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 та рядкові змінні. Текстові файли. Стандартна бібліотека. Створення й використання бібліотек.»

***Виконала:***

студент групи ШІ-12

Онаць Максим Володимирович

# **Тема роботи:**

Ознайомлення з роботою з файлами в С++. Принцип роботи з бінарними та текстовими файлами.

# **Мета роботи:**

Теоретично та практично освоїти роботу з файлами.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Робота з файлами.
* Тема №2: Бінарні й текстові файли.
* Тема №3: Робота з символьними та рядковими змінними.

1. Індивідуальний план опрацювання теорії:

* Тема №1: Робота з файлами.
  + Джерела Інформації
  + <https://youtu.be/FeNqHytI0fA?si=mErTTwb3H348xzXO>
  + Що опрацьовано:
    - Створення, видалення, перейменування файлів, внесення, зчитування інформації та ін.
  + Статус: Ознайомлений
  + Початок опрацювання теми: 30.11
  + Звершення опрацювання теми: 01.12
* Тема №2: Бінарні й текстові файли.
  + Джерела Інформації:
  + <https://www.easeus.com/knowledge-center/what-is-the-difference-between-binary-and-text-files.html>
  + Що опрацьовано:
    - Різниця між текстовими і бінарними файлами.
  + Статус: Ознайомлений
  + Початок опрацювання теми: 30.11
  + Звершення опрацювання теми: 01.12
* Тема №3 Робота з символьними та рядковими змінними.
  + Джерела Інформації:
  + <http://www.kievoit.ippo.kubg.edu.ua/kievoit/2016/73_C++/index.html>
  + Що опрацьовано:
    - Основні методи роботи з символьними та рядковими змінними.
  + Статус: Ознайомлений
  + Початок опрацювання теми: 30.11
  + Звершення опрацювання теми: 03.12

# **Виконання роботи:**

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

Завдання №1 VNS Lab 6.

Варіант 12.

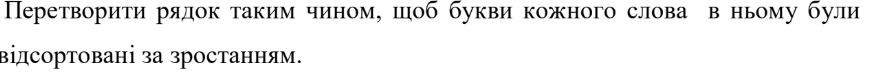
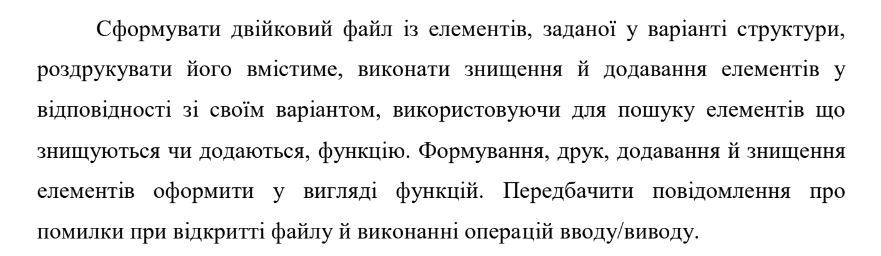


Рисунок 1. Умова завдання 1.

Завдання №2 VNS Lab 8.

Варіант 12.



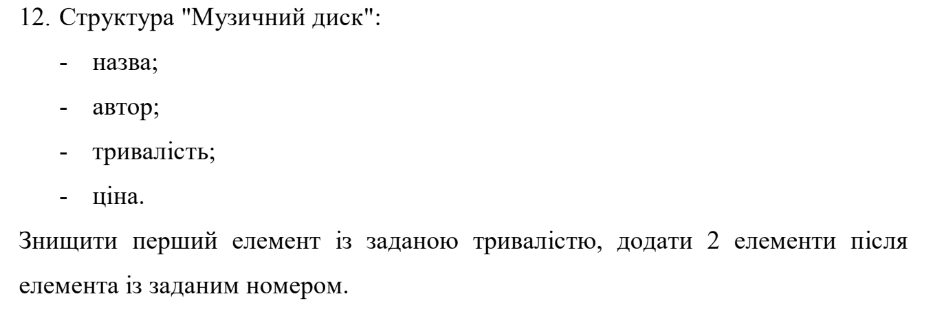


Рисунок 2. Умова завдання 2.

Завдання №3 VNS Lab 9.

Варіант 12.

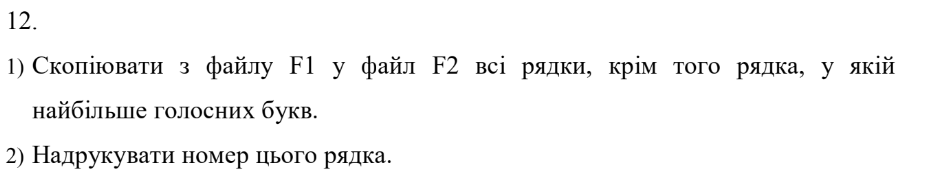


Рисунок 3. Умова завдання 3.

Завдання №4 Algotester Lab 4.

Варіант 2.

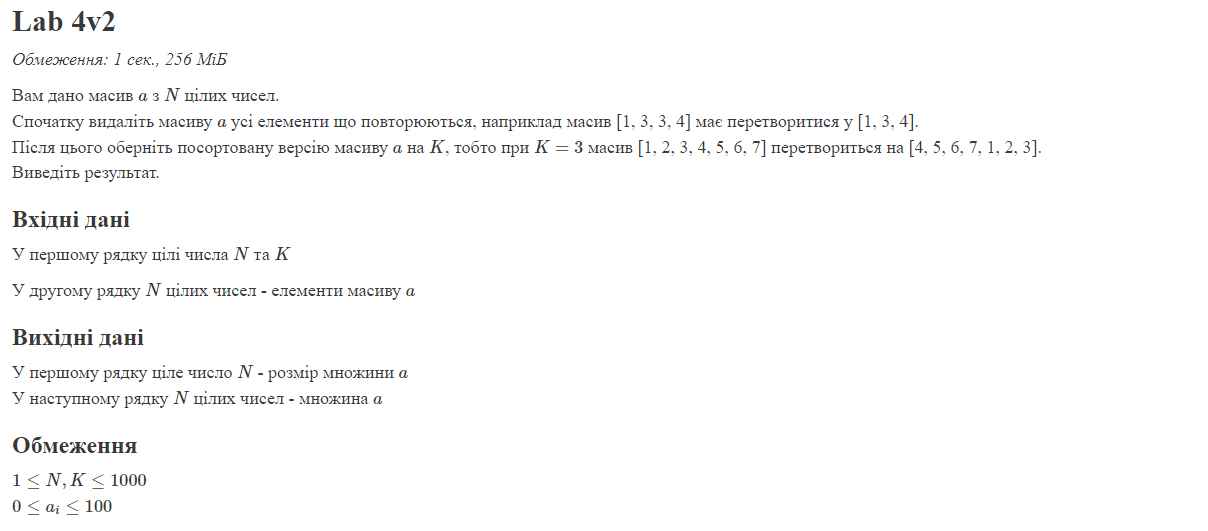


Рисунок 4. Умова завдання 4.

Важливо врахувати обмеження під час імплементації програми.

Завдання №5 Algotester Lab 6.

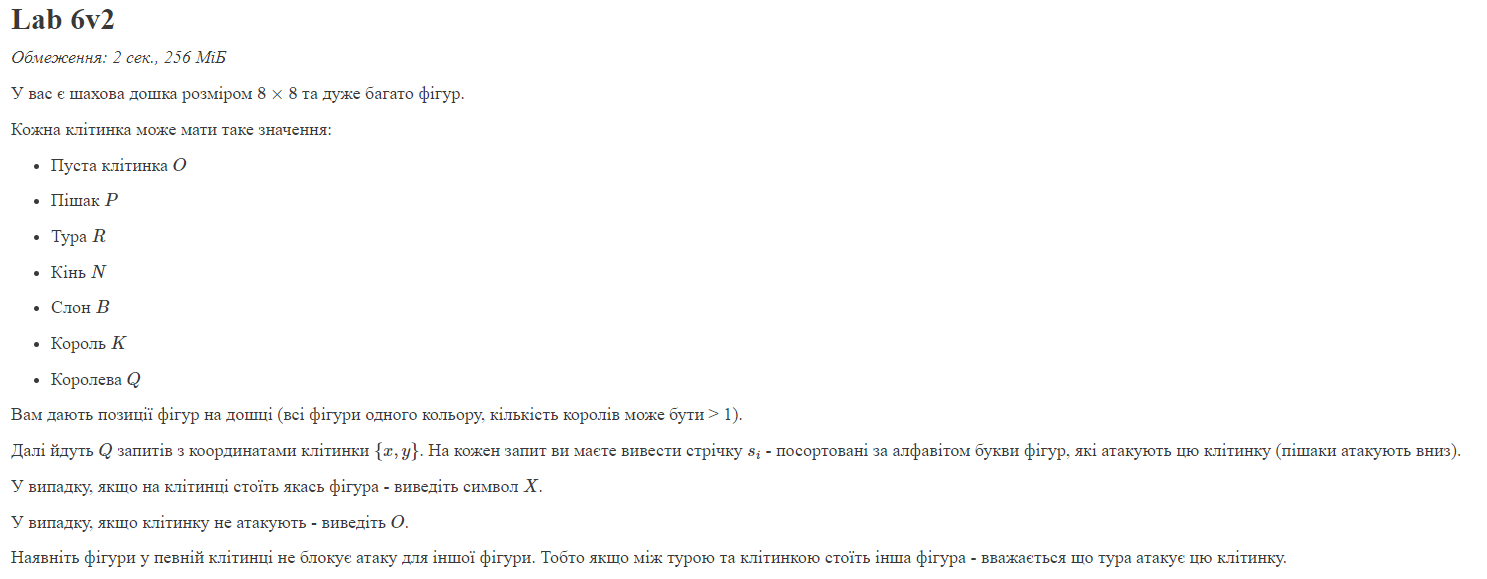


Рисунок 5. Умова завдання 5.

Важливо врахувати обмеження під час імплементації програми.

Завдання №6 Class Practice Task 1.

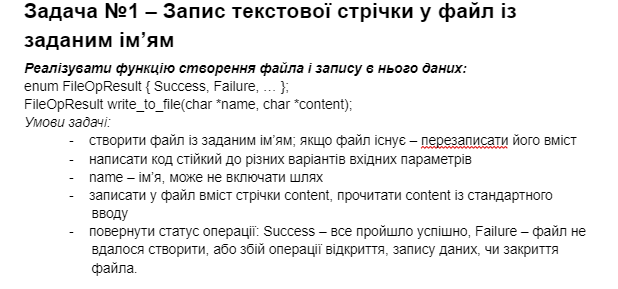


Рисунок 6. Умова завдання 6.

Завдання №7 Class Practice Task 2.

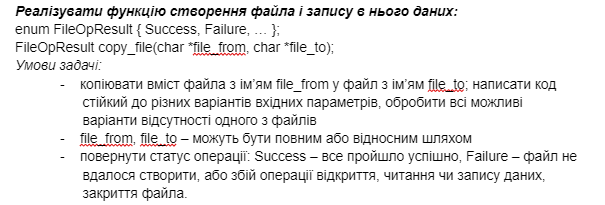


Рисунок 7. Умова завдання 7.

Завдання №8 Self-Practice Task.

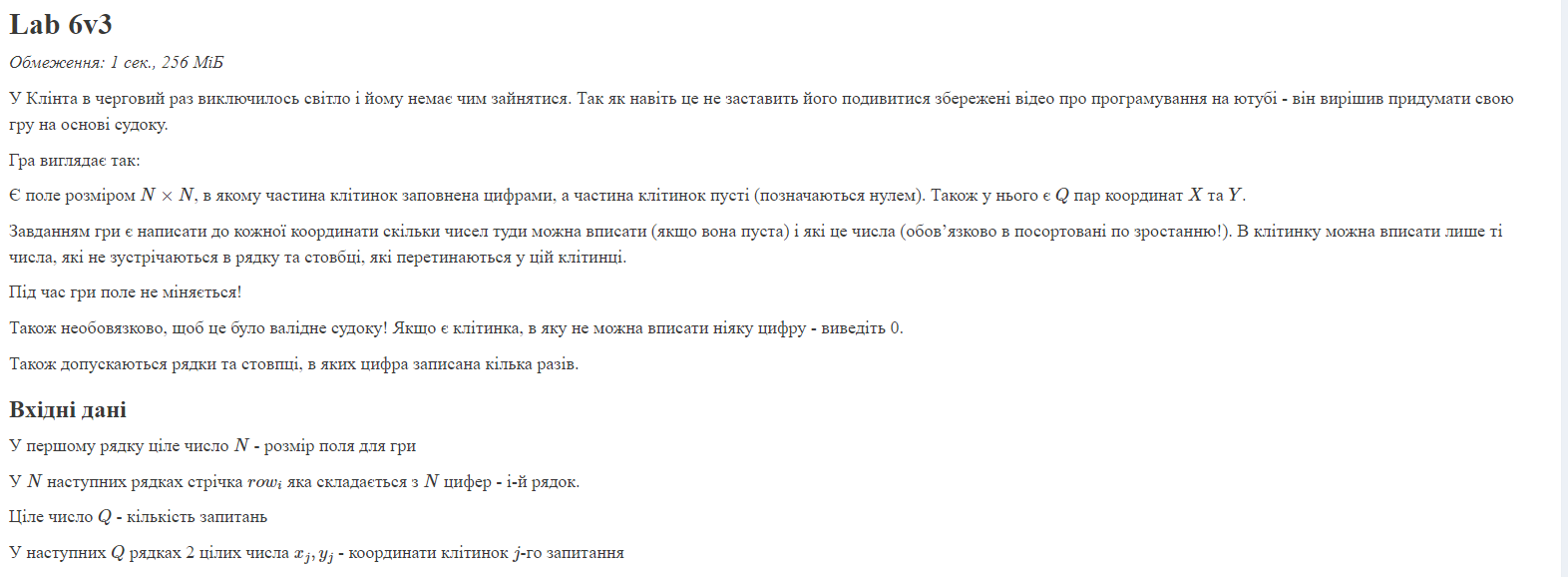


Рисунок 8. Умова завдання 8.

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

Програма №1 VNS Lab 6.

Планований час на виконання – 1,5 год

Програма №2 VNS Lab 8.

Планований час на виконання – 2 год

Програма №3 VNS Lab 9.

Планований час на виконання – 1,5 год

Програма №4 Algotester Lab 4.

Планований час на виконання – 1,5 год

Програма №5 Algotester Lab 6.

Планований час на виконання – 3 год

Програма №6 Class Practice Task 1.

Планований час на виконання – 40 хв

Програма №7 Class Practice Task 2.

Планований час на виконання – 40 хв

Програма №8 Self-Practice Task.

Планований час на виконання – 40 хв

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

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

Завдання №1

#include <iostream>

#include <cstring>

#include <algorithm>

using namespace std;

void sortCharsInWords(char\* str) {

int len = strlen(str);

// Проходимо по кожному символу рядка

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

// Якщо знайдено початок слова

if (isalpha(str[i])) {

// Знаходимо кінцевий індекс слова

int j = i;

while (isalpha(str[j + 1])) {

j++;

}

// Сортуємо букви між початковим і кінцевим індексами слова

sort(str + i, str + j + 1);

// Пересуваємо індекс до кінця слова

i = j;

}

}

}

int main() {

const int MAX\_LENGTH = 255;

char str[MAX\_LENGTH];

cout << "Введіть рядок: ";

cin.getline(str, MAX\_LENGTH);

// Виклик функції для сортування букв в кожному слові

sortCharsInWords(str);

cout << "Перетворений рядок: " << str << endl;

return 0;

}

Завдання №2

#include <iostream>

#include <fstream>

#include <string>

#include <cstring>

using namespace std;

// Структура "Музичний диск"

struct MusicDisc {

string title;

string author;

int duration;

float price;

};

// Функція для видалення елемента з файлу

void deleteDisc(const string& filename, int index) {

fstream file(filename, ios::binary | ios::in | ios::out);

if (!file) {

cerr << "An error occurred while opening the file." << endl;

return;

}

// Обчислення розміру структури

int size = sizeof(MusicDisc);

// Встановлення позиції для видалення елемента

file.seekp(index \* size, ios::beg);

// Запис нульових байтів для елемента, щоб він був видалений

char\* buffer = new char[size];

memset(buffer, 0, size);

file.write(buffer, size);

delete[] buffer;

file.close();

cout << "An element with an index " << index << " was deleted." << endl;

}

// Функція для додавання нового елемента до файлу

void addDisc(const string& filename, const MusicDisc& disc) {

fstream file(filename, ios::binary | ios::in | ios::out | ios::app);

if (!file) {

cerr << "An error occurred while opening the file." << endl;

return;

}

// Запис нового елемента в кінець файлу

file.write(reinterpret\_cast<const char\*>(&disc), sizeof(MusicDisc));

file.close();

cout << "A new item has been added to the file." << endl;

}

// Функція для друку вмісту файлу

void printFileContent(const string& filename) {

ifstream file(filename, ios::binary);

if (!file) {

cerr << "An error occurred while opening the file." << endl;

return;

}

// Переміщаємося до початку файлу

file.seekg(0, ios::beg);

// Отримуємо розмір структури

int size = sizeof(MusicDisc);

MusicDisc disc;

int index = 0;

// Читаємо кожен елемент з файлу та виводимо його

while (file.read(reinterpret\_cast<char\*>(&disc), size)) {

cout << "Element " << index << ":" << endl;

cout << "Name: " << disc.title << endl;

cout << "Author: " << disc.author << endl;

cout << "Duration: " << disc.duration << " Sec" << endl;

cout << "Price: " << disc.price << " UAH" << endl;

cout << endl;

index++;

}

file.close();

}

int main() {

string filename = "music.bin";

// Створення двійкового файлу з елементами структури "Музичний диск"

ofstream file(filename, ios::binary);

if (!file) {

cerr << "An error occurred while opening the file" << endl;

return 1;

}

MusicDisc disc1 = { "Title 1", "Author 1", 180, 9.99 };

MusicDisc disc2 = { "Title 2", "Author 2", 210, 12.99 };

MusicDisc disc3 = { "Title 3", "Author 3", 240, 14.99 };

file.write(reinterpret\_cast<const char\*>(&disc1), sizeof(MusicDisc));

file.write(reinterpret\_cast<const char\*>(&disc2), sizeof(MusicDisc));

file.write(reinterpret\_cast<const char\*>(&disc3), sizeof(MusicDisc));

file.close();

// Друк вмісту файлу

cout << "Contents of the file after creation:" << endl;

printFileContent(filename);

cout << endl;

// Видалення елемента з індексом 1

deleteDisc(filename, 1);

// Друк вмісту файлу після видалення

cout << "File contents after deletion:" << endl;

printFileContent(filename);

cout << endl;

// Додавання нового елемента

MusicDisc disc4 = { "Title 4", "Author 4", 195, 11.49 };

addDisc(filename, disc4);

// Друк вмісту файлу після додавання

cout << "File content after adding:" << endl;

printFileContent(filename);

cout << endl;

return 0;

}

Завдання №3

#include <iostream>

#include <fstream>

#include <string>

int countVowels(const std::string& line) {

int vowelsCount = 0;

std::string vowels = "aeiouAEIOU";

for (char c : line) {

if (vowels.find(c) != std::string::npos) {

vowelsCount++;

}

}

return vowelsCount;

}

int main() {

std::ifstream inputFile("F1.txt");

std::ofstream outputFile("F2.txt");

std::string line;

int maxVowels = 0;

int maxVowelsLine = 0;

int lineNumber = 0;

while (std::getline(inputFile, line)) {

lineNumber++;

int vowelsCount = countVowels(line);

if (vowelsCount > maxVowels) {

maxVowels = vowelsCount;

maxVowelsLine = lineNumber;

}

// Копіюємо рядки в файл F2, крім рядка з найбільшою кількістю голосних букв

if (lineNumber != maxVowelsLine) {

outputFile << line << std::endl;

}

}

inputFile.close();

outputFile.close();

std::cout << "Line number with the largest number of vowels: " << maxVowelsLine << std::endl;

return 0;

}

Завдання №4

#include <iostream>

#include <vector>

#include <algorithm>

int main() {

int N, K;

std::cin >> N >> K;

std::vector<int> a(N);

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

std::cin >> a[i];

}

// Видалення повторюючихся елементів

std::sort(a.begin(), a.end());

auto last = std::unique(a.begin(), a.end());

a.erase(last, a.end());

// Обертання масиву на K

K %= a.size();

std::rotate(a.begin(), a.begin() + K, a.end());

// Виведення результату

std::cout << a.size() << std::endl;

for (int i : a) {

std::cout << i << " ";

}

std::cout << std::endl;

return 0;

}

Завдання №5

#include <iostream>

#include <vector>

using namespace std;

void bishop(int &y, int &x, char desk[8][8], vector<char> &result) {

int dir1[4][2] = {{1, 1}, {-1, -1}, {1, -1}, {-1, 1}};

for (int i = 0; i < 4; ++i) {

for (int j = 1; y + j \* dir1[i][0] >= 0 && y + j \* dir1[i][0] < 8 && x + j \* dir1[i][1] >= 0 && x + j \* dir1[i][1] < 8; ++j) {

if (desk[y + j \* dir1[i][0]][x + j \* dir1[i][1]] == 'B') {

result.push\_back('B');

return;

}

}

}

}

void king(int &y, int &x, char desk[8][8], vector<char> &result) {

int dir2[8][2] = {{-1, 0}, {-1, 1}, {0, 1}, {1, 1}, {1, 0}, {1, -1}, {0, -1}, {-1, -1}};

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

int ny = y + dir2[i][0], nx = x + dir2[i][1];

if (ny >= 0 && ny < 8 && nx >= 0 && nx < 8 && desk[ny][nx] == 'K') {

result.push\_back('K');

break;

}

}

}

void queen(int &y, int &x, char desk[8][8], vector<char> &result) {

int dir4[4][2] = {{1, 1}, {1, -1}, {-1, -1}, {-1, 1}};

for (int i = 0; i < 4; ++i) {

for (int j = 1; y + j \* dir4[i][0] >= 0 && y + j \* dir4[i][0] < 8 && x + j \* dir4[i][1] >= 0 && x + j \* dir4[i][1] < 8; ++j) {

if (desk[y + j \* dir4[i][0]][x + j \* dir4[i][1]] == 'Q') {

result.push\_back('Q');

return;

}

}

}

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

if ((y >= 0 && y < 8 && desk[y][j] == 'Q') || (j >= 0 && j < 8 && desk[j][x] == 'Q')) {

result.push\_back('Q');

return;

}

}

}

void rook(int &y, int &x, char desk[8][8], vector<char> &result) {

int dir5[4][2] = {{0, 1}, {1, 0}, {0, -1}, {-1, 0}};

for (int i = 0; i < 4; ++i) {

for (int j = 1; y + j \* dir5[i][0] >= 0 && y + j \* dir5[i][0] < 8 && x + j \* dir5[i][1] >= 0 && x + j \* dir5[i][1] < 8; ++j) {

if (desk[y + j \* dir5[i][0]][x + j \* dir5[i][1]] == 'R') {

result.push\_back('R');

return;

}

}

}

}

void CheckFigureOnSquare(int y, int x, char desk[8][8], vector<char> &result) {

bishop(y, x, desk, result);

king(y, x, desk, result);

int dir3[8][2] = {{-2, 1}, {-1, 2}, {1, 2}, {2, 1}, {2, -1}, {1, -2}, {-1, -2}, {-2, -1}};

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

int ny = y + dir3[i][0], nx = x + dir3[i][1];

if (ny >= 0 && ny < 8 && nx >= 0 && nx < 8 && desk[ny][nx] == 'N') {

result.push\_back('N');

break;

}

}

if ((y - 1 >= 0 && x + 1 < 8 && desk[y - 1][x + 1] == 'P') || (y - 1 >= 0 && x - 1 >= 0 && desk[y - 1][x - 1] == 'P'))

result.push\_back('P');

queen(y, x, desk, result);

rook(y, x, desk, result);

}

int main() {

char desk[8][8];

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

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

cin >> desk[i][j];

int q;

cin >> q;

int inputs[q][2];

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

for (int j = 0; j < 2; ++j)

cin >> inputs[i][j];

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

int y = inputs[i][0] - 1, x = inputs[i][1] - 1;

vector<char> result;

if (desk[y][x] != 'O') cout << 'X' << endl;

else {

CheckFigureOnSquare(y, x, desk, result);

if (result.empty()) cout << "O";

else for (char a : result) cout << a;

cout << endl;

}

}

return 0;

}

Завдання №6

#include <fstream>

#include <iostream>

#include <string>

using namespace std;

enum FileOpResult { Success, Failure };

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

ofstream file(name);

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

return Failure;

}

file << content;

if (file.fail()) {

file.close();

return Failure;

}

file.close();

return Success;

}

int main() {

const char \*name = "example.txt";

const char \*content = "Hello, this is the content of the file.";

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

if (result == Success) {

cout << "Content written to file successfully!" << endl;

} else {

cout << "Failed to write to file." << endl;

}

return 0;

}

Завдання №7

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

enum FileOpResult { Success, Failure };

FileOpResult copy\_file(const char \*file\_from, const char \*file\_to) {

ifstream source(file\_from);

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

return Failure; // Unable to open source file

}

ofstream destination(file\_to);

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

source.close();

return Failure; // Unable to open or create destination file

}

string line;

while (getline(source, line)) {

destination << line << endl;

}

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

source.close();

destination.close();

return Failure; // Failed to read from source or write to destination

}

source.close();

destination.close();

return Success;

}

int main() {

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

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

FileOpResult result = copy\_file(file\_from, file\_to);

if (result == Success) {

cout << "File copied successfully!" << endl;

} else {

cout << "Failed to copy file." << endl;

}

return 0;

}

Завдання №8

#include <iostream>

#include <vector>

#include <set>

#include <algorithm>

using namespace std;

void process(vector<vector<int>> &field, int x, int y) {

if (field[x - 1][y - 1] == 0) {

set<int> numbers;

for (int i = 1; i <= field.size(); i++) {

numbers.insert(i);

}

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

numbers.erase(field[i][y - 1]);

numbers.erase(field[x - 1][i]);

}

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

for (int a : numbers) {

cout << a << " ";

}

} else {

cout << "1" << endl << field[x - 1][y - 1];

}

cout << endl;

}

int main() {

int N, Q;

cin >> N;

const int rows = N;

const int cols = N;

vector<vector<int>> field;

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

vector<int> rowOfField(cols);

for (int j = 0; j < cols; ++j) {

char number;

cin >> number;

rowOfField[j] = number - '0';

}

field.push\_back(rowOfField);

}

cin >> Q;

vector<pair<int, int>> coordinates;

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

int x, y;

cin >> x >> y;

coordinates.emplace\_back(x, y);

}

for (auto coordinate : coordinates) {

process(field, coordinate.first, coordinate.second);

}

return 0;

}

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

Завдання №1



Рисунок 9. Результат виконання програми 1.

Час затрачений на виконання завдання – 2 год

Завдання №2

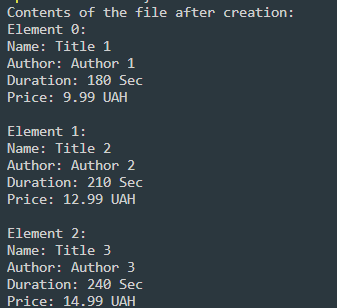


Рисунок 10. Результат виконання програми 2.

Час затрачений на виконання завдання – 2 год

Завдання №3



Рисунок 11. Результат виконання програми 3.

Час затрачений на виконання завдання – 2 год

Завдання №4

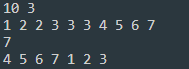


Рисунок 12. Результат виконання програми 4.

Час затрачений на виконання завдання – 2 год

Завдання №5

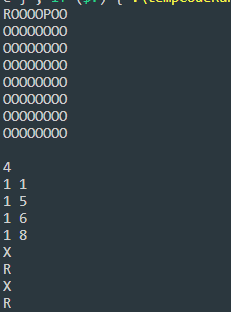


Рисунок 13. Результат виконання програми 5.

Час затрачений на виконання завдання – 3 год

Завдання №6



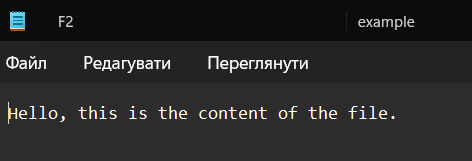


Рисунок 14. Результат виконання програми 6.

Час затрачений на виконання завдання – 1,5 год

Завдання №7



Рисунок 15. Результат виконання програми 7.

Час затрачений на виконання завдання – 1,5 год

Завдання №8

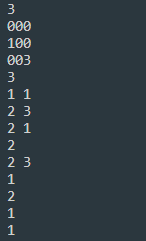
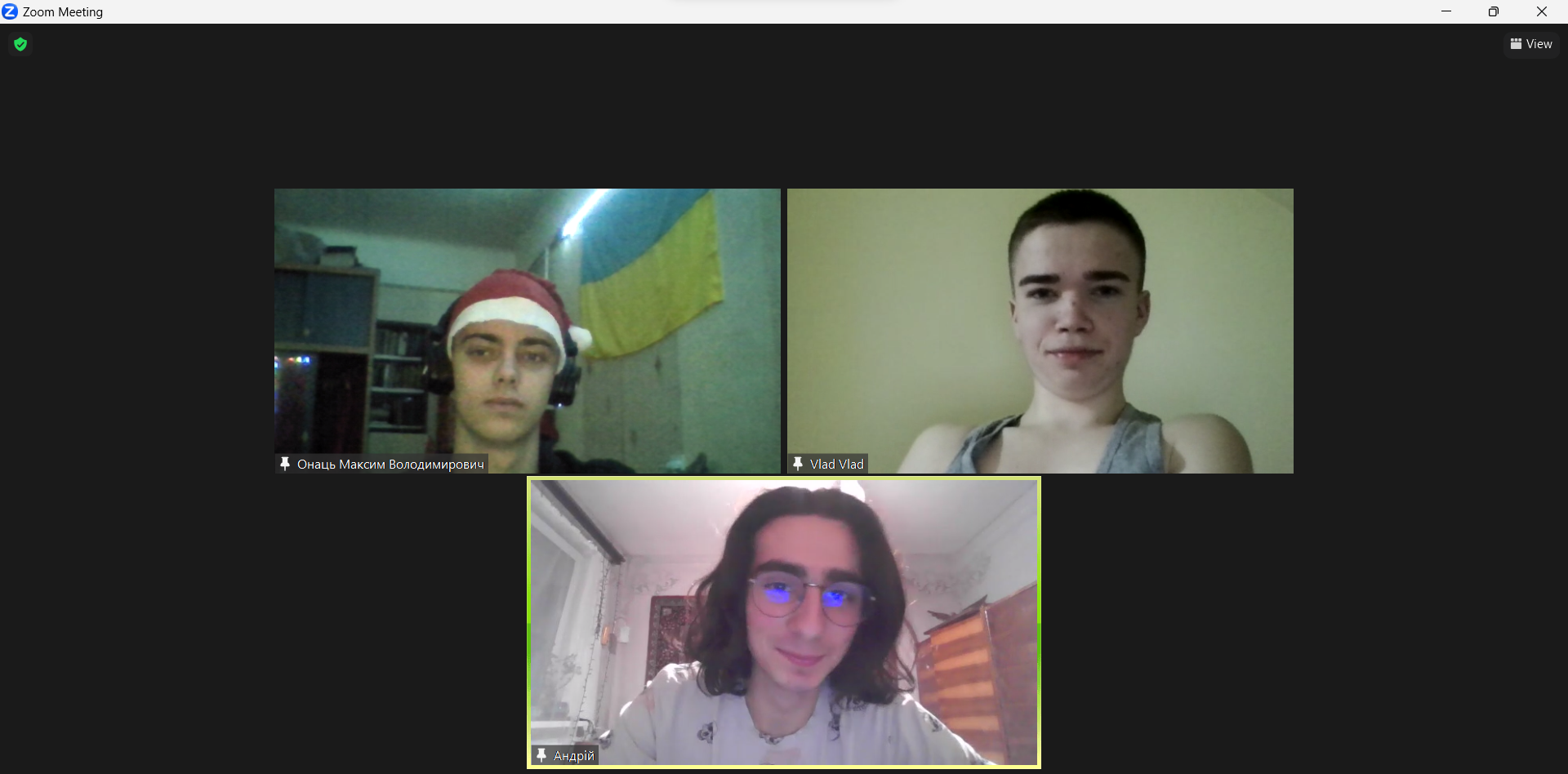
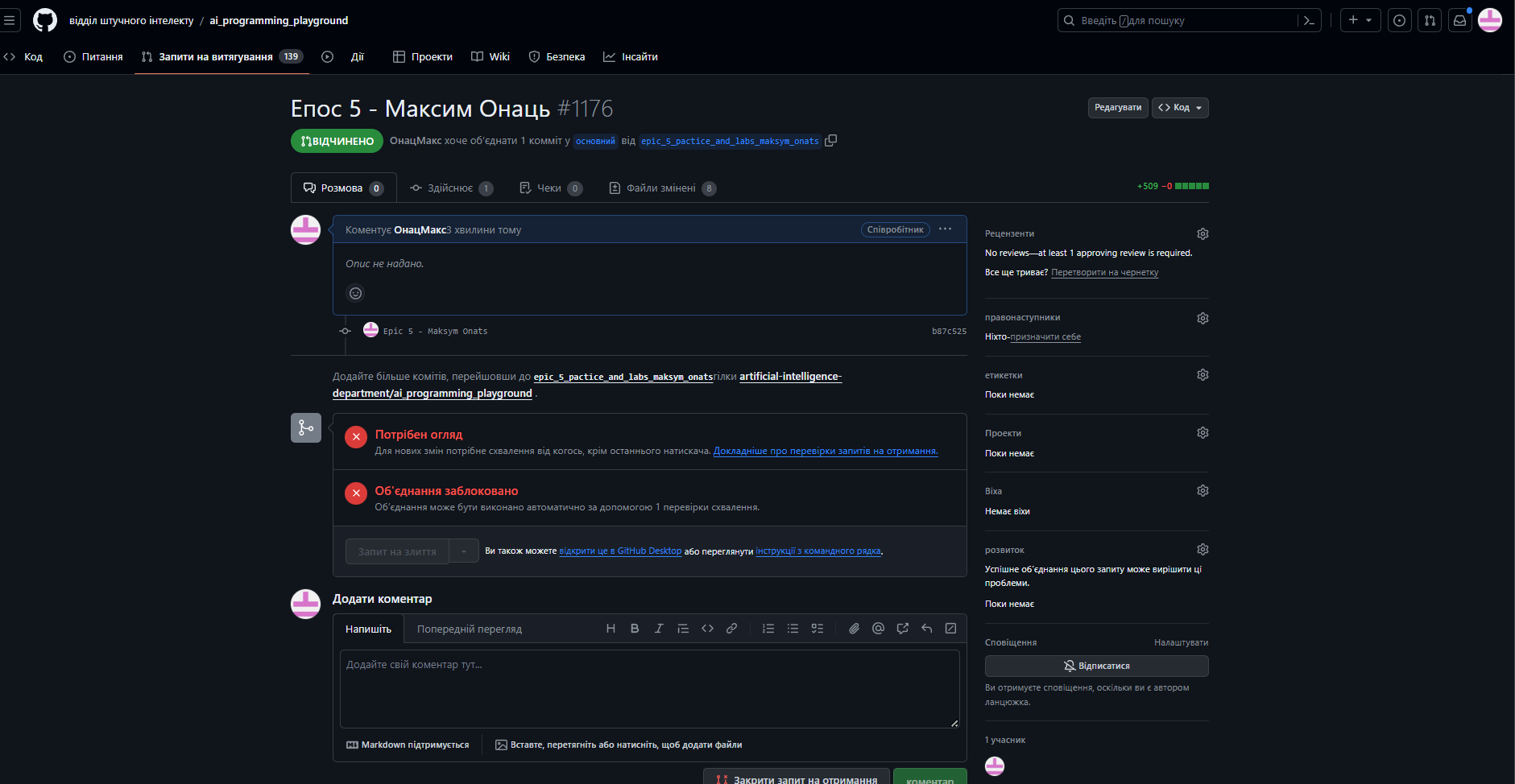


Рисунок 16. Результат виконання програми 8.

Час затрачений на виконання завдання – 40 хв

## **6. Кооперація з командою:**





# **Висновки:**

Протягом епіку 5 ознайомилася з роботою з файлами. Повторила системи числення. Вивчила різницю та особливості роботи з текстовими та бінарними файлами. Ознайомилася зі створенням та використанням бібліотек.