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-14

Грицеляк Маркіян Орестович

# **Тема роботи:**

Бінарні та текстові файли. Робота з файлами у С++. Символи та рядкові змінні. Зчитування з файлу та запис у файл. Створення й використання бібліотек

# **Мета роботи:**

Ознайомитися з файлами та з деталями роботи з ними у С++. Опрацювати зчитування та запис у файл. Ознайомитися з символами та рядковими змінним

# **Теоретичні відомості:**

* 1) Теоретичні відомості з переліком важливих тем:
* - Тема №1: Файли. Бінарні та текстові файли. Використання файлів. Робота з файлами.
* - Тема №2: Стандартна бібліотека Створення й використання бібліотек»
* - Тема №3: Символи та рядкові змінні.

1. Індивідуальний план опрацювання теорії:

* Індивідуальний план опрацювання теорії:
* Тема №1: Файли. Бінарні та текстові файли. Використання файлів. Робота з файлами.
  + Джерела Інформації:
    - Лекційний матеріал ВНС
    - <https://www.w3schools.in/cplusplus/working-with-files>
  + Що опрацьовано:
    - Бінарні та текстові файли і різниця між ними.Способи запису та читання у\з файл\у,копіювання та видалення елментів з файлу
  + Статус: Ознайомлений
  + Початок опрацювання теми: 01.11.2023
  + Звершення опрацювання теми: 20.12.2023
* Тема №2: Стандартна бібліотека Створення й використання бібліотек»
  + Джерела Інформації:
    - Лекційний матеріал ВНС
    - <https://acode.com.ua/urok-205-kontejnery-stl/>
  + Що опрацьовано:
    - Методи роботи з стандартною бібліотекою.Функції **sort, unique, rotate** ,модуль **vector** і тд.
    - Зрозумів наскільки класно можна спростити собі життя
  + Статус: Ознайомлений
  + Початок опрацювання теми: 01.11.2023
  + Звершення опрацювання теми: 20.12.2023
* Тема №3: Символи та рядкові змінні.
  + Джерела Інформації:
    - Лекційний матеріал ВНС
  + Що опрацьовано:
    - Зчитування та запис рядків
    - Функції **getline,strcmp,strtok,strle**n та ін.
  + Статус: Ознайомлений
  + Початок опрацювання теми: 01.11.2023
  + Звершення опрацювання теми: 20.12.2023

# **Виконання роботи:**

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

Lab# programming: VNS Lab 6

1. Надрукувати найдовше й найкоротше слово в цьому рядку.

2. Надрукувати всі слова, які не містять голосних букв.

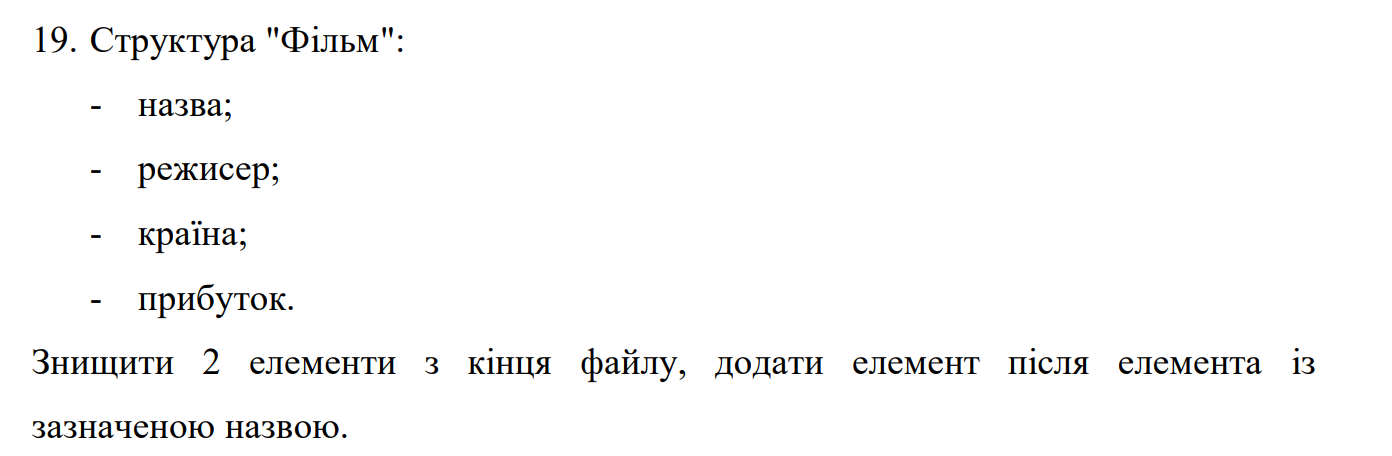
3. Надрукувати всі слова, які містять по одній цифрі.

4. Надрукувати всі слова, які співпадають з її першим словом.

5. Перетворити рядок таким чином, щоб спочатку в ньому були надруковані тільки букви, а потім тільки цифри, не міняючи порядку проходження символів у рядку.

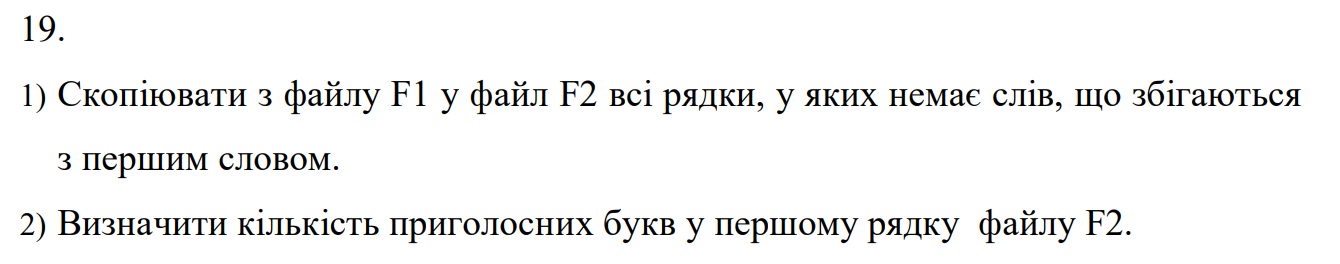
19. Знищити з рядка всі слова, які не є ідентифікаторами.

Lab# programming: VNS Lab 8



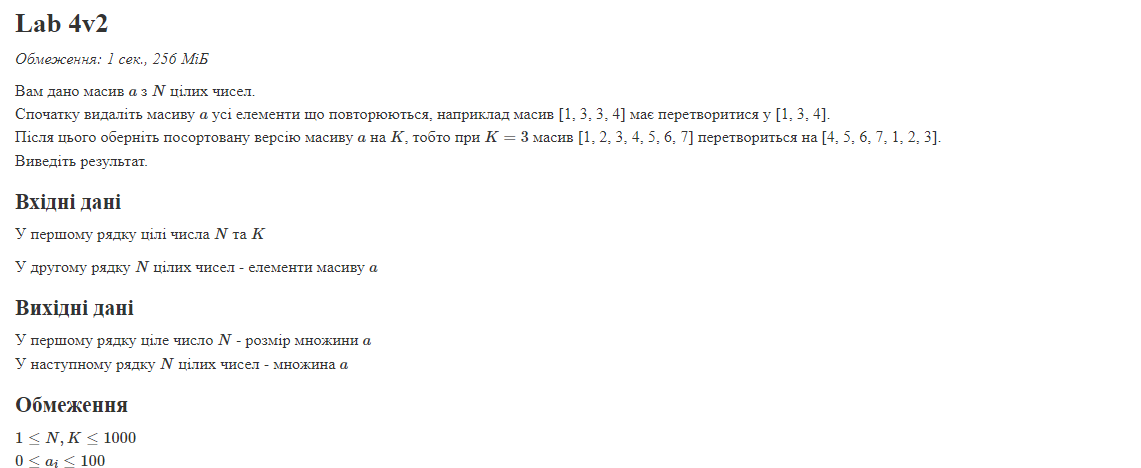
*Рисунок 2 Lab# programming: VNS Lab 8*

Lab# programming: VNS Lab 9



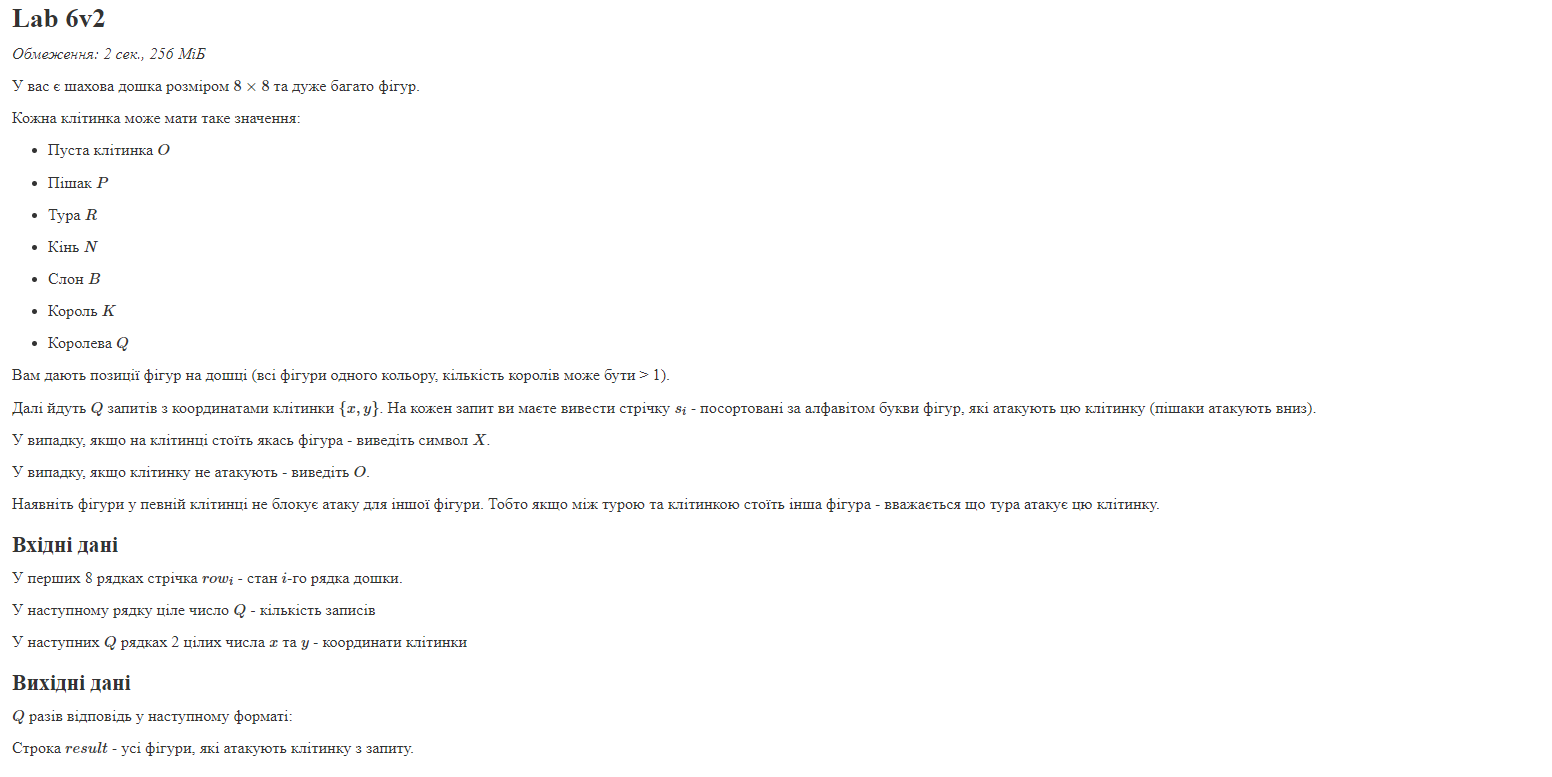
*Рисунок 3 Lab# programming: VNS Lab 9*

Lab# programming: Algotester Lab 4



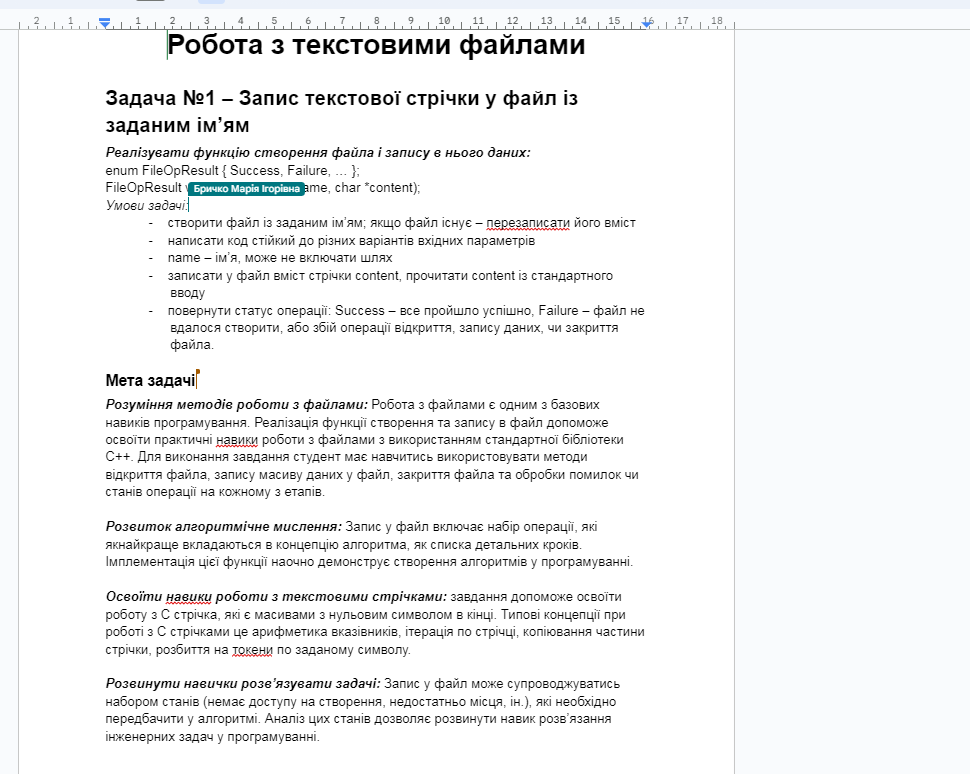
*Lab# programming: Algotester Lab 4*

Lab# programming: Algotester Lab 6

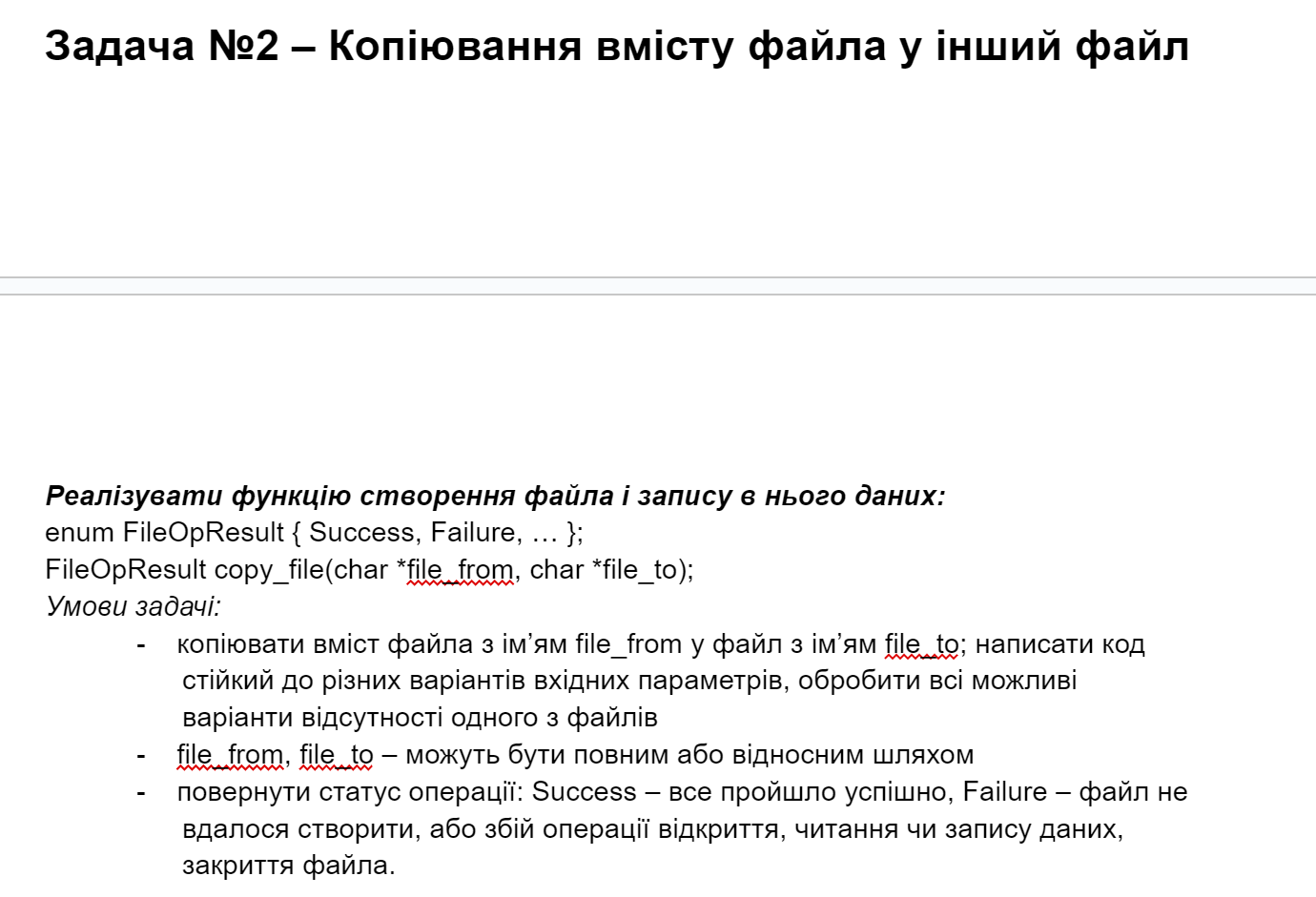


*Lab# programming: Algotester Lab 6*

Practice# programming: Class Practice Task 1-2

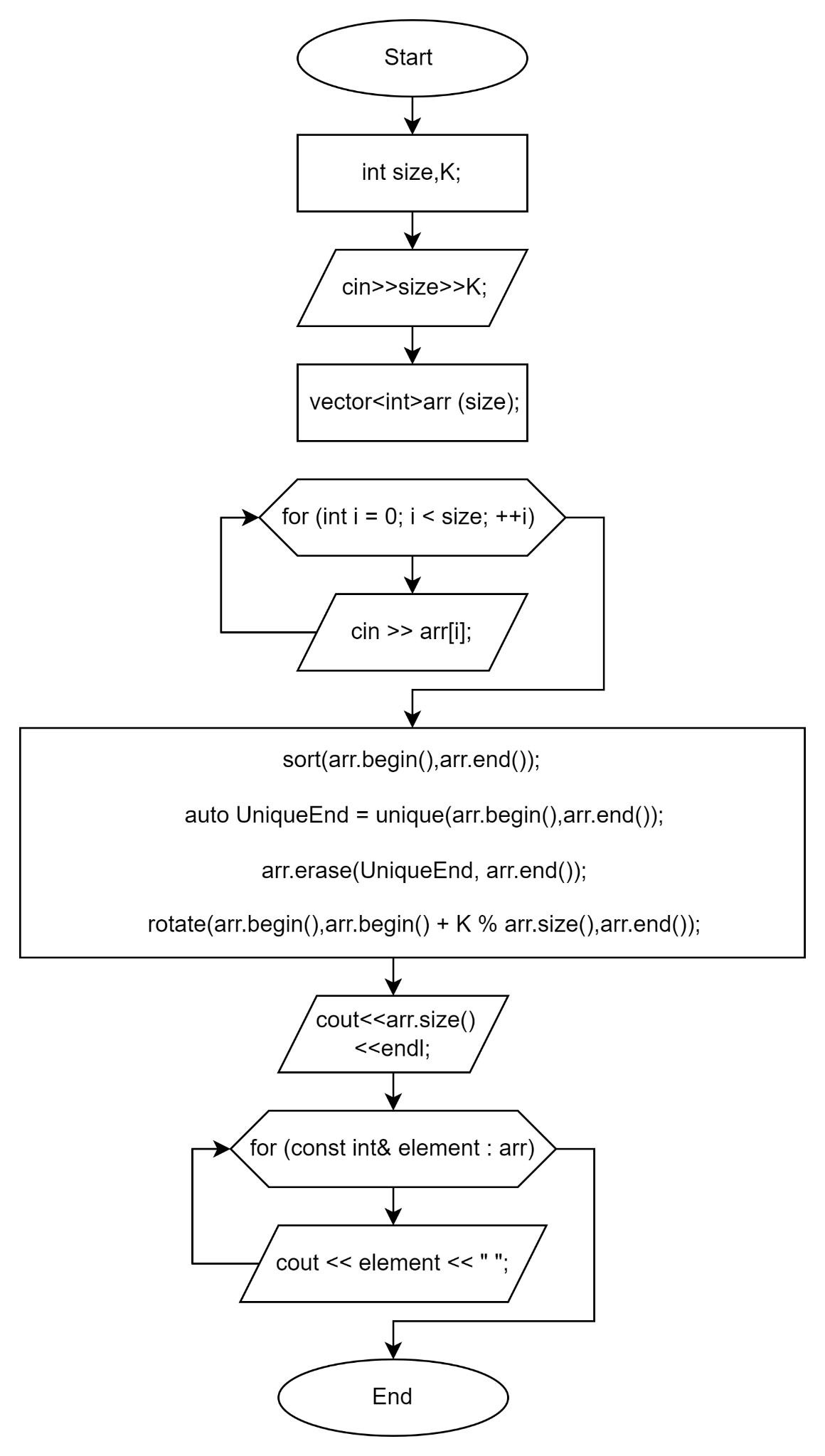


*Practice# programming: Class Practice Task 1*

**

*Practice# programming: Class Practice Task 2*

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



*Рисунок блок-схема до Algotester lab 4*

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

Середовище сконфігуровано з часу здачі епіку 1

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

Lab# programming: VNS Lab 6

#include <iostream>

#include <cstring>

#include <cctype>

#include <string>

const int MAX\_LENGTH = 255;

using namespace std;

bool isIdentifier(const char\* word) {

if (!isalpha(word[0]) && word[0] != '\_') {

return false;

}

for (int i = 1; i < strlen(word); ++i) {

if (!isalnum(word[i]) && word[i] != '\_') {

return false;

}

}

return true;

}

void delete\_non\_identifiers(char\* str) {

char\* token = strtok(str, " ");

string result;

while (token != nullptr) {

if (isIdentifier(token)) {

result += token;

result += ' ';

}

token = strtok(nullptr, " ");

}

if (!result.empty() && result.back() == ' ') {

result.pop\_back();

}

strcpy(str, result.c\_str());

}

int main() {

char str[MAX\_LENGTH];

cout << "Enter a string (up to 255 characters): ";

cin.getline(str, MAX\_LENGTH);

delete\_non\_identifiers(str);

cout << "Output string only with identifiers: " << str;

return 0;

}

Lab# programming: VNS Lab 8

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <algorithm>

using namespace std;

struct Film

{

string title;

string director;

string country;

double revenue;

};

void printFilm(const Film& film) {

cout << "Title: " << film.title << endl;

cout << "Director: " << film.director << endl;

cout << "Country: " << film.country << endl;

cout << "Revenue: " << film.revenue <<"billion $"<< endl;

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

}

void writeFilmToFile(const Film& film, const string& filename) {

ofstream file(filename, ios::binary | ios::app);

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

cerr << "Error opening file for writing!" << endl;

return;

}

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

file.close();

}

void readAndPrintFileContents(const string& filename) {

ifstream file(filename, ios::binary);

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

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

return;

}

Film film;

while (file.read(reinterpret\_cast<char\*>(&film), sizeof(Film))) {

printFilm(film);

}

file.close();

}

void removeLastTwoFilms(const string& filename) {

ifstream fileIn(filename, ios::binary);

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

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

return;

}

vector<Film> films;

Film film;

while (fileIn.read(reinterpret\_cast<char\*>(&film), sizeof(Film))) {

films.push\_back(film);

}

fileIn.close();

// Check if there are two films to remove

if (films.size() >= 2) {

// Remove the last two films

films.pop\_back();

films.pop\_back();

} else {

cerr << "Not enough films to remove!" << endl;

return;

}

// Write the remaining films back to the file

ofstream fileOut(filename, ios::binary | ios::trunc);

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

cerr << "Error opening file for writing!" << endl;

return;

}

for (const Film& f : films) {

fileOut.write(reinterpret\_cast<const char\*>(&f), sizeof(Film));

}

fileOut.close();

}

void addFilmAfterTitle(const string& filename, const string& targetTitle, const Film& newFilm) {

ifstream fileIn(filename, ios::binary);

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

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

return;

}

// Read all films into a vector

vector<Film> films;

Film film;

while (fileIn.read(reinterpret\_cast<char\*>(&film), sizeof(Film))) {

films.push\_back(film);

}

fileIn.close();

// Find the target film by title

auto it = find\_if(films.begin(), films.end(), [&targetTitle](const Film& f) {

return f.title == targetTitle;

});

if (it != films.end()) {

// Insert the new film after the target film

films.insert(it + 1, newFilm);

} else {

cerr << "Film with the specified title not found!" << endl;

return;

}

// Write the updated films back to the file

ofstream fileOut(filename, ios::binary | ios::trunc);

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

cerr << "Error opening file for writing!" << endl;

return;

}

for (const Film& f : films) {

fileOut.write(reinterpret\_cast<const char\*>(&f), sizeof(Film));

}

fileOut.close();

}

int main() {

const string filename = "C:/Users/Gil/OneDrive/Documents/GitHub/desktop-tutorial/Programming/Code/Epic 5/films.bin";

Film films[] = {

{"Oppenheimer", "Cristopher Nolan", "USA", 0.9},

{"Titanic", "James Cameron", "USA", 0.5},

{"Inglorious Basterds", "Quentin Tarantino", "France", 0.5},

{"Killers of the flower moon", "Martin Scorseze", "Usa", 0.4},

{"Napoleon", "Ridley Scott", "France", 0.4}

};

for (const Film& film : films) {

writeFilmToFile(film, filename);

}

cout << "Initial file contents:" << endl;

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

readAndPrintFileContents(filename);

return 0;

}

Lab# programming: VNS Lab 9

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <cctype>

using namespace std;

bool isVowel(char c) {

return (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' ||

c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U');

}

int countConsonants(const string& str) {

int consonantCount = 0;

for (char c : str) {

if (isalpha(c) && !isVowel(c)) {

consonantCount++;

}

}

return consonantCount;

}

void createAndWriteToFile(const string& filename, int lines) {

ofstream file(filename);

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

cerr << "Error opening file " << filename << " for writing!" << endl;

return;

}

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

string line;

cout << "Enter line " << (i + 1) << ": ";

getline(cin, line);

file << line << endl;

}

file.close();

}

void copyLinesWithoutMatchingWord(const string& inputFilename, const string& outputFilename) {

ifstream inputFile(inputFilename);

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

cerr << "Error opening file " << inputFilename << " for reading!" << endl;

return;

}

ofstream outputFile(outputFilename);

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

cerr << "Error opening file " << outputFilename << " for writing!" << endl;

inputFile.close();

return;

}

string wordToMatch;

cout << "Enter word to match: ";

cin >> wordToMatch;

string line;

while (getline(inputFile, line)) {

// Check if the line doesn't contain the specified word

if (line.find(wordToMatch) == string::npos) {

outputFile << line << endl;

}

}

inputFile.close();

outputFile.close();

}

int main() {

string filename1 = "File1.txt";

string filename2 = "File2.txt";

const string path = "C:/Users/Gil/OneDrive/Documents/GitHub/desktop-tutorial/Programming/Code/Epic 5/";

string file1 = path+filename1;

string file2 = path+filename2;

const int numLines = 10;

// Task 1: Create File1 and write 10 lines

createAndWriteToFile(file1, numLines);

// Task 2: Copy lines from File1 to File2 without matching word

copyLinesWithoutMatchingWord(file1, file2);

// Task 3: Count consonants in the first line of File2

ifstream file2Stream(file2);

if (!file2Stream.is\_open()) {

cerr << "Error opening file " << file2 << " for reading!" << endl;

return 1;

}

string firstLine;

getline(file2Stream, firstLine);

int consonantCount = countConsonants(firstLine);

cout << "Number of consonants in the first line of " << filename2 << ": " << consonantCount << endl;

file2Stream.close();

return 0;

}

Lab# programming: Algotester Lab 4

#include <iostream>

#include <algorithm>

#include <vector>

using namespace std;

int main()

{

int size,K;

cin>>size>>K;

vector<int>arr (size);

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

{

cin >> arr[i];

}

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

auto UniqueEnd = unique(arr.begin(),arr.end());

arr.erase(UniqueEnd, arr.end());

rotate(arr.begin(),arr.begin() + K % arr.size(),arr.end());

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

for (const int& element : arr)

{

cout << element << " ";

}

return 0;

}

Lab# programming: Algotester Lab 6

#include <iostream>

#include <vector>

#include <algorithm>

#include <set>

#include <string>

using namespace std;

vector<char> get\_attacking\_figures(vector<string>& board, int x, int y) {

set<char> attacking\_figures;

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

return {'X'};

}

else {

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

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

char figure = board[i][j];

if (figure != 'O') {

if (figure == 'P' && (abs(i - x == -1 && abs(j - y) == 1))) {

attacking\_figures.insert(figure);

}

else if (figure == 'R' && (i == x || j == y)) {

attacking\_figures.insert(figure);

}

else if (figure == 'N' && ((abs(i - x) == 2 && abs(j - y) == 1) || (abs(i - x) == 1 && abs(j - y) == 2))) {

attacking\_figures.insert(figure);

}

else if (figure == 'B' && abs(i - x) == abs(j - y) && board[x][y]!='B' ) {

attacking\_figures.insert(figure);

}

else if (figure == 'K' && abs(i - x) <= 1 && abs(j - y) <= 1 && (abs(i - x) != 0 || abs(j - y) != 0)) {

attacking\_figures.insert(figure);

}

else if (figure == 'Q' && ((i == x || j == y || abs(i - x) == abs(j - y)) && board[x][y]!='Q')) {

attacking\_figures.insert(figure);

}

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

attacking\_figures.insert('O');

}

}

}

}

if(attacking\_figures.size()>1) {

attacking\_figures.erase('O');

}

vector<char> result(attacking\_figures.begin(), attacking\_figures.end());

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

return result;

}

int main() {

vector<string> board(9);

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

cin >> board[i];

}

int Q;

cin >> Q;

int count=(Q\*2)-1;

int arr[count];

int x, y;

for (int q=0; q < count; q+=2) {

cin >> x >> y;

arr[q-1]=x;

arr[q]=y;

vector<char> result = get\_attacking\_figures(board, arr[q-1]-1, arr[q]-1);

for (char figure : result) {

cout << figure;

}

cout << endl;

}

return 0;

}

Lab# programming: Class practice task 1-2

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

enum FileOpResult { Success, Failure };

FileOpResult write\_to\_file(const string& name, const string& content)

{

ofstream file(name);

if (!file.is\_open())

{

cerr << "Error opening file for writing: " << name << endl;

return FileOpResult::Failure;

}

file << content;

if (file.fail())

{

cerr << "Error writing content to file: " << name << endl;

file.close();

return FileOpResult::Failure;

}

file.close();

return FileOpResult::Success;

}

FileOpResult copy\_from\_file(const string& source, const string& destination) {

ifstream sourceFile(source, ios::binary);

ofstream destFile(destination, ios::binary);

if (!sourceFile.is\_open())

{

cerr << "Error opening source file for reading: " << source << endl;

return FileOpResult::Failure;

}

if (!destFile.is\_open())

{

cerr << "Error opening destination file for writing: " << destination << endl;

sourceFile.close();

return FileOpResult::Failure;

}

destFile << sourceFile.rdbuf();

if (destFile.fail())

{

cerr << "Error copying content from source to destination" << endl;

sourceFile.close();

destFile.close();

return FileOpResult::Failure;

}

sourceFile.close();

destFile.close();

return FileOpResult::Success;

}

int main() {

string filename, content;

cout << "Enter the filename: ";

cin >> filename;

string relativePath = "C:/Users/Gil/OneDrive/Documents/GitHub/desktop-tutorial/Programming/Code/Epic 5/" + filename;

cout << "Enter the content: ";

cin.ignore();

getline(cin, content);

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

if (result == FileOpResult::Success)

{

cout << "File write successful!" << endl;

string destinationPath = "C:/Users/Gil/OneDrive/Documents/GitHub/desktop-tutorial/Programming/Code/Epic 5/copy\_" + filename;

FileOpResult copyResult = copy\_from\_file(relativePath, destinationPath);

if (copyResult == FileOpResult::Success)

{

cout << "File copy successful!" << endl;

}

else

{

cout << "File copy failed!" << endl;

}

}

else

{

cout << "File write failed!" << endl;

}

return 0;

}

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

Lab# programming: VNS Lab 6

IEnter a string (up to 255 characters): Good Afternoon! My name is Gustavo Fring and welcome to the Los Pollos Hermanos! We serve only exceptional food! there 4ers 6toid

Output string only with identifiers: Good My name is Gustavo Fring and welcome to the Los Pollos We serve only exceptional there

Затрачений час:2 години

Lab# programming: VNS Lab 8

Initial file contents:

-----------------------------

Title: Oppenheimer

Director: Cristopher Nolan

Country: USA

Revenue: 0.9billion $

-----------------------------

Title: Titanic

Director: James Cameron

Country: USA

Revenue: 0.5billion $

-----------------------------

Title: Inglorious Basterds

Director: Quentin Tarantino

Country: France

Revenue: 0.5billion $

-----------------------------

Title: Killers of the flower moon

Director: Martin Scorseze

Country: Usa

Revenue: 0.4billion $

-----------------------------

Title: Napoleon

Director: Ridley Scott

Country: France

Revenue: 0.4billion $

-----------------------------



Затрачений час: дофіга годин

Lab# programming: VNS Lab 9

Enter line 1: Today is sunny

Enter line 2: we play football

Enter line 3: it's time to cook

Enter line 4: let's watch a film

Enter line 5: Breaking Bad

Enter line 6: we need to cook

Enter line 7: captain cook

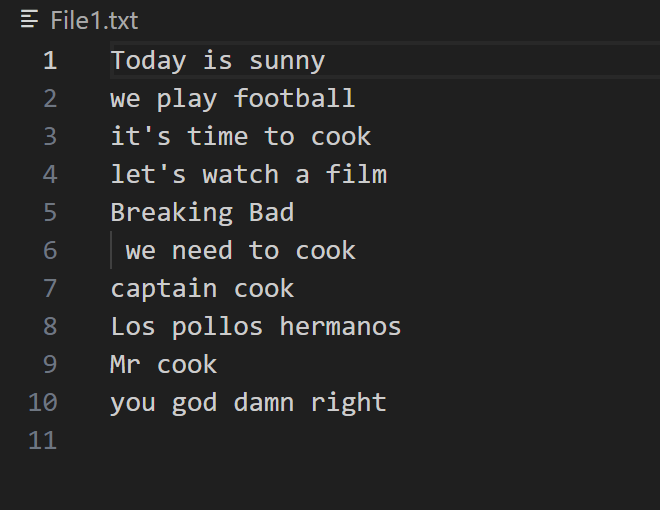
Enter line 8: Los pollos hermanos

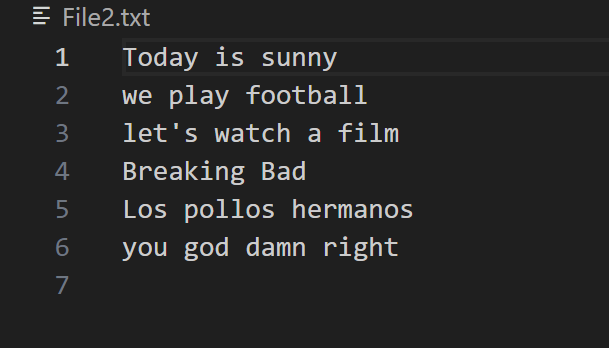
Enter line 9: Mr cook

Enter line 10: you god damn right

Enter word to match: cook

Number of consonants in the first line of C:/Users/Gil/OneDrive/Documents/GitHub/desktop-tutorial/Programming/Code/Epic 5/File2.txt: 8





Затрачений час 3 години

Lab# programming: Algotester Lab 4

7 3

1 3 5 7 7 4 4

5

5 7 1 3 4



*Lab# programming: Algotester Lab 4*

Затрачений час 2 години

Lab# programming: Algotester Lab 6

OOOOOOOO

OROOOOOO

OONOOOOO

OOOOPOOO

OOOOOOOO

OOOOOOOO

KOQOOOOO

OOOOOOOR

7

8 1

KR

1 2

NR

5 4

NP

5 1

Q

6 2

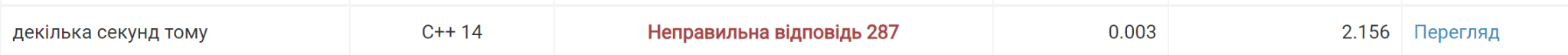
KQR

8 4

QR

6 7

O



*Lab# programming: Algotester Lab 6*

Затрачений час: супер дофіга і не зайшло на алго:(

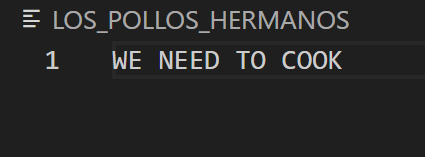
Practice# programming: Class Practice Task 1-2

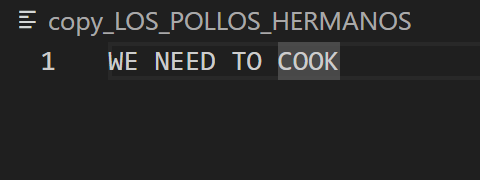
Enter the filename: LOS\_POLLOS\_HERMANOS

Enter the content: WE NEED TO COOK

File write successful!

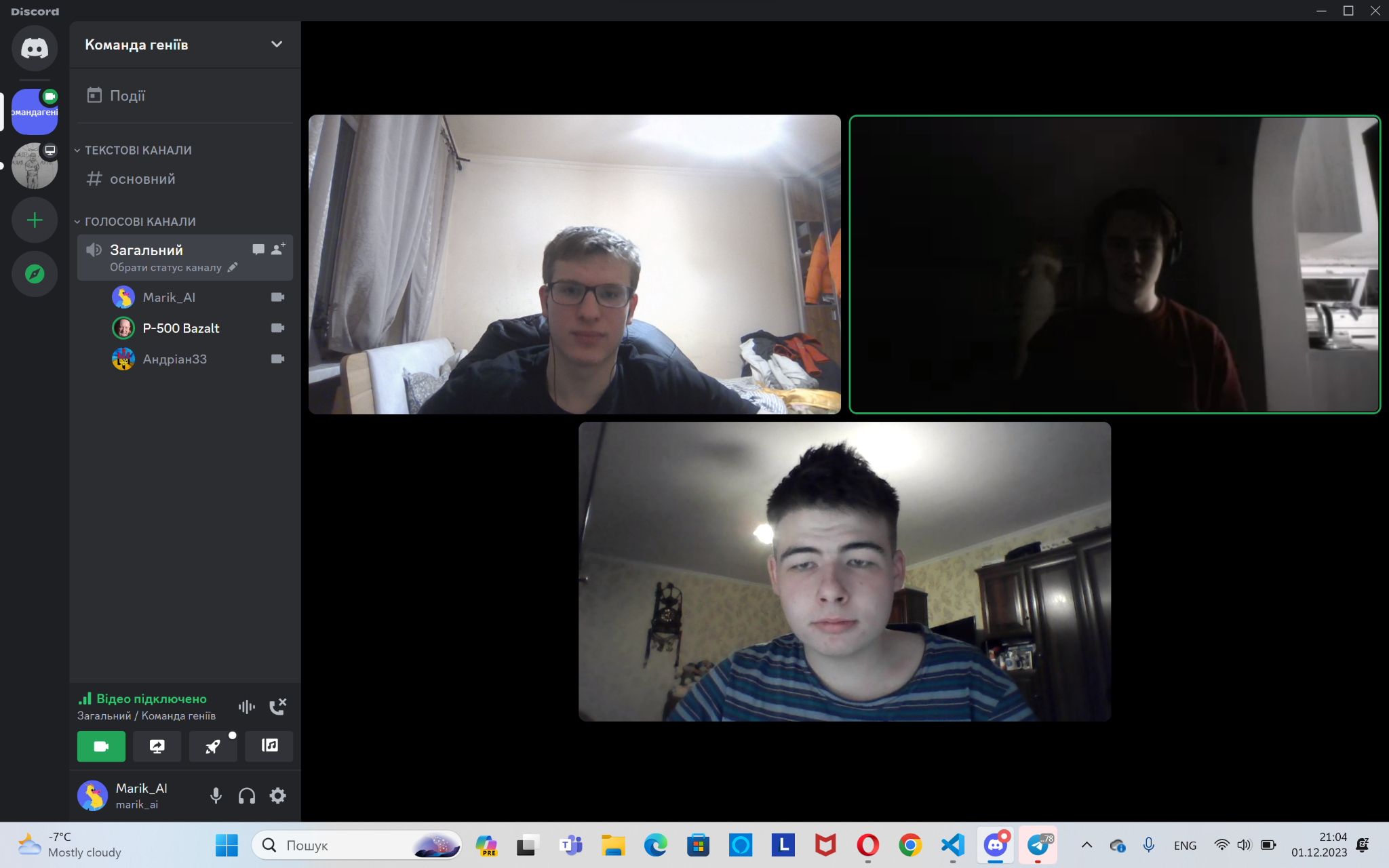
File copy successful!





Затрачений час 2 години

## **6. Кооперація з командою:**



*зустріч з командою*

# **Висновки:**

Ознайомився з файлами та з деталями роботи з ними у С++. Опрацював зчитування та запис у файл. Ознайомився з символами та рядковими змінним , ітераторами .