## Міністерство освіти і науки України Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



Звіт

## про виконання

**Лабораторних та практичних робіт №6**

***з дисципліни:*** «Мови та парадигми програмування»

## *з розділу*: «Епік 6: Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур»

***Виконав:***

студент групи ШІ-13

Гаргай Юрій

Львів 2023

# Тема роботи:

Робота із простими структурами даних та алгоритмами обробки. Робота з одновимірними та двовимірними масивами.

**Задачі:**

1. *VNS Lab 10*
2. *Algotester Lab 5*
3. *Algotester Lab 78*
4. *Practice task3*
5. *Practice task4*
6. *Practice task5*

# Мета роботи:

Організація динамічних структур: черги, стеку, списку і дерева. Створення прикладних алгоритмів для їх обробки.

# Теоретичні відомості:

* 1. Теоретичні відомості з переліком важливих тем:
* Тема №1: Динамічний масив
* Тема №2: Однозв’язний та двозв’язний списки.
* Тема №3: Бінарне дерево
  1. Індивідуальний план опрацювання теорії:
* Тема №1: Динамічний масив
* Джерела Інформації:
  + - https://www.geeksforgeeks.org/how-do-dynamic-arrays-work/
    - <https://ua.udemy.com/>

Beginning C++ Programming - From Beginner to Beyond

* Що опрацьовано:
* Ознайомився з динамічним масивом
  + Статус: Ознайомлений
  + Початок опрацювання теми: 12.11.2023
  + Звершення опрацювання теми: 15.11.2023
* Тема №2: Однозв’язний та двозв’язний списки.
* Джерела Інформації:
  + - https://www.geeksforgeeks.org/data-structures/linked-list/doubly-linked-list/
    - <https://ua.udemy.com/>

Beginning C++ Programming - From Beginner to Beyond

* + Що опрацьовано**:**
    - Ознайомився з однозв’язними та двозв’язними списками
  + Статус: Ознайомлений
  + Початок опрацювання теми:06.12.2023
  + Звершення опрацювання теми: 08.12.2023
* Тема №3: Бінарне дерево
  + Джерела Інформації:
    - <https://ua.udemy.com/>

Beginning C++ Programming - From Beginner to Beyond

* + Що опрацьовано:
    - Ознайомився з бінарним деревом
  + Статус: Ознайомлений
  + Початок опрацювання теми: 08.12.2023
  + Звершення опрацювання теми: 10.12.2023

# Виконання роботи:

**Завдання №1**: VNS Lab 10

Деталі завдання

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.

Варіант завдання: **13**

Записи в лінійному списку містять ключове поле типу \*char (рядок

символів). Сформувати двонаправлений список. Знищити з нього К перших

елементів. Додати елемент після елемента, що починається із зазначеного

символу.

**Планована оцінка часу виконання завдань:** 1 день

**Завдання №2** Algotester Lab 5

* Варіант завдання: **2**

В пустелi iснує незвичайна печера, яка є двохвимiрною. Її висота це N, ширина - M.

Всерединi печери є пустота, пiсок та камiння. Пустота позначається буквою , пiсок S i камiння X;

Одного дня стався землетрус i весь пiсок посипався вниз. Вiн падає на найнижчу клiтинку з пустотою, але вiн не може пролетiти через камiння.

Ваше завдання сказати як буде виглядати печера пiсля землетрусу.

**Планована оцінка часу виконання завдань:** 1 день

**Завдання №3**: Algotester Lab 78

Варіант завдання: **3**

**Деталі завдання**

Ваше завдання - власноруч реалiзувати структуру даних "Двiйкове дерево пошуку".

Ви отримаєте Q запитiв, кожен запит буде починатися зi слова-iдентифiкатора, пiсля якого

йдуть його параметри.

Вам будуть поступати запити такого типу:

• Вставка:

Iдентифiкатор - insert

Ви отримуєте цiле число value - число, яке треба вставити в дерево.

• Пошук:

Iдентифiкатор - contains

Ви отримуєте цiле число value - число, наявнiсть якого у деревi необхiдно перевiрити.

Якщо value наявне в деревi - ви виводите Y es, у iншому випадку No.

• Визначення розмiру:

Iдентифiкатор - size

Ви не отримуєте аргументiв.

Ви виводите кiлькiсть елементiв у деревi.

• Вивiд дерева на екран

Iдентифiкатор - print

Ви не отримуєте аргументiв.

Ви виводите усi елементи дерева через пробiл.

Реалiзувати використовуючи перегрузку оператора <<

**Планована оцінка часу виконання завдань:** 1 день

**Завдання №4** Practice task3

* Деталі завдання:
* 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.

**Планована оцінка часу виконання завдань:** 1 день

**Завдання №5** Practice task4

* Деталі завдання:
* 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

**Планована оцінка часу виконання завдань:** 1 день

**Завдання №6** Practice task5

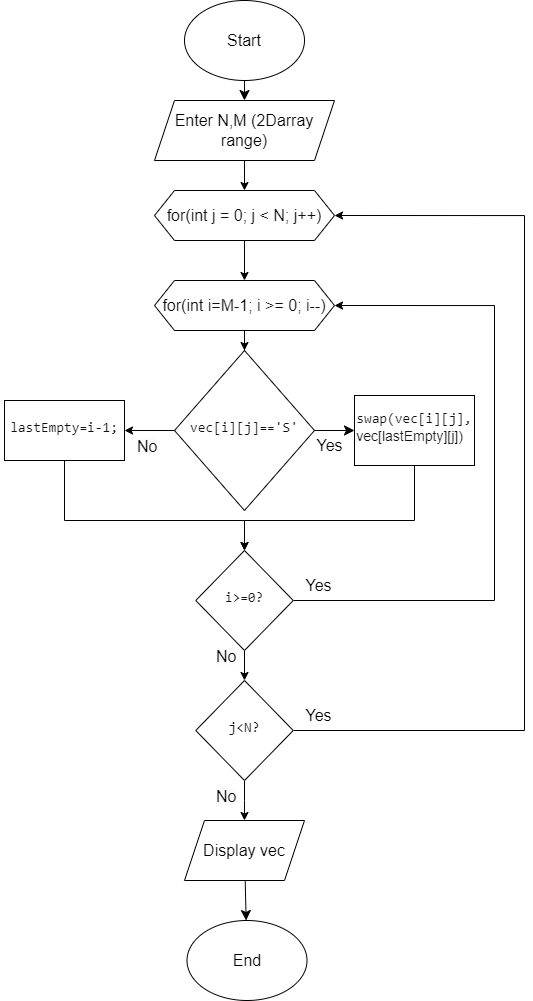
* Деталі завдання:
* використовувати цілочисельні значення у вузлах дерева;
* реалізувати функцію, яка ітеративно проходить по бінарному дереві і записує у батьківський вузол суму значень підвузлів
* вузол-листок не змінює значення
* значення змінюються від листків до кореня дерева

**Планована оцінка часу виконання завдань:** 1 день

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

Завдання №2Algotester Lab 5

* + Блок-схема



ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss

Рисунок 1 блок схема

Планований час на реалізацію: 1 день sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss

### Код програм з результатом та посиланням на зовнішні ресурси:

Завдання №1 VNS Lab 10

1)Код програми

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

struct Node {

    string key;

    Node\* next;

    Node\* prev;

};

Node\* createNode(string key) {

    Node\* newNode = new Node();

    if (!newNode) {

        cout << "Memory Error\n";

        return NULL;

    }

    newNode->key = key;

    newNode->next = NULL;

    newNode->prev = NULL;

    return newNode;

}

Node\* insertNode(Node\* head, string key) {

    if (head == NULL) {

        head = createNode(key);

    } else {

        Node\* temp = head;

        while (temp->next != NULL) {

            temp = temp->next;

        }

        Node\* newNode = createNode(key);

        temp->next = newNode;

        newNode->prev = temp;

    }

    return head;

}

Node\* deleteFirstKNodes(Node\* head, int K) {

    Node\* temp = head;

    Node\* nextNode;

    for (int i = 0; i < K && temp != NULL; i++) {

        nextNode = temp->next;

        delete temp;

        temp = nextNode;

    }

    if (temp != NULL) {

        temp->prev = NULL;

    }

    return temp;

}

Node\* insertAfterKey(Node\* head, char symbol, string newKey) {

    Node\* temp = head;

    while (temp != NULL && temp->key[0] != symbol) {

        temp = temp->next;

    }

    if (temp != NULL) {

        Node\* newNode = createNode(newKey);

        newNode->next = temp->next;

        newNode->prev = temp;

        if (temp->next != NULL) {

            temp->next->prev = newNode;

        }

        temp->next = newNode;

    }

    return head;

}

Node\* readListFromFile(const string& filename) {

    ifstream file(filename);

    if (!file) {

        cout << "Can't open the file\n";

        return NULL;

    }

    Node\* head = NULL;

    string key;

    while (getline(file, key)) {

        head = insertNode(head, key);

    }

    file.close();

    return head;

}

void printList(Node\* head);

void deleteList(Node\* &head);

void writeListToFile(Node\* head, const string& filename);

int main() {

    Node\* head = NULL;

    string key;

    char symbol;

    int K;

    // Введення елементів

    cout << "Enter elements, enter 'stop' when finish:\n";

    while (true) {

        cin >> key;

        if (key == "stop") break;

        head = insertNode(head, key);

    }

    printList(head);

    // Введення K

    cout << "Enter K:\n";

    cin >> K;

    head = deleteFirstKNodes(head, K);

    printList(head);

    // Введення символу і нового ключа

    cout << "Enter symbol:\n";

    cin >> symbol;

    cout << "Enter new key:\n";

    cin >> key;

    head = insertAfterKey(head, symbol, key);

    writeListToFile(head, "listss.txt");

    deleteList(head);

    head = readListFromFile("listss.txt");

    printList(head);

    deleteList(head);

    return 0;

}

void printList(Node\* head) {

    if (head == NULL) {

        cout << "List is empty\n";

        return;

    }

    Node\* temp = head;

    while (temp != NULL) {

        cout << temp->key << " -> ";

        temp = temp->next;

    }

    cout << "NULL\n";

}

void writeListToFile(Node\* head, const string& filename) {

    ofstream file(filename);

    if (!file) {

        cout << "Can't open the file\n";

        return;

    }

    Node\* temp = head;

    while (temp != NULL) {

        file << temp->key << "\n";

        temp = temp->next;

    }

    file.close();

}

void deleteList(Node\* &head) {

    Node\* nextNode;

    while (head != NULL) {

        nextNode = head->next;

        delete head;

        head = nextNode;

    }

}

Код програми

2) Результат

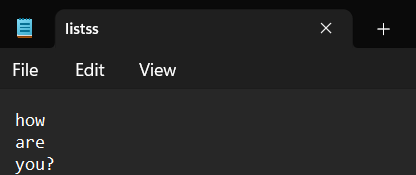


Рисунок 2 результат

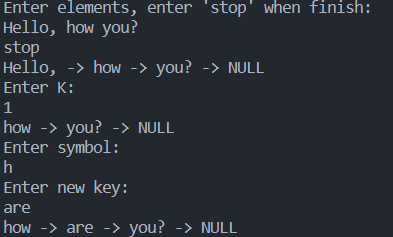


Рисунок 3 результат

3) Посилання на файл програми у пул-запиті GitHub

[Pull-request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/903/files#diff-ee4a8a6eb08f89423496bc29e6ae9a1b9db74eeee1157204f3c8c4b68d97cd41)

ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss

Завдання №2 Algotester Lab 5

1)Код програми

#include <iostream>

#include <vector>

using namespace std;

int main() {

    int M, N;

    cin >> M;

    cin >> N;

    vector<vector<char>> vec(M, vector<char>(N));

    for (int i = 0; i < M; i++) {

        for (int j = 0; j < N; j++) {

            cin >> vec[i][j];

        }

    }

    // Моделювання землетрусу

    for (int j = 0; j < N; j++) {

        int lastEmpty = M - 1;

        for (int i = M - 1; i >= 0; i--) {

            if (vec[i][j] == 'S') {

                swap(vec[i][j], vec[lastEmpty][j]);

                lastEmpty--;

            }

            else if (vec[i][j] == 'X') {

                lastEmpty = i - 1;

            }

        }

    }

    for (int i = 0; i < M; i++) {

        for (int j = 0; j < N; j++) {

            cout << vec[i][j];

        }

        cout << endl;

    }

    return 0;

}

Код програми

2) Результат



Рисунок 4 результат в алготестері

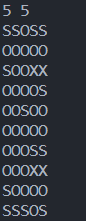


Рисунок 5 результат

3) Посилання на файл програми у пул-запиті GitH ub

[Pull-Request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/903/files#diff-f5caeb3caf357acb8b58273085fe46f089bd54e322d43b7b41e1372ba9260f78)

Завдання №3 Algotester Lab 78

1)Код програми

#include <iostream>

#include <string>

using namespace std;

struct Node {

    int value;

    Node\* left;

    Node\* right;

};

class BinaryTree {

    Node\* root;

    int size;

public:

    BinaryTree() : root(nullptr), size(0) {}

    void insert(int value) {

        root = insert(root, value);

    }

    bool contains(int value) {

        return contains(root, value);

    }

    void print() {

        print(root);

        cout << endl;

    }

    int getSize() {

        return size;

    }

private:

    Node\* insert(Node\* node, int value) {

        if (node == nullptr) {

            node = new Node{value, nullptr, nullptr};

            size++;

        } else if (value < node->value) {

            node->left = insert(node->left, value);

        } else if (value > node->value) {

            node->right = insert(node->right, value);

        }

        return node;

    }

    bool contains(Node\* node, int value) {

        if (node == nullptr) {

            return false;

        } else if (value < node->value) {

            return contains(node->left, value);

        } else if (value > node->value) {

            return contains(node->right, value);

        } else {

            return true;

        }

    }

    void print(Node\* node) {

        if (node != nullptr) {

            print(node->left);

            cout << node->value << ' ';

            print(node->right);

        }

    }

};

int main() {

    BinaryTree tree;

    int Q;

    cin >> Q;

    for (int i = 0; i < Q; i++) {

        string command;

        cin >> command;

        if (command == "insert") {

            int value;

            cin >> value;

            if (!tree.contains(value)) {

                tree.insert(value);

            }

        } else if (command == "contains") {

            int value;

            cin >> value;

            cout << (tree.contains(value) ? "Yes" : "No") << endl;

        } else if (command == "size") {

            cout << tree.getSize() << endl;

        } else if (command == "print") {

            tree.print();

        }

    }

    return 0;

}

Код програми

2) Результат



Рисунок 6 результат в алготестері

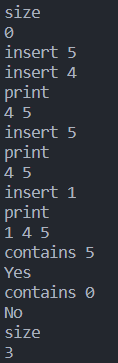


Рисунок 7 результат

3) Посилання на файл програми у пул-запиті GitHub

[Pull-Request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/903/files#diff-3818d8323821523c1fa0aa6a221827edb08f27d44a26a19d96ad0554d18b97ab)

Завдання №4 Practice task3

1)Код програми

#include <iostream>

using namespace std;

struct Node {

    int data;

    Node\* next;

};

Node\* newNode(int data);

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

void printList(Node \*node);

void deleteList(Node\* &head);

int main() {

    Node\* n1 = newNode(0);

    n1->next = newNode(4);

    n1->next->next = newNode(8);

    n1->next->next->next = newNode(2);

    n1->next->next->next->next = newNode(6);

    cout << "First List is ";

    printList(n1);

    Node\* n2 = newNode(5);

    n2->next = newNode(9);

    cout << "Second List is ";

    printList(n2);

    Node\* final = add(n1, n2);

    cout << "Final list is ";

    printList(final);

    deleteList(n1);

    deleteList(n2);

    deleteList(final);

   return 0;

}

Node\* newNode(int data) {

    Node\* new\_node = new Node;

    new\_node->data = data;

    new\_node->next = NULL;

    return new\_node;

}

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

    Node\* result = NULL;

    Node\* temp, \*prev = NULL;

    int carry = 0, sum;

    while (n1 != NULL || n2 != NULL) {

        sum = carry + (n1? n1->data: 0) + (n2? n2->data: 0);

        carry = (sum >= 10)? 1 : 0;

        sum = sum % 10;

        temp = newNode(sum);

        if(result == NULL) {

            result = temp;

        } else {

            prev->next = temp;

        }

        prev = temp;

        if(n1) n1 = n1->next;

        if(n2) n2 = n2->next;

    }

    if(carry > 0) {

        temp = newNode(carry);

        prev->next = temp;

    }

    return result;

}

void printList(Node \*node) {

    while(node != NULL) {

        cout << node->data << " ";

        node = node->next;

    }

    cout << endl;

}

void deleteList(Node\* &head) {

    Node\* current = head;

    Node\* next = NULL;

    while (current != NULL) {

        next = current->next;

        delete current;

        current = next;

    }

    head = NULL;

}

Код програми

2) Результат



Рисунок 8 результат

3) Посилання на файл програми у пул-запиті GitHub

[Pull-Request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/903/files#diff-6eae2403427d152169aff2dff7fa8a3c03afeb42ac4e996d97607d51af5d773e)

Завдання № 5 Practice task4

1)Код програми

#include <iostream>

using namespace std;

struct Node {

    int data;

    Node\* left;

    Node\* right;

};

Node\* Tree(int data) {

    Node\* node = new Node();

    node->data = data;

    node->left = NULL;

    node->right = NULL;

    return(node);

}

void printInorder(Node\* node);

Node\* swapLeftRight(Node\* root) {

    if (root == NULL)

        return root;

    Node\* temp = root->left;

    root->left = root->right;

    root->right = temp;

    if (root->left)

        swapLeftRight(root->left);

    if (root->right)

        swapLeftRight(root->right);

    return root;

}

int main() {

    Node \*root = Tree(1);

    root->left = Tree(2);

    root->right = Tree(3);

    root->left->left = Tree(4);

    root->left->right = Tree(5);

    cout << "Inorder print of tree: ";

    printInorder(root);

    root = swapLeftRight(root);

    cout << "\nInorder print of mirrored tree: ";

    printInorder(root);

    return 0;

}

void printInorder(Node\* node) {

    if (node == NULL)

        return;

    printInorder(node->left);

    cout << node->data << " ";

    printInorder(node->right);

}

Код програми

2) Результат



Рисунок 9 результат

3) Посилання на файл програми у пул-запиті GitHub

[Pull-Request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/903/files#diff-2088c8c6ee5b09e4157d87df58eaaf20e2cbc64eaf3e8e8e64f05305c42e5a83)

Завдання № 6 Practice task5

1)Код програми

#include <iostream>

using namespace std;

struct Node {

    int data;

    Node\* left;

    Node\* right;

};

Node\* Tree(int data) {

    Node\* node = new Node();

    if (node == NULL) {

        cout << "Memory allocation failed" << endl;

        return NULL;

    }

    node->data = data;

    node->left = node->right = NULL;

    return node;

}

int updateTree(Node\* node) {

    if (node == NULL)

        return 0;

    if (node->left == NULL && node->right == NULL)

        return node->data;

    int old\_val = node->data;

    int left\_sum  = updateTree(node->left);

    int right\_sum = updateTree(node->right);

    node->data = left\_sum + right\_sum;

    return node->data + old\_val;

}

int main() {

    Node\* root = Tree(0);

    root->left = Tree(-12);

    root->right = Tree(13);

    root->left->left = Tree(4);

    root->left->right = Tree(0);

    root->right->left = Tree(1);

    root->right->right = Tree(-2);

    updateTree(root);

    cout << "Updated tree root value is: " << root->data << endl;

    return 0;

}

Код програми

2) Результати



Рисунок 10 результат

3) Посилання на файл програми у пул-запиті GitHub

[Pull-Request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/903/files#diff-5017cc1f5726ded57fb795a3735087556fa5fc0a10510ea23e401a6056781984)

***Міт з командою***

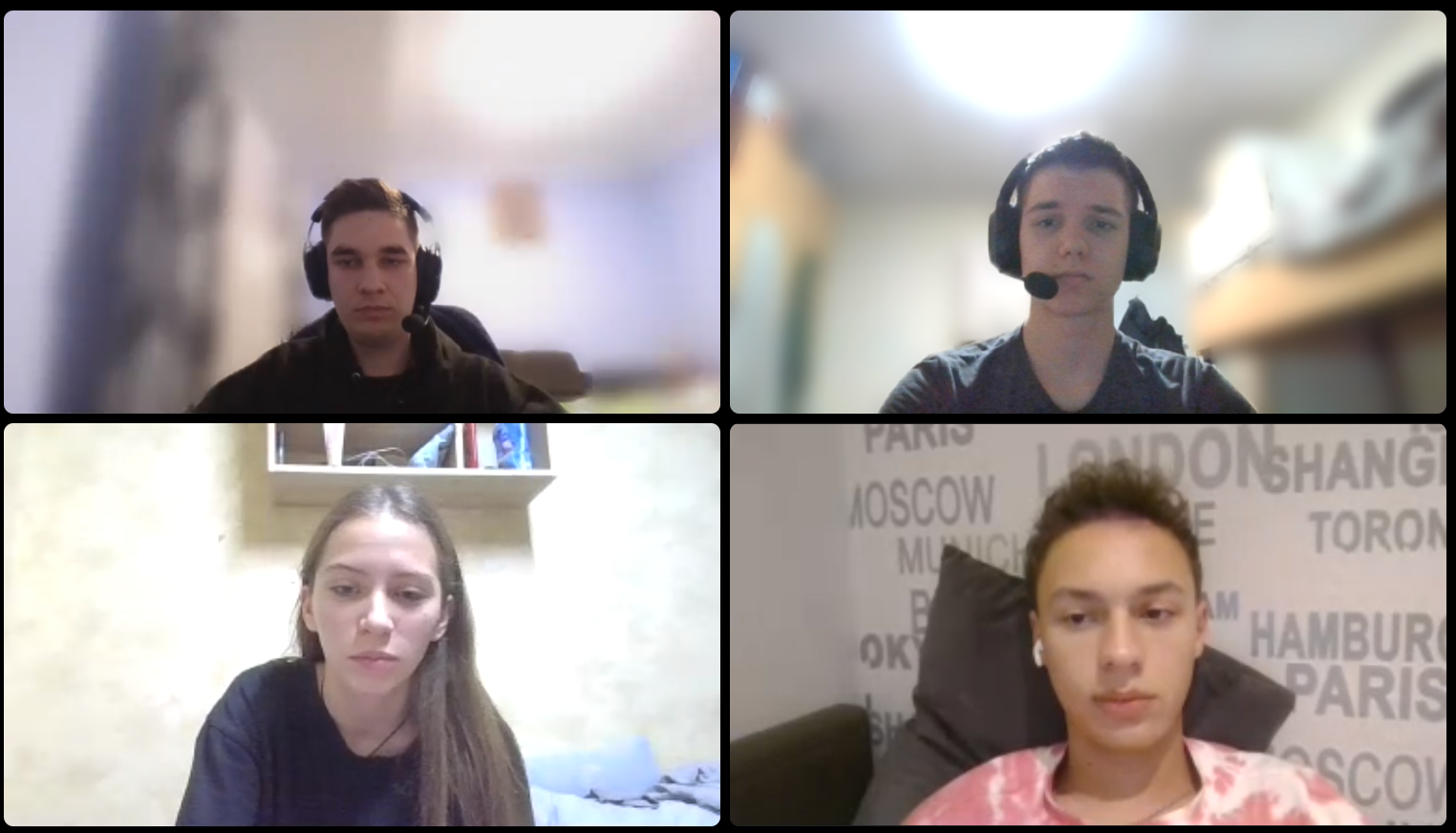
******

Рисунок 11 Міт з командою в Discord

# Висновок:

У процесі цього епіку я успішно оволодів динамічними структурами (Черга, Стек, Списки, Дерево). Виконуючи різноманітні завдання, такі як Algotester Lab 78 мені вдалося застосувати отримані знання на практиці. Це мій ще один крок у свiт програмування і впевнений, що ці набутi навички будуть корисними для подальшого навчання та програмування.