Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

Описание: Описание: Описание: A blue and white logo

Description automatically generated

**Звіт**

**про виконання лабораторних та практичних робіт блоку № 6**

***з дисципліни:*** «Основи програмування»

**Виконав:**

Студент групи ШІ-11

Лопатін Володимир Дмитрович

Львів 2024

**Тема:**

Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур. Поняття класу та його застосування

**Мета роботи:**  
Ознайомитися з принципами реалізації та використання динамічних структур даних (черга, стек, списки, дерево) у програмуванні. Розвинути навички проектування алгоритмів для обробки динамічних структур, а також засвоїти поняття класу, принципи його реалізації та застосування для створення об'єктно-орієнтованих рішень.

**Теоретичні відомості:**

* Динамічні структури даних
* Робота з динамічними структурами
* Поняття класу
* Практичне застосування класів в ООП

Динамічні структури даних:

Не був знайомий, на парах отримав базове поняття, дорозібрався через різні ресурси.

Витрачено 40 хв.

Робота з динамічними структурами:

Уперше зіткнувся в цьому епіку.

Витратив 1 годину.

Поняття класу:

Був знайомий, проте потрібно було нагадати.

Витратив 45 хвилин.

Практичне застосування класів в ООП:

Мав певне уявленняЮ але на практиці кл=раще зрозумів.

Для повного розуміння потрібно було 45 хвилин.

**Виконання роботи:**

1. Опрацювання завдання та вимог до програм та серидовища:

**Завдання №1 з ВНС**

«Лабораторна № 10 варіант 19»

Потрібно створити клас двозв'зний список та додати методи додавання на початок, додавання в кінець, видалення за індексом, виводу елементів, збереження в файл та читання з файлу.

**Завдання №2 з Algotester**

«Лабораторна №5 варіант 1»

Потрібно створити програму, яка приймає стан дошки в байтах, виконує закляття над однією клітинкою n разів і потім виводить стан дошки в байтах.

**Завдання №3**

**«**Практичне завдання**»**

**Однозв'язний список:**

*Реалізувати метод реверсу однозв’язного списку:* Node\* reverse(Node \*head);

bool compare(Node \*h1, Node \*h2) – метод порівняння двох списків

Node\* add(Node \*n1, Node \*n2) – додавання чисел через список

**Бінарне дерево:**

TreeNode \*create\_mirror\_flip(TreeNode \*root) – метод віддзеркалення дерева

void tree\_sum(TreeNode \*root) – метод знаходження кожному батьківському вузлу суми підвузлів.

Обчислення варто виконувати з використанням проміжних змінних.

Порівняти й пояснити отримані результати.

**Завдання №4**

«Лабораторна 7-8 з Algotester»

Ваше завдання - власноруч реалізувати структуру даних "Динамічний масив".  
Ви отримаєте Q запитів, кожен запит буде починатися зі слова-ідентифікатора, після якого йдуть його аргументи.

**Завдання №5**

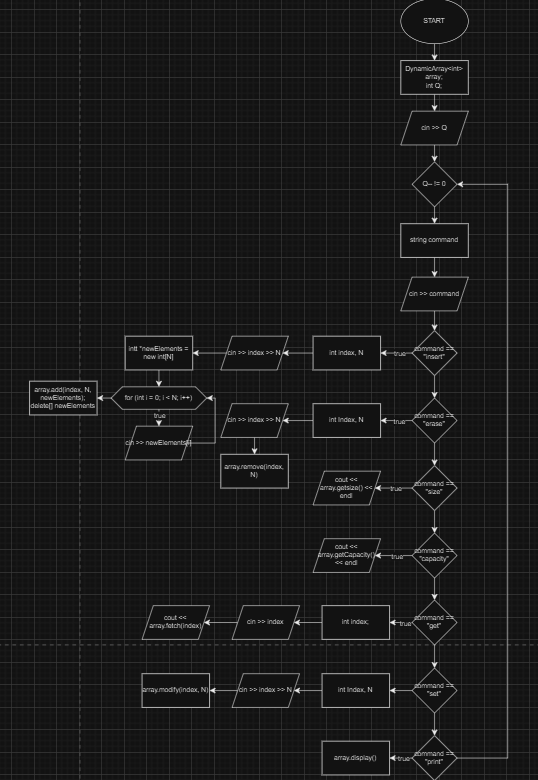
# «Self practice з Algotester»

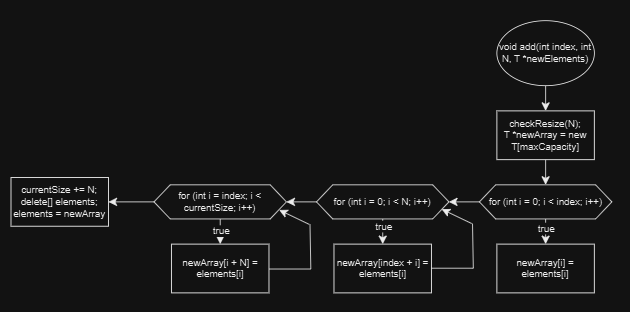
Задача полягала в тому, щоб закодувати рядок використовуючи букву і кількість разів її повторень.

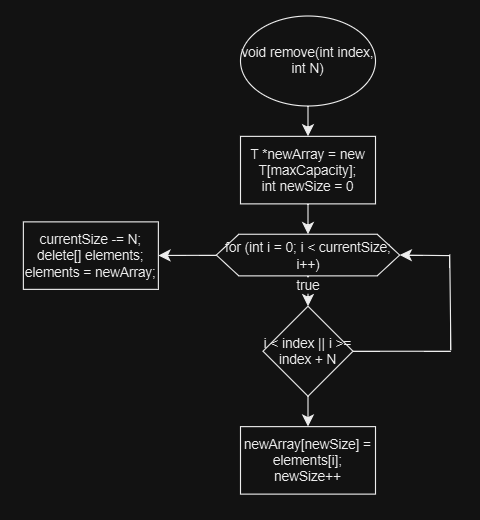
1. Дизайн та планова оцінка часу виконання завдань:

**Завдання №4**

Хотів зробити за 2 години.

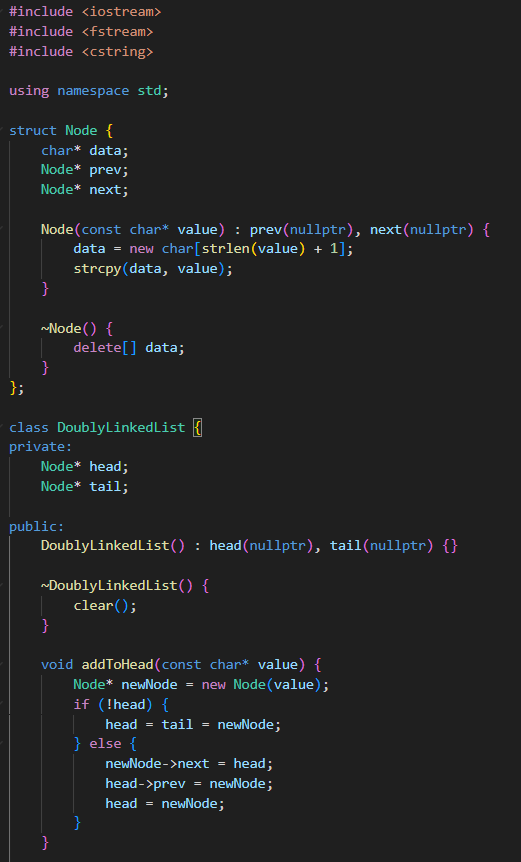


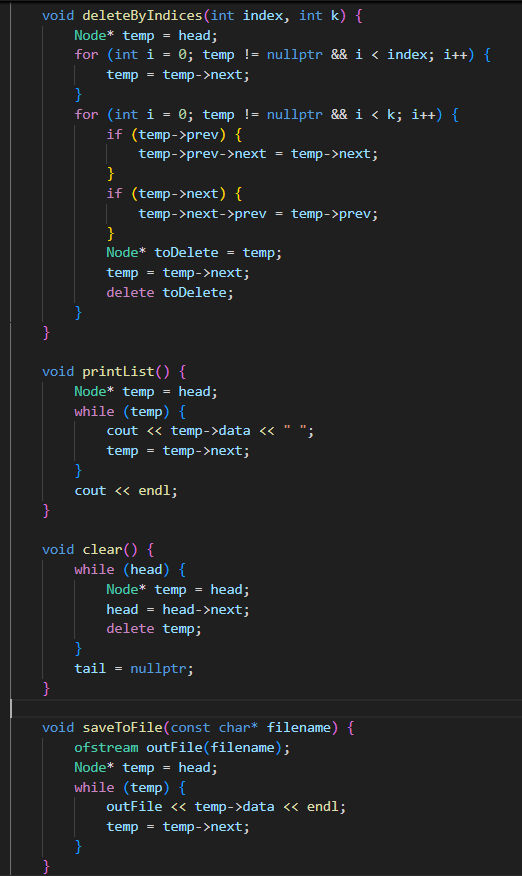


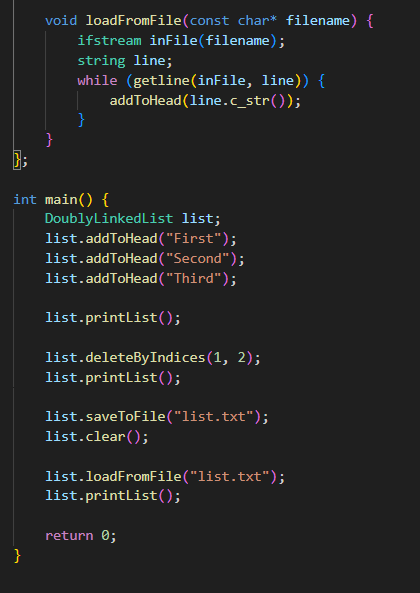


1. Код програм з посиланням на зовнішні ресурси:

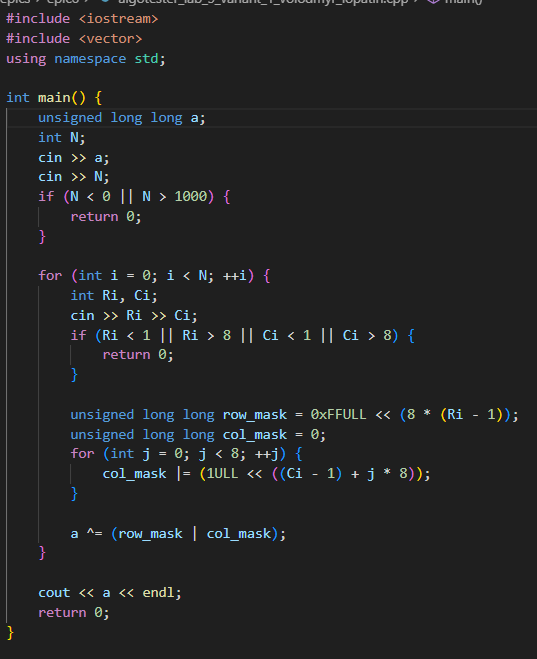
**Завдання №1**



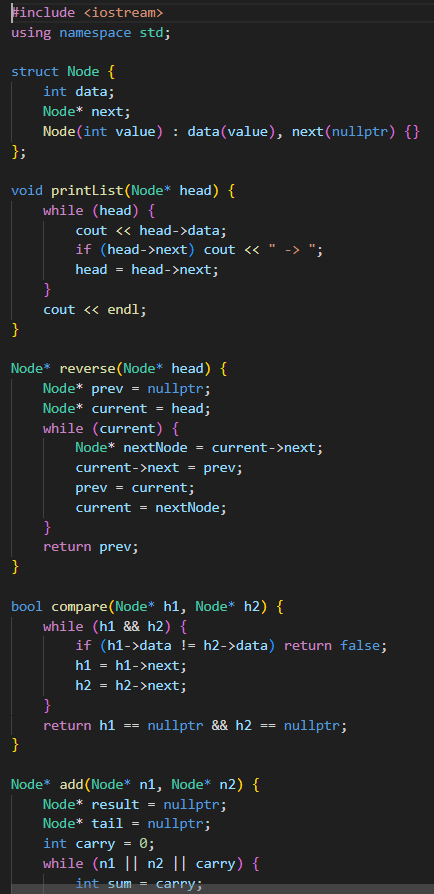
****

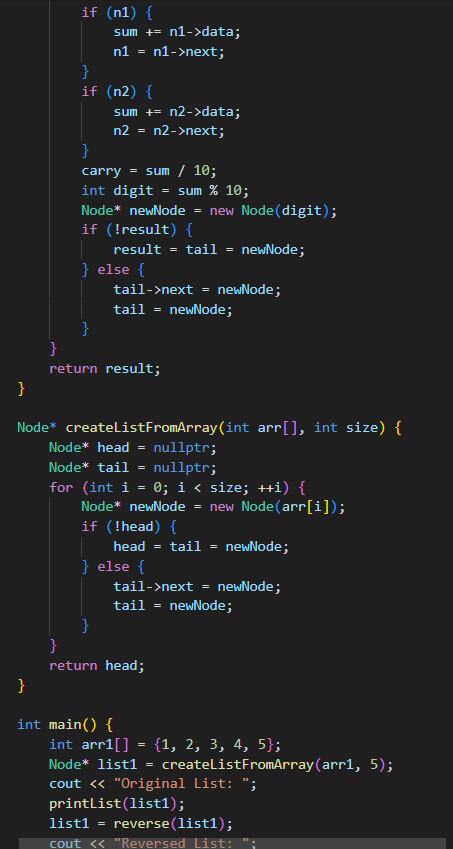
****

**Завдання №2**

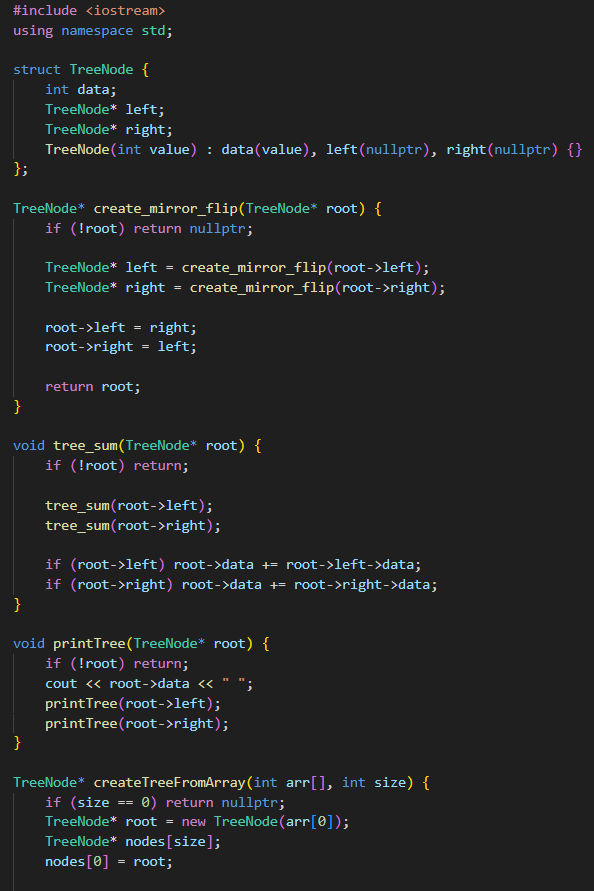
****

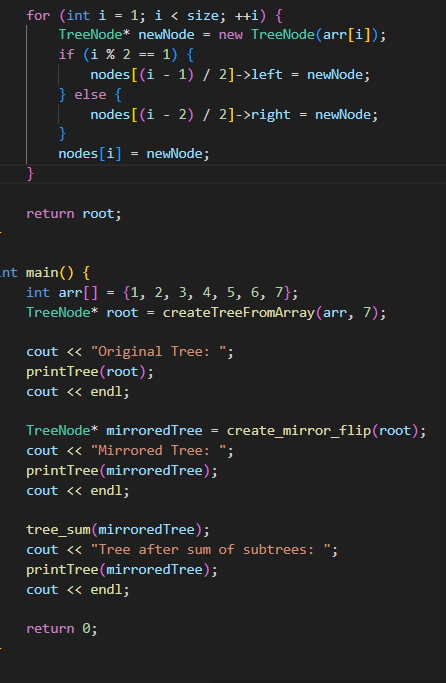
**Завдання №3**

****

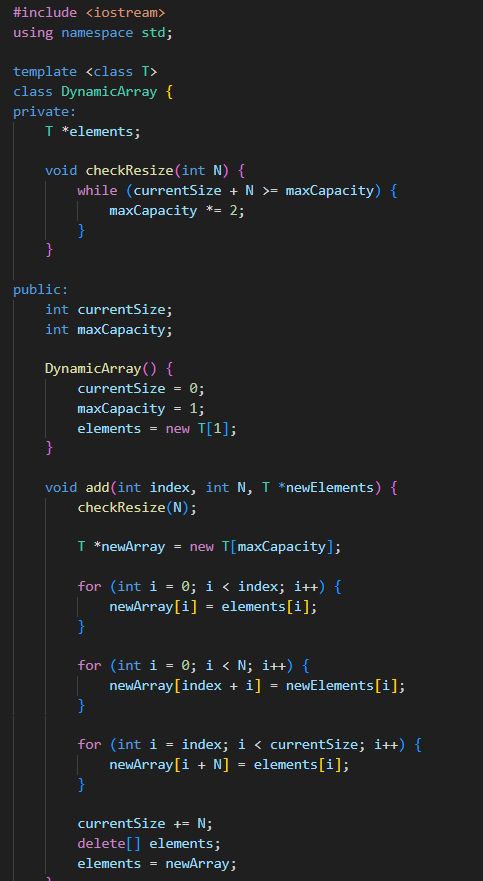


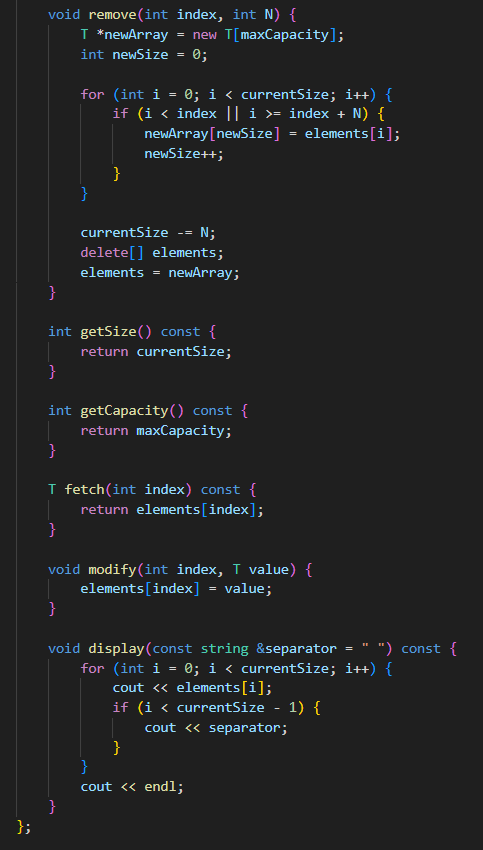


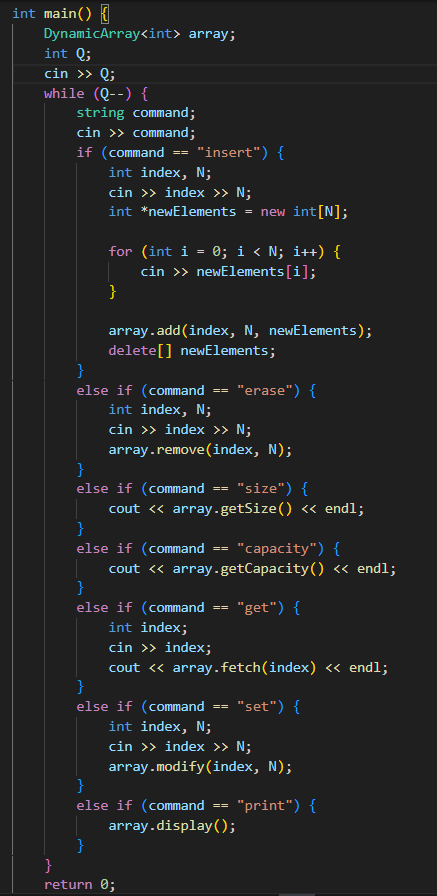
****

****

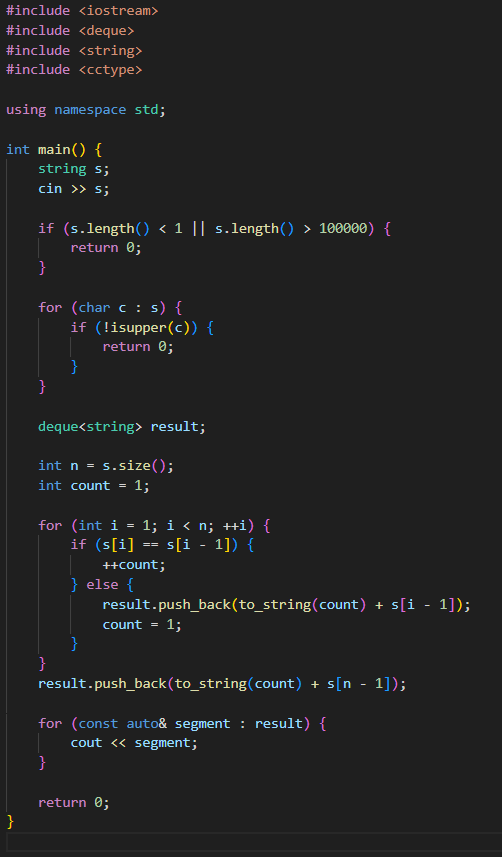
**Завдання №4**

****

****

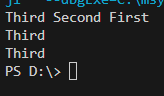


**Завдання №5**

****

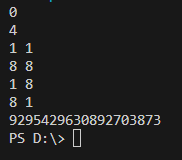
1. Результати виконання завдань, тестування та фактично затрачений час

**Завдання №1**



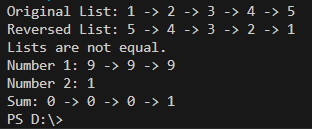
Витратив 40 хвилин.

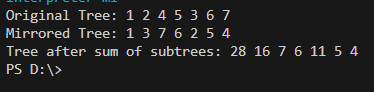
**Завдання №2**

****

Витратив на завдання близько 1 години.

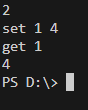
**Завдання №3**

****



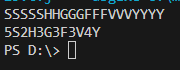
На це завдання пішло 45 хвилин.

**Завдання №4**

****

Витратив на завдання приблизно 60 хвилин.

**Завдання №5**

****

На завдання пішло пів години.

**Висновки:**

У ході виконання лабораторної роботи було досліджено та освоєно основи роботи з динамічними структурами даних, такими як черга, стек, списки та дерево. Вивчені алгоритми обробки цих структур, а також важливість вибору відповідної структури для конкретної задачі. Завдяки застосуванню об’єктно-орієнтованого підходу та створенню класів для реалізації зазначених структур, було поглиблено розуміння принципів інкапсуляції, абстракції та модульності в програмуванні. Отримані знання дозволяють ефективно використовувати ці структури даних для вирішення широкого спектра практичних задач, а також сприяють розвитку навичок розробки оптимізованих алгоритмів.