Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

Зображення, що містить текст, Шрифт, коло, логотип

Автоматично згенерований опис

**Звіт**

про виконання

**Лабораторних та практичних робіт № 6**

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

На тему: «Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур.»

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

***Виконав:***

студент групи ШІ – 11

Яровой Павло Олегович

Львів 2023

# **Тема роботи:**

*Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур.*

# **Мета роботи:**

# *Ознайомлення та вивчення використання динамічних структур даних, таких як Черга, Стек, Списки та Дерево, а також освоєння алгоритмів обробки даних в дереві. Розгляд особливостей застосування кожної структури та їх використання для оптимальної ефективності в конкретних завданнях.*

# **Теоретичні відомості:**

1. Індивідуальний план опрацювання теорії:

Тема №1: Динамічні структури (Черга, Стек, Списки, Дерево).

* + Джерела Інформації
    - <https://www.geeksforgeeks.org/data-structures/>
    - <https://www.boardinfinity.com/blog/guide-to-5-data-structures-in-c/>

Тема №2: Алгоритми обробки Дерев.

* + Джерела Інформації:
    - <https://www.programiz.com/dsa/trees>

# **Виконання роботи:**

## **1. Опрацювання завдання та вимог до програм та середовища:**

**TASK № 1 VNS LAB 10 Variant 20**

*Написати програму, у якій створюються динамічні структури й виконати їхню обробку у відповідності зі своїм варіантом. Для кожного варіанту розробити такі функції:*

*1. Створення списку.*

*2. Додавання елемента в список (у відповідності зі своїм варіантом).*

*3. Знищення елемента зі списку (у відповідності зі своїм варіантом).*

*4. Друк списку.*

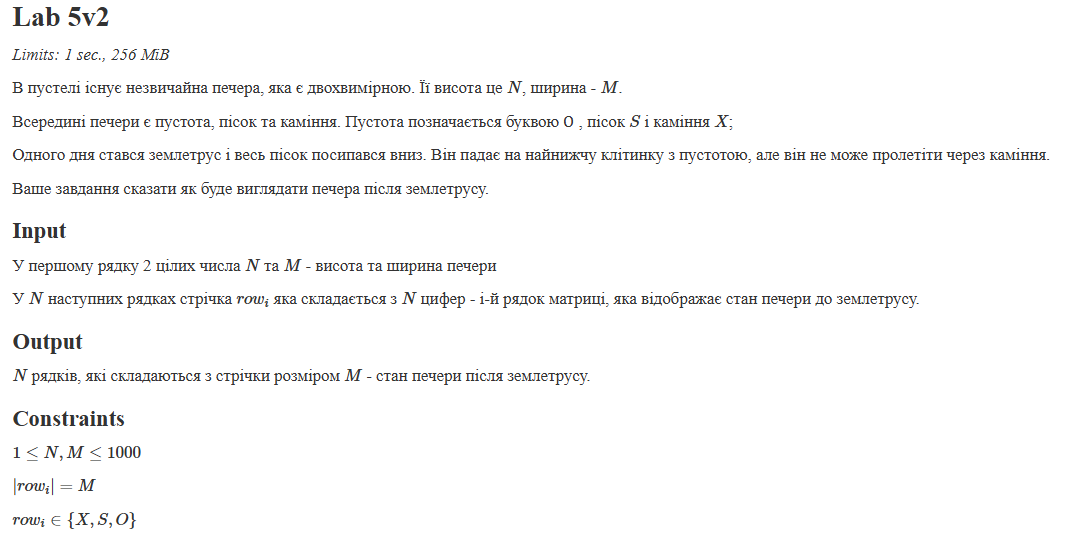
*5. Запис списку у файл.*

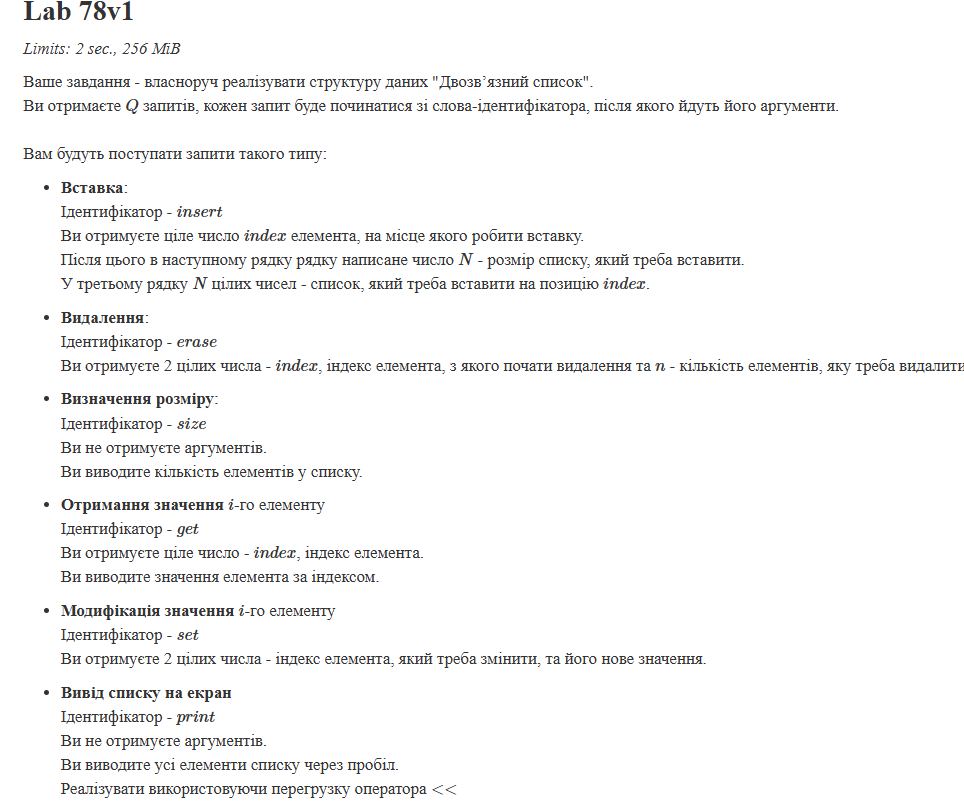
*6. Знищення списку.*

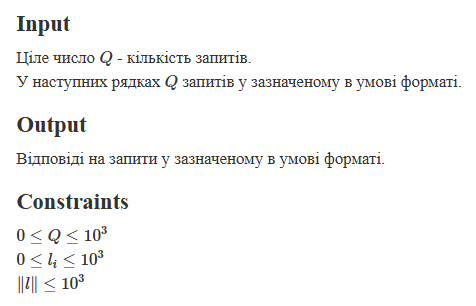
*7. Відновлення списку з файлу.*

*20.Записи в лінійному списку містять ключове поле типу \*char (рядок символів). Сформувати двонаправлений список. Знищити елемент із заданим ключем. Додати по К елементів на початок й в кінець списку.*

**TASK № 2 Algotester Lab 5v2**

**  
TASK№ 3 Algotester Lab 78v1 (100%)**

****

****

**TASK№ 4 Class Practice Task (1-5)**

## **Задача №1 - Реверс списку (Reverse list)**

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

*Умови задачі:*

- використовувати цілочисельні значення в списку;

- реалізувати метод реверсу;

- реалізувати допоміжний метод виведення вхідного і обернутого списків;

## **Задача №2 - Порівняння списків**

bool compare(Node \*h1, Node \*h2);

*Умови задачі:*

- використовувати цілочисельні значення в списку;

- реалізувати функцію, яка ітеративно проходиться по обох списках і порівнює дані в кожному вузлі;

- якщо виявлено невідповідність даних або якщо довжина списків різна (один список закінчується раніше іншого), функція повертає ***false***.

## **Задача №3 – Додавання великих чисел**

Node\* add(Node \*n1, Node \*n2);

*Умови задачі:*

- використовувати цифри від 0 до 9 для значень у списку;

- реалізувати функцію, яка обчислює суму двох чисел, які збережено в списку; молодший розряд числа записано в голові списка (напр. 379 ⟹ 9→7→3);

- функція повертає новий список, передані в функцію списки не модифікуються.

## **Задача №4 - Віддзеркалення дерева**

TreeNode \*create\_mirror\_flip(TreeNode \*root);

*Умови задачі:*

- використовувати цілі числа для значень у вузлах дерева

- реалізувати функцію, що проходить по всіх вузлах дерева і міняє місцями праву і ліву вітки дерева

- функція повертає нове дерево, передане в функцію дерево не модифікується

## **Задача №5 - Записати кожному батьківському вузлу суму підвузлів**

void tree\_sum(TreeNode \*root);

*Умови задачі:*

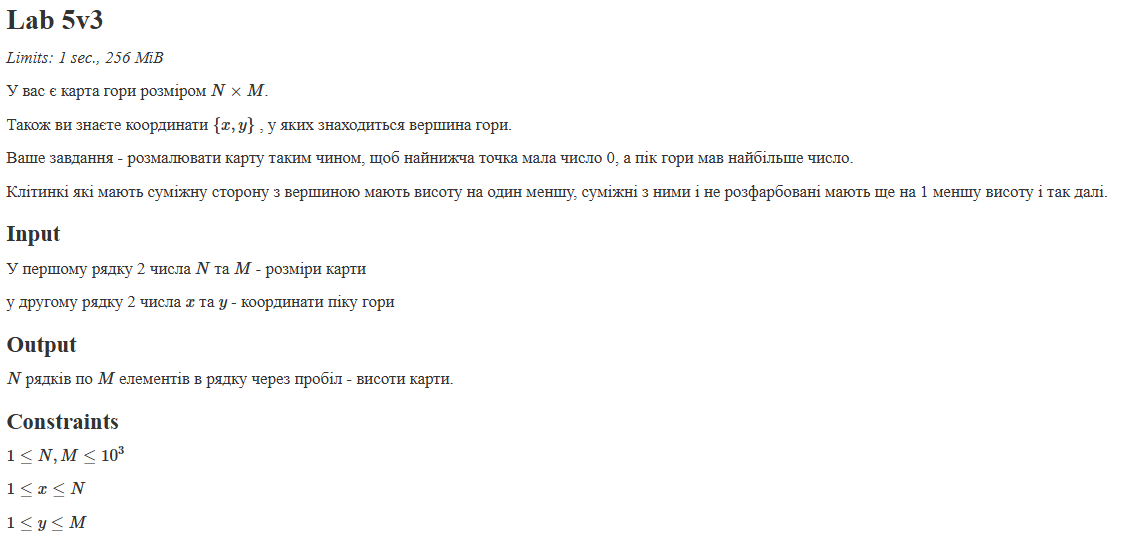
- використовувати цілочисельні значення у вузлах дерева;

- реалізувати функцію, яка ітеративно проходить по бінарному дереві і записує у батьківський вузол суму значень підвузлів

- вузол-листок не змінює значення

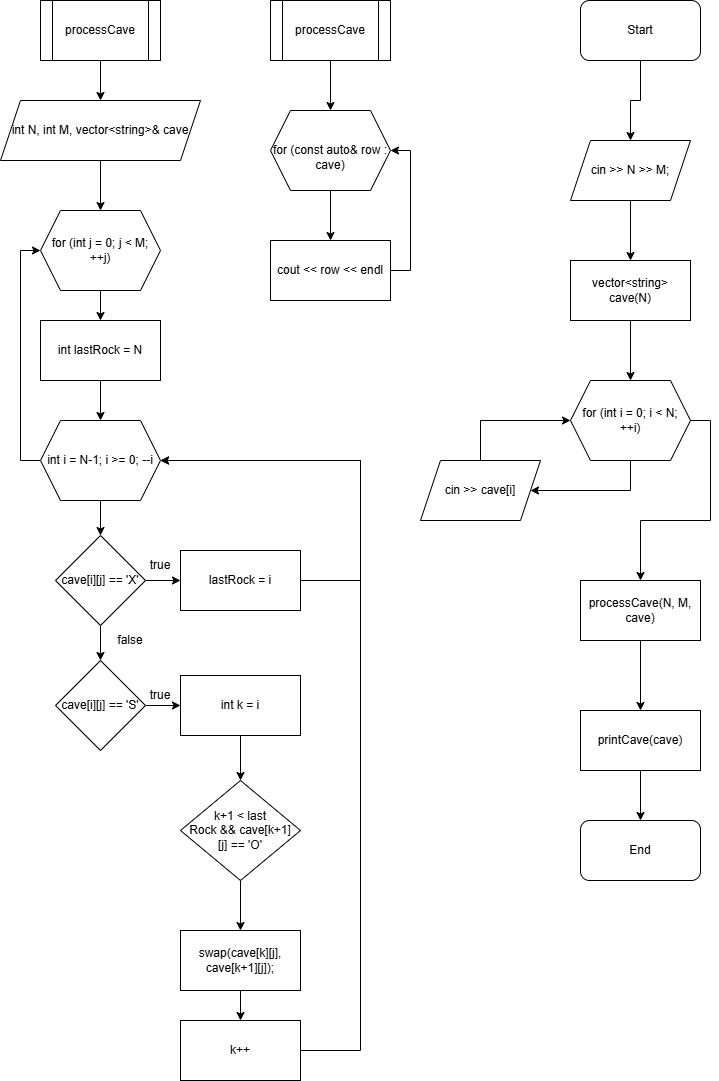
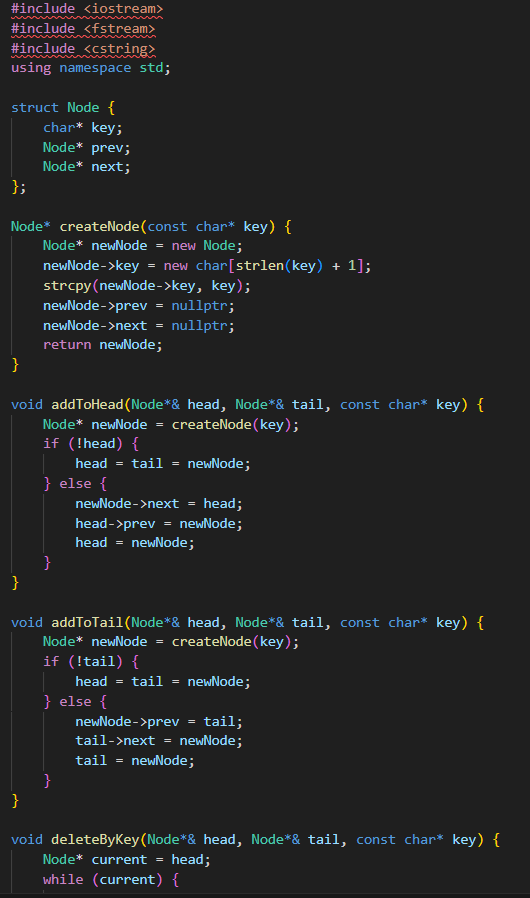
- значення змінюються від листків до кореня дерева

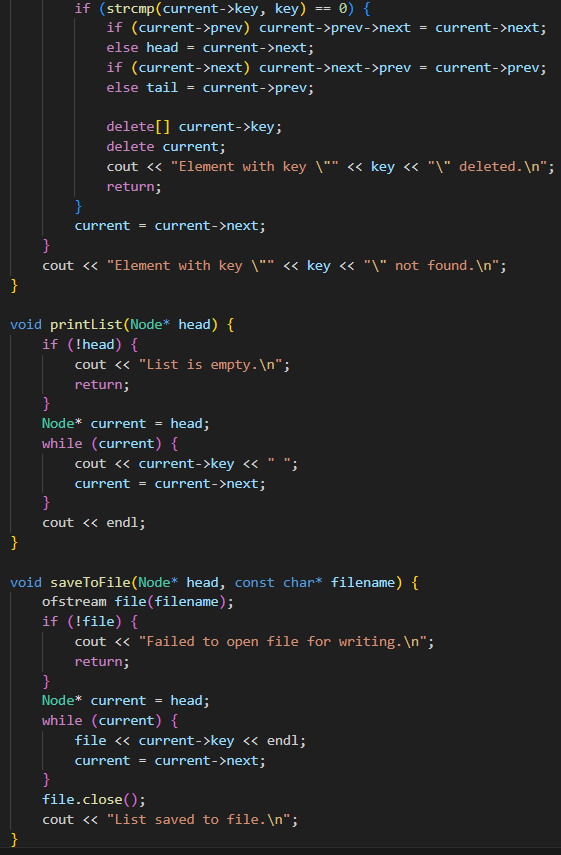
**TASK№ 5 Self Practice Task**

****

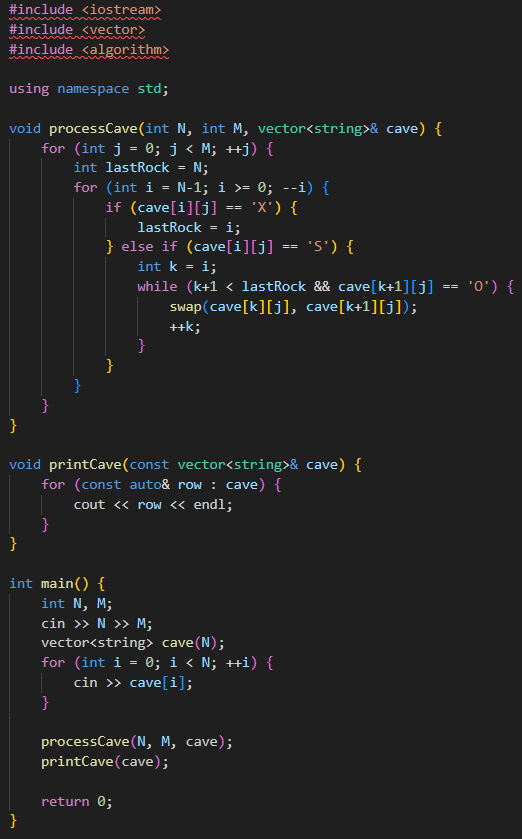
## **2. Дизайн та запланована оцінка часу виконання завдань:**

**Програма №\_1 Algotester Lab 5**

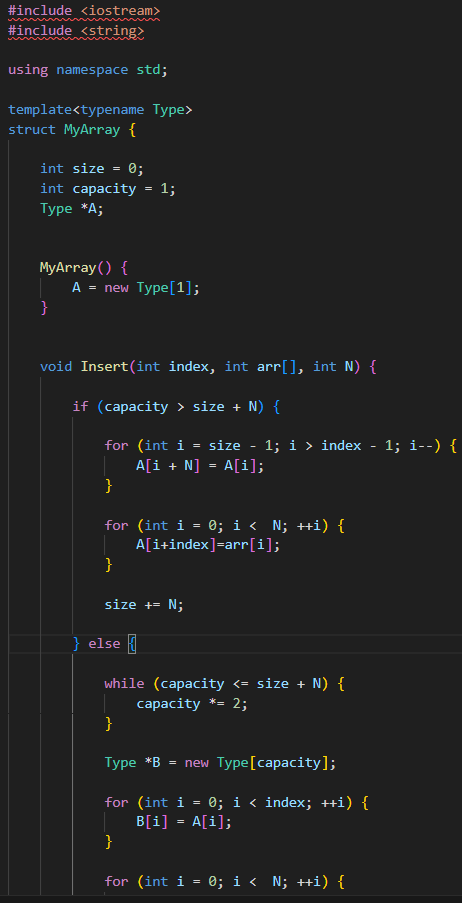
**3. Код програм з посиланням на зовнішні ресурси:TASK № 1 VNS LAB 10**  


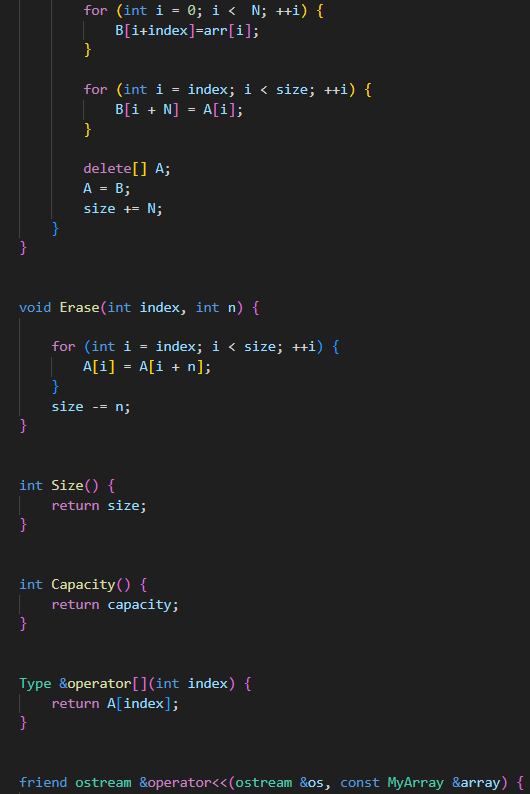


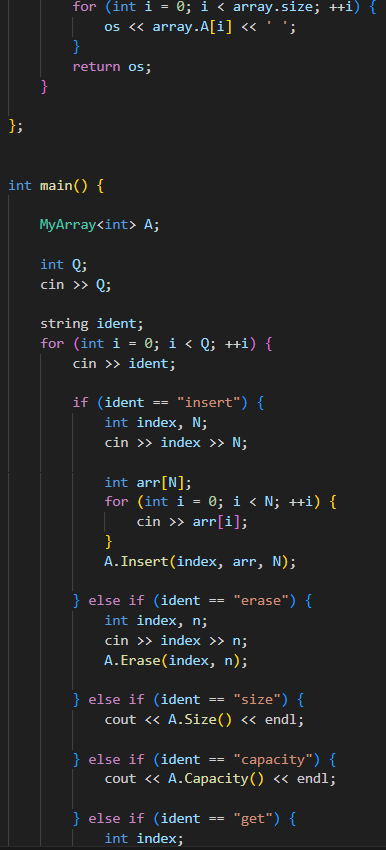
**TASK № 2 Algotester Lab 5v2**

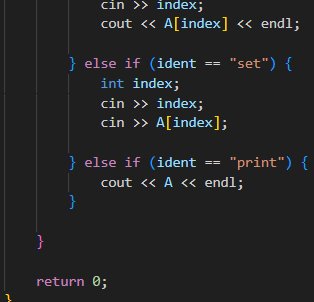


**TASK№ 3 Algotester Lab 78v1 (100%)**

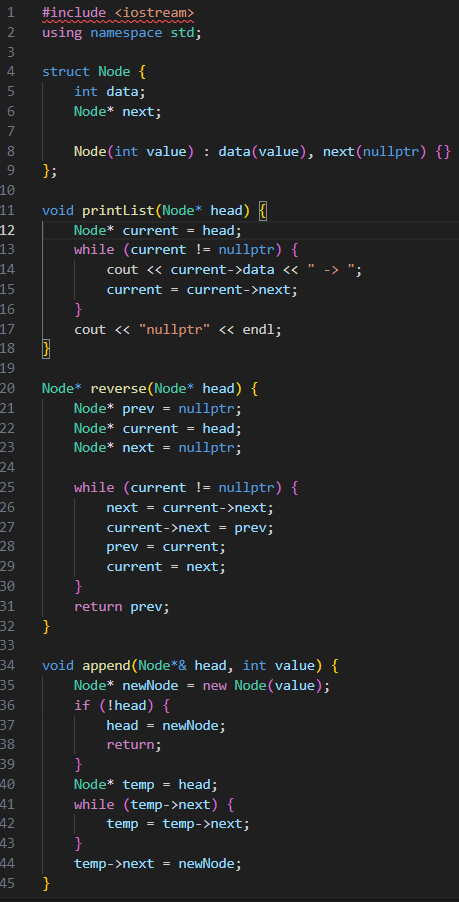
****

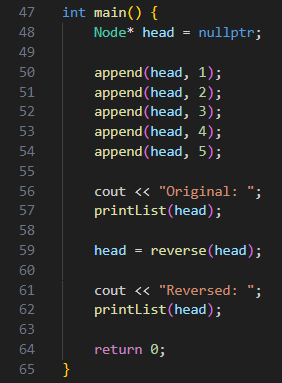
****

****

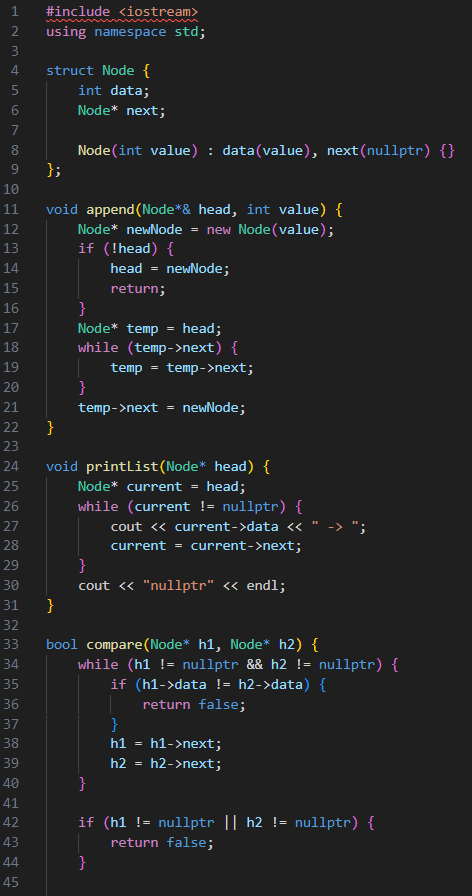
****

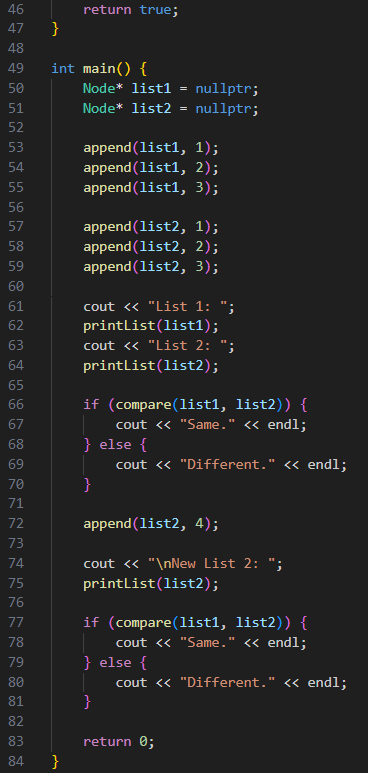
**TASK№ 4 Class Practice Task 1.1**

****

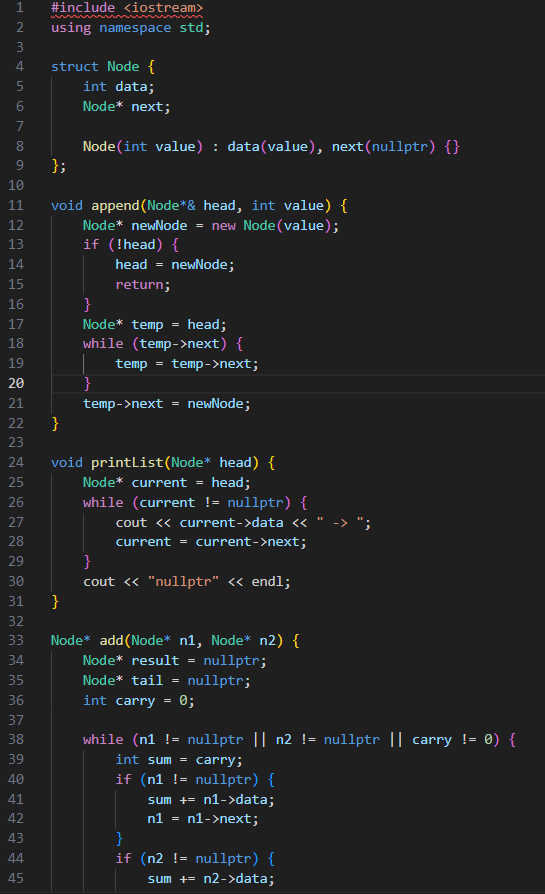
****

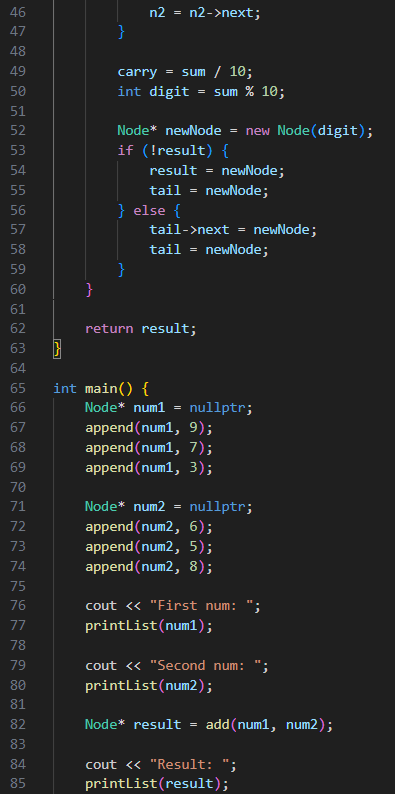
**TASK№ 4 Class Practice Task 1.2**

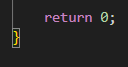
****

****

**TASK№ 4 Class Practice Task 1.3**

****

****

****

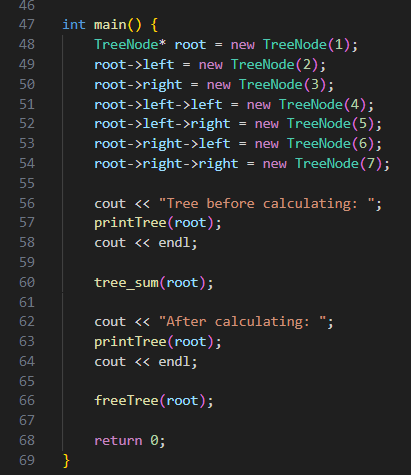
**TASK№ 4 Class Practice Task 1.4**

****

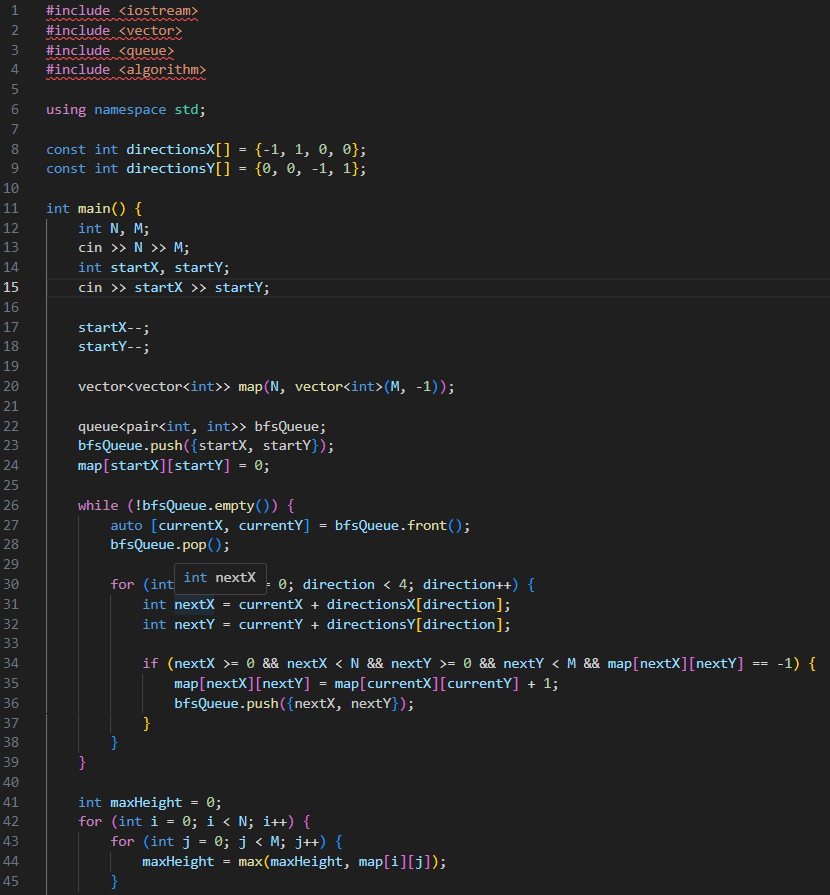
****

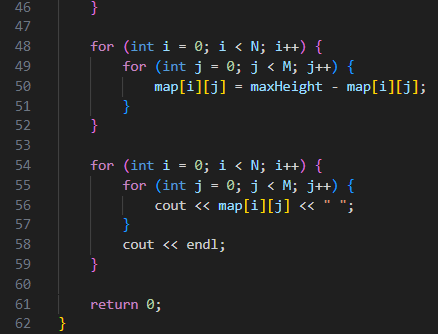
**TASK№ 4 Class Practice Task 1.5**

****

****

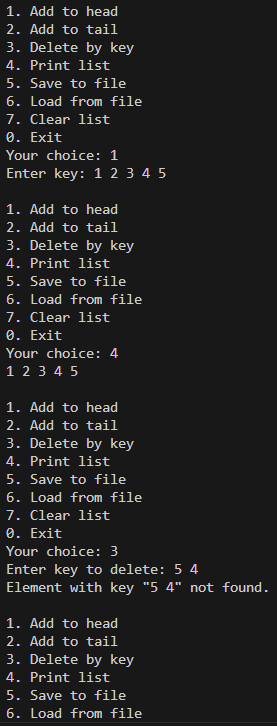
**TASK№ 5 Self Practice Task**

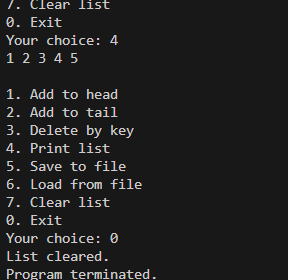
****

****

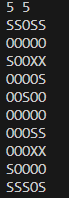
## **4. Результати виконання завдань, тестування та фактично витрачений час:**

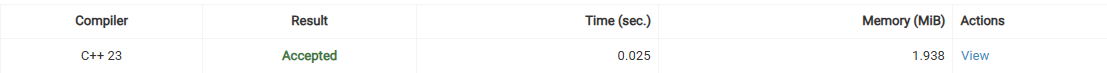
**TASK № 1 VNS LAB 10 Variant 20**

****

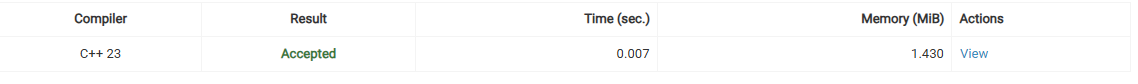
****

**TASK № 2 Algotester Lab 5v2**

****

**  
TASK№ 3 Algotester Lab 78v1 (100%)**

****

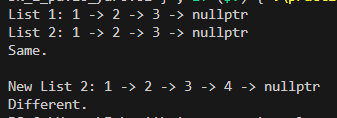
****

**TASK№ 4 Class Practice Task (1-5)**

**1.1**

****

**1.2**

****

**1.3**

****

**1.4**

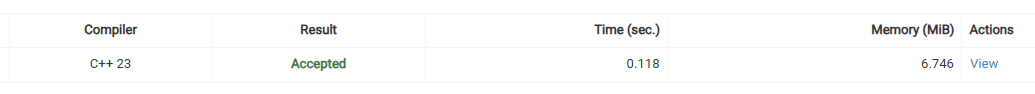
****

**1.5**

****

**TASK№ 5 Self Practice Task**

****

****

# **Висновки:**

Отримавши практичний досвід у використанні динамічних структур даних, я реалізував зв’язний список та впровадив алгоритми обробки дерев. Отримані навички вдосконалили моє розуміння алгоритмів і їхню практичну реалізацію в програмах. Процес вирішення завдань сприяв розвитку логічного мислення та допоміг оптимізувати код. Різноманітні практичні задачі, такі як обчислення суми великих чисел за допомогою зв’язних списків, робота з алгоритмами обробки дерев, дозволили застосовувати теоретичні знання у реальних сценаріях програмування.