Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

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

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

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

до:

ВНС Лабораторної Роботи № 10

Алготестер Лабораторної Роботи № 5

Алготестер Лабораторної Роботи № 7-8

Практичних Робіт до блоку № 6

**Виконала:**

Студентка групи ШІ-12

Хвостова Олександра Андріївна

**Тема роботи:**

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

**Мета роботи:**

Мета цього навчального блоку — ознайомити студентів з основами та особливостями динамічних структур даних. Це включає розуміння того, що таке динамічні структури даних, як вони використовуються для ефективного зберігання і обробки інформації, а також як реалізовувати основні операції над ними. Вивчення цих тем допоможе студентам розробляти ефективні алгоритми і структури даних, які є важливими у багатьох практичних завданнях програмування.

**Теоретичні відомості:**

1. Основи Динамічних Структур Даних:
   1. Вступ до динамічних структур даних: визначення та важливість
   2. Виділення пам'яті для структур даних (stack і heap)
   3. Приклади простих динамічних структур: динамічний масив

Відео. <https://www.youtube.com/watch?v=NyOjKd5Qruk&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=58>

Стаття. <https://drukarnia.com.ua/articles/stack-ta-heap-_c5UU>

Стаття. <https://acode.com.ua/urok-90-dynamichni-masyvy/>

* + Статус: Ознайомлений
  + Початок опрацювання теми: 27.11.2024
  + Завершення опрацювання теми: 28.11.2024

1. Стек:
   1. Визначення та властивості стеку
   2. Операції push, pop, top: реалізація та використання
   3. Приклади використання стеку: обернений польський запис
   4. Переповнення стеку

Стаття. <https://www.geeksforgeeks.org/stack-in-cpp-stl/?ref=header_outind>

Стаття. <https://www.geeksforgeeks.org/stack-push-and-pop-in-c-stl/>

Стаття. <https://www.geeksforgeeks.org/evaluate-the-value-of-an-arithmetic-expression-in-reverse-polish-notation-in-java/>

Стаття. <https://en.wikipedia.org/wiki/Reverse_Polish_notation>

Відео. <https://www.youtube.com/watch?v=gtErSTjL4vE>

Стаття. <https://uk.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%BF%D0%BE%D0%B2%D0%BD%D0%B5%D0%BD%D0%BD%D1%8F_%D1%81%D1%82%D0%B5%D0%BA%D0%B0>

* + Статус: Ознайомлений
  + Початок опрацювання теми: 26.11.2024
  + Завершення опрацювання теми: 27.11.2024

1. Черга:
   1. Визначення та властивості черги
   2. Операції enqueue, dequeue, front: реалізація та застосування
   3. Приклади використання черги: обробка подій, алгоритми планування
   4. Розширення функціоналу черги: пріоритетні черги

Стаття. <https://studfile.net/preview/6889815/page:4/>

Відео. <https://www.youtube.com/watch?v=Yhw8NbjrSFA&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=142>

* + Статус: Ознайомлений
  + Початок опрацювання теми: 24.11.2024
  + Завершення опрацювання теми: 25.11.2024

1. Зв'язні Списки:
   1. Визначення однозв'язного та двозв'язного списку
   2. Принципи створення нових вузлів, вставка між існуючими, видалення, створення кільця(circular linked list)
   3. Основні операції: обхід списку, пошук, доступ до елементів, об'єднання списків
   4. Приклади використання списків: управління пам'яттю, FIFO та LIFO структури

Відео. <https://www.youtube.com/watch?v=3MfhrHWBdgo&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=138>

Відео. <https://www.youtube.com/watch?v=-25REjF_atI&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=139>

* + Статус: Ознайомлений
  + Початок опрацювання теми: 21.11.2024
  + Завершення опрацювання теми: 23.11.2024

1. Дерева:
   1. Вступ до структури даних "дерево": визначення, типи
   2. Бінарні дерева: вставка, пошук, видалення
   3. Обхід дерева: в глибину (preorder, inorder, postorder), в ширину
   4. Застосування дерев: дерева рішень, хеш-таблиці
   5. Складніші приклади дерев: AVL, Червоно-чорне дерево

Відео. <https://www.youtube.com/watch?app=desktop&v=alxzyWswCVg&t=0s>

Відео. <https://youtu.be/qBFzNW0ALxQ?si=o_QP6uPfbXygyKst>

Відео. <https://www.youtube.com/watch?v=87-lAYP9KCA>

Відео. <https://youtu.be/__66xMXz7wc?si=fEphxzcHedgeh3p0>

Відео. <https://youtu.be/qvZGUFHWChY?si=o2_yh67M4GD5ht2F>

Відео. <https://youtu.be/DB1HFCEdLxA?si=3-7GVYCgUp2paojA>

* + Статус: Ознайомлений
  + Початок опрацювання теми: 15.11.2024
  + Завершення опрацювання теми: 21.11.2024

1. Алгоритми Обробки Динамічних Структур:
   1. Основи алгоритмічних патернів: ітеративні, рекурсивні
   2. Алгоритми пошуку, сортування даних, додавання та видалення елементів
   * Статус: Ознайомлений
   * Початок опрацювання теми: 12.11.2024
   * Завершення опрацювання теми: 20.11.2024

**Виконання роботи:**

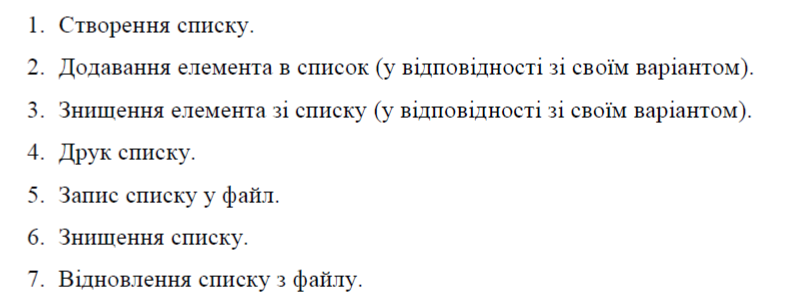
**1. Опрацювання завдання та вимог до програм та середовища:**

Завдання №1. VNS Lab 10 - Task 7

* Варіант завдання: 7
* Планований час на виконання: 1 година
* Деталі завдання

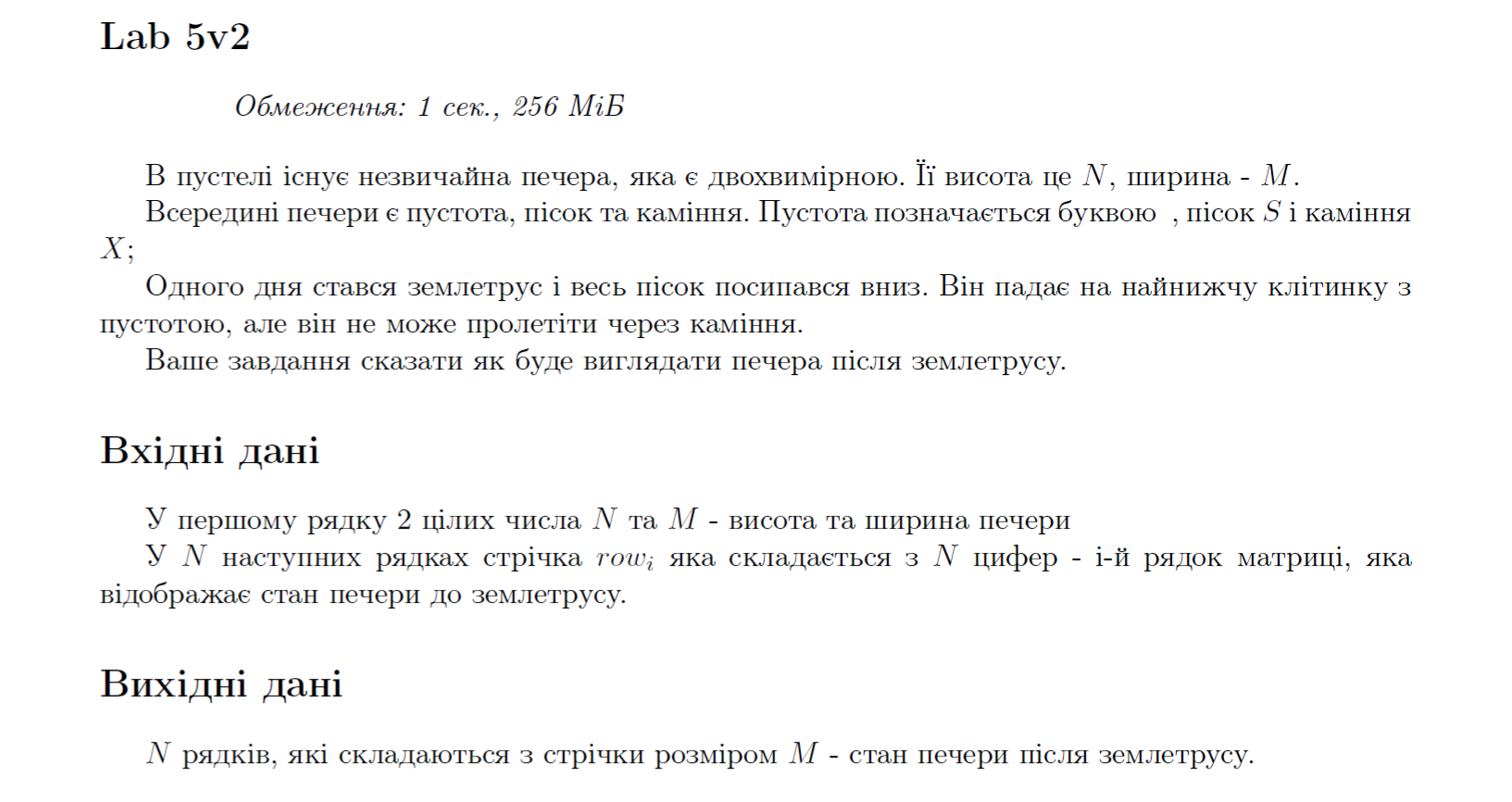


* Важливі деталі для врахування в імплементації програми



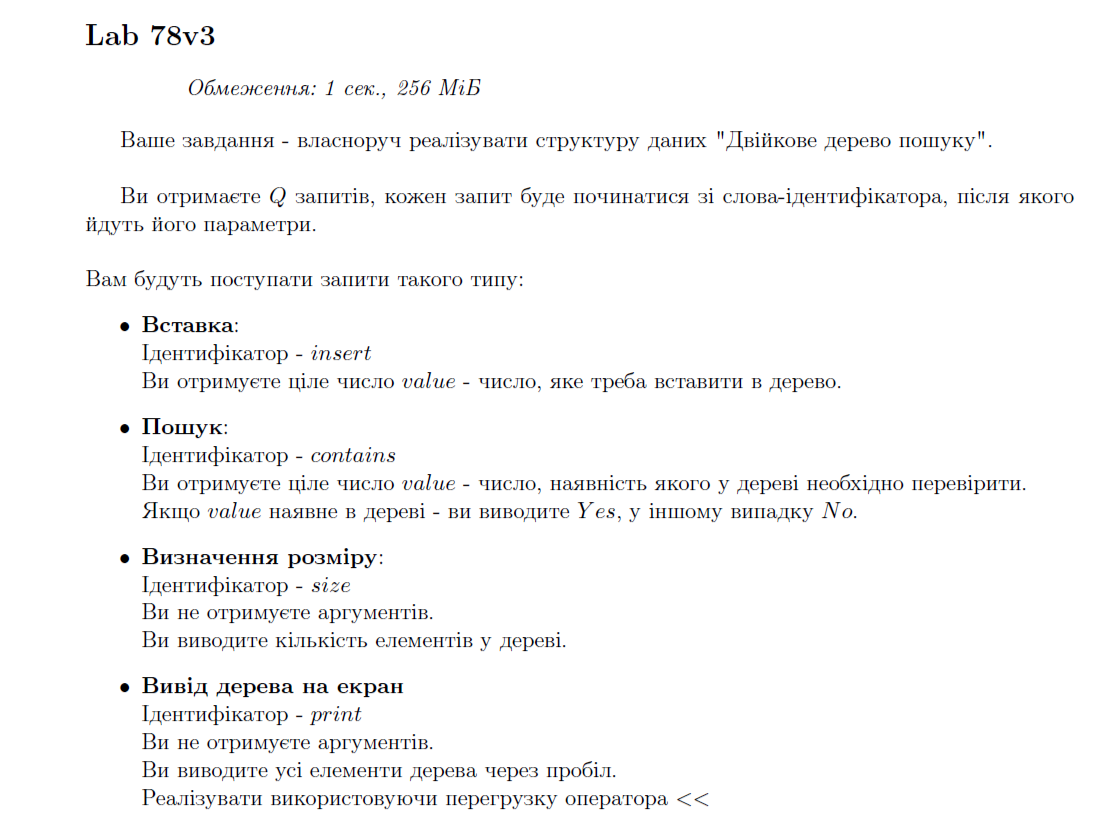
Завдання №2. Algotester. Lab 5. V2

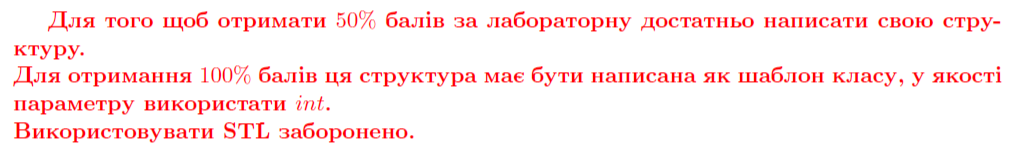
* Планований час виконання: 3 години



Завдання №3-4. Algotester. Lab 78. V3

* Планований час виконання: 3 години





## Завдання №5. Class Practice Work

* Планований час виконання: 3 години

## 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;

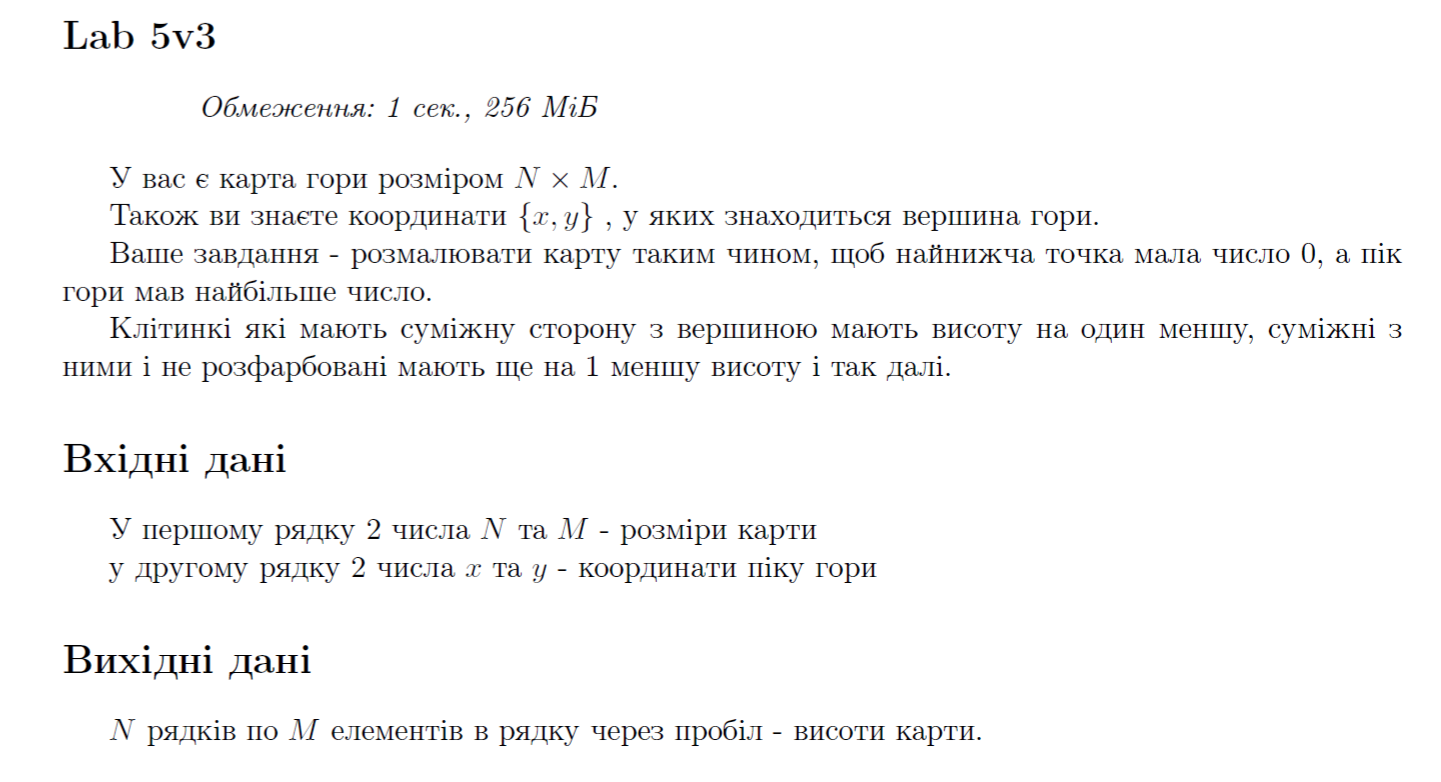
- реалізувати функцію, яка ітеративно проходить по бінарному дереві і записує у батьківський вузол суму значень підвузлів

- вузол-листок не змінює значення

- значення змінюються від листків до кореня дерева

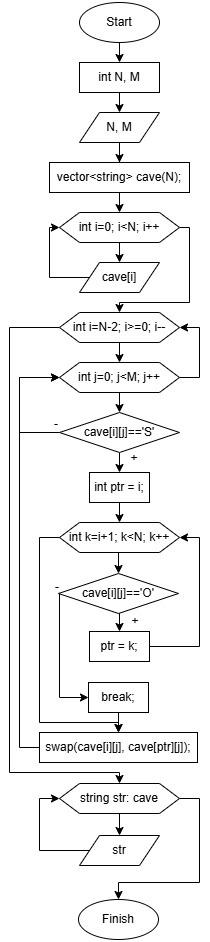
Завдання №6. Self Practice Work

* Планований час виконання: 3 години



**2. Дизайн та планована оцінка часу виконання завдань:**

Завдання №2. Algotester. Lab 5. V2

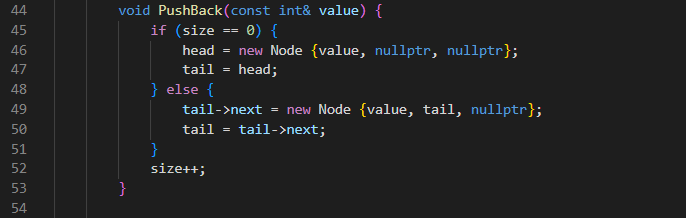
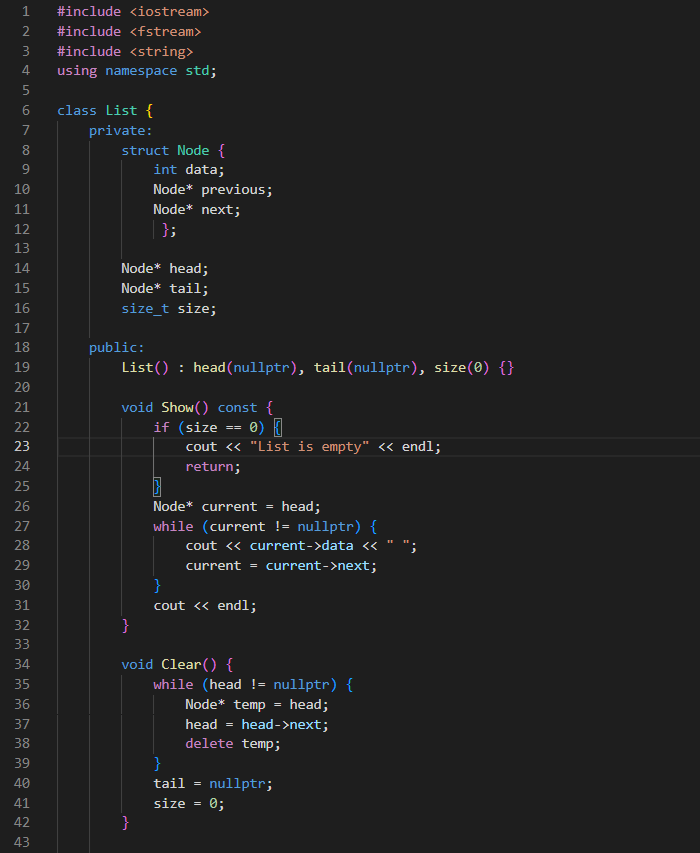


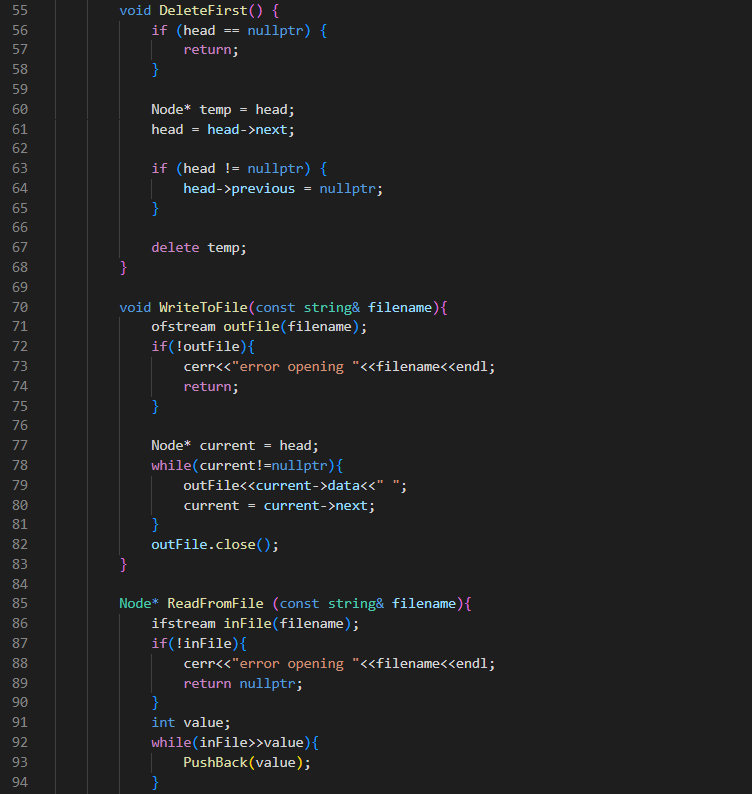
**4. Код програм з посиланням на зовнішні ресурси:**

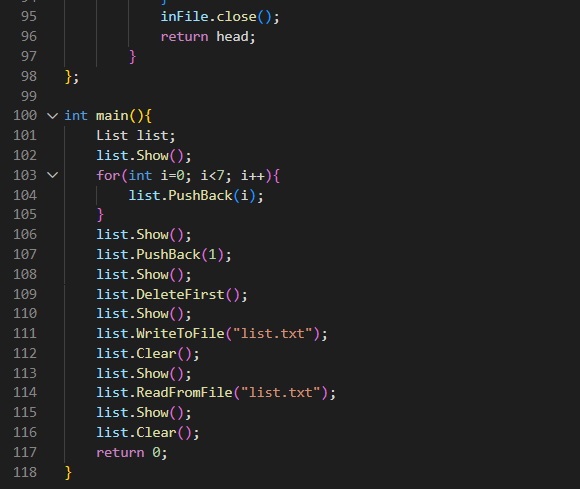
Завдання №1. VNS Lab 10 - Task 7

* Варіант завдання: 7

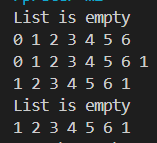
<https://github.com/artificial-intelligence-department/ai_programming_playground_2024/blob/db21ed7dfa6888a84a9c4da046edbcdf3ed4aa12/ai_12/oleksandra_khvostova/epic_6/vns_lab_10_task_oleksandra_khvostova.cpp>





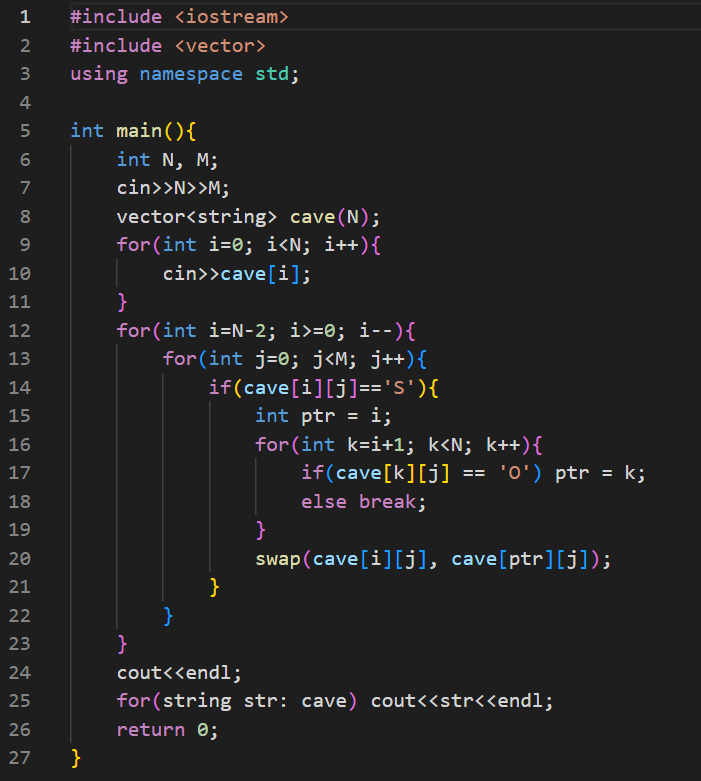


Час затрачений на виконання завдання: 2 години

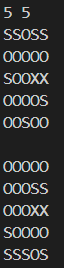


Завдання №2. Algotester. Lab 5. V2

<https://github.com/artificial-intelligence-department/ai_programming_playground_2024/blob/db21ed7dfa6888a84a9c4da046edbcdf3ed4aa12/ai_12/oleksandra_khvostova/epic_6/algotester_lab_5_oleksandra_khvostova.cpp>



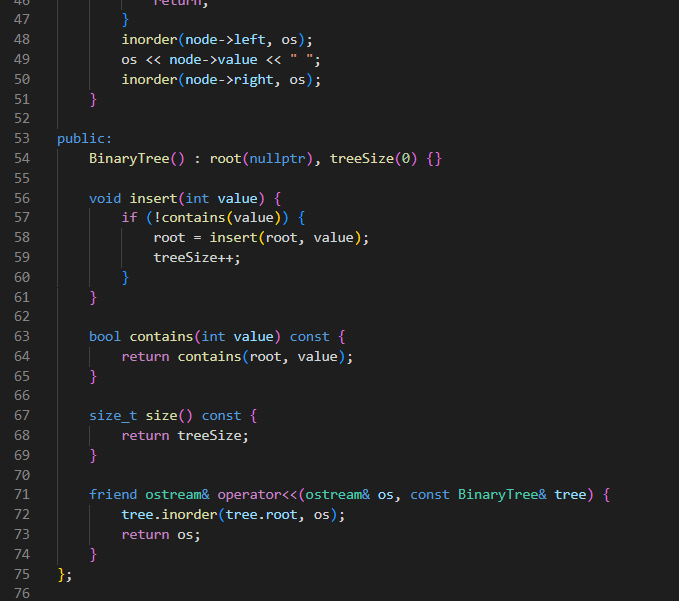
Час затрачений на виконання завдання: 2 години

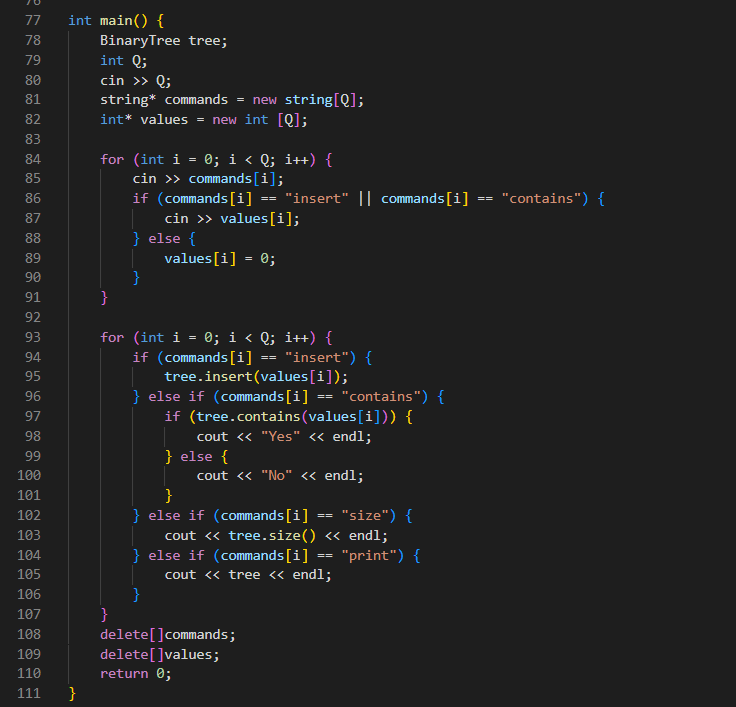


Завдання №3. Algotester. Lab 78. V3

<https://github.com/artificial-intelligence-department/ai_programming_playground_2024/blob/db21ed7dfa6888a84a9c4da046edbcdf3ed4aa12/ai_12/oleksandra_khvostova/epic_6/algotester_lab_78_1_oleksandra_khvostova.cpp>

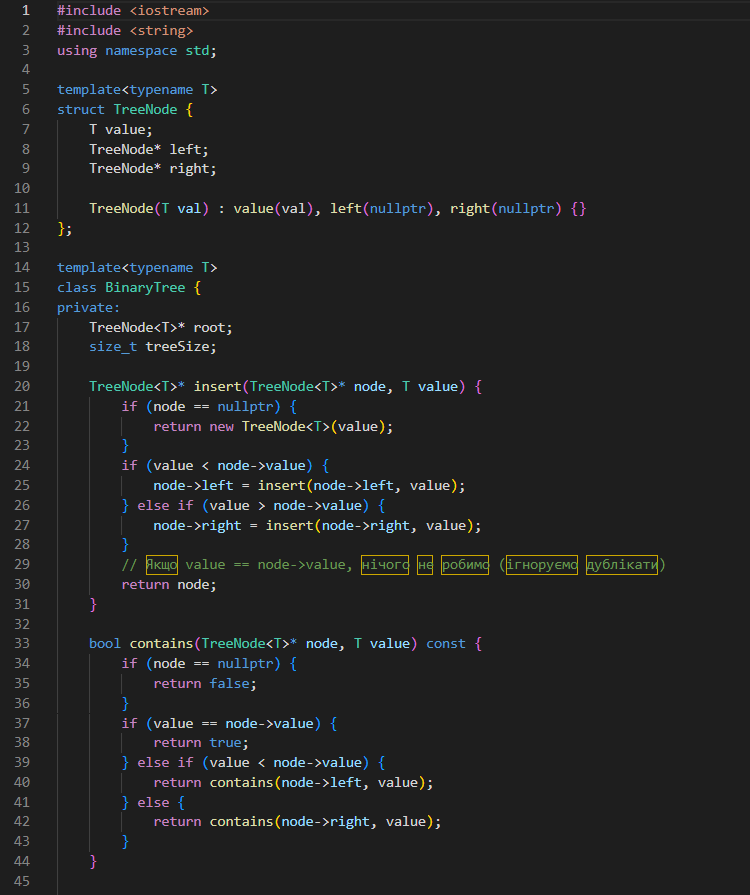


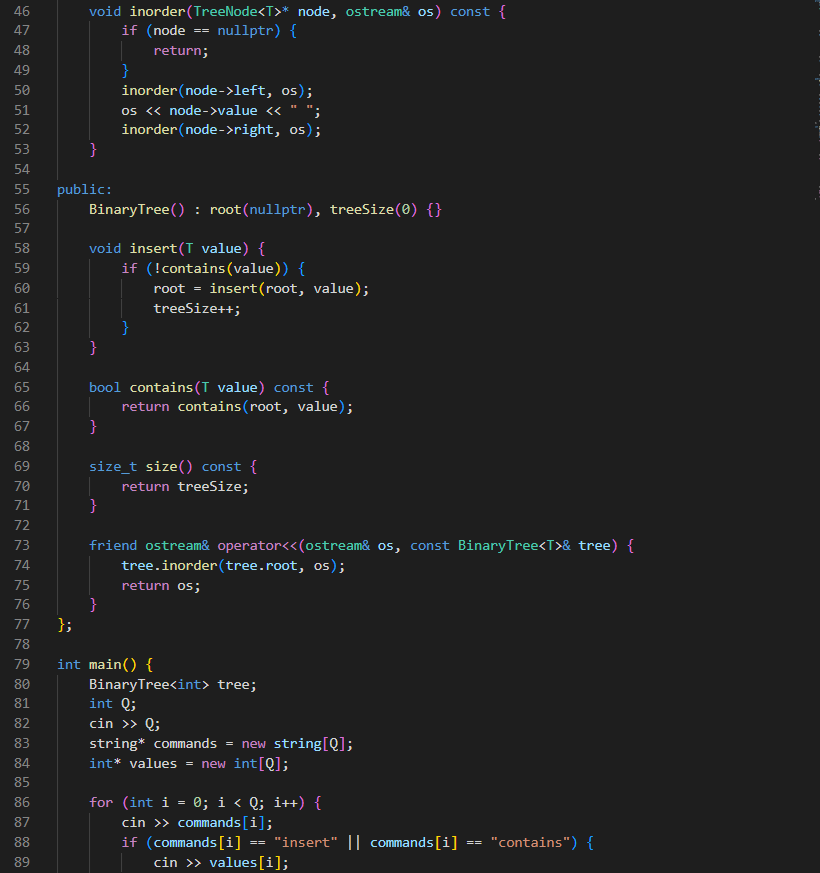


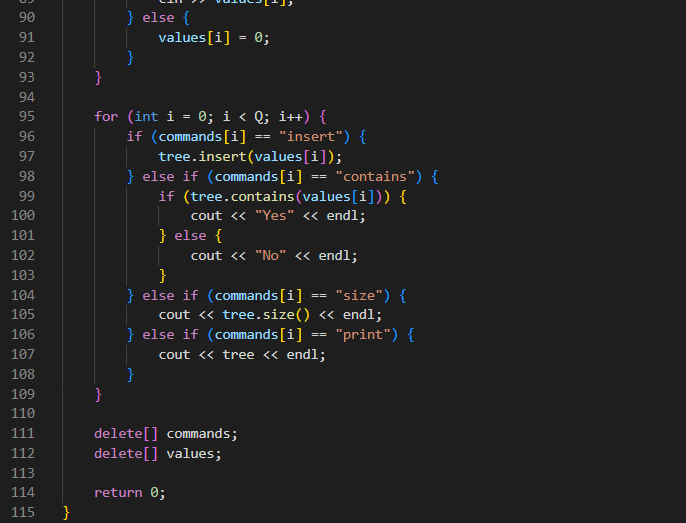


Завдання №4. Algotester. Lab 78. V3

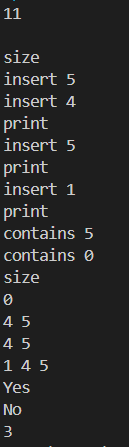
<https://github.com/artificial-intelligence-department/ai_programming_playground_2024/blob/db21ed7dfa6888a84a9c4da046edbcdf3ed4aa12/ai_12/oleksandra_khvostova/epic_6/algotester_lab_78_2_oleksandra_khvostova.cpp>







Час затрачений на виконання завдання: 4 години



## Завдання №5. Class Practice Work

## <https://github.com/artificial-intelligence-department/ai_programming_playground_2024/blob/db21ed7dfa6888a84a9c4da046edbcdf3ed4aa12/ai_12/oleksandra_khvostova/epic_6/practice_work_task_oleksandra_khvostova.cpp>

## 

## 

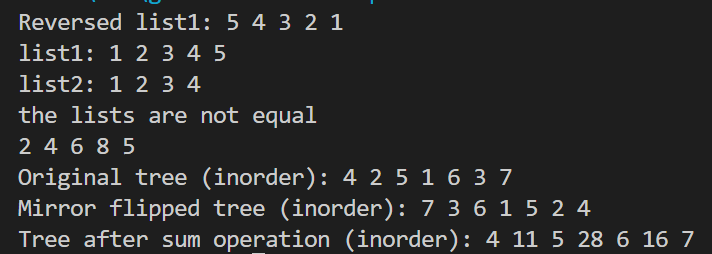
## 

## 

## 

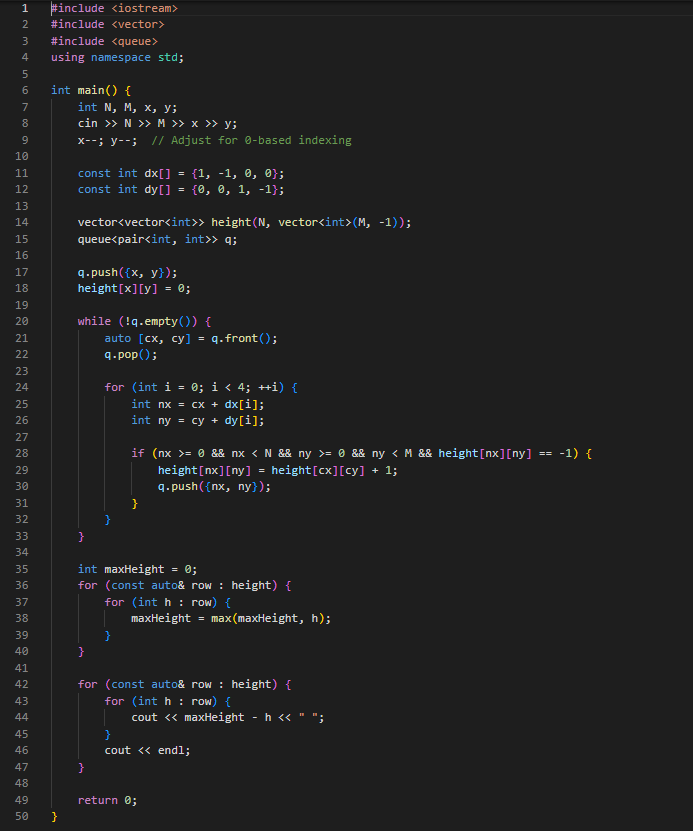
## 

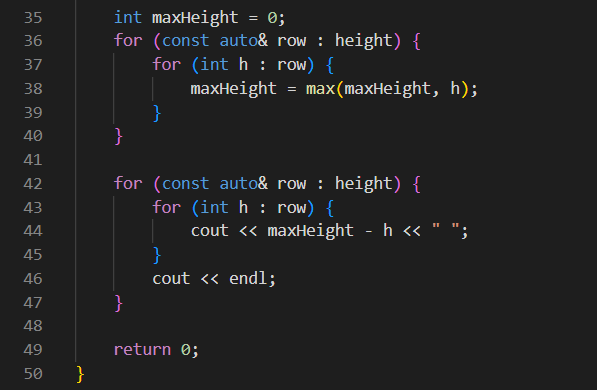
Час затрачений на виконання: 4 години



Завдання №6. Self Practice Work

<https://github.com/artificial-intelligence-department/ai_programming_playground_2024/blob/db21ed7dfa6888a84a9c4da046edbcdf3ed4aa12/ai_12/oleksandra_khvostova/epic_6/practice_work_self_algotester_tasks_oleksandra_khvostova.cppv>



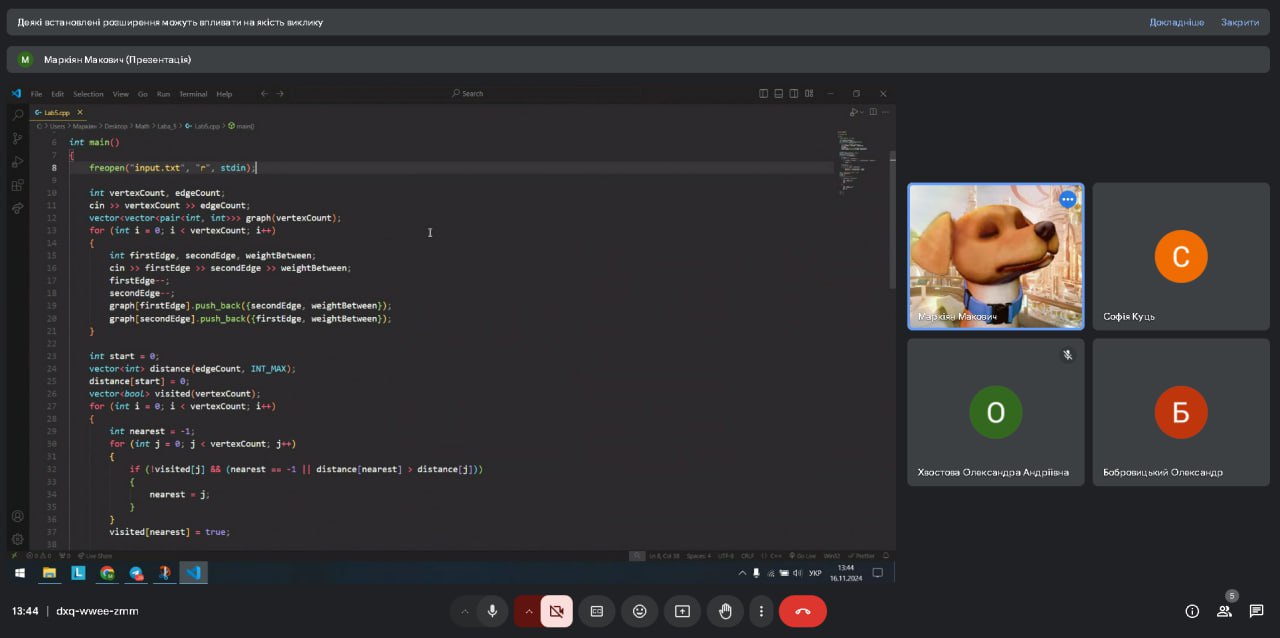


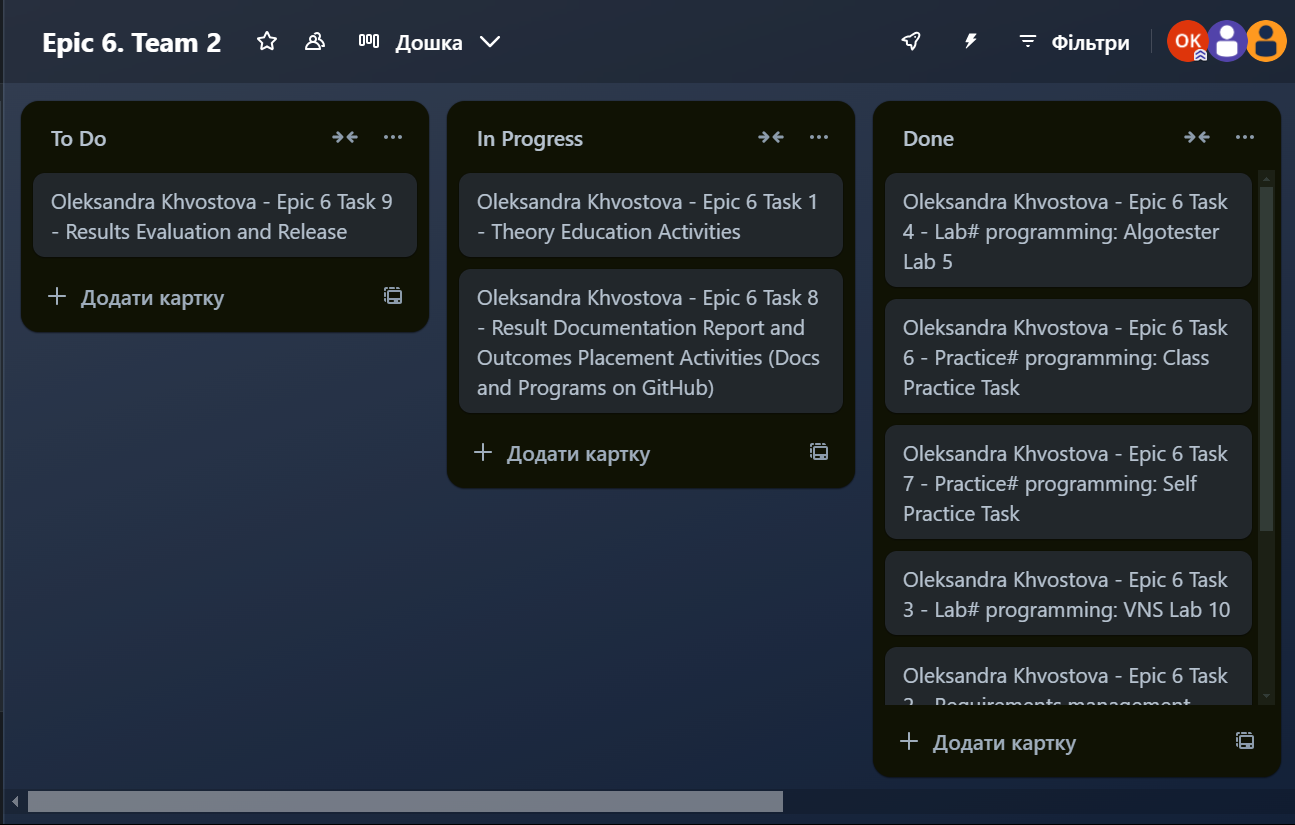
## Час затрачений на виконання: 2 години

## 

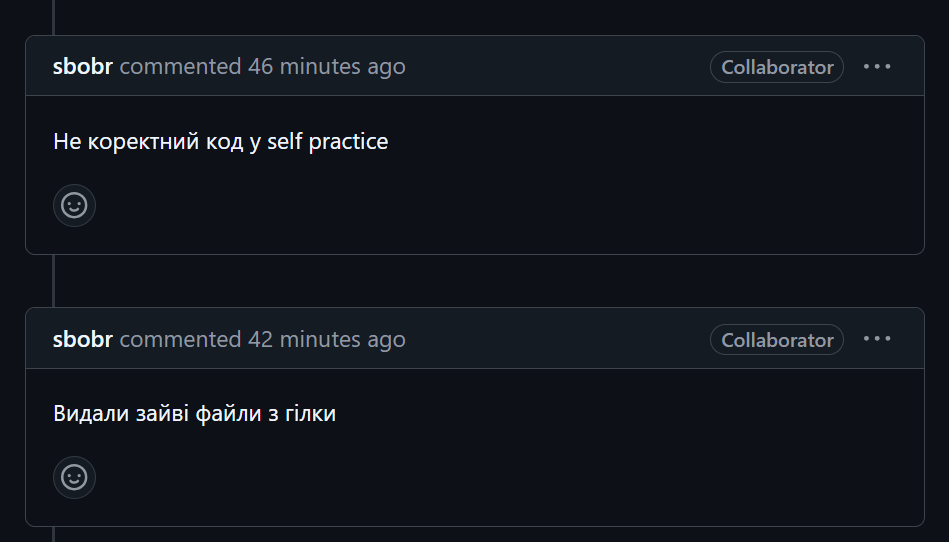
**6. Кооперація з командою:**

* Скрін з 1-ї зустрічі по обговоренню задач Епіку та Скрін прогресу по Трелло





Скрін з 2-му коментарями від учасників команди на пул реквесті з Ревю Роботи



**Висновки:**

Динамічні структури даних дозволяють ефективно керувати пам'яттю, забезпечуючи змінний розмір і можливість динамічного виділення пам'яті. Стек працює за принципом LIFO і використовується для збереження тимчасових даних або рекурсивних операцій. Черга працює за принципом FIFO і підходить для обробки подій та алгоритмів планування. Зв'язні списки дозволяють ефективно управляти пам'яттю і реалізовувати структури FIFO та LIFO. Дерева використовуються для ефективного зберігання та пошуку інформації, забезпечуючи швидкі операції вставки, пошуку і видалення елементів. Алгоритми пошуку, сортування і додавання/видалення елементів є ключовими для обробки даних у динамічних структурах даних.