Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Лабораторних та практичних робіт № 6, 8, 9**

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

***з розділу***: «Файли. Системи числення. Бінарні файли. Символи і рядкові змінні та текстові файли. Стандартна бібліотека та деталі/методи роботи з файлами. Створення та використання бібліотек»

***Виконав:***

студент групи ШІ-12

Олександр Олександрович Воронін

# **Тема роботи:**

Ознайомлення та робота з символьними та рядковими змінними в мові програмування C++. Принцип роботи з бінарними та текстовими файлами.

# **Мета роботи:**

Навчитися працювати з символьними та рядковими змінними в мові C++. Застосувати набуті знання на практиці.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Текстові та бінарні файли.
* Тема №2: Основні методи роботи з файлами в C++.
* Тема №3: Стандартна бібліотека в C++.

1. Індивідуальний план опрацювання теорії:

* Тема №1: Текстові та бінарні файли.
  + Джерела Інформації
    - Стаття: <https://www.easeus.com/knowledge-center/what-is-the-difference-between-binary-and-text-files.html>
  + Статус: Ознайомлений
  + Початок опрацювання теми: 9.12.2023
  + Звершення опрацювання теми: 11.12.2023
* Тема №2: Основні методи роботи з файлами в C++.
  + Джерела Інформації:
    - Стаття: <https://www.geeksforgeeks.org/file-handling-c-classes/>
    - <https://www.youtube.com/watch?v=SSNJ7alki-E&t=3617s&ab_channel=%D0%91%D0%BB%D0%BE%D0%B3%D0%B0%D0%BD%E2%A6%81%D0%A3%D1%80%D0%BE%D0%BA%D0%B8%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F>
    - <https://www.w3schools.com/cpp/cpp_files.asp>
  + Статус: Ознайомлений
  + Початок опрацювання теми: 9.12.2023
  + Звершення опрацювання теми: 11.12.2023
* Тема №3: Стандартна бібліотека C++.
  + Джерела Інформації:
    - Відео: <https://www.youtube.com/watch?v=m-WJikuZGuU&ab_channel=%D0%91%D0%BB%D0%BE%D0%B3%D0%B0%D0%BD%E2%A6%81%D0%A3%D1%80%D0%BE%D0%BA%D0%B8%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F>

* + Статус: Ознайомлений
  + Початок опрацювання теми: 10.12.2023
  + Звершення опрацювання теми: 11.12.2023

# **Виконання роботи:**

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

Завдання №1 VNS Lab 6

Варіант 24

Завдання №2 VNS Lab 8

Варіант 24

Завдання №3 VNS Lab 9

Варіант 24

Завдання №4 Algotester Lab 4

Варіант 2

Завдання №5 Algotester Lab 6

Варіант 2

Завдання №6 Class Practice Work

Запис текстової стрічки у файл із заданим ім’ям

Завдання №7 Self Practice Work

Варіант 1

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

Програма №1 VNS Lab 6

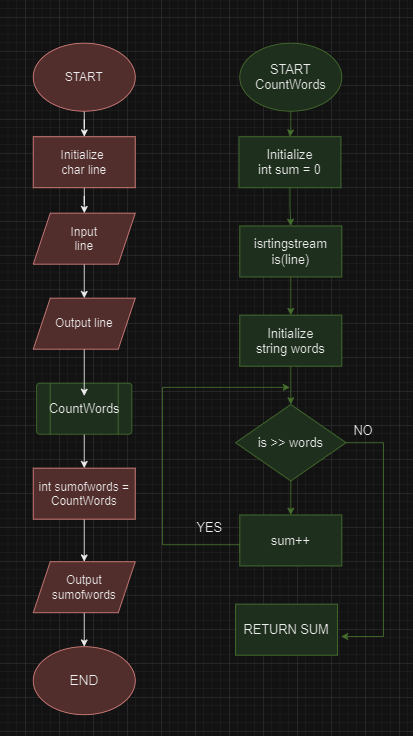
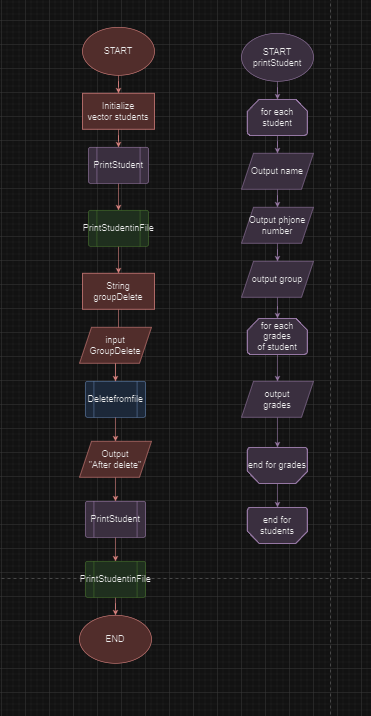
* Блок-схема
* 

Рисунок 1.Блок-Схема VNS LAB 6

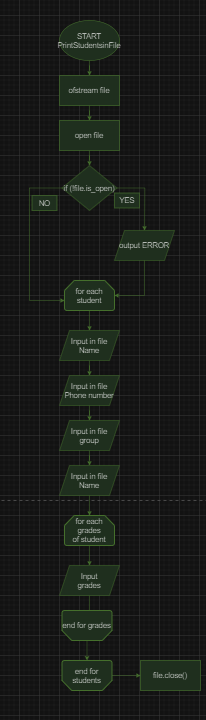
* Планований час на реалізацію: 1 година

Програма №2 VNS Lab 8

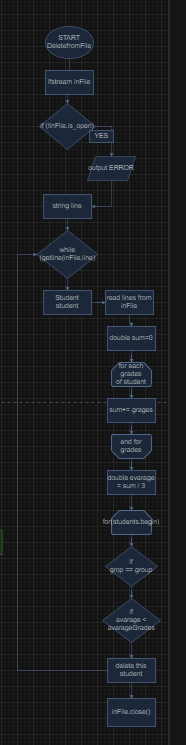
* Блок-схема



Блок-схема VNS lab8 1



Блок-схема VNS lab8 2



Блок-схема VNS lab8 3

* Планований час на реалізацію: 2 година

Програма №3 VNS Lab 9

* Планований час на реалізацію: 2 година

Програма №4 Algotester Lab 4

* Планований час на реалізацію: 1 година

Програма №5 Algotester Lab 6

* Планований час на реалізацію: 3 година

Програма №6 Class Practice Work

* Планований час на реалізацію: 40 хвилин

Програма №7 Self Practice Work

* Планований час на реалізацію: 1 година

## **3. Конфігурація середовища до виконання завдань:**

не потрібна

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

Завдання №1 VNS Lab 6

#include <iostream>

#include <sstream>

#include <string>

using namespace std;

int CountWords(char\* line){

int sum = 0;

istringstream is(line);

string words;

 while (is >> words) {

        sum++;

    }

return sum;

}

int main (){

    char line[255];

    cout << "Enter elements "<<endl;

    cin.getline(line,'.');

    cout << line <<endl;

    int sumofwords = CountWords(line);

    cout << sumofwords;

}

Завдання №2 VNS Lab 8

#include <iostream>

#include <vector>

#include <fstream>

using namespace std;

double avarageGrades= 3 ;

struct Student

{

    string name;

    string number;

    string group;

    double grades[3];

};

void printStudent(const vector<Student>& students){

    for (const Student& student : students) {

    cout << "Name "<< student.name <<endl;

    cout << "Phone number "<<student.number<< endl;

    cout << "group "<<student.group<<endl;

    cout << "Grades "<<endl;

    for (const double& grade : student.grades){

        cout << grade<<" ";

    }

    cout << " "<<endl;

    cout << "-----------------------" << endl;

    }

}

void PrintStudentsinFile(const vector<Student>& students, const string& filename){

    ofstream file;

file.open("Students.bin");

file.is\_open();

if(!file.is\_open()){

    cout << "ERROR"<<endl;

}

 for (const Student& student : students){

    file << "Name "<< student.name <<endl;

    file << "Phone number "<<student.number<< endl;

    file << "group "<<student.group<<endl;

    file << "Grades "<<endl;

    for (const double& grade : student.grades){

        file << grade<<" ";

    }

    file << " "<<endl;

 }

file.close();

}

void DeleteformFIle(vector<Student>& students,const string &group, double minAverage){

     ifstream inFile("Students.bin");

    if (!inFile.is\_open())

    {

        cout << "ERROR" << endl;

        return;

    }

    string line;

    while (getline(inFile, line))

    {

        Student student;

        inFile >> student.name >> student.number >> student.group;

        for (int i = 0; i < 3; ++i)

        {

            inFile >> student.grades[i];

        }

        double sum = 0;

        for (double grade : student.grades)

        {

            sum += grade;

        }

        double average = sum / 3;

        for (auto it = students.begin(); it != students.end(); it++){

        if (it->group == group)

        {

            if(average<avarageGrades)

            {

                students.erase(it);

            }

        }

        }

    }

    inFile.close();

}

int main(){

vector<Student> students{

{"Olexandr Oleksandrovych Voronin","9643433215","SHI-12",{5,3.5,3}},

{"Volodymyr Volodymyrovych Dashavski","968526076","SHI-12",{3,3,3}},

{"Dmytro Dmytrovych Sydor","974343615","SHI-12",{3.5,4,5}},

{"Anton Antonov Antan","956323615","SHI-11",{3.5,4,5}},

{"Ivan Ivasik Telesik","956321122","SHI-11",{3.5,2,2}}

};

printStudent(students);

PrintStudentsinFile(students,"Students.bin");

string groupDelete;

cout << "Enter group where you want to delete students: ";

cin >> groupDelete;

DeleteformFIle(students, groupDelete, avarageGrades);

cout << "After delete " << endl;

printStudent(students);

PrintStudentsinFile(students,"Students.bin");

}

Завдання №3 VNS Lab 9

#include <iostream>

#include <fstream>

#include <vector>

#include <algorithm>

#include <string>

#include <sstream>

#include <set>

using namespace std;

vector<string> splitstring(string& line){

    vector<string> w;

    istringstream iss(line);

    string word;

    while (iss >> word){

        w.push\_back(word);

    }

    return w;

}

void Longestword(vector<string>& words){

    sort(words.begin(), words.end());

    string longestword = words.front();

    cout << longestword;

}

int main(){

vector<string> lines;

string line;

string FILE1= "file1.txt";

string FILE2="file2.txt";

fstream fl1;

ofstream fl2;

fl1.open(FILE1, fstream::in | fstream::out);

fl2.open(FILE2);

if(!(fl1.is\_open()|| fl2.is\_open())){

    cout << "ERROR";

    }

 while (getline(fl1, line)) {

        lines.push\_back(line);

    }

   for(auto& line : lines){

    vector<string>words = splitstring(line);

    set<string> uniqueWords(words.begin(), words.end());

       if(uniqueWords.size()<words.size()){

            fl2 << line <<"\n";

       }

    Longestword(words);

   }

fl1.close();

fl2.close();

}

Завдання №4 Algotester Lab 4

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main(){

    int size,k;

    cin >> size >> k;

    vector<int> N;

    for(int i=0;i<size;i++){

        int element;

        cin >> element;

        N.push\_back(element);

    }

    sort(N.begin(), N.end());

    auto x = unique(N.begin(), N.end());

    N.erase(x,N.end());

    if(k > N.size()){

        k = k % N.size();

    }

    rotate(N.begin(), N.begin() + k, N.end());

    cout<< N.size()<<endl;

    for(int i = 0;i <N.size();i++){

        cout << N[i]<<" ";

    }

    cout <<endl;

}

Завдання №5 Algotester Lab 6

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

struct ChessPiece {

    char type;

    int x, y;

    ChessPiece(char t, int xpos, int ypos) : type(t), x(xpos), y(ypos) {}

};

bool Pawn(int x, int y, const vector<ChessPiece>& pieces) {

    for (const auto& piece : pieces) {

        if (piece.type == 'P' && ((x - 1 == piece.x && y - 1 == piece.y) || (x - 1 == piece.x && y + 1 == piece.y))) {

            return true;

        }

    }

    return false;

}

bool Rook(int x, int y, const vector<ChessPiece>& pieces) {

    for (const auto& piece : pieces) {

        if (piece.type == 'R' && (x == piece.x || y == piece.y)) {

            return true;

        }

    }

    return false;

}

bool Knight(int x, int y, const vector<ChessPiece>& pieces) {

    int knight\_dx[] = {-2, -1, 1, 2, 2, 1, -1, -2};

    int knight\_dy[] = {1, 2, 2, 1, -1, -2, -2, -1};

    for (int i = 0; i < 8; ++i) {

        int nx = x + knight\_dx[i];

        int ny = y + knight\_dy[i];

        for (const auto& piece : pieces) {

            if (nx == piece.x && ny == piece.y && piece.type == 'N') {

                return true;

            }

        }

    }

    return false;

}

bool Bishop(int x, int y, const vector<ChessPiece>& pieces) {

    int dx[] = {-1, -1, 1, 1};

    int dy[] = {-1, 1, -1, 1};

    for (int k = 0; k < 4; ++k) {

        for (int j = 1; j < 8; ++j) {

            int nx = x + dx[k] \* j;

            int ny = y + dy[k] \* j;

            for (const auto& piece : pieces) {

                if (nx == piece.x && ny == piece.y && piece.type == 'B') {

                    return true;

                }

            }

        }

    }

    return false;

}

bool King(int x, int y, const vector<ChessPiece>& pieces) {

    int king\_dx[] = {-1, -1, -1, 0, 0, 1, 1, 1};

    int king\_dy[] = {-1, 0, 1, -1, 1, -1, 0, 1};

    for (int i = 0; i < 8; ++i) {

        int nx = x + king\_dx[i];

        int ny = y + king\_dy[i];

        for (const auto& piece : pieces) {

            if (nx == piece.x && ny == piece.y && piece.type == 'K') {

                return true;

            }

        }

    }

    return false;

}

bool Queen(int x, int y, const vector<ChessPiece>& pieces) {

    for (const auto& piece : pieces) {

        if (piece.type == 'Q' && (x == piece.x || y == piece.y ||

                                   abs(x - piece.x) == abs(y - piece.y))) {

            return true;

        }

    }

    return false;

}

int main() {

    vector<ChessPiece> pieces;

    for (int i = 0; i < 8; ++i) {

        string row;

        cin >> row;

        for (int j = 0; j < 8; ++j) {

            if (row[j] != 'O') {

                pieces.push\_back(ChessPiece(row[j], i, j));

            }

        }

    }

    int Q;

    cin >> Q;

    vector<pair<int, int>> queries;

    for (int m = 0; m < Q; m++) {

        int a, b;

        cin >> a >> b;

        queries.push\_back({a - 1, b - 1});

    }

    for (const auto& query : queries) {

        int x = query.first;

        int y = query.second;

        string answer;

        bool isOccupied = any\_of(pieces.begin(), pieces.end(), [x, y](const ChessPiece& piece) {

            return piece.x == x && piece.y == y;

        });

        if (isOccupied) {

            cout << "X" << endl;

        } else {

            for (char figure : "PRNBKQ") {

                switch (figure) {

                    case 'P':

                        if (Pawn(x, y, pieces)) {

                            answer += figure;

                        }

                        break;

                    case 'R':

                        if (Rook(x, y, pieces)) {

                            answer += figure;

                        }

                        break;

                    case 'N':

                        if (Knight(x, y, pieces)) {

                            answer += figure;

                        }

                        break;

                    case 'B':

                        if (Bishop(x, y, pieces)) {

                            answer += figure;

                        }

                        break;

                    case 'K':

                        if (King(x, y, pieces)) {

                            answer += figure;

                        }

                        break;

                    case 'Q':

                        if (Queen(x, y, pieces)) {

                            answer += figure;

                        }

                        break;

                }

            }

            if (answer.empty()) {

                cout << "O" << endl;

            } else {

                sort(answer.begin(), answer.end());

                cout << answer << endl;

            }

        }

    }

    return 0;

}

Завдання №6 Class Practice Work

#include <iostream>

#include <fstream>

using namespace std;

enum FileOpResult { Success, Failure };

FileOpResult write\_to\_file(const char \*name, const char \*content) {

    ofstream fl(name);

    if (!fl.is\_open()) {

        return Failure;

    }

    fl << content << endl;

    fl.close();

    return Success;

}

int main(){

    const char \*file\_name = "file.txt";

    char content[200];

    cout << "Enter content to write into file: ";

    cin.getline(content,200);

  FileOpResult result = write\_to\_file(file\_name, content);

    if(result == Success)

    {

        ifstream fl(file\_name);

        string line;

        while(getline(fl, line)){

            cout << line << endl;

        }

        fl.close();

        cout << "successfully printed!" << endl;

    }

}

Завдання №7 Self Practice Work

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main(){

    int N,M;

    vector<int> first;

    vector<int> second;

    cin >> N;

    for(int i=0;i< N;i++){

        int b;

        cin >> b;

        first.push\_back(b);

    }

    cin >> M;

    for(int j=0;j< M;j++){

        int a;

        cin >> a;

        second.push\_back(a);

    }

   sort(first.begin(),first.end());

   sort(second.begin(),second.end());

   vector<int> diference1;

   set\_difference(first.begin(),first.end(),second.begin(),second.end(),back\_inserter(diference1));

    sort(diference1.begin(),diference1.end());

    cout << diference1.size()<<endl;

    for(int i=0;i<diference1.size();i++){

        cout <<diference1[i]<<" ";

    }

    cout << endl;

   vector<int> diference2;

   set\_difference(second.begin(),second.end(),first.begin(),first.end(),back\_inserter(diference2));

   sort(diference2.begin(),diference2.end());

    cout << diference2.size()<<endl;

    for(int i=0;i<diference2.size();i++){

        cout <<diference2[i]<<" ";

    }

    cout << endl;

   vector<int> intersection;

   set\_intersection(first.begin(), first.end(), second.begin(), second.end(),back\_inserter(intersection));

    sort(intersection.begin(),intersection.end());

    cout << intersection.size()<<endl;

    for(int i=0;i<intersection.size();i++){

        cout <<intersection[i]<<" ";

    }

    cout << endl;

   vector<int> uni;

   set\_union(first.begin(), first.end(), second.begin(), second.end(),back\_inserter(uni));

    sort(uni.begin(),uni.end());

    cout << uni.size()<<endl;

    for(int i=0;i<uni.size();i++){

        cout <<uni[i]<<" ";

    }

    cout << endl;

   vector<int> semetric\_dif;

   set\_symmetric\_difference(first.begin(),first.end(),second.begin(),second.end(),back\_inserter(semetric\_dif));

   sort(semetric\_dif.begin(),semetric\_dif.end());

   cout << semetric\_dif.size()<<endl;

    for(int i=0;i<semetric\_dif.size();i++){

        cout <<semetric\_dif[i]<<" ";

    }

    cout << endl;

}

## **5. Результати виконання завдань, тестування та фактично затрачений час:**

Завдання №1 VNS Lab 6

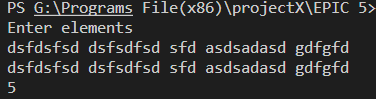


Рисунок2.Результат програми VNS Lab 6

Час затрачений на виконання завдання: 1 година

Завдання №2 VNS Lab 8

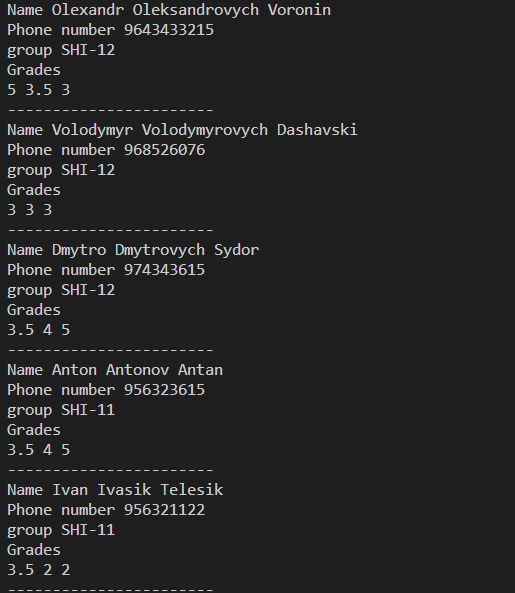


Рисунок 3 Результат програми VNS Lab 8

Час затрачений на виконання завдання:

Завдання №3 VNS Lab 9

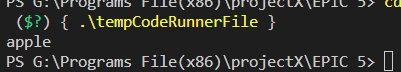


Рисунок 4 результат програми VNS Lab9

Час затрачений на виконання завдання: 2 години

Завдання №4 Algotester Lab 4

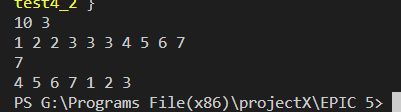


Рисунок 5 Результат програми Algotester 4v2

Час затрачений на виконання завдання: 1 година

Завдання №5 Algotester Lab 6

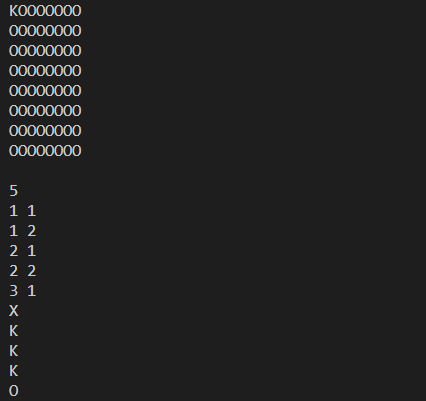


Рисунок 6 результат програми Algotester 6v2

Час затрачений на виконання завдання: 7 годин

Завдання №6 Class Practice Work Task

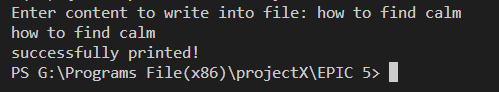


Рисунок 7 результат програми

Час затрачений на виконання завдання: 1 година

Завдання №7 Self Practice Work

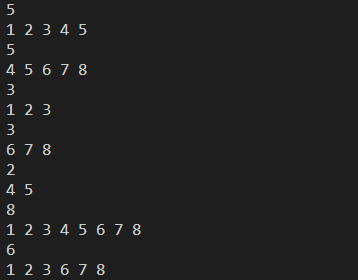


Рисунок 8 результат програми Self practice algotester 4v1

Час затрачений на виконання завдання: 1 година

## **6. Кооперація з командою:**

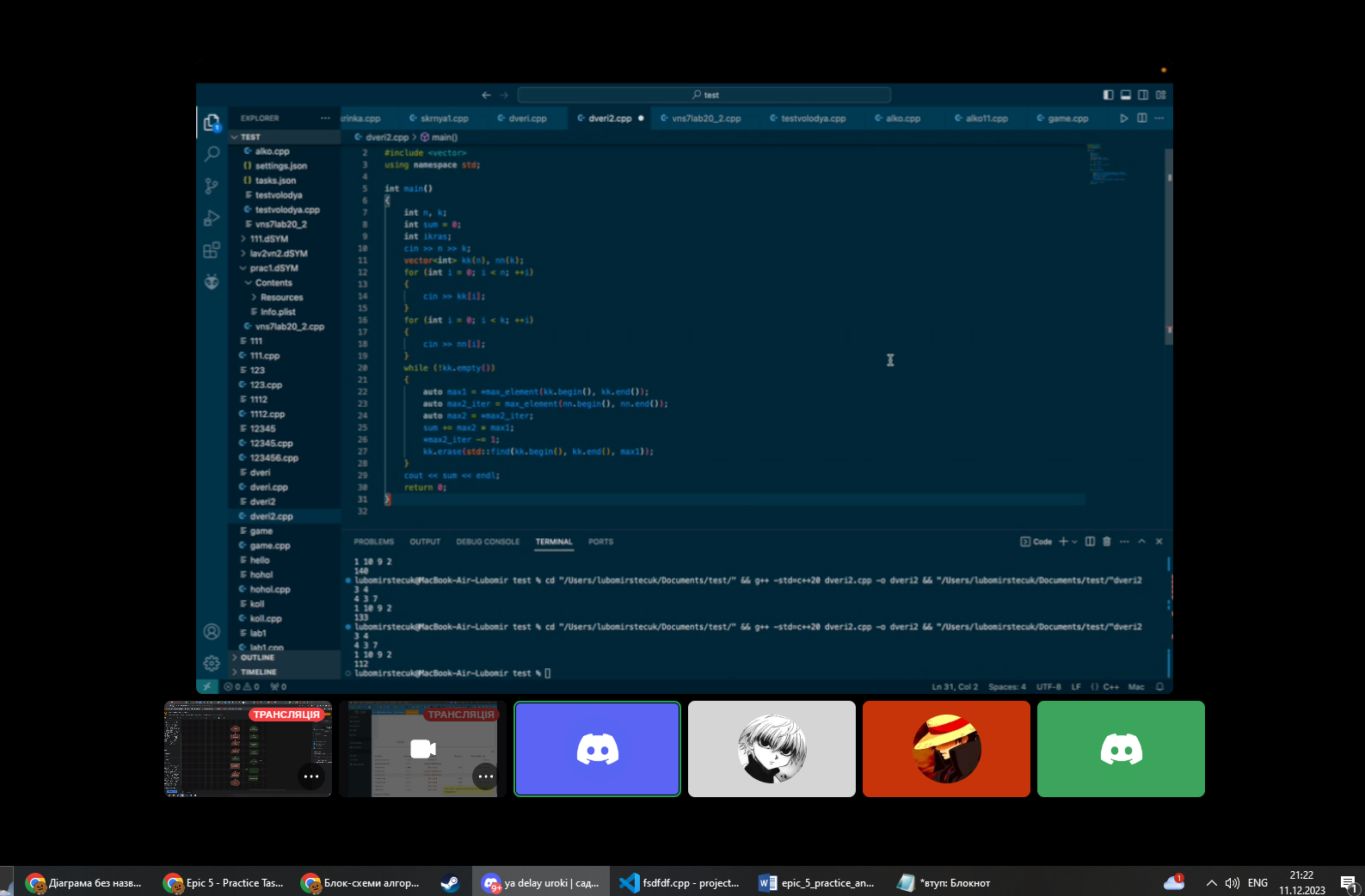


Рисунок 9.Перша зустріч з командою

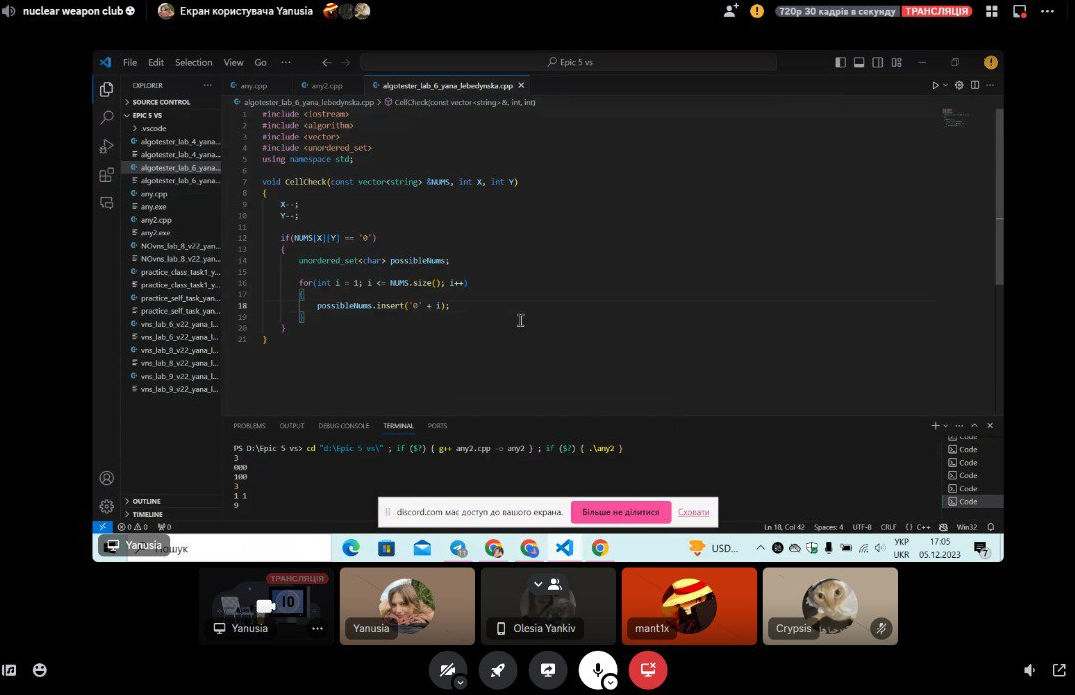


Рисунок 10 Друга зустріч з командою

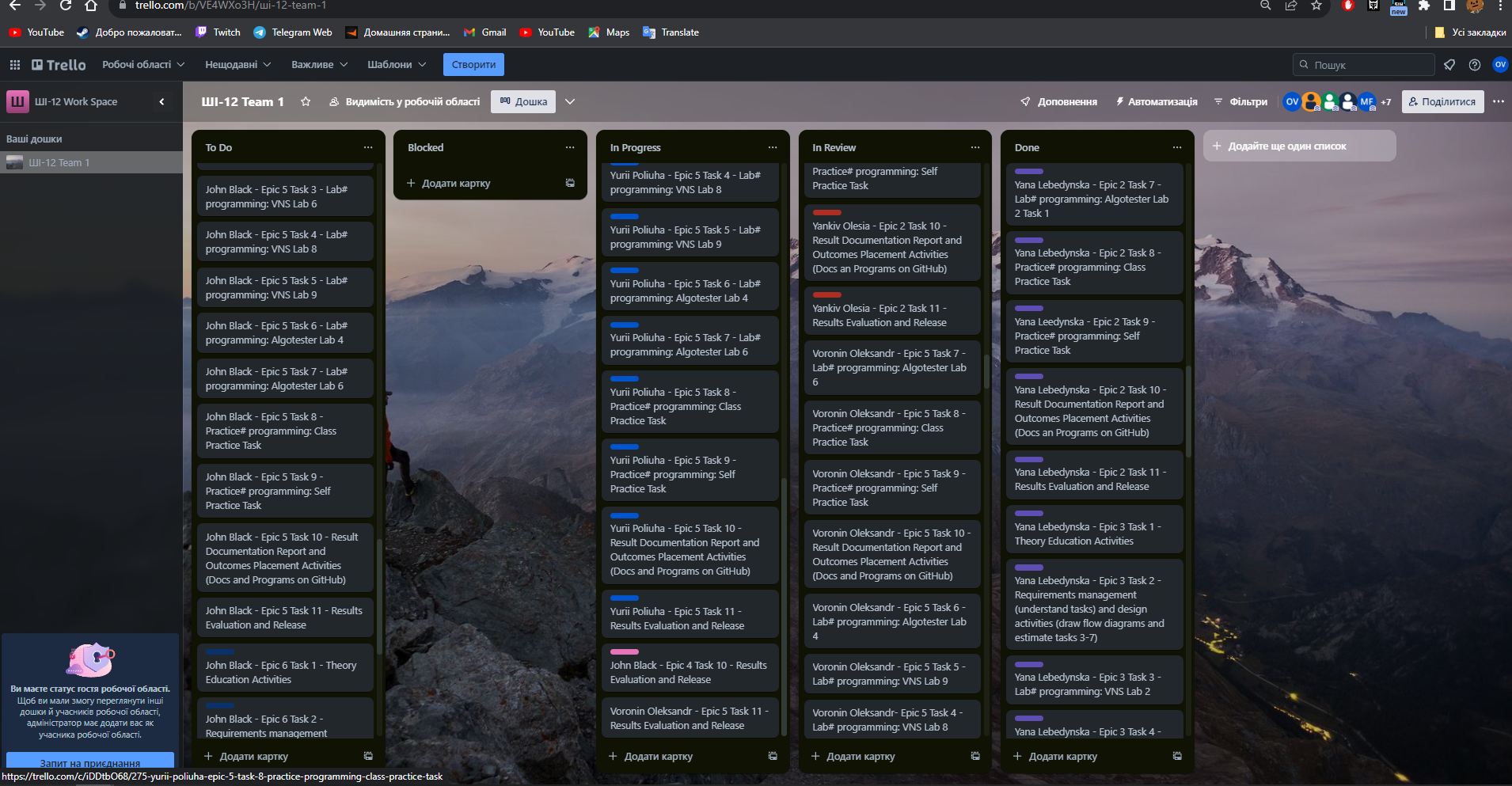


Рисунок 3 Дошка трело

# **Висновки:**

Під час опрацювання теоретичного матеріалу та роботи над завданнями розділу я навчився працювати з файлами, рядками та символьними елементами в мові C++.. Весь пройдений матеріал закріплено практично завдяки виконанні лабораторних та практичних робіт а також самопрактиці.