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

Яцишин Ігор Васильович

# **Тема роботи:**

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

# **Мета роботи:**

Навчитися працювати з символьними та рядковими змінними в мові C++. Вивчити основні алгоритми роботи з файлами та бібліотеками. 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++.
* Тема №4: Робота з символьними та рядковими змінними.

1. Індивідуальний план опрацювання теорії:

* Тема №1: Текстові та бінарні файли.
  + Джерела Інформації
    - Стаття: What Is The Difference Between Binary and Text Files? [Full Information] [What Is The Difference Between Binary and Text Files? [Full Information]](https://www.easeus.com/knowledge-center/what-is-the-difference-between-binary-and-text-files.html)
  + Що опрацьовано:
    - Опрацьовано теоретичний матеріал про основні відмінності між текстовими та бінарними файлами, їх особливості
  + Статус: Ознайомлений
  + Початок опрацювання теми: 29.11.2023 22.20
  + Звершення опрацювання теми: 29.11.2023 23.00
* Тема №2: Основні методи роботи з файлами в C++.
  + Джерела Інформації:
    - Стаття: File Handling Through C++ Classes [File Handling through C++ Classes - GeeksforGeeks](https://www.geeksforgeeks.org/file-handling-c-classes/)
    - Стаття: C++ Files and Streams [C++ Files and Streams](https://www.tutorialspoint.com/cplusplus/cpp_files_streams.htm)
    - Відео: C++ Теорія ⦁ Урок 166 ⦁ Робота з файлами (стиль мови C)

[C++ Теорія ⦁ Урок 166 ⦁ Робота з файлами (стиль мови C)](https://www.youtube.com/watch?v=FeNqHytI0fA&t=3573s&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)

* + Що опрацьовано:
    - Вивчено теоретичні основи роботи з файлами та потоками в мові програмування С++
    - Опрацьовано приклади практичного застосування файлів в програмному коді мовою C++
  + Статус: Ознайомлений
  + Початок опрацювання теми: 30.11.2023 18.00
  + Звершення опрацювання теми: 30.11.2023 21.25
* Тема №3: Стандартна бібліотека C++.
  + Джерела Інформації:
    - Відео: C++ Теорія ⦁ Урок 79 ⦁ Стандартна бібліотека C++ [C++ Теорія ⦁ Урок 79 ⦁ Стандартна бібліотека C++](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)
    - Стаття: Створення та використання C++ бібліотек <https://pllug-blog.blogspot.com/2016/05/c_31.html#creating-static-library>
  + Що опрацьовано:
    - Опрацьовано поняття стандартної бібліотеки мови C++ та її використання
    - Розглянуто приклади створення власних бібліотек та подальшої роботи з ними
  + Статус: Ознайомлений
  + Початок опрацювання теми: 30.11.2023 22.30
  + Звершення опрацювання теми: 30.11.2023 23.20
* Тема №4: Робота з символьними та рядковими змінними.
  + Джерела Інформації:
    - Стаття: C++ char [C++ char Type (Characters)](https://www.programiz.com/cpp-programming/char-type)
    - Стаття: C++ Strings [C++ Strings](https://www.w3schools.com/cpp/cpp_strings.asp)
    - Відео: C++ Теорія ⦁ Урок 137 ⦁ string [C++ Теорія ⦁ Урок 137 ⦁ string](https://www.youtube.com/watch?v=1FkTJYm-T34&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)
  + Що опрацьовано:
    - Опрацьовано та вивчено основні методи роботи з символьними змінними та рядками
    - Вивчено принцип роботи з функціями, спрямованими на обробку рядків у мові C++
  + Статус: Ознайомлений
  + Початок опрацювання теми: 01.12.2023 16.00
  + Звершення опрацювання теми: 01.12.2023 18.10

# **Виконання роботи:**

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

Завдання №1 VNS Lab 6

* Варіант завдання: 4
* 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. Надрукувати всі слова, які співпадають з її першим словом.
* Важливі деталі для врахування в імплементації програми:

Використання функції gets() для отримання рядка з клавіатури

Завдання №2 VNS Lab 8

* Варіант завдання: 4
* 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.

Структура "Людина":

- прізвище, ім'я, по батькові;

- домашня адреса;

- номер телефону;

- вік.

Знищити усі елементи із заданим віком, додати елемент після елемента із

заданим номером.

* Важливі деталі для врахування в імплементації програми:

Дії над файлами та структурами мають виконуватися через функції

Завдання №3 VNS Lab 9

* Варіант завдання: 4
* Деталі завдання: Створити текстовий файл F1 не менше, ніж з 10 рядків і записати в нього інформацію

1) Скопіювати з файлу F1 у файл F2 рядки, починаючи з 4.

2) Підрахувати кількість символів в останньому слові F2.

* Важливі деталі для врахування в імплементації програми

Використання фунrції getline() для отримання стрічки з клавіатури

Завдання №4 Algotester Lab 4

* Варіант завдання: 2
* Деталі завдання: <https://algotester.com/en/ContestProblem/DisplayWithEditor/134641>
* Важливі деталі для врахування в імплементації програми

Завдання виконано без використання засобів STL, а натомість описано власні функції, які є аналогами функцій стандартної бібліотеки

Завдання №5 Algotester Lab 6

* Варіант завдання: 2
* Деталі завдання: <https://algotester.com/en/ContestProblem/DisplayWithEditor/134743>
* Важливі деталі для врахування в імплементації програми

Використання власноручно написаної функції для перевірки атаки фігур

Завдання №6 Class Practice Work Task 1

* Варіант завдання: немає
* Деталі завдання: Запис текстової стрічки у файл із заданим ім’ям

Реалізувати функцію створення файла і запису в нього даних:

enum FileOpResult { Success, Failure, … };

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

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.

* Важливі деталі для врахування в імплементації програми

Використання функції write\_to\_file() та вивід успішного/неуспішного статусу виконання програми

Завдання №7 Class Practice Work Task 2

* Варіант завдання: немає
* Деталі завдання: Копіювання вмісту файла у інший файл

Реалізувати функцію створення файла і запису в нього даних:

enum FileOpResult { Success, Failure, … };

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

Умови задачі:

- копіювати вміст файла з ім’ям file\_from у файл з ім’ям file\_to; написати код стійкий до різних варіантів вхідних параметрів, обробити всі можливі варіанти відсутності одного з файлів

- file\_from, file\_to – можуть бути повним або відносним шляхом

- повернути статус операції: Success – все пройшло успішно, Failure – файл не вдалося створити, або збій операції відкриття, читання чи запису даних, закриття файла.

* Важливі деталі для врахування в імплементації програми

Використання функції copy\_file() та вивід успішного/неуспішного статусу виконання програми

Завдання №8 Self Practice Work

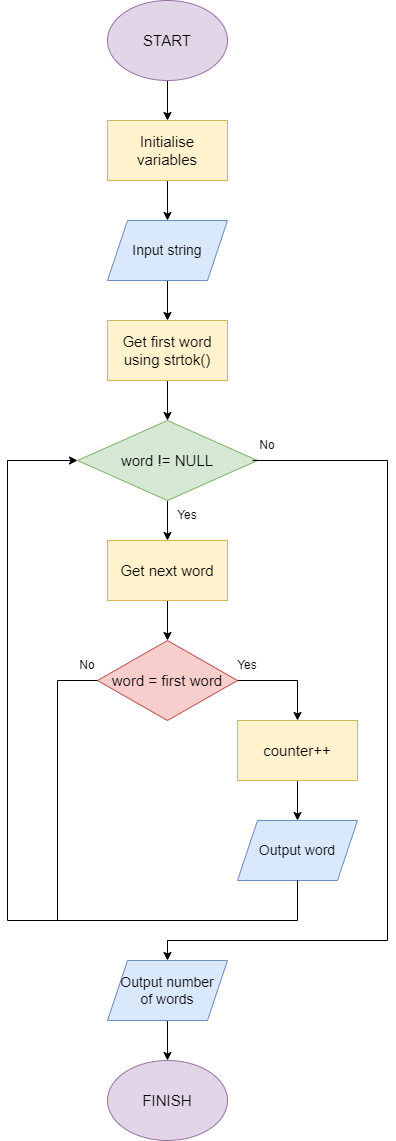
* Варіант завдання: немає
* Деталі завдання:Дано два текстових файли. Чи мають їх відповідні рядки однакову довжину?
* Важливі деталі для врахування в імплементації програми

Використання функції getline() для отримання стрічки з клавіатури

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

Програма №1 VNS Lab 6

* Блок-схема



*Рисунок 1. Блок-схема до програми №1*

* Планований час на реалізацію: 55 хвилин

Програма №2 VNS Lab 8

* Планований час на реалізацію: 80 хвилин

Програма №3 VNS Lab 9

* Планований час на реалізацію: 60 хвилин

Програма №4 Algotester Lab 4

* Планований час на реалізацію: 120 хвилин

Програма №5 Algotester Lab 6

* Планований час на реалізацію: 120 хвилин

Програма №6 Class Practice Work Task 1

* Планований час на реалізацію: 40 хвилин

Програма №7 Class Practice Work Task 2

* Планований час на реалізацію: 40 хвилин

Програма №8 Self Practice Work

* Планований час на реалізацію: 40 хвилин

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

Для виконання поставлених завдань додаткова конфігурація середовища не є необхідною

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

Завдання №1 VNS Lab 6

#include <iostream>

#include <string.h>

using namespace std;

int main()

{

char string[256];

int counter = 0;

cout << "Enter your string: ";

gets(string);

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

char \*first = word;

while(word != NULL)

{

word = strtok(NULL, " ");

if(word != NULL && strcmp(word, first) == 0)

{

cout << word << endl;

counter++;

}

}

cout << "Total: " << counter << " words";

return 0;

}

Програма отримує стрічку шляхом введення з клавіатури. Далі у циклі while кожне слово стрічки порівнюється з першим і у разі співпадіння виводиться на екран. Сумарна кількість слів, що дорівнюють першому виводиться на екран.

Посилання на файл програми у пул-запиті GitHub

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/705

Завдання №2 VNS Lab 8

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

struct Person

{

char name[100];

char address[100];

char mobile[20];

int age;

};

void write\_to\_file(const char\* filename, int &n);

void read\_from\_file(const char\* filename, int &n);

void delete\_person(const char\* filename, int age);

void add\_person(const char\* filename, int index);

int main()

{

int n;

cout << " How much structures you want to enter?";

cin >> n;

write\_to\_file("file.txt");

read\_from\_file("file.txt");

add\_person("file.txt", 1);

delete\_person("file.txt", 16);

read\_from\_file("file.txt");

return 0;

}

void write\_to\_file(const char\* filename, int &n)

{

Person person[n];

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

{

cout << "Enter name: ";

cin.getline(person[i].name, 100);

cout << "Enter address: ";

cin.getline(person[i].address, 100);

cout << "Enter mobile: ";

cin.getline(person[i].mobile, 20);

cout << "Enter age: ";

cin >> person[i].age;

cin.ignore(); // Ignore the newline character left in the buffer

}

ofstream file1(filename, ios::out | ios::binary);

if (!file1.is\_open())

{

cout << "Failure" << endl;

}

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

{

file1.write((char\*)&person[i], sizeof(Person));

}

file1.close();

}

void read\_from\_file(const char\* filename, int &n)

{

Person person[n];

ifstream file2(filename, ios::in | ios::binary);

if (!file2.is\_open())

{

cout << "Failure" << endl;

}

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

{

file2.read((char\*)&person[i], sizeof(Person));

}

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

{

cout << person[i].name << endl;

cout << person[i].address << endl;

cout << person[i].mobile << endl;

cout << person[i].age << endl;

}

file2.close();

}

void delete\_person(const char\* filename, int age)

{

Person person;

ifstream file1(filename, ios::in | ios::binary);

ofstream file2("temporary.txt", ios::out | ios::binary);

if (!file1.is\_open() || !file2.is\_open())

{

cout << "Failure" << endl;

return;

}

while (file1.read((char\*)&person, sizeof(Person)))

{

if (person.age != age)

{

file2.write((char\*)&person, sizeof(Person));

}

}

file1.close();

file2.close();

remove(filename);

rename("temporary.txt", filename);

}

void add\_person(const char\* filename, int index)

{

Person new\_person;

cout << "Enter name: ";

cin.getline(new\_person.name, 100);

cout << "Enter address: ";

cin.getline(new\_person.address, 100);

cout << "Enter mobile: ";

cin.getline(new\_person.mobile, 20);

cout << "Enter age: ";

cin >> new\_person.age;

cin.ignore(); // Ignore the newline character left in the buffer

Person person;

ifstream file1(filename, ios::in | ios::binary);

ofstream file2("temporary.txt", ios::out | ios::binary);

if (!file1.is\_open() || !file2.is\_open())

{

cout << "Failure" << endl;

return;

}

int current\_index = 0;

while (file1.read((char\*)&person, sizeof(Person)))

{

file2.write((char\*)&person, sizeof(Person));

if (current\_index == index)

{

file2.write((char\*)&new\_person, sizeof(Person));

}

current\_index++;

}

file1.close();

file2.close();

remove(filename);

rename("temporary.txt", filename);

}

За допомогою функції write\_to\_file() програма отримує дані про людину та записує їх у файл. Функція read\_from\_file() зчитує дані з файлу та виводить їх в консоль. Функція delete\_person() стирає дані про особу з певним заданим віком. Також функція add\_person() додає особу після особи з вказаним індексом. Програма передбачає вивід повідомлення у разі помилки при відкритті чи створенні файлу

Посилання на файл програми у пул-запиті GitHub

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/705

Завдання №3 VNS Lab 9

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

int main()

{

ofstream first\_file("F1.txt");

if (!first\_file.is\_open())

{

cout << "Failure" << endl;

return 1;

}

int rows;

cout << "Enter number of rows you want to fill: ";

cin >> rows;

cin.ignore();

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

{

cout << "Enter information of row number " << i + 1 << " in file F1.txt: ";

string string;

getline(cin, string);

first\_file << string << endl;

}

first\_file.close();

ifstream first\_file\_read("F1.txt");

ofstream second\_file("F2.txt");

if (!first\_file\_read.is\_open() || !second\_file.is\_open())

{

cout << "Failure" << endl;

return 1;

}

string line;

int line\_number = 0;

while (getline(first\_file\_read, line))

{

++line\_number;

if (line\_number >= 4)

{

second\_file << line << endl;

}

}

first\_file\_read.close();

second\_file.close();

ifstream second\_file\_read("F2.txt");

if (!second\_file\_read.is\_open())

{

cout << "Failure" << endl;

return 1;

}

string last\_word;

while (!second\_file\_read.eof())

{

second\_file\_read >> last\_word;

}

second\_file\_read.close();

cout << "Last word in F2: " << last\_word << endl;

cout << "Number of charackters in the word: " << last\_word.length() << endl;

return 0;

}

Програма запитує в користувача кількість рядків, яку він хоче ввести та відповідно самі рядки. Далі рядки записуються в файл і починаючи з четвертого копіюються у інший файл. Також здійснюється підрахунок символів у останньому слові другого файлу. Результат виконання виводиться в консоль

Посилання на файл програми у пул-запиті GitHub

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/705

Завдання №4 Algotester Lab 4

#include <iostream>

#include <vector>

using namespace std;

void sort\_vector(vector<int> &vector, int size);

void remove\_duplicates(vector<int> &vector, int &size);

void rotate\_vector(vector<int> &vector, int size, int k);

int main()

{

int n, k;

cin >> n >> k;

vector<int> array(n);

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

cin >> array[i];

}

sort\_vector(array, n);

remove\_duplicates(array, n);

cout << n << endl;

rotate\_vector(array, n , k);

return 0;

}

void sort\_vector(vector<int> &vector, int size)

{

for (int i = 0; i < size - 1; i++)

{

for(int j = 0; j < size - i - 1; j++)

{

if (vector[j] > vector[j + 1])

{

int temp = vector[j];

vector[j] = vector[j + 1];

vector[j + 1] = temp;

}

}

}

}

void remove\_duplicates(vector<int> &vector, int &size)

{

int counter = 0;

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

{

if(counter == 0 || vector[i] != vector[counter - 1])

{

vector[counter++] = vector[i];

}

}

size = counter;

}

void rotate\_vector(vector<int> &vector, int size, int k)

{

k %= size;

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

{

cout << vector[i] << " ";

}

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

{

cout << vector[i] << " ";

}

cout << endl;

}

У програмі описано власноруч створені функції для сортування вектора, видалення у ньому елементів що повторюються та обертання масиву. Результати виконання виводяться в консоль за допомогою cout.

Посилання на файл програми у пул-запиті GitHub

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/705

Завдання №5 Algotester Lab 6

#include <iostream>

#include <algorithm>

#include <vector>

using namespace std;

vector<char> attacks(char board[8][8], int x, int y);

int main()

{

char board[8][8];

int Q;

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

{

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

{

cin >> board[i][j];

}

}

cin >> Q;

vector<pair<int, int>> coordinates;

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

{

int x, y;

cin >> x >> y;

coordinates.push\_back({x, y});

}

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

{

int x = coordinates[i].first;

int y = coordinates[i].second;

vector<char> figures = attacks(board, x, y);

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

if (board[x - 1][y - 1] != 'O')

{

cout << "X";

}

else if (figures.empty())

{

cout << "O";

}

else

{

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

{

cout << figures[i];

if (i + 1 < figures.size() && figures[i] == figures[i + 1])

{

while (i + 1 < figures.size() && figures[i] == figures[i + 1])

{

i++;

}

}

}

}

cout << endl;

}

return 0;

}

vector<char> attacks(char board[8][8], int x, int y)

{

vector<char> result;

if (x - 2 >= 0 && x - 2 < 8 && y - 2 >= 0 && y - 2 < 8 && board[x - 2][y - 2] == 'P')

{

result.push\_back('P');

}

else if (x - 2 >= 0 && x - 2 < 8 && y >= 0 && y < 8 && board[x - 2][y] == 'P')

{

result.push\_back('P');

}

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

{

if (board[i][y - 1] == 'R')

{

result.push\_back('R');

}

}

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

{

if (board[x - 1][i] == 'R')

{

result.push\_back('R');

}

}

for (int i = 1; x - i >= 1 && y - i >= 1; ++i)

{

if (board[x - i - 1][y - i - 1] == 'B')

{

result.push\_back('B');

break;

}

}

for (int i = 1; x - i >= 1 && y + i <= 8; ++i)

{

if (board[x - i - 1][y + i - 1] == 'B')

{

result.push\_back('B');

break;

}

}

for (int i = 1; x + i <= 8 && y - i >= 1; ++i)

{

if (board[x + i - 1][y - i - 1] == 'B')

{

result.push\_back('B');

break;

}

}

for (int i = 1; x + i <= 8 && y + i <= 8; ++i)

{

if (board[x + i - 1][y + i - 1] == 'B')

{

result.push\_back('B');

break;

}

}

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

{

if (board[i][y - 1] == 'Q' || board[x - 1][i] == 'Q')

{

result.push\_back('Q');

}

}

for (int i = 1; x - i >= 1 && y - i >= 1; ++i)

{

if (board[x - 1 - i][y - 1 - i] == 'Q')

{

result.push\_back('Q');

break;

}

}

for (int i = 1; x - i >= 1 && y + i <= 8; ++i)

{

if (board[x - 1 - i][y - 1 + i] == 'Q')

{

result.push\_back('Q');

break;

}

}

for (int i = 1; x + i <= 8 && y - i >= 1; ++i)

{

if (board[x - 1 + i][y - 1 - i] == 'Q')

{

result.push\_back('Q');

break;

}

}

for (int i = 1; x + i <= 8 && y + i <= 8; ++i)

{

if (board[x - 1 + i][y - 1 + i] == 'Q')

{

result.push\_back('Q');

break;

}

}

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

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

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

{

int nx = x + knight\_x[i];

int ny = y + knight\_y[i];

if (nx >= 1 && nx <= 8 && ny >= 1 && ny <= 8)

{

if (board[nx - 1][ny - 1] == 'N')

{

result.push\_back('N');

}

}

}

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

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

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

{

int nx = x + king\_x[i];

int ny = y + king\_y[i];

if (nx >= 1 && nx <= 8 && ny >= 1 && ny <= 8)

{

if (board[nx - 1][ny - 1] == 'K')

{

result.push\_back('K');

}

}

}

return result;

}

У функції attacks() здійснюється обробка станів шахової дошки та запис атакуючих фігур у динамічний вектор. Далі основна програма виводить усі фігури, які можуть атакувати клітинку

Посилання на файл програми у пул-запиті GitHub

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/705

Завдання №6 Class Practice Work Task 1

#include <iostream>

#include <fstream>

using namespace std;

enum FileOpResult { Success, Failure };

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

int main()

{

char \*name, \*content;

cout << "Enter file name (with extension): ";

cin >> name;

cout << "Enter content that should be written to file: ";

cin >> content;

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

if (result == FileOpResult::Success)

{

cout << "Success" << endl;

} else {

cout << "Failure" << endl;

}

return 0;

}

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

{

ofstream file(name);

if (!file.is\_open())

{

cout << "Failure" << endl;

}

file << content;

if (file.fail())

{

cout << "Failure" << endl;

file.close();

}

file.close();

return FileOpResult::Success;

}

Функція write\_to\_file()записує у файл стрічку, попередньо введену користувачем. Програма передбачає вивід повідомлення про успішність виконання алгоритму.

Посилання на файл програми у пул-запиті GitHub

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/705

Завдання №7 Class Practice Work Task 2

#include <iostream>

#include <fstream>

using namespace std;

enum FileOpResult { Success, Failure };

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

int main()

{

char \*name\_to, \*name\_from;

cout << "Enter file name that should be coppied: ";

cin >> name\_from;

cout << "Enter file name that should be destination: ";

cin >> name\_to;

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

if (result == FileOpResult::Success)

{

cout << "Success" << endl;

} else {

cout << "Failure" << endl;

}

return 0;

}

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

{

ifstream file1(file\_from);

if (!file1.is\_open())

{

return FileOpResult::Failure;

}

ofstream file2(file\_to);

if (!file2.is\_open())

{

file1.close();

return FileOpResult::Failure;

}

file2 << file1.rdbuf();

if (file2.fail())

{

file1.close();

file2.close();

return FileOpResult::Failure;

}

return FileOpResult::Success;

}

Програма запитує в користувача імена файлів, з якого і в який потрібно здійснити копіювання вмісту. Далі програма за допомогою функції copy\_file() безпосередньо виконує копіювання. Програма виводить результат успішності виконання всіх маніпуляцій.

Посилання на файл програми у пул-запиті GitHub

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/705

Завдання №8 Self Practice Work

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

int main()

{

ofstream F1("First\_file.txt");

ofstream F2("Second\_file.txt");

if (!F1.is\_open() || !F2.is\_open())

{

cout << "Failure" << endl;

return 1;

}

int n;

cout << "How many lines do you want to enter? ";

cin >> n;

cin.ignore();

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

{

cout << "Enter content for First\_file.txt: ";

string str;

getline(cin, str);

F1 << str << endl;

}

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

{

cout << "Enter content for Second\_file.txt: ";

string str;

getline(cin, str);

F2 << str << endl;

}

F1.close();

F2.close();

ifstream F1\_("First\_file.txt");

ifstream F2\_("Second\_file.txt");

if (!F1\_.is\_open() || !F2\_.is\_open())

{

cout << "Failure" << endl;

return 1;

}

string first\_file\_line, second\_file\_line;

int counter = 0;

while (getline(F1\_, first\_file\_line) && getline(F2\_, second\_file\_line))

{

counter++;

if (first\_file\_line.length() != second\_file\_line.length())

{

std::cout << "Files have lines with different length" << endl;

return 0;

}

}

std::cout << "All lines in files have the same length" << endl;

F1\_.close();

F2\_.close();

return 0;

}

Програма запитує у користувача кількість рядків, які він хоче ввести та самі рядки для першого та другого файлів. Далі програма по черзі перевіряє відповідні стрічки файлів у циклі. Якщо стрічки різної довжини, то програма завершує роботу, виводячи відповідне повідомлення. Також програма виводить повідомлення, якщо всі стрічки обох файлів відповідно мають однакову довжину.

Посилання на файл програми у пул-запиті GitHub

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/705

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

Завдання №1 VNS Lab 6

INPUT:

Enter your string: apple is a company, not just an apple

OUTPUT:

apple

Total: 1 words

Час затрачений на виконання завдання: 60 хвилин

Завдання №2 VNS Lab 8

INPUT/OUTPUT:

How much structures you want to enter?2

Enter name: Anna

Enter address: Lviv

Enter mobile: 444

Enter age: 45

Enter name: Vasyl

Enter address: Ternopil

Enter mobile: 777

Enter age: 26

Anna

Lviv

444

45

Vasyl

Ternopil

777

26

Enter name: Petro

Enter address: Lviv

Enter mobile: 555

Enter age: 65

Anna

Lviv

444

45

Vasyl

Ternopil

777

26

Час затрачений на виконання завдання: 150 хвилин

Завдання №3 VNS Lab 9

INPUT:

Enter number of rows you want to fill: 12

Enter information of row number 1 in file F1.txt: qwerty

Enter information of row number 2 in file F1.txt: qwerty

Enter information of row number 3 in file F1.txt: qwerty

Enter information of row number 4 in file F1.txt: qwerty

Enter information of row number 5 in file F1.txt: qwerty

Enter information of row number 6 in file F1.txt: qwerty

Enter information of row number 7 in file F1.txt: qwerty

Enter information of row number 8 in file F1.txt: qwerty

Enter information of row number 9 in file F1.txt: qwerty

Enter information of row number 10 in file F1.txt: qwerty

Enter information of row number 11 in file F1.txt: qwerty

Enter information of row number 12 in file F1.txt: Lviv Polytechinc National University

OUTPUT:

Last word in F2: University

Number of charackters in the word: 10

Час затрачений на виконання завдання: 80 хвилин

Завдання №4 Algotester Lab 4

INPUT:

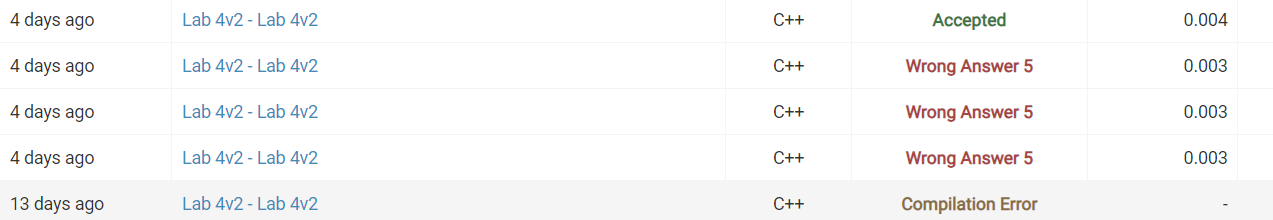
10 3

1 2 2 3 3 3 4 5 6 7

OUTPUT:

7

4 5 6 7 1 2 3



*Рисунок 2. Результат зарахування програми №4 на Алготестері*

Час затрачений на виконання завдання: 180 хвилин

Завдання №5 Algotester Lab 6

INPUT(EXAMPLE):

KOOOOOOO

OOOOOOOO

OOOOOOOO

OOOOOOOO

OOOOOOOO

OOOOOOOO

OOOOOOOO

OOOOOOOO

5

1 1

1 2

2 1

2 2

3 1

OUTPUT:

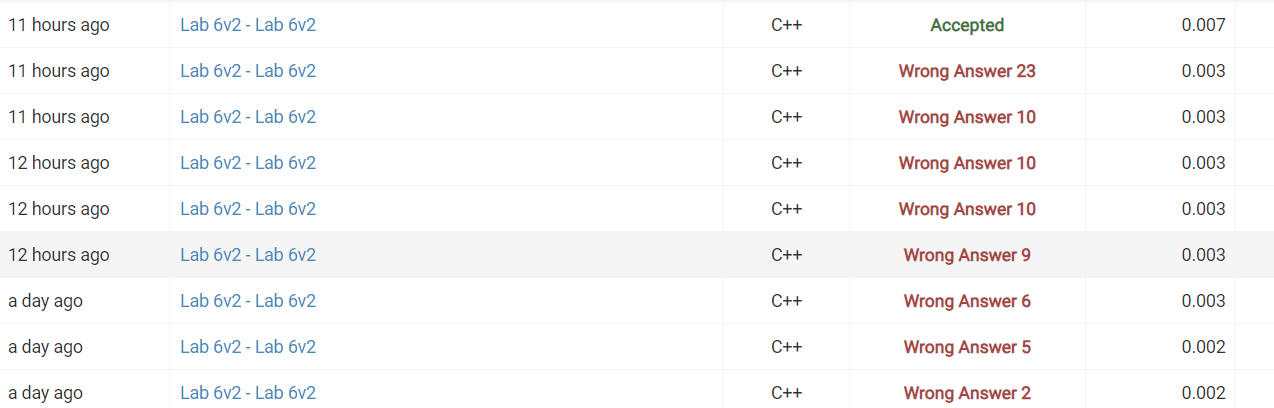
X

K

K

K

O



*Рисунок 3. Результат зарахування програми №5 на Алготестері*

Час затрачений на виконання завдання: 250+ хвилин

Завдання №6 Class Practice Work Task 1

INPUT:

Enter file name (with extension): File.txt

Enter content that should be written to file: qwerty

OUTPUT:

Success

Час затрачений на виконання завдання: 50 хвилин

Завдання №7 Class Practice Work Task 2

INPUT:

Enter file name that should be coppied: File1.txt

Enter file name that should be destination: File2.txt

OUTPUT:

Success

Час затрачений на виконання завдання: 60 хвилин

Завдання №8 Class Practice Work 8

INPUT:

How many lines do you want to enter? 3

Enter content for First\_file.txt: C++

Enter content for First\_file.txt: Java

Enter content for First\_file.txt: Python

Enter content for Second\_file.txt: C++

Enter content for Second\_file.txt: JS

Enter content for Second\_file.txt: Python

OUTPUT:

Files have lines with different length

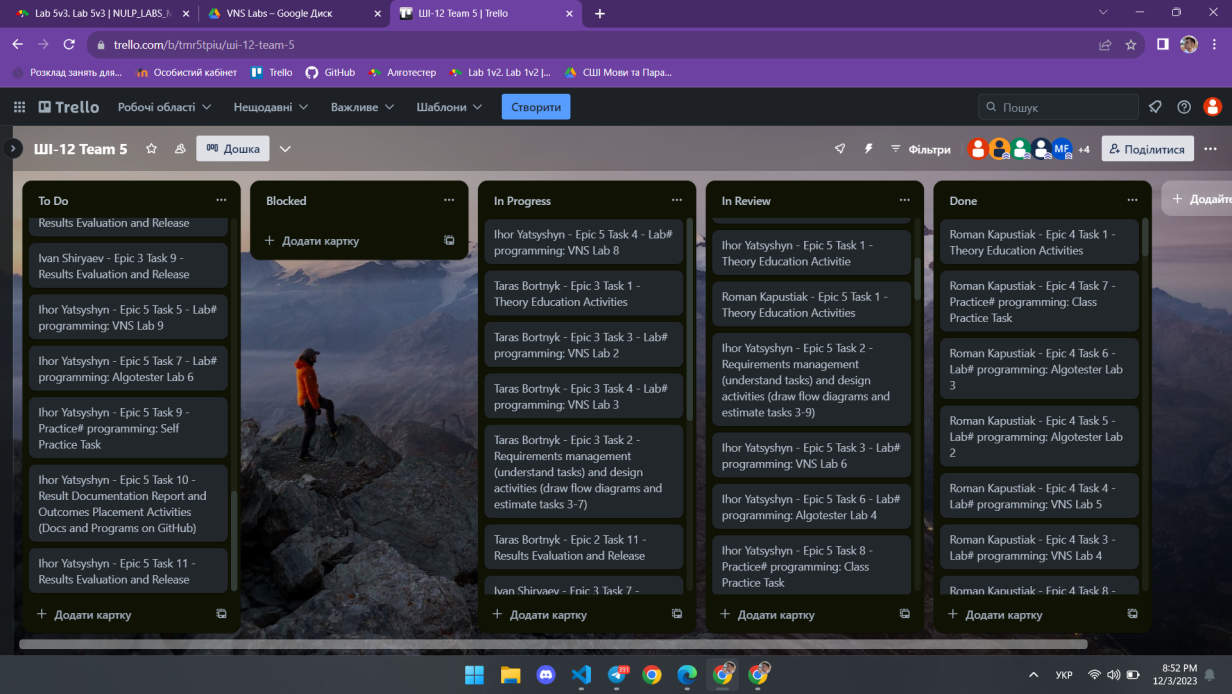
Час затрачений на виконання завдання: 60 хвилин

## **6. Кооперація з командою:**

* Скрін з 1-ї зустрічі по обговоренню задач Епіку та Скрін прогресу по Трелло



*Рисунок 4. Скріншот першої зустрічі*

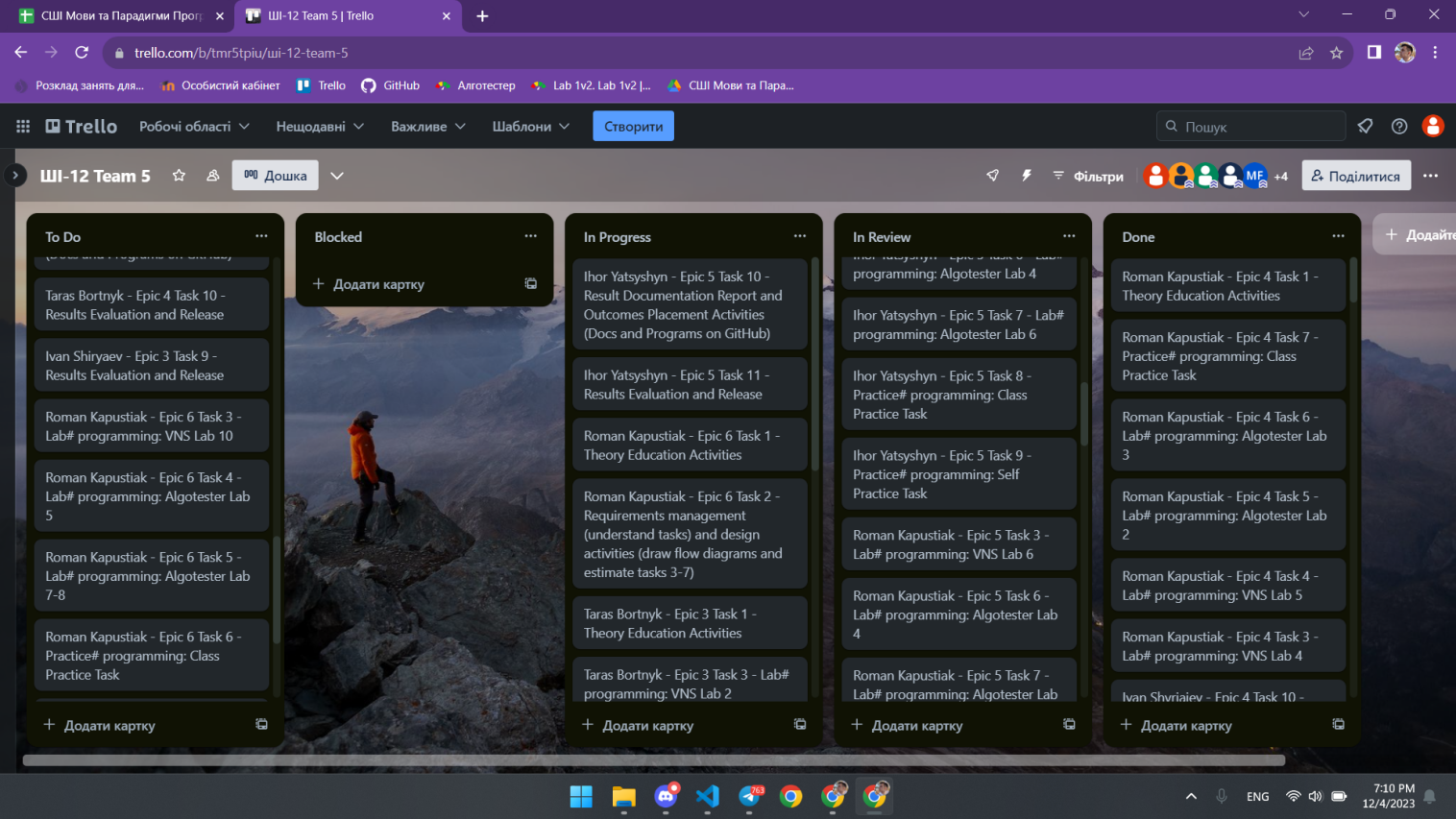
**

*Рисунок 5. Скріншот Trello на час першої зустрічі*

* Скрін з 2-ї зустрічі по обговоренню задач Епіку та Скрін прогресу по Трелло

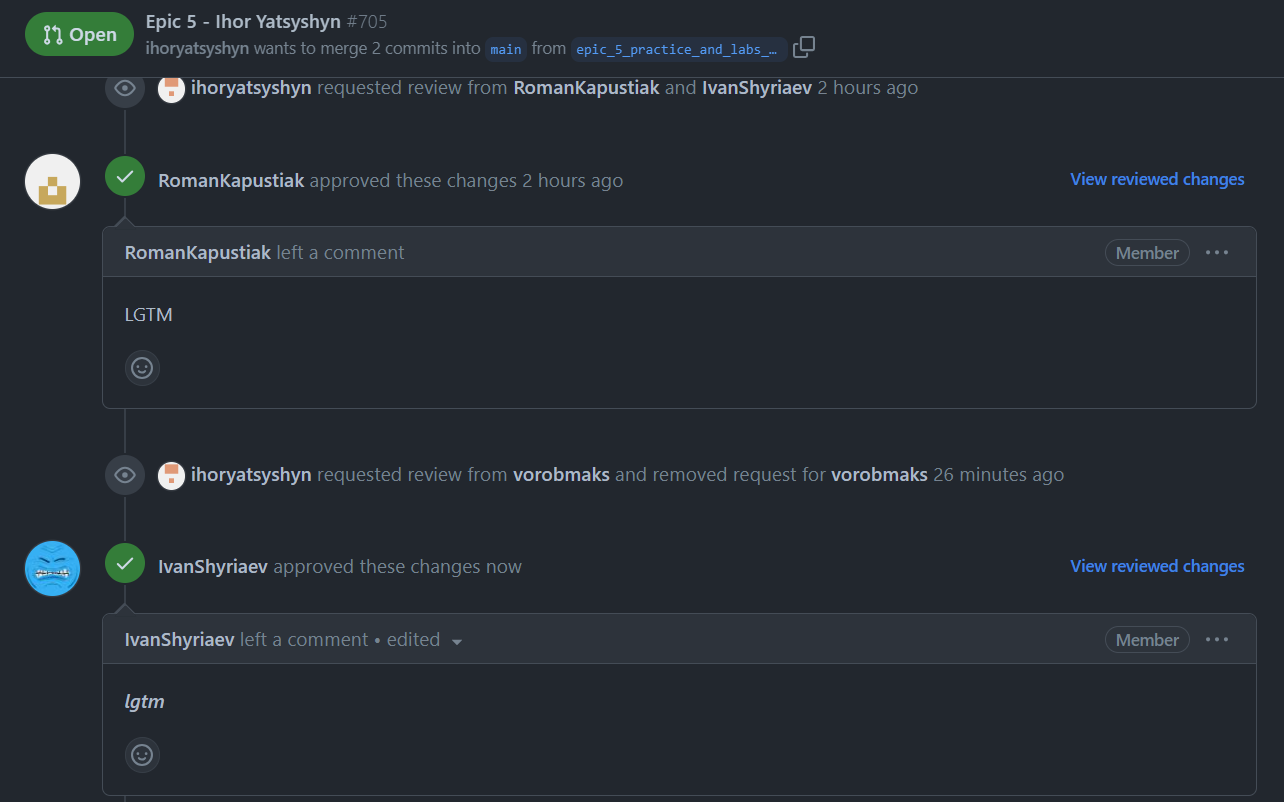


*Рисунок 6. Скріншот другої зустрічі*

**

*Рисунок 7. Скріншот Trello на час другої зустрічі*

* Скрін з коментарями від учасників команди на пул-реквесті з Рев’ю Роботи



*Рисунок 8. Скріншот коментарів від учасників команди до пул-реквесту*

# **Висновки:**

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