович

**Перевірив:**

ас. Баран М. О.

Львів 2025

**Мета:** Ознайомитися з принципами роботи динамічних структур даних та набути практичних навичок реалізації однозв’язних і двозв’язних списків. Навчитись програмно реалізовувати стек, чергу та двозв’язний список із базовими операціями додавання, видалення, пошуку та виведення елементів.

**Обладнання:**

Комп'ютер з встановленим програмним забезпеченням(Intellij IDEA Ultimate).

**Теоретичні відомості**

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

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.

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.

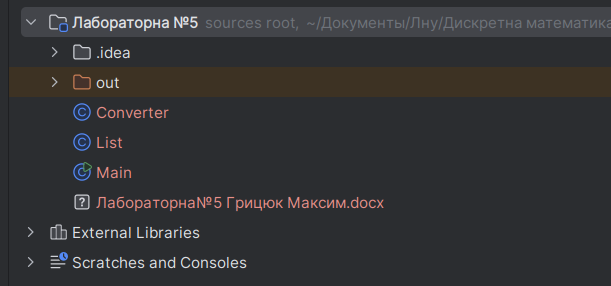
(a + b) X (c + d ) ^ x – інфіксна форма запису;

X + ab ^ +cdx – відповідна префіксна форма запису;

ab + cd + x ^ X – відповідна постфіксна форма запису.

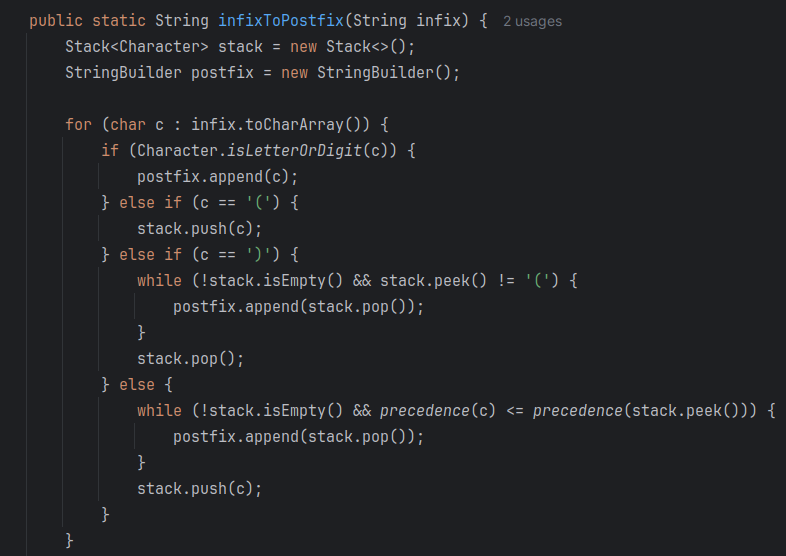
**Хід роботи**

**Завдання №1.1 Створити нову бібліотеку Converter (файли Converter.java).**



Створив нову бібліотеку Converter, до якої підключив власноруч реалізовану структуру стеку на базі зв’язаного списку (клас Stack, що працює з символами/рядками).

****Завдання №1.2** Реалізувати функції InfixToPostfix() та InfixToPrefix().**



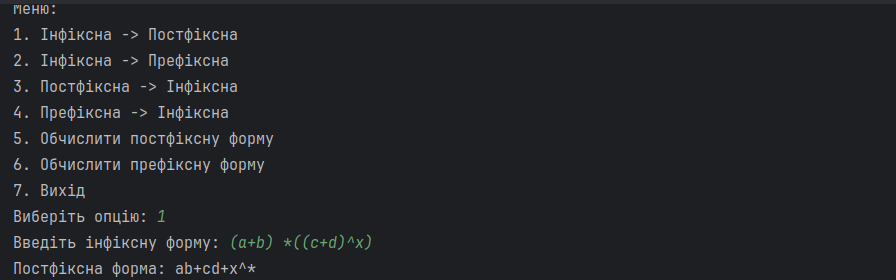
Ці методи приймають рядок з виразом у інфіксній формі та повертають відповідно постфіксний або префіксний варіант. Для реалізації алгоритмів використовується стек та клас Character для обробки символів.

****Завдання №1.**3 Створити проект з меню.**



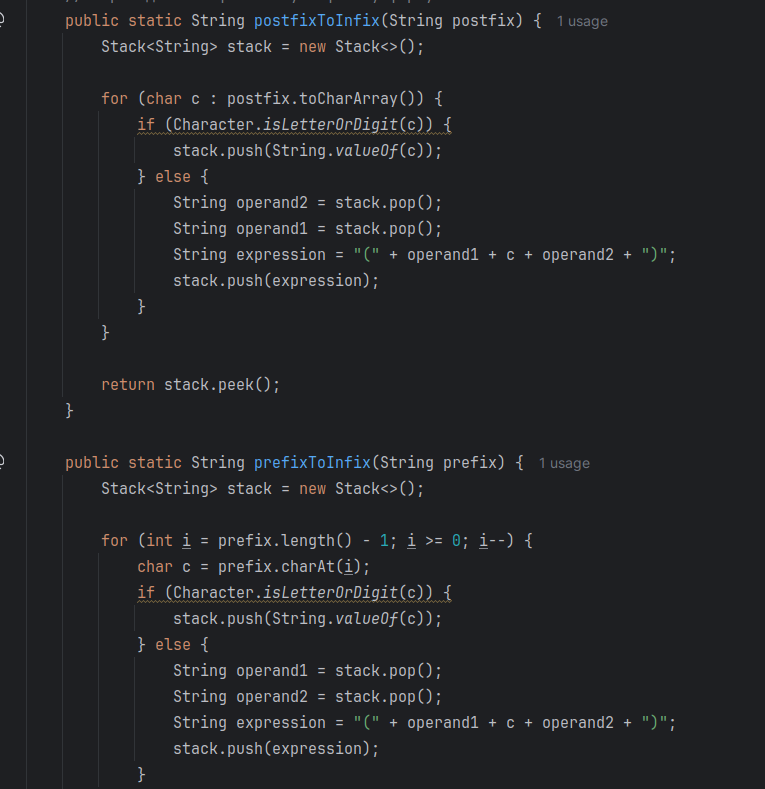
У програмі реалізував головний клас із методом main(), в якому через консольне меню користувач може вибрати потрібну операцію.

****Завдання №1.4** Отримати вираз у інфіксній формі та вручну перевести його в повністю одужкований вигляд.**



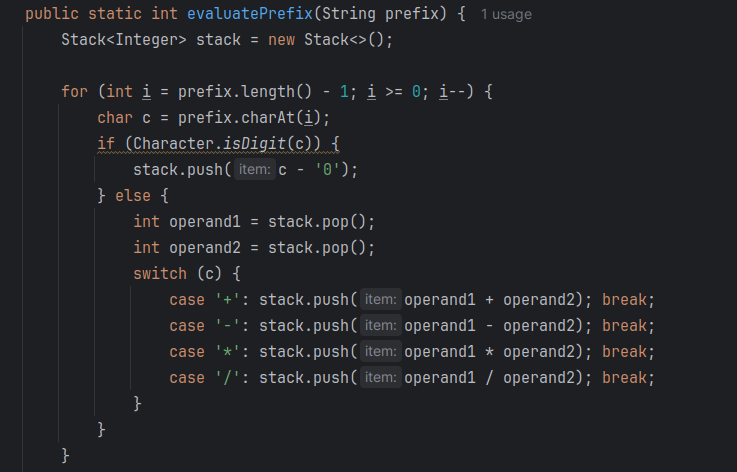
Перевірку правильності виконав вручну та продемонстрував результат у програмі.

**Завдання №2.1 Реалізувати функції PostfixToInfix() та PrefixToInfix().**



Методи аналізують вираз, використовуючи стек, та відновлюють відповідний інфіксний варіант (з дужками для коректного порядку операцій).

****Завдання №2.2** Реалізувати функції обчислення значення виразів.**



Методи приймають рядок виразу (де операнди - числа), виконують відповідні арифметичні дії та повертають числовий результат.

****Завдання №2.3** Розширити меню програми Lab\_5.**

До меню додано опції: префікс - інфікс, постфікс - інфікс, обчислення префіксного виразу, обчислення постфіксного виразу.

****Висновок:** У цій лабораторній роботі я ознайомився з різними формами запису математичних виразів та реалізував алгоритми їх перетворення між інфіксною, префіксною та постфіксною формами. Також я навчився використовувати стек для аналізу виразів і обчислення їх значень, що дало мені краще розуміння структури виразів і принципів роботи з ними.**