Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

**Звіт**

**про виконання лабораторних та практичних робіт блоку № 5**

На тему:  «Файли. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами. Створення й використання бібліотек.»

***з дисципліни:*** «Основи програмування»

до:

ВНС Лабораторної Роботи № 6

ВНС Лабораторної Роботи № 8

ВНС Лабораторної Роботи № 9

Алготестер Лабораторної Роботи №4

Алготестер Лабораторної Роботи №6

Практичних Робіт до блоку №5

**Виконала:**

Студентка групи ШІ-12

Ляшко Леся Ігорівна

Львів 2024

**Тема роботи:** Файли. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами. Створення й використання бібліотек.

**Мета роботи:** Ознайомитися з різними аспектами роботи з файлами у C++. Також метою було розібратися з роботою з char та string, стрічковими літералами, екрануванням символів, порівнянням та пошуком у рядках. Важливо було зрозуміти особливості читання та запису текстових файлів, парсинг текстових файлів, форматування тексту при записі, обробку помилок, а також відмінності

між бінарними та текстовими файлами. Також передбачалося ознайомитися з оглядом

стандартної бібліотеки для роботи з файлами та потоками вводу/виводу. Нарешті, метою

було ознайомитися зі створенням власних бібліотек у C++, їх документуванням та

використанням сторонніх бібліотек у проектах.

**Теоретичні відомості:**

1. **Теоретичні відомості з переліком важливих тем:**

* **Тема №1: Practice# programming: Class Practice Task.**
* **Тема №2: Lab# programming:VNS Lab6 Task .**
* **Тема №3: Lab# programming: VNS Lab 8 Task.**
* **Тема №4: Lab# programming: Algotester Lab 4 Variant3.**
* **Тема №5: Lab# programming: Algotester Lab 6 Variant1.**
* **Тема №6: Practice# programming:  Self Practice Task.**
* **Тема №7: Lab# programming: VNS Lab 9 Task.**

**2) Індивідуальний план опрацювання теорії:**

* **Тема №1: Practice# programming: Class Practice Task.**

*Ознайомлена та опрацьовано.*

*Опрацьовувала протягом: 19.11.2024*

**с++ files:** <https://www.w3schools.com/cpp/cpp_files.asp>

<https://acode.com.ua/urok-220-bazovyj-fajlovyj-vvid-i-vyvid/>

<https://www.learncpp.com/cpp-tutorial/basic-file-io/>

* **Тема №2: Lab# programming:VNS Lab6 Task .**

*Ознайомлена та опрацьовано.*

*Опрацьовувала протягом: 19.11.2024*

**Бібліотеки:** <https://www.w3schools.com/cpp/cpp_ref_cstring.asp>

<https://en.cppreference.com/w/cpp/header/cctype>

**C-style string:** <https://www.w3schools.com/cpp/cpp_strings_cstyle.asp>

<https://w3schoolsua.github.io/c/c_strings.html#gsc.tab=0>

* **Тема №3: Lab# programming: VNS Lab 8 Task.**

*Ознайомлена та опрацьовано.*

*Опрацьовувала протягом: 20.11.2024*

<https://www.geeksforgeeks.org/vector-insert-function-in-cpp-stl/>

* **Тема №4: Lab# programming: Algotester Lab 4 Variant3.**

*Ознайомлена та опрацьовано.*

*Опрацьовувала протягом: 20.11.2024*

**Stl:** <https://www.geeksforgeeks.org/the-c-standard-template-library-stl/>

* **Тема №6: Practice# programming:  Self Practice Task.**

*Ознайомлена та опрацьовано.*

*Опрацьовувала протягом: 21.11.2024*

***Бібліотеки:*** *<https://acode.com.ua/statychni-i-dynamichni-biblioteky/>*

***Як створити бібліотеку:*** *<https://snih.zapisi.cx.ua/ukraincyam/yak-stvoriti-svoyu-biblioteku-v-s-pokrokove-kerivnictvo.html>*

* **Тема №7: Lab# programming: VNS Lab 9 Task.**

*Ознайомлена та опрацьовано.*

*Опрацьовувала протягом: 21.11.2024*

*<https://www.programiz.com/cpp-programming/library-function/cctype/tolower>*

**Практичне виконання**

**1.Class Practice Task.**

*Очікуваний час на виконання: 1 год 30 хв.*

*Реальність: 2 год.*

#include <iostream>

#include <fstream>

#include <string>

#include <cstring>

using namespace std;

enum FileOpResult1 { Failure1, Success1 };

FileOpResult1 const write\_to\_file(const char \*name,const char \*content){

fstream file(name, fstream::out);

if(!file) return Failure1;

file.write(content, strlen(content));

file.close();

return Success1;

}

enum FileOpResult2 { Success2, Failure2};

FileOpResult2 const copy\_file(const char \*file\_from, const char \*file\_to){

fstream file1(file\_from, fstream::in);

if(!file1) return Failure2;

fstream file2(file\_to, fstream::out);

if(!file2) return Failure2;

string line;

while(getline(file1, line)){

line += "\n";

file2.write(line.c\_str(), line.size());

}

file1.close();

file2.close();

return Success2;

}

int main(){

string name, content;

cout << "Enter file name : ";

getline(cin, name);

name += ".txt";

cout << "Enter file content: ";

getline(cin, content);

FileOpResult1 result1 = write\_to\_file(name.c\_str(), content.c\_str());

if(result1 == Success1) cout << " success\n";

else cout << "Failure";

string file1, file2;

cout << "Enter name of file to copy from: ";

getline(cin, file1);

cout << "Enter name of file in which to copy: ";

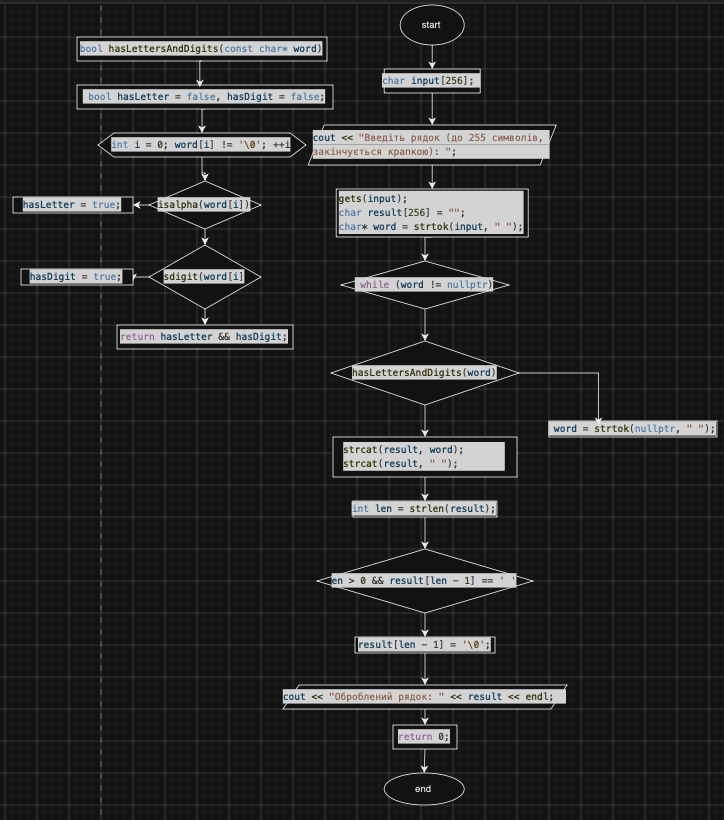
getline(cin, file2);

file1 += ".txt";

file2 += ".txt";

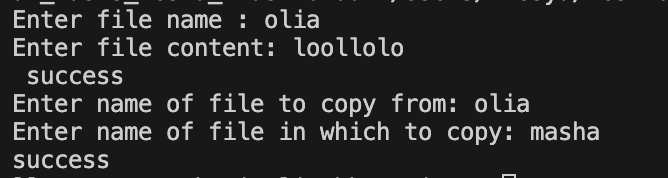
FileOpResult2 result2 = copy\_file(file1.c\_str(), file2.c\_str());

if(result2 == Success2) cout << "success\n";

else cout << "Failure";

return 0;

}



**2.VNS Lab6 Task .**

*Очікуваний час на виконання: 1 год 30 хв.*

*Реальність: 1 год 30 хв.*

#include <iostream>

#include <cstring>

#include <cctype>

using namespace std;

bool hasLettersAndDigits(const char\* word) {

bool hasLetter = false, hasDigit = false;

for (int i = 0; word[i] != '\0'; ++i) {

if (isalpha(word[i])) hasLetter = true;

if (isdigit(word[i])) hasDigit = true;

}

return hasLetter && hasDigit;

}

int main() {

char input[256];

cout << "Введіть рядок (до 255 символів, закінчується крапкою): ";

gets(input);

char result[256] = "";

char\* word = strtok(input, " ");

while (word != nullptr) {

if (hasLettersAndDigits(word)) {

strcat(result, word);

strcat(result, " ");

}

word = strtok(nullptr, " ");

}

int len = strlen(result);

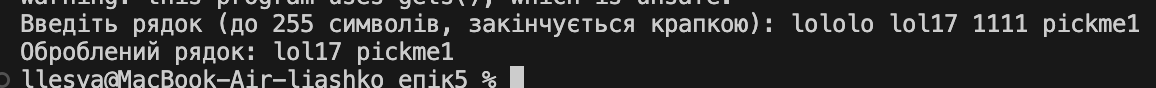
if (len > 0 && result[len - 1] == ' ') {

result[len - 1] = '\0';

}

cout << "Оброблений рядок: " << result << endl;

return 0;

}

**3. VNS Lab 8 Task.**

*Очікуваний час на виконання: 1 год 30 хв.*

*Реальність: 1 год.*

#include <iostream>

#include <fstream>

#include <vector>

#include <cstring>

using namespace std;

struct Stadium {

char name[50]; // Назва

char address[100]; // Адреса

int capacity; // Місткість

char sports[50]; // Види спорту

};

void writeToFile(const char\* filename, const vector<Stadium>& stadiums) {

ofstream file(filename, ios::binary);

if (!file) {

cout << "Помилка запису у файл!\n";

return;

}

for (const auto& stadium : stadiums) {

file.write((char\*)&stadium, sizeof(Stadium));

}

file.close();

}

void printFromFile(const char\* filename) {

ifstream file(filename, ios::binary);

if (!file) {

cout << "Помилка читання файлу!\n";

return;

}

Stadium stadium;

while (file.read((char\*)&stadium, sizeof(Stadium))) {

cout << "Назва: " << stadium.name

<< ", Адреса: " << stadium.address

<< ", Місткість: " << stadium.capacity

<< ", Види спорту: " << stadium.sports << endl;

}

file.close();

}

void deleteFromFile(const char\* filename, const char\* nameToDelete) {

ifstream file(filename, ios::binary);

if (!file) {

cout << "Помилка читання файлу!\n";

return;

}

vector<Stadium> stadiums;

Stadium stadium;

while (file.read((char\*)&stadium, sizeof(Stadium))) {

if (strcmp(stadium.name, nameToDelete) != 0) {

stadiums.push\_back(stadium);

}

}

file.close();

writeToFile(filename, stadiums);

}

void addToFileAfter(const char\* filename, int index, const Stadium& s1, const Stadium& s2) {

ifstream file(filename, ios::binary);

if (!file) {

cout << "Помилка читання файлу!\n";

return;

}

vector<Stadium> stadiums;

Stadium stadium;

while (file.read((char\*)&stadium, sizeof(Stadium))) {

stadiums.push\_back(stadium);

}

file.close();

if (index >= 0 && index < stadiums.size()) {

stadiums.insert(stadiums.begin() + index + 1, s1);

stadiums.insert(stadiums.begin() + index + 2, s2);

writeToFile(filename, stadiums);

} else {

cout << "Невірний номер елемента!\n";

}

}

Stadium inputStadium() {

Stadium stadium;

cin.ignore();

cout << "Введіть назву: ";

cin.getline(stadium.name, 50);

cout << "Введіть адресу: ";

cin.getline(stadium.address, 100);

cout << "Введіть місткість: ";

cin >> stadium.capacity;

cin.ignore();

cout << "Введіть вид спорту: ";

cin.getline(stadium.sports, 50);

return stadium;

}

int main() {

const char\* filename = "stadiums.bin";

vector<Stadium> stadiums;

int n;

cout << "Скільки стадіонів додати? ";

cin >> n;

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

cout << "Стадіон " << i + 1 << ":\n";

stadiums.push\_back(inputStadium());

}

writeToFile(filename, stadiums);

cout << "\nПоточний список стадіонів:\n";

printFromFile(filename);

cout << "\nВведіть назву стадіону для видалення: ";

char nameToDelete[50];

cin.ignore();

cin.getline(nameToDelete, 50);

deleteFromFile(filename, nameToDelete);

cout << "\nСписок після видалення:\n";

printFromFile(filename);

cout << "\nВведіть номер стадіону, після якого додати нові: ";

int index;

cin >> index;

cout << "Перший новий стадіон:\n";

Stadium newStadium1 = inputStadium();

cout << "Другий новий стадіон:\n";

Stadium newStadium2 = inputStadium();

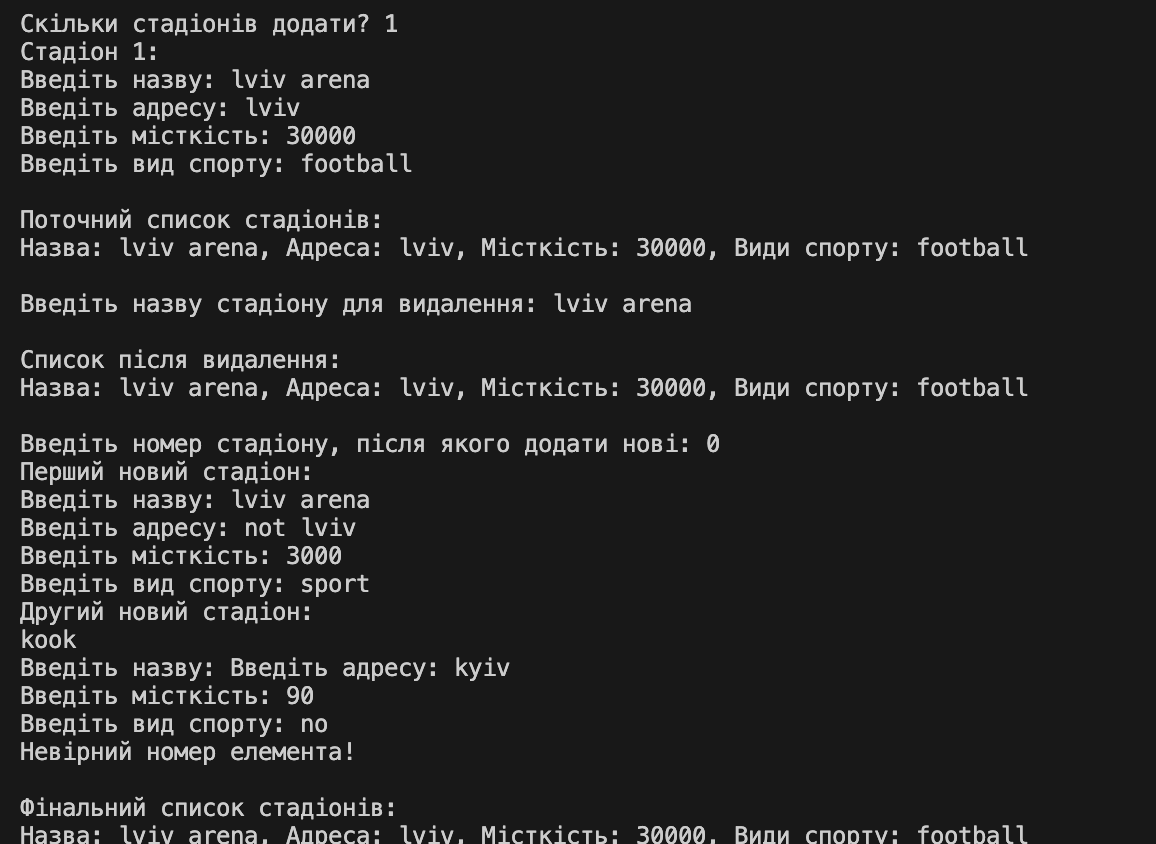
addToFileAfter(filename, index - 1, newStadium1, newStadium2);

cout << "\nФінальний список стадіонів:\n";

printFromFile(filename);

return 0;

}



**4.Algotester Lab 4 Variant3.**

*Очікуваний час на виконання: 1 год 30 хв.*

*Реальність: 45 хв.*

#include <iostream>

#include <algorithm>

#include <vector>

#include <unordered\_set>

using namespace std;

vector<int> difference(const vector<int>& a, const vector<int>& b) {

unordered\_set<int> b\_elements(b.begin(), b.end());

vector<int> result;

for (int num : a) {

if (b\_elements.find(num) == b\_elements.end()) {

result.push\_back(num);

}

}

return result;

}

vector<int> intersection(const vector<int>& a, const vector<int>& b) {

unordered\_set<int> a\_elements(a.begin(), a.end());

vector<int> result;

for (int num : b) {

if (a\_elements.find(num) != a\_elements.end()) {

result.push\_back(num);

}

}

return result;

}

vector<int> unionSets(const vector<int>& a, const vector<int>& b) {

unordered\_set<int> elements(a.begin(), a.end());

elements.insert(b.begin(), b.end());

vector<int> result(elements.begin(), elements.end());

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

return result;

}

vector<int> symmetricDifference(const vector<int>& a, const vector<int>& b) {

unordered\_set<int> a\_elements(a.begin(), a.end());

unordered\_set<int> b\_elements(b.begin(), b.end());

vector<int> result;

for (int num : a) {

if (b\_elements.find(num) == b\_elements.end()) {

result.push\_back(num);

}

}

for (int num : b) {

if (a\_elements.find(num) == a\_elements.end()) {

result.push\_back(num);

}

}

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

return result;

}

int main() {

int N, M;

cin >> N;

vector<int> a(N);

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

cin >> a[i];

}

cin >> M;

vector<int> b(M);

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

cin >> b[i];

}

vector<int> diff\_a\_b = difference(a, b);

vector<int> diff\_b\_a = difference(b, a);

vector<int> intersect = intersection(a, b);

vector<int> union\_ab = unionSets(a, b);

vector<int> sym\_diff = symmetricDifference(a, b);

cout << diff\_a\_b.size() << endl;

for (int num : diff\_a\_b) {

cout << num << " ";

}

cout << endl;

cout << diff\_b\_a.size() << endl;

for (int num : diff\_b\_a) {

cout << num << " ";

}

cout << endl;

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

for (int num : intersect) {

cout << num << " ";

}

cout << endl;

cout << union\_ab.size() << endl;

for (int num : union\_ab) {

cout << num << " ";

}

cout << endl;

cout << sym\_diff.size() << endl;

for (int num : sym\_diff) {

cout << num << " ";

}

cout << endl;

return 0;

}



*Очікуваний час на виконання: 1 год 30 хв.*

*Реальність: 1 год.*

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main()

{

int n;

std::cin >> n;

vector<int> arr(n);

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

{

cin >> arr[i];

}

auto mod0\_end = partition(arr.begin(), arr.end(), [](int x) { return x % 3 == 0; });

auto mod1\_end = partition(mod0\_end, arr.end(), [](int x) { return x % 3 == 1; });

auto mod2\_end = arr.end();

// Сортування

sort(arr.begin(), mod0\_end); // ост 0 за зр

sort(mod0\_end, mod1\_end, greater<int>()); // ост 1 за сп

sort(mod1\_end, mod2\_end); // ост2 за зр

// Видалення повторень

auto unique\_end = unique(arr.begin(), arr.end());

int resultSize = distance(arr.begin(), unique\_end);

cout << resultSize << "\n";

for (auto it = arr.begin(); it != unique\_end; ++it)

{

cout << \*it << " ";

}

cout << "\n";

return 0;

}

**5.Algotester Lab 6 Variant1.**

*Очікуваний час на виконання: 1 год 30 хв.*

*Реальність: 1 год 30 хв.*

#include <iostream>

#include <vector>

#include <unordered\_map>

#include <set>

#include <algorithm>

using namespace std;

int main() {

int N, K;

cin >> N >> K;

cin.ignore();

unordered\_map<string, int> wordCount;

set<char> resultSet

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

string word;

getline(cin, word);

transform(word.begin(), word.end(), word.begin(), ::tolower);

wordCount[word]++;

}

for (const auto &entry : wordCount) {

if (entry.second >= K) {

for (char c : entry.first) {

resultSet.insert(c);

}

}

}

if (resultSet.empty()) {

cout << "Empty!" << endl;

} else {

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

for (auto it = resultSet.rbegin(); it != resultSet.rend(); ++it) {

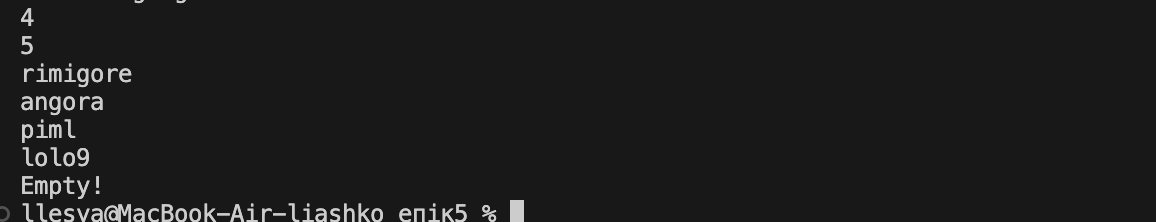
cout << \*it << " ";

}

cout << endl;

}

return 0;

}

**6. Self Practice Task.**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

void createInputFile(const string& inputFile) {

ofstream inFile(inputFile);

if (!inFile) {

cout << "Не вдалося створити вхідний файл!" << endl;

return;

}

inFile << "# Це коментар" << endl;

inFile << "Це перший рядок" << endl;

inFile << "Другий рядок" << endl;

inFile << endl;

inFile << "# Ще один коментар" << endl;

inFile << "Третій рядок" << endl;

inFile << "Четвертий рядок" << endl;

inFile.close();

}

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

ifstream inFile(inputFile);

ofstream outFile(outputFile);

if (!inFile) {

cout << "Не вдалося відкрити вхідний файл: " << inputFile << endl;

return;

}

if (!outFile) {

cout << "Не вдалося створити вихідний файл: " << outputFile << endl;

return;

}

string line;

int removedLines = 0;

int totalLines = 0;

while (getline(inFile, line)) {

totalLines++;

if (line.empty() || line[0] == '#') {

removedLines++;

continue;

}

outFile << line << endl;

}

inFile.close();

outFile.close();

cout << "Загальна кількість рядків: " << totalLines << endl;

cout << "Кількість видалених рядків: " << removedLines << endl;

cout << "\nТекст у файлі " << outputFile << ":\n";

ifstream resultFile(outputFile);

while (getline(resultFile, line)) {

cout << line << endl;

}

resultFile.close();

}

int main() {

string inputFile = "input.txt";

string outputFile = "output.txt";

cout << "Створюємо вхідний файл " << inputFile << "...\n";

createInputFile(inputFile);

processFile(inputFile, outputFile);

return 0;

}

**7. VNS Lab 9 Task.**

*Очікуваний час на виконання: 1 год 30 хв.*

*Реальність: 1 год.*

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <cctype>

using namespace std;

void createFileF1(const string& filename) {

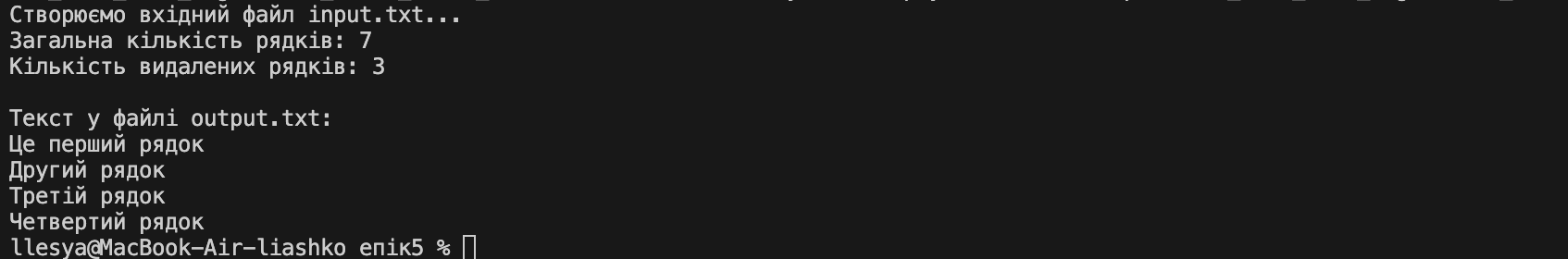
ofstream file(filename);

if (!file) {

cout << "Не вдалося створити файл F1!\n";

return;

}

 cout << "Введіть 10 або більше рядків тексту (для завершення введіть \"END\"):\n";

string line;

int count = 0;

while (getline(cin, line)) {

if (line == "END") break;

file << line << endl;

count++;

}

if (count < 10) {

cout << "Ви ввели менше 10 рядків! Спробуйте додати більше наступного разу.\n";

}

file.close();

}

void copyLines(const string& fileF1, const string& fileF2, int N1, int N2) {

ifstream file1(fileF1);

ofstream file2(fileF2);

if (!file1 || !file2) {

cout << "Не вдалося відкрити файли!\n";

return;

}

string line;

int lineNumber = 0;

vector<string> copiedLines;

while (getline(file1, line)) {

lineNumber++;

if (lineNumber >= N1 && lineNumber <= N2 && line.find('A') == string::npos && line.find('a') == string::npos) {

file2 << line << endl;

copiedLines.push\_back(line);

}

}

file1.close();

file2.close();

cout << "\nСкопійовані рядки:\n";

for (const string& copiedLine : copiedLines) {

cout << copiedLine << endl;

}

}

int findLineWithMostVowels(const string& fileF2) {

ifstream file(fileF2);

if (!file) {

cout << "Не вдалося відкрити файл F2!\n";

return -1;

}

string line;

int maxVowels = 0;

int maxVowelLine = 0;

int lineNumber = 0;

while (getline(file, line)) {

lineNumber++;

int vowelCount = 0;

for (char c : line) {

if (tolower(c) == 'a' || tolower(c) == 'e' || tolower(c) == 'i' ||

tolower(c) == 'o' || tolower(c) == 'u' || tolower(c) == 'y') {

vowelCount++;

}

}

if (vowelCount > maxVowels) {

maxVowels = vowelCount;

maxVowelLine = lineNumber;

}

}

file.close();

return maxVowelLine;

}

int main() {

string fileF1 = "F1.txt";

string fileF2 = "F2.txt";

cout << "Створимо файл F1:\n";

createFileF1(fileF1);

int N1, N2;

cout << "Введіть номер початкового рядка (N1): ";

cin >> N1;

cout << "Введіть номер кінцевого рядка (N2): ";

cin >> N2;

cout << "\nКопіюємо рядки з F1 у F2...\n";

copyLines(fileF1, fileF2, N1, N2);

cout << "Копіювання завершено. Результат у файлі F2.\n";

int lineWithMostVowels = findLineWithMostVowels(fileF2);

if (lineWithMostVowels != -1) {

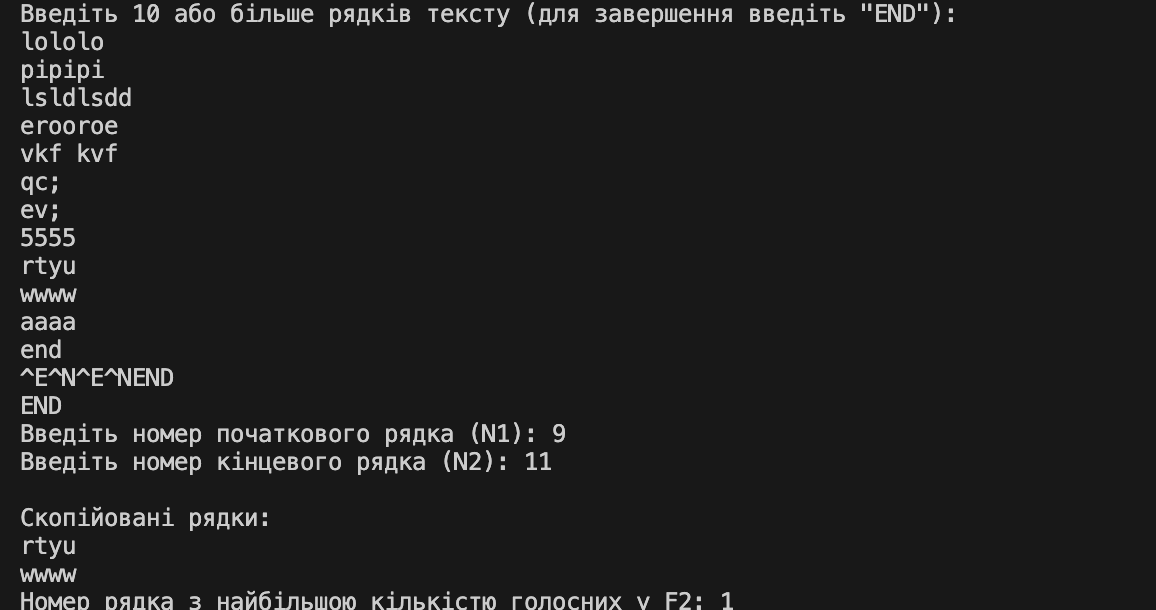
cout << "Номер рядка з найбільшою кількістю голосних у F2: " << lineWithMostVowels << endl;

} else {

cout << "Файл F2 порожній або не існує.\n";

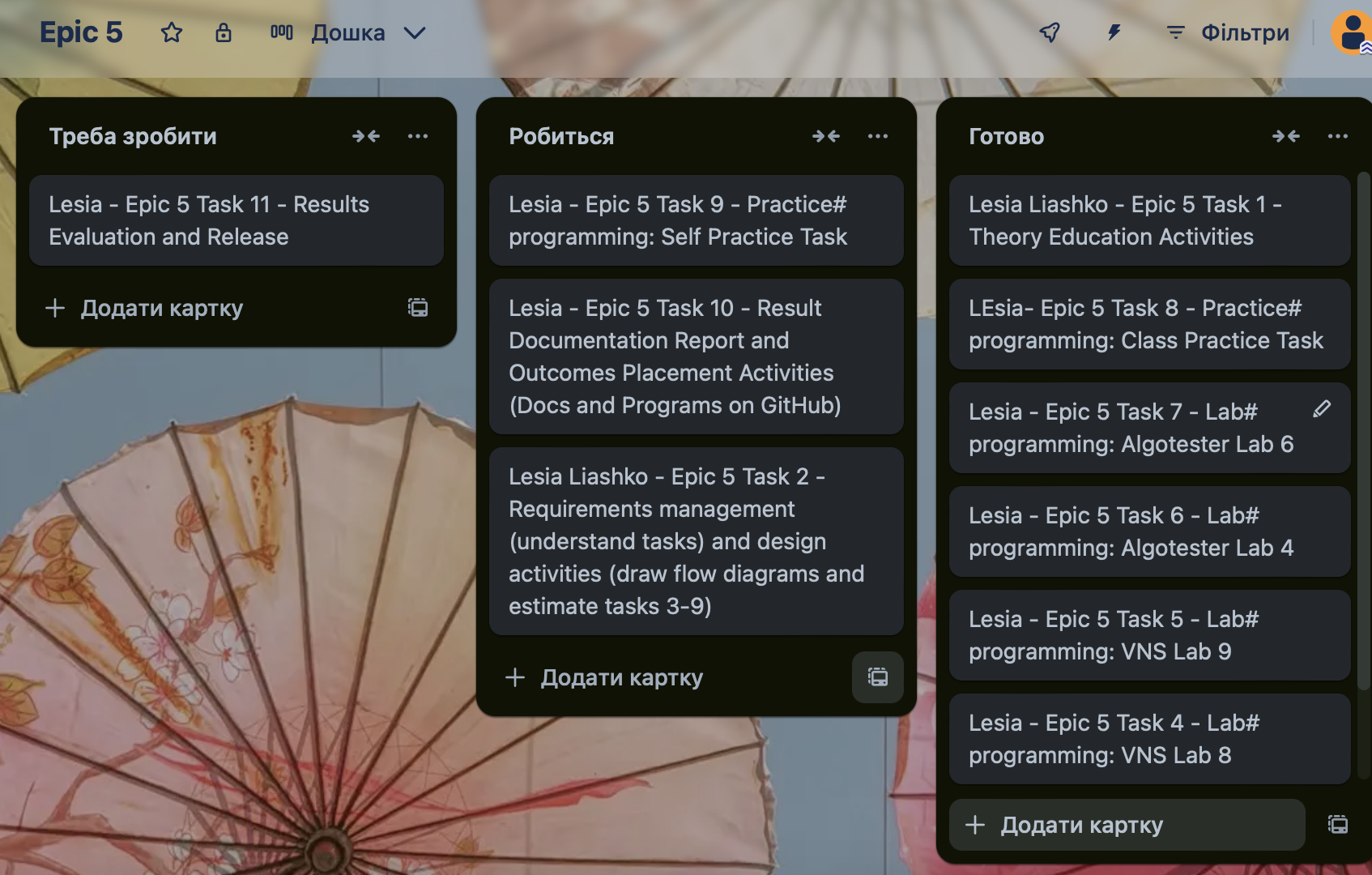
}

return 0;

}

**Робота з командою**

Наші зустрічі виходять на новий рівень, і тепер ми бачимося фізично. Обговорили практичне завдання з 5 епіку та трохи заглянули наперед, а саме обговорили завдання з саги та про лайвкодінг. Познайомилися ближче та сходили до храму. На момент зустрічі мала ось такий перелік завдань, що були готові та, які потрібно було ще зробити:





**Висновки**: Знання основних операцій з файловими дескрипторами та C-style читання і запису допомагає краще розуміти низькорівневі аспекти роботи з файлами. Розуміння символів і рядкових змінних, а також методів їх обробки, є ключовим для ефективного маніпулювання текстовими даними. Текстові файли дозволяють зберігати дані у зручному для читання форматі, а методи форматування тексту дозволяють створювати структуровані файли. Створення власних бібліотек дозволяє розширювати функціональність програм та використовувати повторюваний код у різних проектах, що підвищує

ефективність розробки.