Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Лабораторних та практичних робіт № 5**

***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 й використання бібліотек»

***Виконав:***

студент групи ШІ-11 Мацько Ілля Феліксович

Львів 2023

# **Тема роботи:**

Прості структури даних. Одновимірні масиви. Двовимірні Масиви. Алгоритми обробки.

# **Мета роботи:**

Опанувати роботу з простими структурами даних. Засвоїти одновимірні та двовимірні масиви. Вивчити алгоритми обробки.

# **Теоретичні відомості:**

1) Теоретичні відомості з переліком важливих тем:

* Тема №1: Файли. Бінарні файли
* Тема №2: Системи числення
* Тема №3: Стандартна бібліотека та деталі/методи роботи з файлами
* Тема №4: Створення й використання бібліотек

2) Індивідуальний план опрацювання теорії

**Тема №1: Файли. Бінарні файли**

Робота з файлами у C++ є ключовою для збереження та обробки даних. Використання бінарних файлів дозволяє ефективно зберігати та відновлювати структуровані дані.

Посилання:

1. [cplusplus.com - File Input/Output](http://www.cplusplus.com/doc/tutorial/files/)

2. [cppreference.com - File I/O](https://en.cppreference.com/w/cpp/io)

3. [GeeksforGeeks - Binary File Handling in C++](https://www.geeksforgeeks.org/binary-file-handling-c/))

**Тема №2: Системи числення**

C++ використовує стандартні системи числення, такі як десяткова, двійкова, вісімкова та шістнадцяткова. Робота з ними реалізується за допомогою маніпуляторів потоків та функцій стандартної бібліотеки.

Посилання:

1. [cplusplus.com - Input/Output Manipulators](http://www.cplusplus.com/reference/iomanip/)

2. [cppreference.com - Integer manipulators](https://en.cppreference.com/w/cpp/io/manip)

**Тема №3: Стандартна бібліотека та деталі/методи роботи з файлами**

Стандартна бібліотека C++ надає класи та функції для роботи з файлами. Використовуються класи `ifstream`, `ofstream` та `fstream` для введення та виведення текстових та бінарних файлів.

Посилання:

1. [cplusplus.com - Files in C++](http://www.cplusplus.com/doc/tutorial/files/)

2. [cppreference.com - File Input/Output](https://en.cppreference.com/w/cpp/io)

3. [GeeksforGeeks - File Handling in C++](https://www.geeksforgeeks.org/file-handling-c-classes/)

**Тема №4: Створення й використання бібліотек**

Створення та використання бібліотек у C++ передбачає роботу з заголовочними файлами та визначенням функцій та класів. Включення бібліотеки у проект відбувається за допомогою директиви `#include`.

Посилання:

1. [cplusplus.com - Header Files](http://www.cplusplus.com/doc/tutorial/headerfiles/)

2. [cppreference.com - Header Files](https://en.cppreference.com/w/cpp/preprocessor/include)

3. [GeeksforGeeks - C++ Libraries](https://www.geeksforgeeks.org/cpp-libraries/)

# **Виконання роботи:**

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

Завдання №1 VNS Lab 6 - Task 1

* + Варіант завдання - 23
  + Деталі завдання:

Задано рядок, що складається із символів. Символи поєднуються в слова.

Слова одне від одного відокремлюються одним або декількома пробілами.

Наприкінці тексту ставиться крапка. Текст містить не більше 255 символів.

Виконати ввід рядка, використовуючи функцію gets(s) і здійснити обробку

рядка у відповідності зі своїм варіантом.

Завдання №2 VNS Lab 8 - Task 1

* + Варіант завдання - 23
  + Деталі завдання:

Сформувати двійковий файл із елементів, заданої у варіанті структури,

роздрукувати його вмістиме, виконати знищення й додавання елементів у

відповідності зі своїм варіантом, використовуючи для пошуку елементів що

знищуються чи додаються, функцію. Формування, друк, додавання й знищення

елементів оформити у вигляді функцій. Передбачити повідомлення про

помилки при відкритті файлу й виконанні операцій вводу/виводу.

Завдання № 3 VNS Lab 9

* Варіант завдання - 23
* Деталі завдання:

Створити текстовий файл F1 не менше, ніж з 10 рядків і записати в нього інформацію

Виконати завдання:

1) Скопіювати з файлу F1 у файл F2 всі рядки, у яких міститься два однакових

слова.

2) Визначити номер слова, у якому найбільше букв «А».

Завдання № 4 Algotester Lab 4

* Варіант завдання - 1
* Деталі завдання:

Вам дано 2 цілих чисел масиви, розміром N та M. Ваше завдання вивести:

1. Різницю N-M

2. Різницю M-N

3. Їх перетин

4. Їх обєднання

5. Їх симетричну різницю

Завдання № 4 Algotester Lab 6

* Варіант завдання - 1
* Деталі завдання:

Вам дано N слів та число K. Ваше завдання перечислити букви в словах, які зустрічаються в тексті більше-рівне ніж K разів (саме слово, не буква!). Великі та маленькі букви вважаються однаковими, виводити необхідно малі, посортовані від останьої до першої у алфавіті. Букву потрібно виводити лише один раз.

У випадку якщо таких букв немає - вивести "Empty!".

Завдання № 5 Class practice work

* + Деталі завдання:

Умови задачі: створити файл із заданим ім’ям; якщо файл існує – перезаписати його вміст; написати код стійкий до різних варіантів вхідних параметрів; name – ім’я, може не включати шлях; записати у файл вміст стрічки content, прочитати content із стандартного вводу; повернути статус операції: Success – все пройшло успішно, Failure – файл не вдалося створити, або збій операції відкриття, запису даних, чи закриття файла.

Завдання № 6 Self practice work

* + Деталі завдання

У вас є шахова дошка розміром 8×8 та дуже багато фігур. Вам дають позиції фігур на дошці (всі фігури одного кольору, кількість королів може бути > 1).

Далі йдуть Q запитів з координатами клітинки {x,y}. На кожен запит ви маєте вивести стрічку si - посортовані за алфавітом букви фігур, які атакують цю клітинку (пішаки атакують вниз). У випадку, якщо на клітинці стоїть якась фігура - виведіть символ X. У випадку, якщо клітинку не атакують - виведіть O. Наявніть фігури у певній клітинці не блокує атаку для іншої фігури. Тобто якщо між турою та клітинкою стоїть інша фігура - вважається що тура атакує цю клітинку.

## **Дизайн та оцінка часу виконання завдань:**

Програма № 1 VNS Lab 6 - Task 1

* + Запланований час на реалізацію - 1 година

Програма № 2 VNS Lab 8 - Task 1

* + Запланований час на реалізацію - 2 години

Програма № 2 VNS Lab 9 - Task 1

* + Запланований час на реалізацію - 2 години

Програма № 3 Algotester Lab 4v1

* + Запланований час на реалізацію - 2 години

Програма № 4 Algotester Lab 6v1

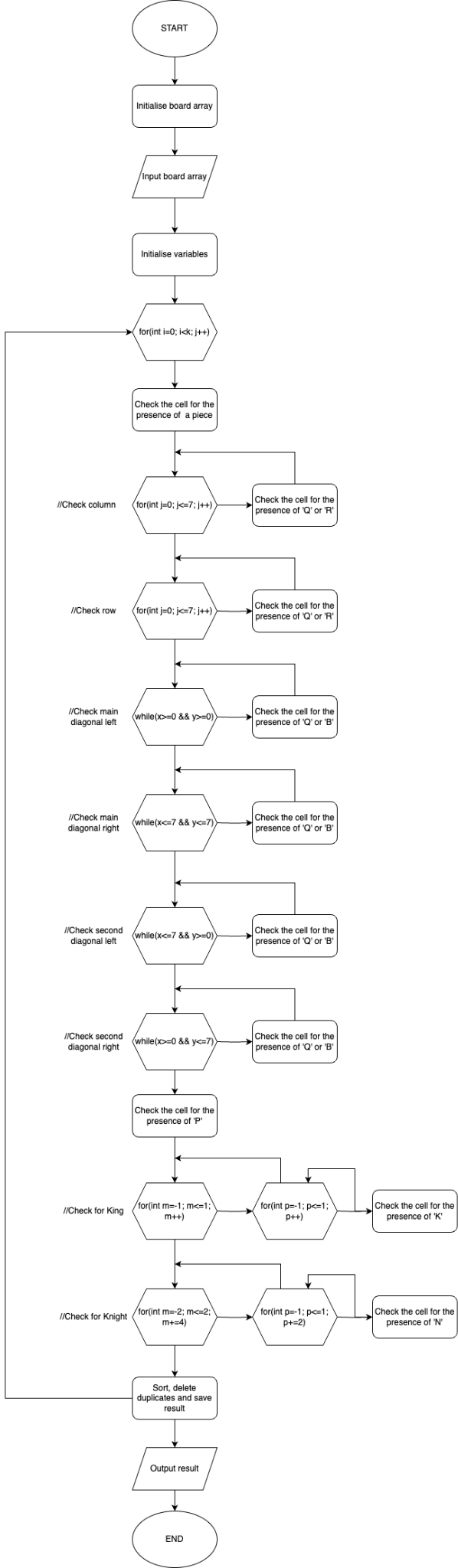
* + Запланований час на реалізацію - 2 години

Програма № 5 Class practice work

* + Запланований час на реалізацію - 2 години

Програма № 6 Self practice(Algotester Lab6v2)

* + Блок-схема



* + Запланований час на реалізацію - 3 години

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

Завдання № 1 [VNS Lab 6 - Task 1](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/667/files#diff-558b317ca830d9633303edb83e994fadeb5e2991661264a32b7695e4883f8d50)

#include <iostream>

#include <stdio.h>

using namespace std;

int main()

{

char str[256];

cin.getline(str, 256);

// gets(str);

int count = 0, min = 256;

for(int i = 0; i < 256; i++)

{

if(str[i] != ' ')

{

count++;

} else {

if(count < min)

{

min = count;

}

count = 0;

}

}

cout << min;

}

Завдання № 2 [VNS Lab 8 - Task 1](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/667/files#diff-04d3cf4f6cb5410bc20b225a1590547a1d688748b7afeadb692cf5a19ba17528)

#include <iostream>

#include <fstream>

#include <vector>

using namespace std;

struct Stadium {

string name;

int yearBuilt;

int numOfFields;

string sports;

};

void printStadiums(const vector<Stadium>& stadiums) {

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

cout << "Name: " << stadiums[i].name << " Year Built: " << stadiums[i].yearBuilt << " Number of Fields: " << stadiums[i].numOfFields << " Sports: " << stadiums[i].sports << endl;

}

}

void destroyStadiums(std::vector<Stadium>& stadiums, int year) {

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

if (it->yearBuilt < year) {

it = stadiums.erase(it);

} else {

it++;

}

}

}

void addStadiums(vector<Stadium>& stadiums, int targetIndex) {

if (targetIndex >= 2 && targetIndex < stadiums.size()) {

Stadium newStadium1 = {"NEW STADIUM 1", 2022, 1, "Handball"};

Stadium newStadium2 = {"NEW STADIUM 2", 2022, 1, "Volleyball"};

stadiums.insert(stadiums.begin() + targetIndex, **{**newStadium1, newStadium2**}**);

} else {

cerr << "Invalid target index for adding stadiums." << endl;

return;

}

}

int main() {

ofstream file1("stadiums.bin", ios::binary);

if (!file1.is\_open()) {

cerr << "Error opening file1 for writing." << endl;

return -1;

}

vector<Stadium> stadiums = **{**

{"Stadium A", 2000, 1, "Football"},

{"Stadium B", 2005, 2, "Soccer"},

{"Stadium C", 2010, 1, "Basketball"},

{"Stadium D", 2015, 3, "Baseball"},

{"Stadium E", 2020, 1, "Football"},

{"Stadium F", 2025, 2, "Soccer"},

{"Stadium G", 2030, 1, "Basketball"},

{"Stadium H", 2035, 3, "Baseball"}

**}**;

file1.write(reinterpret\_cast<char\*>(stadiums.data()), sizeof(Stadium) \* stadiums.size());

file1.close();

ifstream file2("stadiums.bin", ios::binary);

if (!file2.is\_open()) {

cerr << "Error opening file for reading." << endl;

return -1;

}

cout << "Stadiums:" << endl;

printStadiums(stadiums);

int targetYear;

cout << endl << "Enter target year: ";

cin >> targetYear;

destroyStadiums(stadiums, targetYear);

cout << endl << "After destroying stadiums built before " << targetYear << ":" << endl;

printStadiums(stadiums);

int targetIndex;

cout << endl << "Enter target index: ";

cin >> targetIndex;

addStadiums(stadiums, targetIndex);

cout << endl << "After adding new stadiums before index " << targetIndex << ":" << endl;

printStadiums(stadiums);

file2.close();

return 0;

}

Завдання № 2 [VNS Lab 9 - Task 1](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/667/files#diff-87235c7b0632d3c3c5b8e128d2d63c6dac026a884d5ecfe98c54c4214e9d85d7)

#include <iostream>

#include <fstream>

#include <sstream>

#include <algorithm>

#include <map>

using namespace std;

void copyLinesWithSameWords(const string& inputFile, const string& outputFile) {

ifstream inFile(inputFile);

ofstream outFile(outputFile);

if (!inFile.is\_open() || !outFile.is\_open()) {

cerr << "Error opening files." << endl;

return;

}

string line;

while (getline(inFile, line)) {

istringstream iss(line);

string word;

map<string, int> wordCountMap;

while (iss >> word) {

wordCountMap[word]++;

}

for (const auto& entry : wordCountMap) {

if (entry.second >= 2) {

outFile << line << endl;

}

}

}

cout << "Words with count greater than two have been written to " << outputFile << endl;

inFile.close();

outFile.close();

}

int findWordWithMostA(const string& filename) {

ifstream inFile(filename);

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

cerr << "Error opening file." << endl;

return -1;

}

int maxACount = 0;

int wordNumber = 0;

string line;

int currentWordNumber = 1;

while (getline(inFile, line)) {

istringstream iss(line);

string word;

while (iss >> word) {

int aCount = count(word.begin(), word.end(), 'a');

if (aCount > maxACount) {

maxACount = aCount;

wordNumber = currentWordNumber;

}

currentWordNumber++;

}

}

inFile.close();

if (maxACount > 0) {

cout << "The word with the most 'a's is at position: " << wordNumber << endl;

} else {

cout << "No word with 'a' found." << endl;

}

return wordNumber;

}

void displayFileContent(const string& filename) {

ifstream file(filename);

if (file.is\_open()) {

cout << "Content of " << filename << ":\n";

string line;

while (getline(file, line)) {

cout << line << endl;

}

file.close();

} else {

cerr << "Error opening file " << filename << endl;

}

}

int main() {

ofstream f1("F1.txt");

if (f1.is\_open()) {

f1 << "sdhjkgfkldsf dfv jdfj jdgfh" << endl;

f1 << "aaaaaaaa kjdfgk jdhfksg jfhdjdf" << endl;

f1 << "njhjhf njhjhf sdf" << endl;

f1.close();

} else {

cerr << "Error opening file F1.txt for writing." << endl;

return 1;

}

copyLinesWithSameWords("F1.txt", "F2.txt");

findWordWithMostA("F1.txt");

cout << endl;

displayFileContent("F2.txt");

return 0;

}

Завдання № 3 [Algotester Lab 4v1](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/667/files#diff-1c3fd4c306e3b772c3d8a8f8f0f497a3f87801f4b2f8ff024e6af1249d039cef)

#include <iostream>

#include <algorithm>

#include <vector>

using namespace std;

void print(vector<int>& vector)

{

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

{

cout << vector[i] << ' ';

}

cout << endl;

}

int main() {

int n;

cin >> n;

int arr1[n];

for (int i = 0; i < n; i++) {

cin >> arr1[i];

}

int m;

cin >> m;

int arr2[m];

for (int i = 0; i < m; i++) {

cin >> arr2[i];

}

sort(arr1, arr1+n);

sort(arr2, arr2+m);

vector<int> result;

set\_difference(arr1, arr1+n, arr2, arr2+m, back\_inserter(result));

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

print(result);

result.clear();

set\_difference(arr2, arr2+m, arr1, arr1+n, back\_inserter(result));

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

print(result);

result.clear();

set\_intersection(arr1, arr1+n, arr2, arr2+m, back\_inserter(result));

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

print(result);

result.clear();

set\_union(arr1, arr1+n, arr2, arr2+m, back\_inserter(result));

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

print(result);

result.clear();

set\_symmetric\_difference(arr1, arr1+n, arr2, arr2+m, back\_inserter(result));

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

print(result);

return 0;

}

Завдання № 4 [Algotester Lab 6v1](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/667/files#diff-dd29c20680d78afd43e7589451cce66518261477233456000c72f32f43c750c6)

#include <iostream>

#include <map>

#include <vector>

#include <algorithm>

#include <cctype>

#include <locale>

using namespace std;

int main()

{

int n, k;

cin >> n >> k;

map<string, int> words;

for (int i = 0; i < n; i++) {

string input;

cin >> input;

transform(input.begin(), input.end(), input.begin(), [](unsigned char c) {

return tolower(c);

});

words[input]++;

}

map<char, int> output;

for (auto& word : words) {

if (word.second >= k) {

for (char c : word.first) {

output[c]++;

}

}

}

if (output.empty()) {

cout << "Empty!";

return 0;

}

vector<char> chars;

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

for(auto& c : output) {

chars.push\_back(c.first);

}

for(int i = chars.size()-1; i >= 0; i--)

{

cout << chars[i] << ' ';

}

return 0;

}

Завдання № 5 [Class practice work](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/667/files#diff-5770488e75ca8d32f7fc6a33e2a59930c04acfed353881b01e32c89b878bdd5d)

#include <iostream>

#include <fstream>

using namespace std;

enum FileOpResult {*Success*, *Failure*};

FileOpResult write\_to\_file(string name, string content) {

if (name.empty() || content.empty()) {

return *Failure*;

}

ofstream file(name, ios::trunc);

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

return *Failure*;

}

file << content;

if (file.fail()) {

return *Failure*;

}

file.close();

if (file.fail()) {

return *Failure*;

}

return *Success*;

}

FileOpResult copy\_file(string file\_from, string file\_to) {

if (file\_from.empty() || file\_to.empty()) {

return *Failure*;

}

ifstream source(file\_from, ios::binary);

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

return *Failure*;

}

ofstream destination(file\_to, ios::binary | ios::trunc);

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

return *Failure*;

}

destination << source.rdbuf();

if (destination.fail()) {

return *Failure*;

}

source.close();

destination.close();

if (source.fail() || destination.fail()) {

return *Failure*;

}

return *Success*;

}

FileOpResult print\_file\_content(string filename) {

ifstream file(filename);

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

cerr << "Failed to open file: " << filename << endl;

return FileOpResult::*Failure*;

}

string line;

while (getline(file, line)) {

cout << line << endl;

}

file.close();

return FileOpResult::*Success*;

}

int main() {

string fileFrom = "source.txt";

string fileTo = "destination.txt";

cout << "Enter strings(to stop type 'exit'):" << endl;

string content;

string line;

while (getline(cin, line) && line != "exit") {

content += line + "\n";

}

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

if (result == *Success*) {

cout << "Successfully written content to " << fileFrom << "'." << endl;

result = copy\_file(fileFrom, fileTo);

if (result == *Success*) {

cout << "Successfully copied to " << fileTo << "'." << endl;

} else {

cerr << "Error" << endl;

}

} else {

cerr << "Error" << endl;

}

cout << "Second file content:" << endl;

print\_file\_content("destination.txt");

return 0;

}

Завдання № 6 [Self-practice(Algotester Lab6v2)](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/667/files#diff-22685b788a791c3d1043c81cae6f70b532715664af8219b0cb3cad03fd87b3c8)

#include <iostream>

#include <algorithm>

using namespace std;

int main()

{

char board[8][8];

for(int i=0; i<8; i++)

{

for(int j=0; j<8; j++)

{

cin >> board[i][j];

}

}

int k, x, y;

cin >> k;

string result;

for(int i=0; i<k; i++)

{

cin >> x >> y;

x--;

y--;

if(board[x][y] != 'O')

{

result.push\_back('X');

result.push\_back('\n');

continue;

}

string output;

//column

for(int j=0; j<=7; j++)

{

if(board[j][y] == 'R' || board[j][y] == 'Q')

{

output.push\_back(board[j][y]);

}

}

//row

for(int j=0; j<=7; j++)

{

if(board[x][j] == 'R' || board[x][j] == 'Q')

{

output.push\_back(board[x][j]);

}

}

int count = 0;

//main diagonal left

while(x>=0 && y>=0)

{

if(board[x][y] == 'Q' || board[x][y] == 'B')

{

output.push\_back(board[x][y]);

}

y--;

x--;

count++;

}

x+=count;

y+=count;

count=0;

//main diagonal right

while(x<=7 && y<=7)

{

if(board[x][y] == 'Q' || board[x][y] == 'B')

{

output.push\_back(board[x][y]);

}

y++;

x++;

count++;

}

x-=count;

y-=count;

count=0;

//second diagonal left

while(x<=7 && y>=0)

{

if(board[x][y] == 'Q' || board[x][y] == 'B')

{

output.push\_back(board[x][y]);

}

y--;

x++;

count++;

}

x-=count;

y+=count;

count=0;

//second diagonal right

while(x>=0 && y<=7)

{

if(board[x][y] == 'Q' || board[x][y] == 'B')

{

output.push\_back(board[x][y]);

}

y++;

x--;

count++;

}

x+=count;

y-=count;

//pawn

if((x-1>=0 && y-1>=0 && board[x-1][y-1] == 'P') || (x-1>=0 && y+1<=7 && board[x-1][y+1] == 'P'))

{

output.push\_back('P');

}

//king

for(int m=-1; m<=1; m++)

{

for(int p=-1; p<=1; p++)

{

if(x+m>=0 && x+m<=7 && y+p>=0 && y+p<=7 && board[x+m][y+p] == 'K')

{

output.push\_back('K');

}

}

}

//knight

for(int m=-2; m<=2; m+=4)

{

for(int p=-1; p<=1; p+=2)

{

if((x+m>=0 && x+m<=7 && y+p>=0 && y+p<=7 && board[x+m][y+p] == 'N') || (x+p>=0 && x+p<=7 && y+m>=0 && y+m<=7 && board[x+p][y+m] == 'N'))

{

output.push\_back('N');

}

}

}

if(output.empty())

{

result.push\_back('O');

result.push\_back('\n');

continue;

}

sort(begin(output), end(output));

for(int j=0; j<output.size(); j++)

{

result.push\_back(output[j]);

while(output[j] == output[j+1])

{

j++;

}

}

result.push\_back('\n');

}

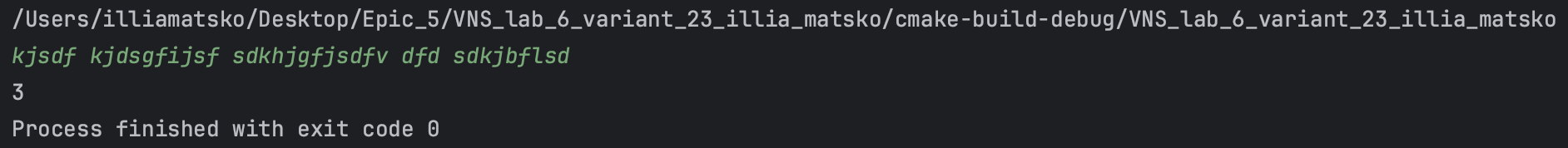
cout << result;

return 0;

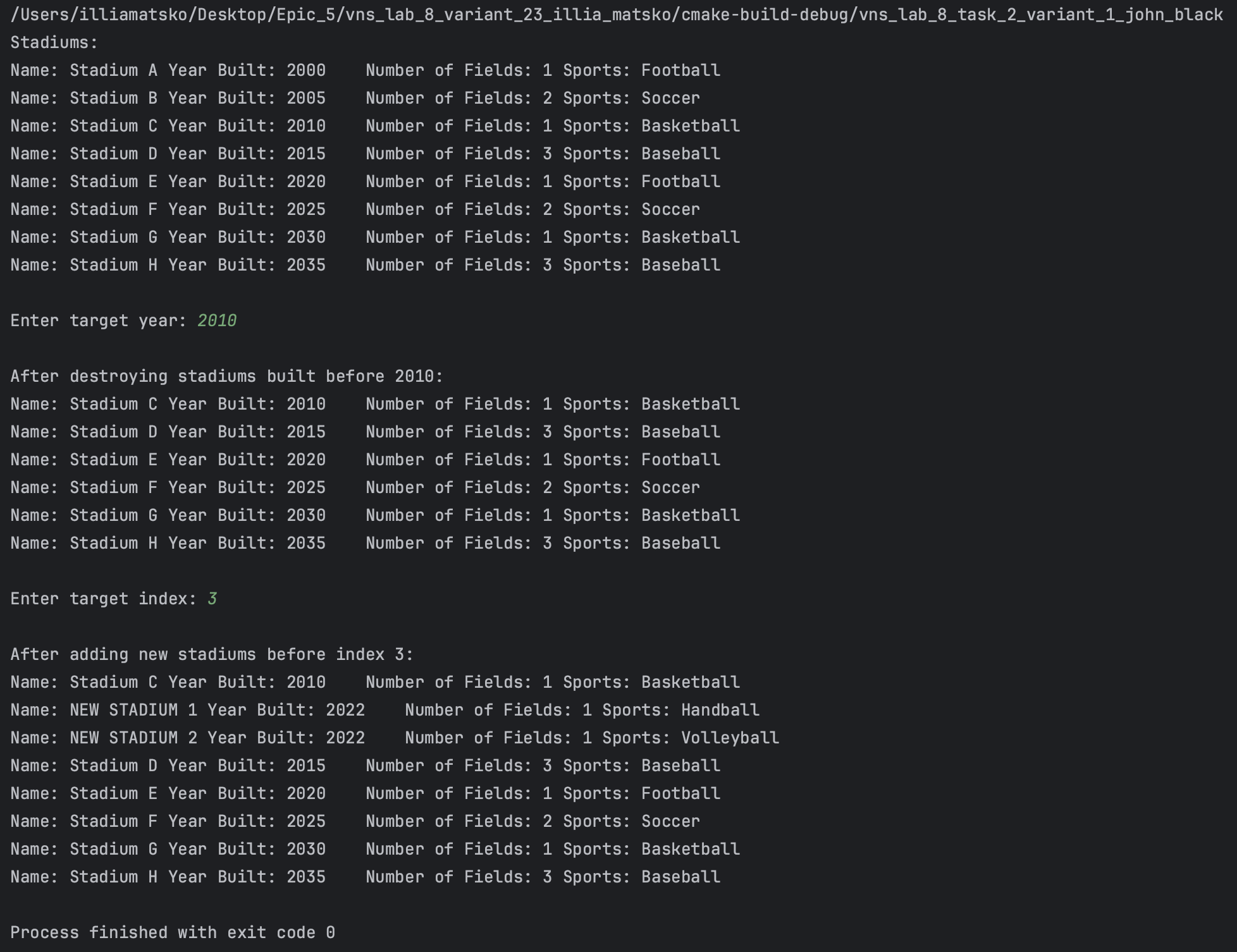
}

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

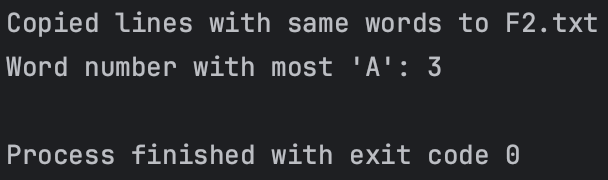
**Завдання № 1**



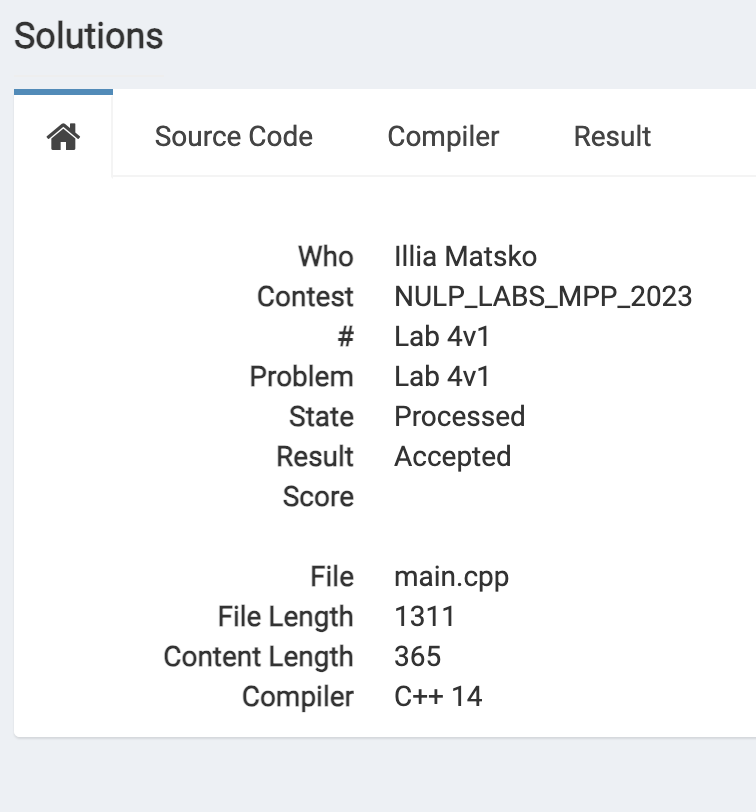
**Завдання № 2**

****

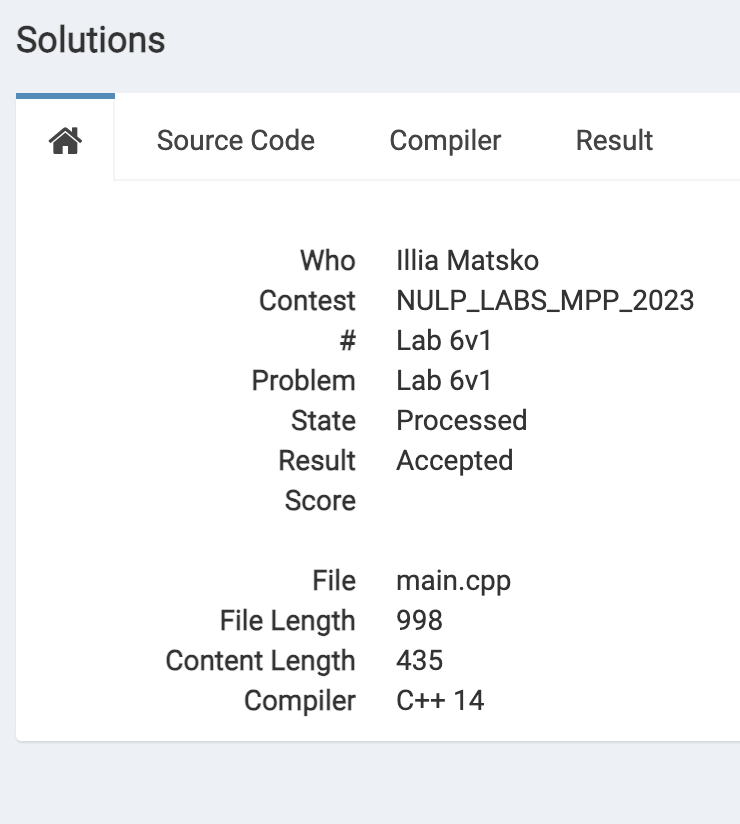
**Завдання № 3**

****

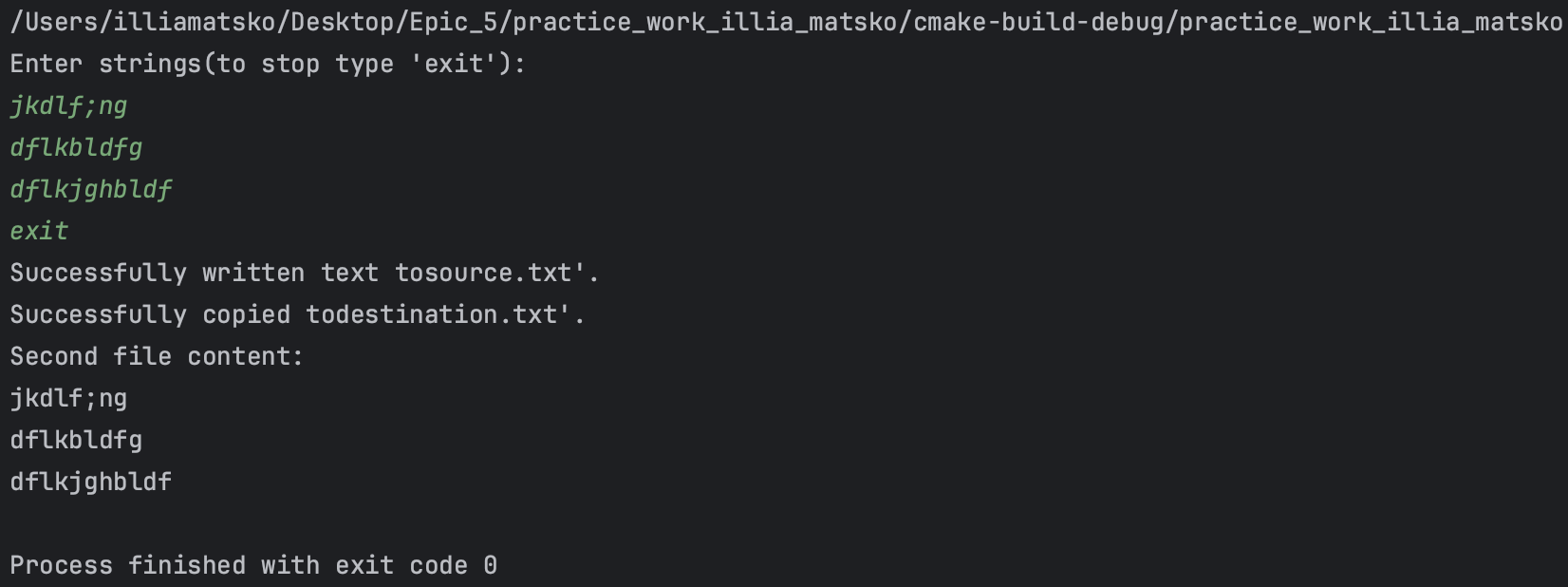
**Завдання № 4**

****

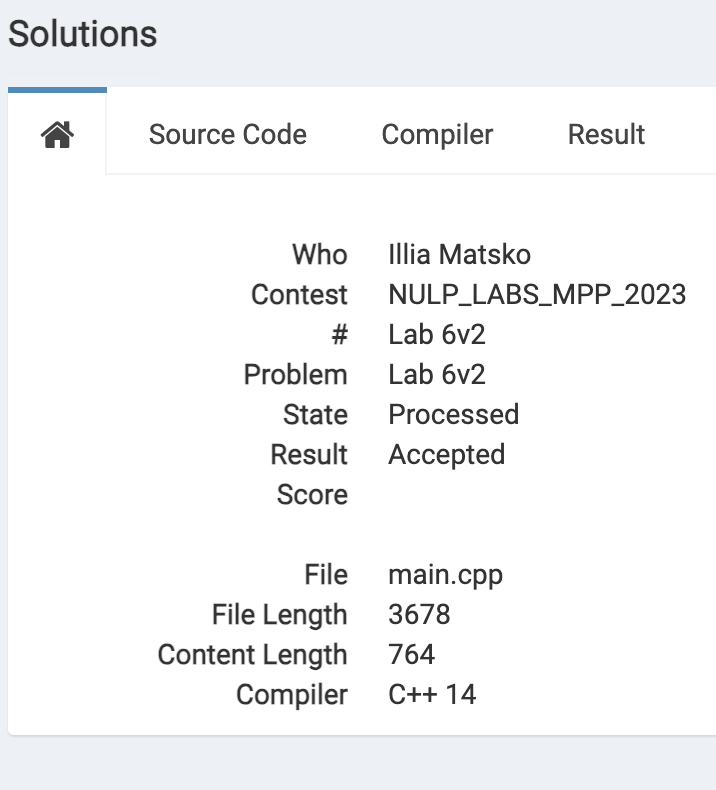
**Завдання № 5**

****

**Завдання № 6**

****

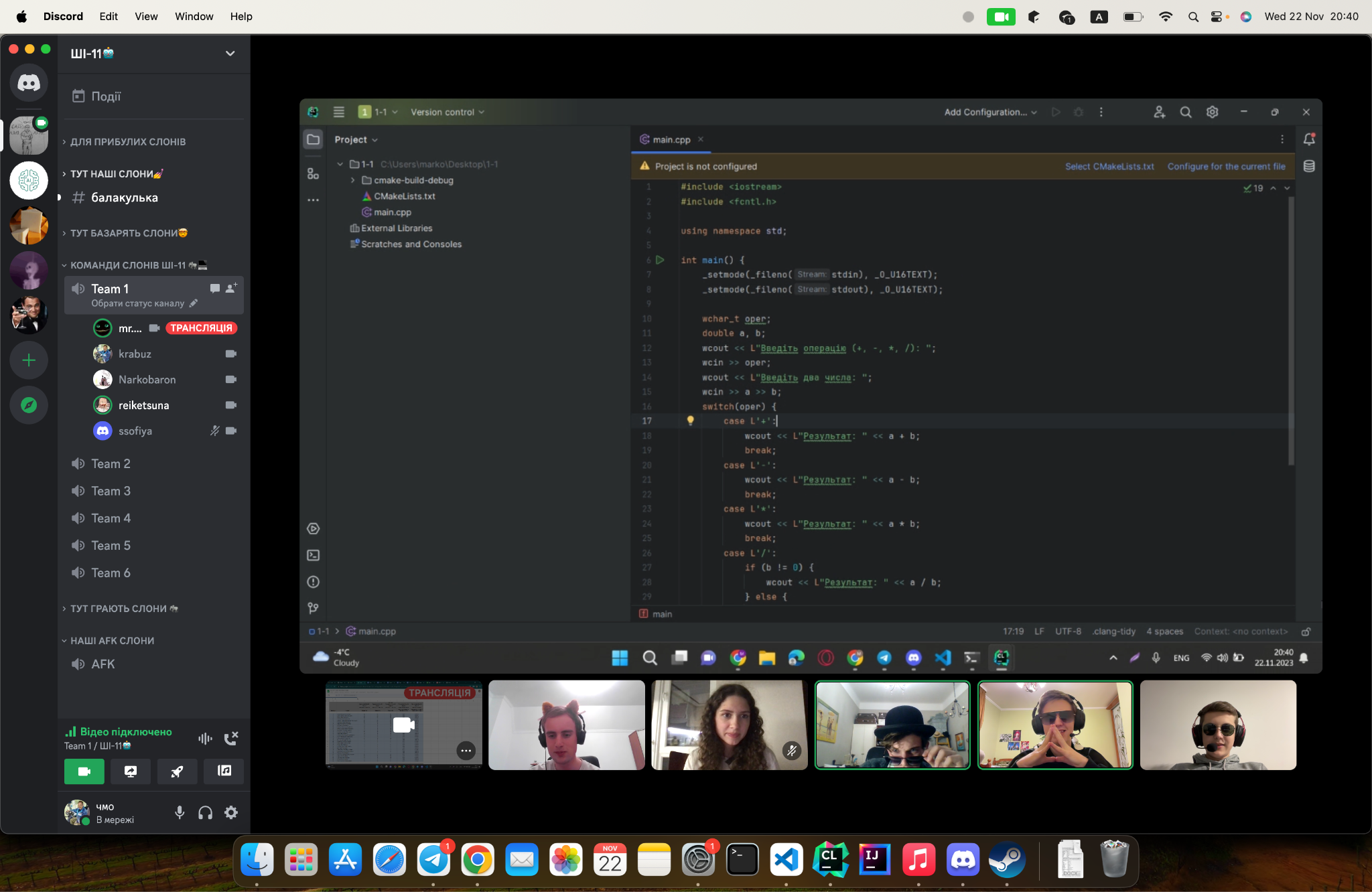
**Завдання № 7**

****

[**Посилання на Pull-request**](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/667)

Робота з командою:





# **Висновки:**

У цій роботі ми розглянули чотири ключові теми в мові програмування C++. Розпочали з роботи з файлами та бінарними файлами, надали посилання на ресурси для додаткового вивчення. Далі детально оглянули системи числення та розкрили їх використання в C++. В третьому розділі дослідили роботу зі стандартною бібліотекою та файлами.

В завершенні вивчили процес створення та використання бібліотек в C++, зазначивши основні етапи та вказавши джерела для додаткової інформації. Ця робота дозволить розширити розуміння базових концепцій мови C++ та ефективно використовувати їх у програмуванні.