Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра систем штучного інтелекту

Звіт

про виконання лабораторних та практичних робіт блоку № 4

На тему: «Одновимірні масиви. Двовимірні Масиви. Вказівники та Посилання. Динамічні масиви. Структури даних. Вкладені структури. Алгоритми обробки та робота з масивами та структурами.»

з дисципліни: «Основи програмування»

до:

ВНС Лабораторної Роботи №4
ВНС Лабораторної Роботи №5
Алготестер Лабораторної Роботи №2
Алготестер Лабораторної Роботи №3
Практичних Робіт до блоку №4

Виконав:
Студент групи ШІ-11
Цяпа Остап Андрійович

Львів 2024

Тема роботи:

Одновимірні масиви. Двовимірні Масиви. Вказівники та Посилання. Динамічні масиви. Структури даних. Вкладені структури. Алгоритми обробки та робота з масивами та структурами.

Мета роботи:

Закріпити на практиці вивчений теоретичний матеріал, зокрема робота з маисивами, двовимірними та одновимірними, а також з вказівниками та посиланнями. Також розібратися з динамічними масивами та динамічною пам’яттю в С++.

Теоретичні відомості:

- Тема №1: Класи пам’яті в С++.
- Тема №2: Вступ до масивів і вказівників.
- Тема №3: Одновимірні масиви.
- Тема №4: Вказівники та посилання.
- Тема №5: Двовимірні масиви.
- Тема №6: Динамічні масиви.
- Тема №7: Структури даних.
- Тема №8: Вкладені структури.
- Тема №9: Використання структур.
- Тема №10: Алгоритми обробки та робота з масивами та структурами.

1) Індивідуальний план опрацювання теорії:

- Тема №1: Класи пам’яті в С++:
  - Джерела інформації:
    - Статті.

    http://cpp.dp.ua/klasy-pam-yati-u-c-builder/
  - Що опрацьовано:
    - Статична пам’ять.
    - Динамічна пам’ять.
    - Поняття стеку.
    - Виділення та вивільнення пам’яті.

    Запланований час на вивчення 30 хвилин.

    Витрачений час 30 хвилин.
- Тема №2: Вступ до Масивів і Вказівників:
  - Джерела інформації:
    - Відео.

    https://www.youtube.com/watch?v=zopWRlYOXWw&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=57
  - Що опрацьовано:
    - Основи масивів: визначення, важливість, приклади використання.
    - Різниця між статичними та динамічними масивами.
    - Основи вказівників: що це таке, як вони працюють.
    - Взаємозв'язок між масивами та вказівниками.
    - Вступ до посилань: основні концепції та відмінності від вказівників.

    Запланований час на вивчення 1 година.

    Витрачений час 1 година.

- Тема №3: Одновимірні масиви:

- Джерела інформації:
  - Відео.

    https://www.youtube.com/watch?v=ULdbOaMBPYc&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=40
- Що опрацьовано
  - Створення та ініціалізація одновимірних масивів.
  - Цикли та обхід масивів.
  - Використання функцій для роботи з масивами.
  - Приклади алгоритмів сортування та пошуку.

  Запланований час на вивчення 1 година.
  Витрачений час 1 година.

- Тема №4: Вказівники та Посилання:
  - Джерела інформації:
    - Відео.

      https://www.youtube.com/watch?v=ULdbOaMBPYc&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=40
    - Що опрацьовано
      - Використання вказівників для доступу до елементів масиву.
      - Різниця між вказівниками та посиланнями в контексті функцій.
      - Динамічне виділення пам'яті з використанням вказівників.
      - Використання вказівників для створення складних структур даних.

      Запланований час на вивчення 2 години.
      Витрачений час 2 години.

- Тема № 5: Двовимірні Масиви:
  - Джерела інформації:
    - Статті.

      https://www.youtube.com/watch?v=hcYgFCgeZzQ
    - Що опрацьовано
      - Оголошення та ініціалізація двовимірних масивів.
      - Практичні приклади використання двовимірних масивів.
      - Передача двовимірних масивів у функції.
      - Застосування двовимірних масивів для розв'язання задач.

      Запланований час на вивчення 1 година.
      Витрачений час 1 година..

- Тема №6: Динамічні Масиви:
  - Джерела інформації:
    - Відео

      https://www.youtube.com/watch?v=OGR9VJEh8Hk&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=60
    - Що опрацьовано
      - Основи динамічного виділення пам'яті.
      - Створення та управління динамічними масивами.
      - Реалізація змінної розмірності масивів.
      - Передача динамічних масивів у функції.

      Запланований час на вивчення 1 година.
      Витрачений час 1 година.

- Тема №7: Структури Даних:
  - Джерела інформації:
    - Відео.

https://www.youtube.com/watch?v=999IE-6b7_s&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=63

- Що опрацьовано
  - Покращив свої знання у темі оголошення та використання структур.
  - Використання масивів та вказівників у структурах.
  - Функції для обробки даних у структурах.
  - Використання структур для представлення складних даних.
  - Вкладені структури та їх використання.

Запланований час на вивчення 2 години.
Витрачений час 2 години.

- Тема №8: Вкладені Структури:
  - Джерела інформації:
    - Відео.

https://www.youtube.com/watch?v=999IE-6b7_s&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=63

- Що опрацьовано
  - Поняття вкладених структур та їх оголошення.
  - Взаємодія з вкладеними структурами.
  - Використання вкладених структур для моделювання складних даних.
  - Передача вкладених структур у функції.
  - Приклади реального використання вкладених структур.

Запланований час на вивчення 1 година.
Витрачений час 1 година.

- Тема №9: Використання Структур:
  - Джерела інформації:
    - Відео

https://www.youtube.com/watch?v=999IE-6b7_s&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=63

- Що опрацьовано
  - Перевантаження операторів у структурі.
  - Вивід/ввід структури (operator<<);
  - Практичні задачі на виведення структур та операції з ними

Запланований час на вивчення 1 година.
Витрачений час 1 година.

- Тема №10: Алгоритми обробки та робота з масивами та структурами:
  - Джерела інформації:
    - Відео.

https://www.youtube.com/watch?v=uQxG9gBROog&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=51

https://www.youtube.com/watch?v=maB87eyn7h8&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=52

- Що опрацьовано
  - Алгоритми пошуку та сортування в масивах.
  - Використання циклів та умовних операторів для роботи з масивами та структурами.
  - Інтеграція масивів та структур у алгоритми.
  - Розв'язання практичних задач з використанням масивів та структур

Запланований час на вивчення 1 година.
Витрачений час 1 година.

Виконання роботи:

1. Опрацювання завдання та вимог до програм.

Завдання №1

VNS LAB 4 – TASK 1 (VARIANT 9)

1) Сформувати одновимірний масив цілих чисел, використовуючи генератор випадкових чисел.
2) Роздрукувати отриманий масив.
3) Знищити перший елемент із заданим значенням.
4) Зсунути масив циклічно на K елементів вправо.
5) Роздрукувати отриманий масив.

Завдання №2

VNS LAB 5 – TASK 1 (VARIANT 9)

Написати функцію для обчислення суми елементів квадратної матриці, які розташовані нижче головної діагоналі. З її допомогою знайти максимальне значення такої суми в n матрицях.

Завдання №3

ALGOTESTER LAB 2 (VARIANT 3)

Вам дано масив цілих чисел розміром N, на першій та останній клітинці розміщено по дрону. Вони одночасно взлітають. На початку кожного ходу швидкість дрону стає рівною значенню клітинки, у якій він знаходиться. Тобто лівий дрон у першу секунду з клітинки з індексом 1 перелетить у клітинку з індексом a1, тобто його наступна позиція рахується як поточна позиція + число у поточній позиції (перегляньте пояснення для візуалізації). Правий робить аналогічно в протилежну сторону.
Вони це роблять до моменту, коли трапиться одна з зазначених подій:
Якшо 2 дрони опиняються в одній клітинці - ви виводите **Collision**.
Якщо лівий дрон опиниться справа від правого - це **Miss.**
У випадку якщо вони зупиняться один навпроти одного, тобто у клітинках ai та ai + 1 – виведіть **Stopped.**
Врахуйте, що перевіряти треба також до взльоту.

Завдання №4

ALGOTESTER LAB 3 (VARIANT 2)

Вам дано 2 масиви розміром N та M. Значення у цих масивах унікальні. Ваше завдання вивести у першому рядку кількість елементів, які наявні в обох масивах одночасно, у другому кількість унікальних елементів в обох масивах разом.

Завдання №5

CLASS PRACTICE WORK

Реалізувати програму, яка перевіряє, чи дане слово чи число є паліндромом за допомогою рекурсії.

Вимоги

1. Визначення функції:
    1. Реалізуйте рекурсивну функцію *isPalindrome*, яка перевіряє, чи заданий рядок є паліндромом.
2. Приклад визначення функції:
    1. *bool isPalindrome(const string& str, int start, int end);*
3. Перевантаження функцій:
    1. Перевантажте функцію *isPalindrome* для роботи з цілими значеннями.
    2. *bool isPalindrome*(ціле число);
4. Рекурсія:
    1. Рекурсивна функція для рядків перевірить символи в поточній початковій і кінцевій позиціях. Якщо вони збігаються, він буде рекурсивно перевіряти наступні позиції, поки початок.

Завдання №6

SELF PRACTICE WORK ALGOTESTER

Ви з'явились у світі під назвою Атод посеред Пустелі Безправ'я. Так сталось, що Ви попали саме в той час і місце, де ведеться битва між чаклункою Ліною і темними силами, які хочуть знищити цей світ. На жаль, трапилась халепа, бо деякі слова із книги чар були пошкоджені під час битви. Одне таке слово можна відновити виконавши ритуал зцілення над пошкодженими буквами. Ритуал зцілення можна виконати на всіх **підряд** розташованих **пошкоджених** буквах. Вам не залишається нічого іншого як допомогти Ліні відновити ці слова і сказати скільки мінімально треба провести таких ритуалів, щоб прочитати одне з наймогутніших у цьому світі заклять - Поневолення Дракона!

Вхідні дані

У першому рядку N - кількість рядків у заклятті.
В наступних N рядкях - набір слів $w_1,\ldots,w_M$ розділених пробілами, де кожне слово може містити малі латинські літери та символ #, який позначає пошкоджену букву.

Вихідні дані

Єдине ціле число - мінімальна кількість ритуалів, які потрібно провести, щоб відновити закляття.

2. Дизайн та планувальна оцінка часу виконання завдань:

Програма №1

- Важливі деталі для реалізації програми.

  Використати одновимірний масив цілих чисел для реалізації, використати при цьому генератор цілих чисел, знайти перший елемент із заданим значенням і видалити його шляхом зсуву всіх наступних елементів вліво, потім вивести результат на екран.

  Плановий час на реалізацію 2 години.

Програма №2

- Важливі деталі для реалізації програми.

Використовуючи функції, знайти суму елементів під головною діагоналлю. Масив повинен передаватися у функцію як параметр. Використати двовимірний масив.

- Плановий час на реалізацію 2 години.

Програма №3

- Важливі деталі для реалізації програми.
  Встановити початкові позиції дронів, у циклі перевірити ситуації "Collision", "Stopped" або "Miss" та вкінці оновити поточні позиції дронів, вивести їх на екран.

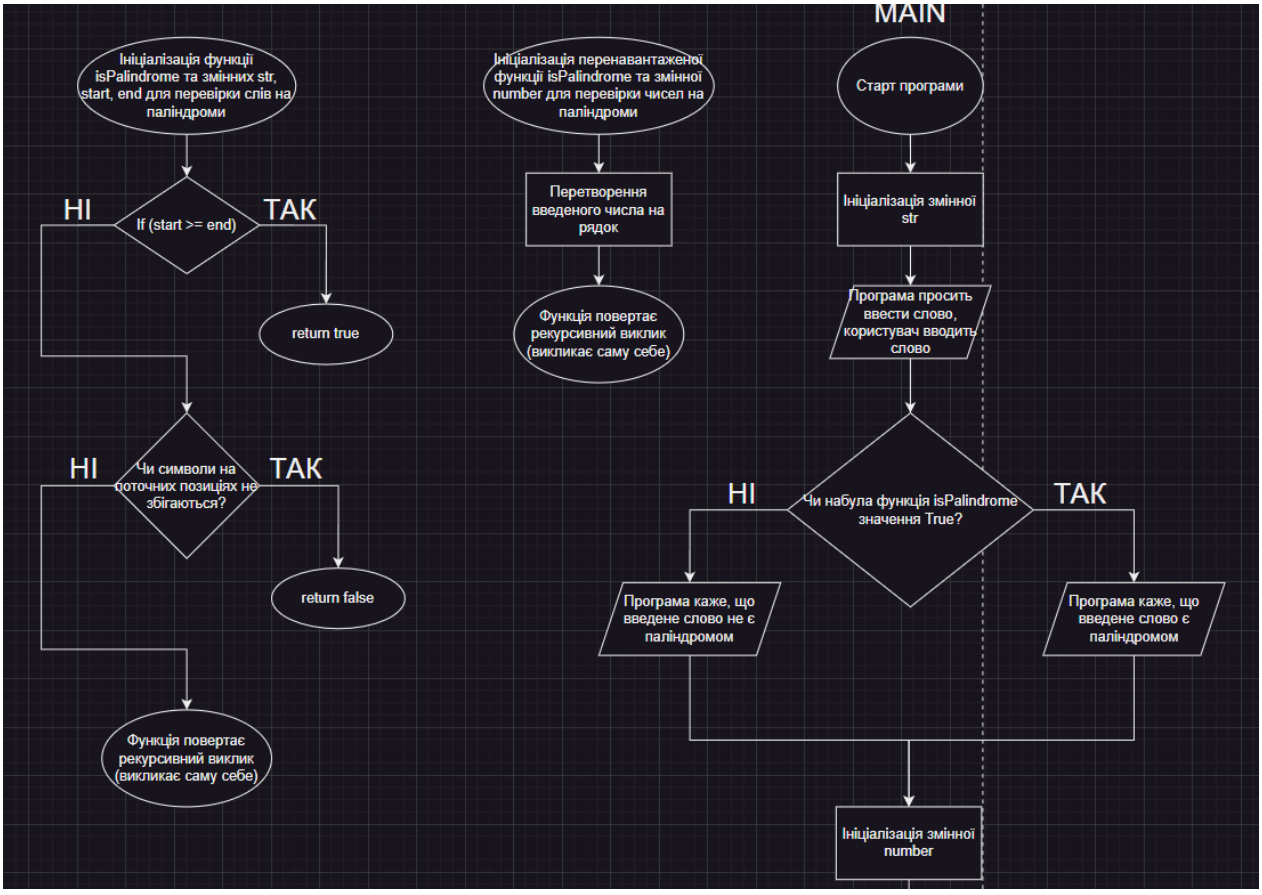
Плановий час на реалізацію 2 години.

Програма №4

- Важливі деталі для реалізації програми.
  Використовувати вектори для зручного написання програми, а також бібліотеку <algorithm>.
- Плановий час на реалізацію 2 години.

Програма №5

- Блок – схема.
- Важливі деталі для реалізації програми.
  Визначити та реалізувати рекурсивну функцію isPalindrome для рядків. Визначити та реалізувати перевантажену функцію isPalindrome для цілих чисел. Використати математичний підхід щоб перевірити чи число є паліндромом.
- Плановий час на реалізацію 3 години.



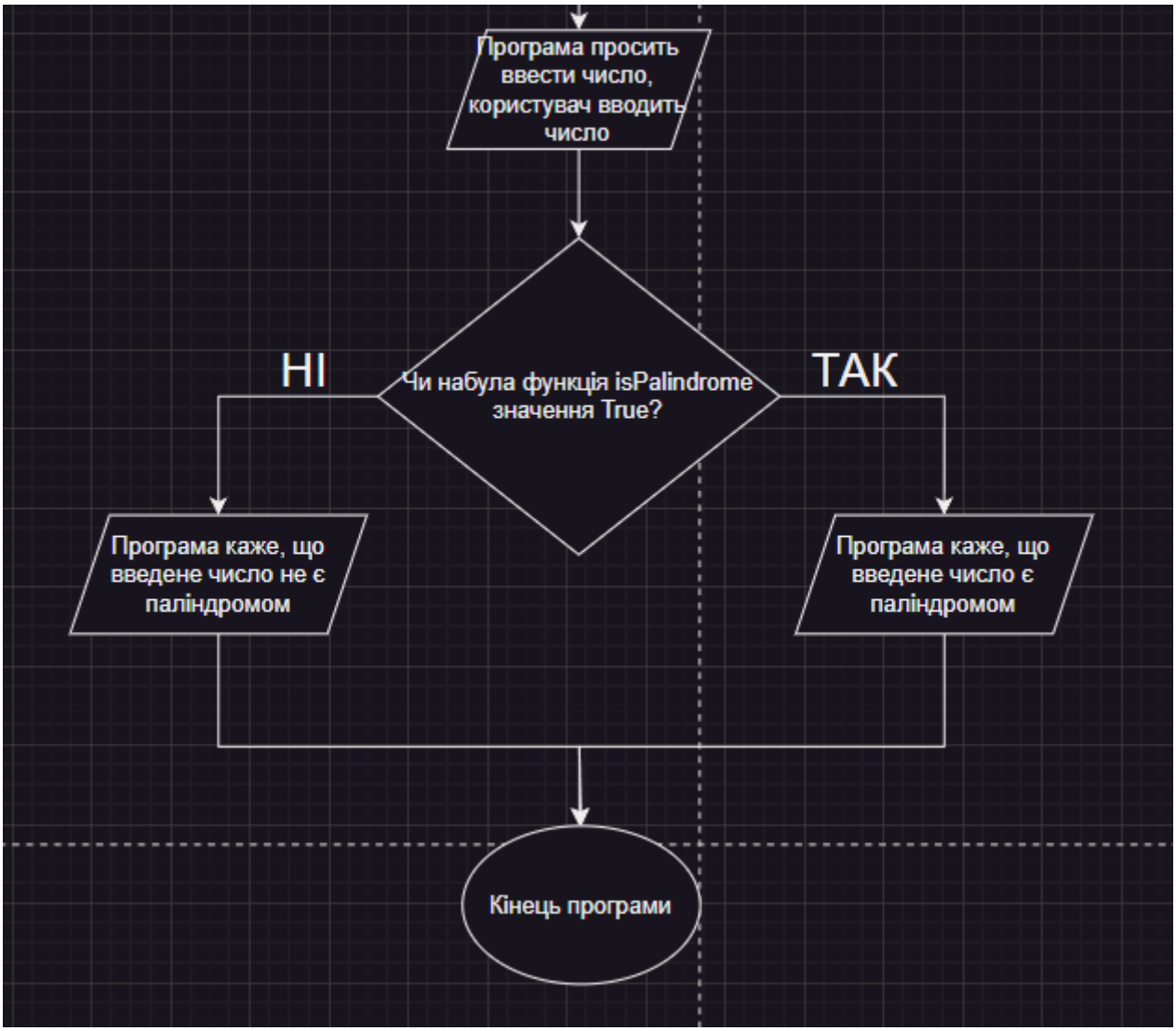


Рисунок 2.1. Блок-схема до завдання №5

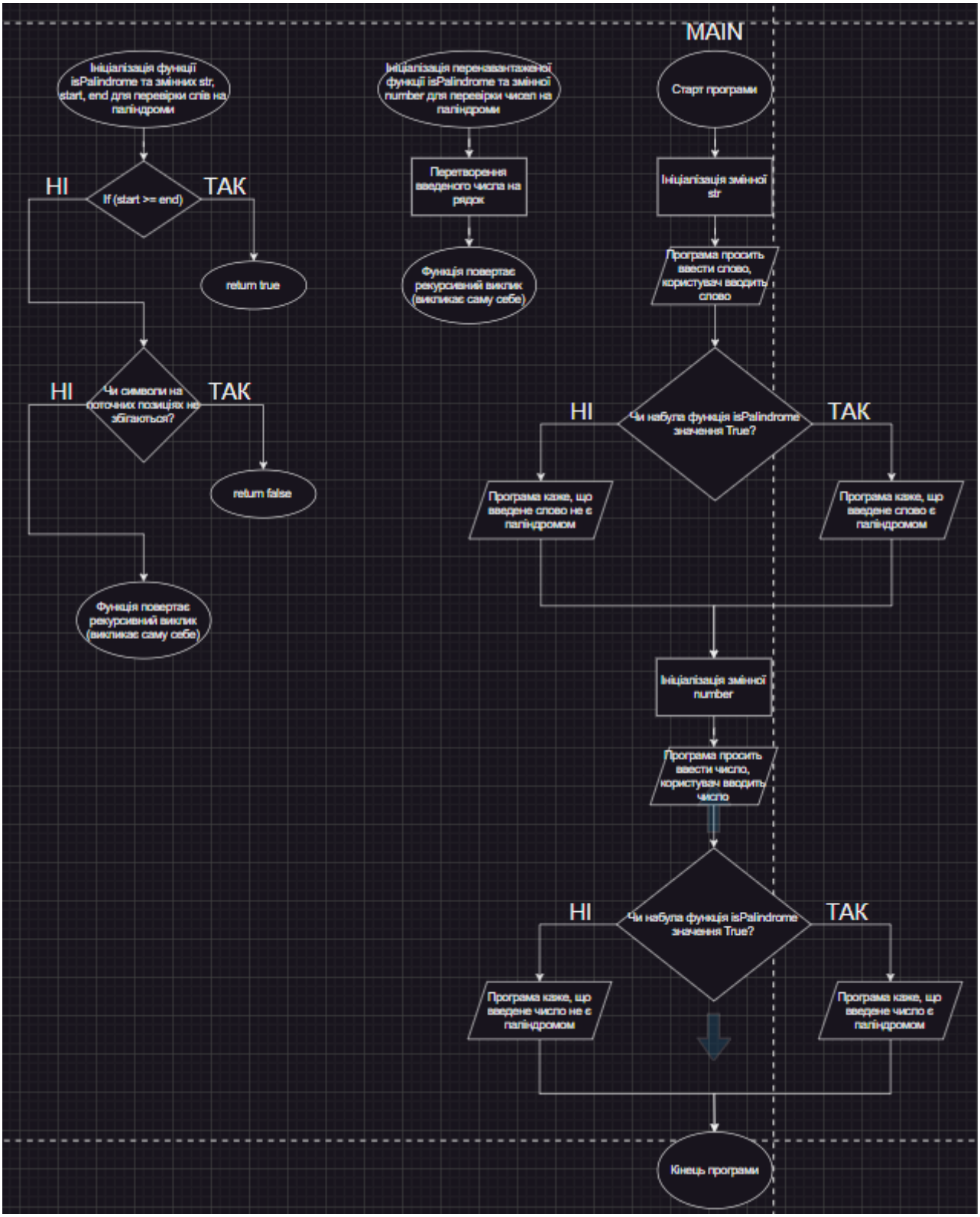


Рисунок 2.2. Загальна блок-схема до завдання №5

Програма №6

- Важливі деталі для реалізації програми.
  Використовуючи бібліотеку <string> та функцію getline перевірити кількість символів “#” в словах кожного рядка, введеного користувачем.
- Плановий час на реалізацію 1 година.

3. Код програм з посиланням на зовнішні ресурси та фактично затрачений час:

Завдання №1

```cpp
#include <iostream>
#include <cstdlib>  // Для функцій rand() та srand()
#include <ctime>    // Для функції time()
using namespace std;

// Функція для зсуву масиву циклічно на K елементів вправо
void shiftArray(int arr[], int n, int k) {
    k = k % n; // Для врахування циклічності, якщо k > n
    int element[n];
    for (int i = 0; i < n; ++i) {
        element[(i + k) % n] = arr[i];
    }
    for (int i = 0; i < n; ++i) {
        arr[i] = element[i];
    }
}

int main() {
    const int N = 100;
    int a[N];
    int realLength;

    // Введення реальної довжини масиву
    cout << "Введіть реальну довжину масиву (не більше " << N << "): ";
    cin >> realLength;
    if (realLength > N) {
        cout << "Довжина масиву перевищує максимальне значення." << endl;
        return 1;
    }

    // Ініціалізація генератора випадкових чисел
    srand(static_cast<unsigned>(time(0)));

    // Формування масиву випадкових чисел
    for (int i = 0; i < realLength; ++i) {
        a[i] = rand() % 100; // Випадкові числа від 0 до 99
    }

    // Вивід початкового масиву
    cout << "Початковий масив:" << endl;
    for (int i = 0; i < realLength; ++i) {
        cout << a[i] << " ";
    }
    cout << endl;

    // Видалення першого елемента із заданим значенням
    int value;
    cout << "Введіть значення, яке потрібно видалити: ";
    cin >> value;
    int index = -1;
    for (int i = 0; i < realLength; ++i) {
        if (a[i] == value) {
            index = i;
            break;
        }
```

```
    }
    if (index != -1) {
        for (int i = index; i < realLength - 1; ++i) {
            a[i] = a[i + 1];
        }
        --realLength;
    } else {
        cout << "Елемент не знайдено." << endl;
    }

    // Зсув масиву циклічно на K елементів вправо
    int k;
    cout << "Введіть кількість елементів для циклічного зсуву вправо: ";
    cin >> k;
    shiftArray(a, realLength, k);

    // Вивід отриманого масиву
    cout << "Отриманий масив після видалення та зсуву:" << endl;
    for (int i = 0; i < realLength; ++i) {
        cout << a[i] << " ";
    }
    cout << endl;

    return 0;
}

```

Рисунок 3.1. Код до програми № 1

```
Введіть реальну довжину масиву (не більше 100): 50
Початковий масив:
19 82 89 95 29 75 80 50 10 32 1 10 79 65 65 80 0 39 5 68 5 96 78 22 80 50 74 27 3 26 13 24 88 70 41 62 62 55 54 3 70 69 61 72 45 3 11 43 79 59
Введіть значення, яке потрібно видалити: 19
Введіть кількість елементів для циклічного зсуву вправо: 10
Отриманий масив після видалення та зсуву:
70 69 61 72 45 3 11 43 79 59 82 89 95 29 75 80 50 10 32 1 10 79 65 65 80 0 39 5 68 5 96 78 22 80 50 74 27 3 26 13 24 88 70 41 62 62 55 54 3
```

Рисунок 3.2. Приклад виконання програми № 1

Спочатку створюємо прототипи функцій, а нижче їх реалізовуємо. За допомогою функції srand будемо заповнювати рандомними числами наш масив. У головній функції (main) будемо їх викликати і робити відповідні перетворення з масивом. Фактично затрачений час 2 години.

Завдання №2

```cpp
#include <iostream>
#include <vector>
using namespace std;

// Функція для обчислення суми елементів нижче головної діагоналі
int suma_pid_diag(const vector<vector<int>>& matrix) {
    int n = matrix.size();
    int suma = 0;
    for (int i = 1; i < n; ++i) {
        for (int j = 0; j < i; ++j) {
            suma += matrix[i][j];
        }
    }
    return suma;
}

int main() {
    int n;
    cout << "Введіть розмір квадратної матриці: ";
    cin >> n;

    int numMatrices;
    cout << "Введіть кількість матриць: ";
    cin >> numMatrices;

    vector<vector<vector<int>>> matrices(numMatrices, vector<vector<int>>(n, vector<int>(n)));

    // Введення матриць
    for (int k = 0; k < numMatrices; ++k) {
        cout << "Введіть елементи матриці " << k + 1 << ":\n";
        for (int i = 0; i < n; ++i) {
            for (int j = 0; j < n; ++j) {
                cin >> matrices[k][i][j];
            }
        }
    }

    // Знаходження максимальної суми
    int maxSuma = INT_MIN;
    for (int k = 0; k < numMatrices; ++k) {
        int currentSuma = suma_pid_diag(matrices[k]);
        if (currentSuma > maxSuma) {
            maxSuma = currentSuma;
        }
    }

    cout << "Максимальна сума елементів, які розташовані нижче головної діагоналі: " << maxSuma << endl;

    return 0;
}
```

Рисунок 3.3. Код до програми № 2

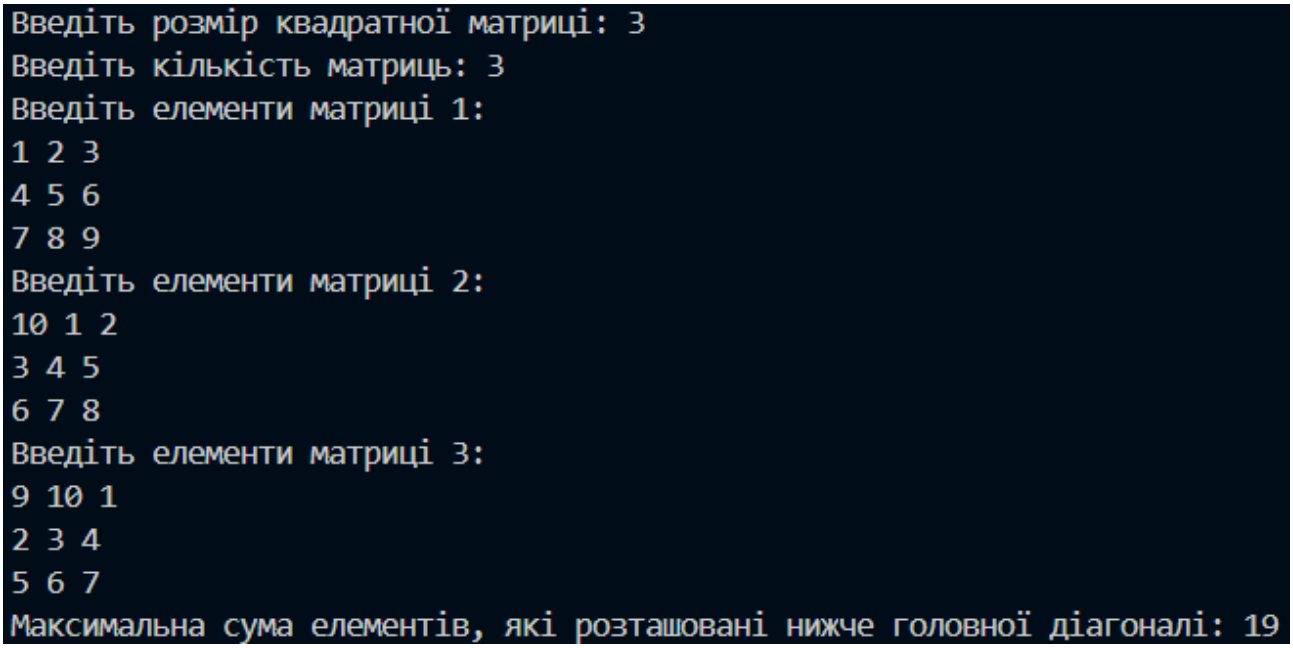


Рисунок 3.4. Приклад виконання програми № 2

Функція sum_pid_diag приймає квадратну матрицю як параметр, обчислює суму елементів під головною діагоналлю та повертає отриманий результат. Після того у головній функції main відбувається зчитування розміру матриць, їх кількості та елементів. Усі ці значення вводить користувач, після чого за допомогою виклику функції sum_pid_diag обчислюється сума елементів під діагоналлю та виводиться максимальне з цих результатів.
Фактично затрачений час 3 години.

Завдання №3

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main() {
    int N;
    cout << "Введіть розмір масиву: ";
    cin >> N;

    vector<int> arr(N);
    cout << "Введіть елементи масиву: ";
    for (int i = 0; i < N; ++i) {
        cin >> arr[i];
    }

    int leftDrone = 0;
    int rightDrone = N - 1;

    while (true) {
        if (leftDrone == rightDrone) {
            cout << leftDrone + 1 << " " << rightDrone + 1 << endl;
            cout << "Collision" << endl;
            break;
        }
        if (leftDrone + 1 == rightDrone) {
            cout << leftDrone + 1 << " " << rightDrone + 1 << endl;
            cout << "Stopped" << endl;
            break;
        }
        if (leftDrone > rightDrone) {
            cout << leftDrone + 1 << " " << rightDrone + 1 << endl;
            cout << "Miss" << endl;
            break;
        }

        leftDrone += arr[leftDrone];
        rightDrone -= arr[rightDrone];
    }

    return 0;
}
```

Рисунок 3.5. Код до програми №3

```
Введіть розмір масиву: 10
Введіть елементи масиву: 1 3 5 2 1 2 1 3 4 1
5 5
Collision
```

Рисунок 3.6. Приклад виконання програми №3

декілька секунд тому	C++ 23	Зараховано	0.003	1.215	Перегляд

Рисунок 3.7. Статус задачі на алготестері

Зчитуємо розмір масиву N та елементи arr, встановлюємо початкові позиції дронів, у циклі перевіряємо ситуації "Collision", "Stopped" або "Miss", оновлюємо позиції дронів, цикл завершується при досягненні однієї з умов.
Фактично затрачений час 30 хвилин.

Завдання №4

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    int N, M;

    // Зчитування першого масиву
    cin >> N;
    vector<int> a(N);
    for (int i = 0; i < N; ++i) {
        cin >> a[i];
    }

    // Зчитування другого масиву
    cin >> M;
    vector<int> b(M);
    for (int i = 0; i < M; ++i) {
        cin >> b[i];
    }

    // Визначення кількості спільних елементів
    int spilne = 0;
    for (int i = 0; i < N; ++i) {
        for (int j = 0; j < M; ++j) {
            if (a[i] == b[j]) {
                ++spilne;
                break;
            }
        }
    }

    // Визначення кількості унікальних елементів
    vector<int> combined = a;
    for (int i = 0; i < M; ++i) {
        combined.push_back(b[i]);
```

```
    }

    // Видалення дублікатів
    sort(combined.begin(), combined.end());
    combined.erase(unique(combined.begin(), combined.end()), combined.end());

    int uniqueElements = combined.size();

    // Вивід результатів
    cout << spilne << endl;
    cout << uniqueElements << endl;

    return 0;
}

```

Рисунок 3.8. Код до програми №4

```
5
1 2 3 4 5
5
5 6 7 8 9
1
9
```

Рисунок 3.9. Приклад виконання програми №4

декілька секунд тому	C++ 23	Зараховано	0.003	1.207	Перегляд

Рисунок 3.10. Статус задачі на алготестері

Зчитуємо розмір і елементи двох масивів a і b, визначаємо кількість спільних елементів за допомогою вкладеного циклу, створюємо об'єднаний масив combined з елементами обох масивів, сортуємо його і видаляємо дублікати, виводимо кількість спільних та унікальних елементів.
Фактично затрачений час 2 години.

Завдання №5

```cpp
#include <iostream>
#include <string>
using namespace std;

bool isPalindrome(const string& str, int start, int end) {
    // Базовий випадок
    if (start >= end) {
        return true;
    }
    // Якщо символи на поточних позиціях не збігаються
    if (str[start] != str[end]) {
        return false;
    }

    return isPalindrome(str, start + 1, end - 1);
}

bool isPalindrome(int number) {
    // Перетворення числа на рядок
    string str = to_string(number);
    // Виклик рекурсивної функції для рядків
    return isPalindrome(str, 0, str.length() - 1);
}

int main() {
    // Перевірка рядка
    string str;
    cout << "Введіть слово: ";
    cin >> str;
    if (isPalindrome(str, 0, str.length() - 1)) {
        cout << "Введене слово є паліндромом!" << endl;
    } else {
        cout << "Введене слово не є паліндромом." << endl;
    }

    // Перевірка числа
    int number;
    cout << "Введіть число: ";
    cin >> number;
    if (isPalindrome(number)) {
        cout << "Введене число є паліндромом!" << endl;
    } else {
        cout << "Введене число не є паліндромом." << endl;
    }

    return 0;
}
```

Рисунок 3.11. Код до програми №5

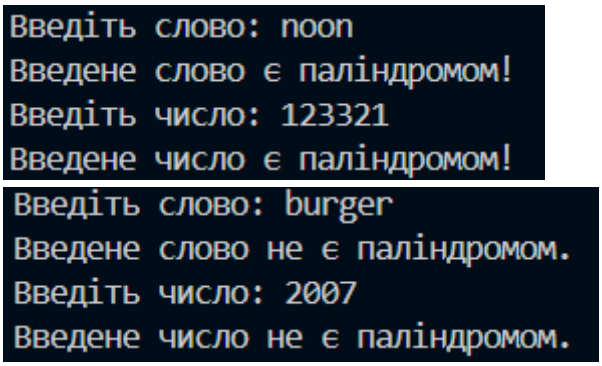


Рисунок 3.12. Приклади виконання програми №5

Створюємо дві перевантажені функції isPalindrom для перевірки числа і слова на паліндром. За допомогою циклу while будемо рухатися з права і зліва до середини й перевіряти чи рівні елементи між собою. У головній функції використовуємо перевірку на те, що користувач буде вводити чи слово, чи число і відповідно будемо викликати наші функції.
Фактично затрачений час 2.5 години.

Завдання №6

```cpp
#include <iostream>
#include <string>

using namespace std;

int main() {
    int N;
    cin >> N;
    cin.ignore(); // Ігноруємо символ нового рядка після N

    int ritualCount = 0;
    bool damagedSymbols = false;

    for (int i = 0; i < N; ++i) {
        string line;
        getline(cin, line);

        for (char c : line) {
            if (c == '#') {
                if (!damagedSymbols) {
                    damagedSymbols = true;
                    ++ritualCount;
                }
            } else {
                damagedSymbols = false;
            }
        }
    }

    cout << ritualCount << endl;
    return 0;
}

```

Рисунок 3.13. Код до програми №6

```
5
Ost#p Tsi#pa
Bohda# Snig#r
B#hdan V#lyaniuk
Arsen#y Kychka
Yulia Danyl#uk
8
```

Рисунок 3.14. Приклад виконання програми №6

декілька секунд тому	C++ 23	Зараховано	0.016	1.336	Перегляд

Рисунок 3.15. Статус задачі на алготестері

Зчитуємо кількість рядків N, ініціалізуємо змінні ritualCount і damagedSymbols, зчитуємо кожний рядок за допомогою getline, перевіряємо кожний символ у рядку, збільшуємо ritualCount і встановлюємо damagedSymbols в true для символів “#”, встановлюємо damagedSymbols в false для інших символів, виводимо кількість необхідних ритуалів.
Фактично затрачений час 1.5 години.

Посилання на пул реквест: [Epic 4 - Ostap Tsiapa by Ostap2007ter · Pull Request #379 · artificial-intelligence-department/ai_programming_playground_2024]

4. Робота з командою:

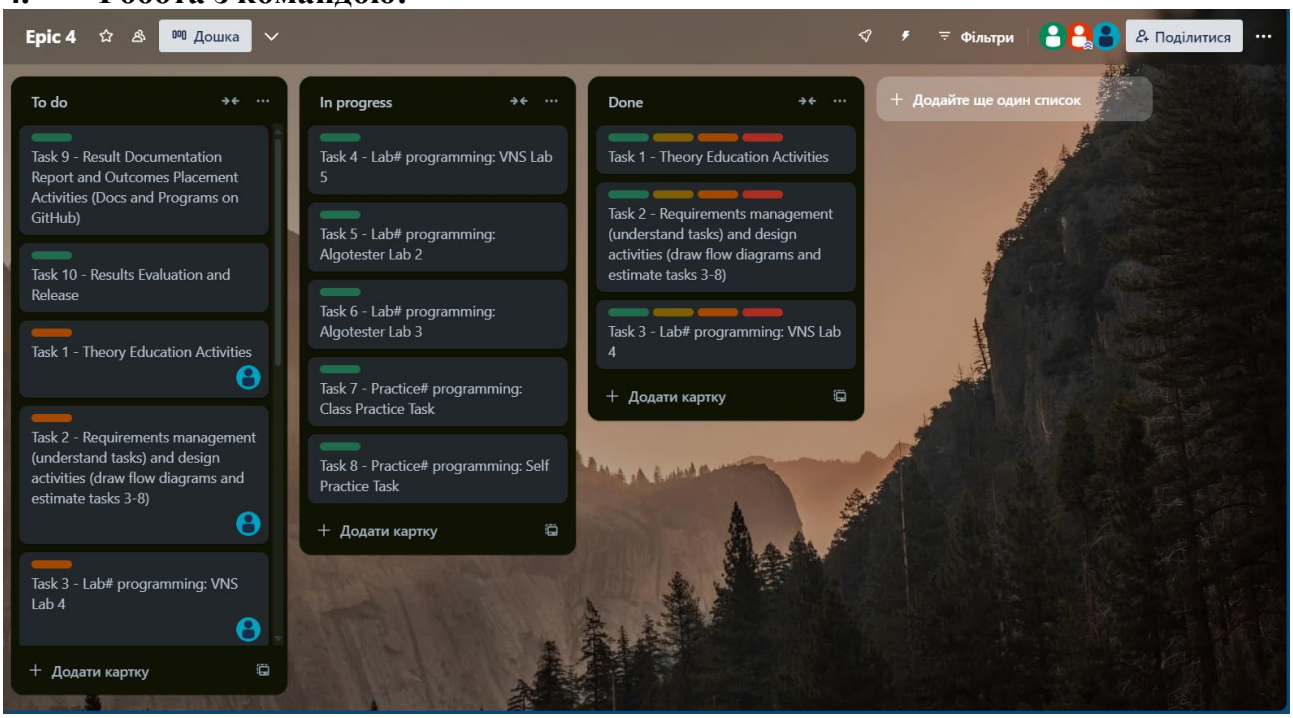


Рисунок 4.1. Командна дошка в Trello

Висновок: У межах практичних та лабораторних робіт блоку №4 я засвоїв багато нового матеріалу, включаючи різні типи масивів (одновимірні та двовимірні), вказівники та посилання, динамічні масиви та структури даних. Практичне застосування цих знань допомогло мені краще зрозуміти, як вони працюють та як їх реалізовувати. Крім того, я створив блок-схему для найскладнішого завдання, що дозволило глибше усвідомити роботу програми. Також організував роботу команди, створивши дошку в Trello.