Міністерство освіти і науки України
Національний університет "Львівська Політехніка"

Кафедра систем штучного інтелекту

Епік №5

з дисципліни
«Основи програмування»

Виконав:
студент групи ШІ-11
Гнатюк Ярослав

Львів – 2024 р.

Епік №5

Тема: Файли. Бінарні Файли. Символи, рядкові змінні та текстові файли. Стандартна бібліотека та деталі/методи роботи з файлами. Створення й використання бібліотек.

Мета роботи: Ознайомитися з основними принципами роботи з файлами в програмуванні, зокрема з бінарними файлами, текстовими файлами, символами та рядковими змінними. Навчитися використовувати стандартну бібліотеку для маніпуляцій із файлами, а також розглянути методи роботи з ними. Продемонструвати процес створення та використання власних бібліотек для оптимізації роботи з файлами й підвищення повторного використання коду.

Теоретичні відомості:

- <stdio.h> fgets() – зчитування вводу користувача
- <stdio.h> getline() – для зчитування тексту з файлів
- <string.h> strcspn() – індекс першого входження специфічного елемента
- Робота з файлами: https://www.w3schools.com/cpp/cpp_files.asp
  - ofstream – Створити та записати інформацію у файли
  - ifstream – Читати інформацію з файлів
  - fstream – Комбінація двох попередніх
- <algorithm> sort() – сортування елементів масиву
- <algorithm> unique() – видалення дублікатів з відсортованого масиву
- <algorithm> rotate() – обертає елементів в діапазоні таким чином, щоб елемент у заданій позиції ставав першим елементом.
- <algorithm> partition() – переставлення елементів в діапазоні таким чином, щоб всі елементи, які задовольняють предикат, розміщувалися перед іншими
- Створення та використання власних бібліотек:
  - https://www.youtube.com/watch?v=mnwDpO4zqLA&t=889s&pp=ygUm0LLQu9Cw0YHQvdGWINCx0ZbQsdC70ZbQvtGC0LXQutC4INGBKys%3D
  - ChatGPT

Виконання роботи
Частина 1

Завдання №1

Назва: Lab 6 Variant 5

Опис: 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), та перетворити рядок таким чином, щоб спочатку в ньому були надруковані тільки букви, а потім тільки цифри, не міняючи порядку проходження символів у рядку.

Вимоги: Використати get(s) для вводу рядка.

Завдання №2

Назва: Lab 8 Variant 5

Опис: 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;
- рік народження;
- ріст;
- вага.

Знищити усі елементи із зазначеним ростом і вагою, додати елемент після елемента із зазначеним прізвищем.

Завдання №3

Назва: Lab 9 Variant 5

Опис: Створити текстовий файл F1 не менше, ніж з 10 рядків і записати в нього інформацію, а також реалізувати наступні дії:

1) Скопіювати з файлу F1 у файл F2 рядки, починаючи з K до K+5.
2) Підрахувати кількість голосних букв у файлі F2.

Завдання №4

Назва: Algotester Lab 4 Variant 2 (1, 2)

Опис: Вам дано масив a з N цілих чисел. Спочатку видаліть масиву a усі елементи що повторюються, наприклад масив [1, 3, 3, 4] має перетворитися у [1, 3, 4]. Після цього оберніть посортовану версію масиву a на K, тобто при K = 3 масив [1, 2, 3, 4, 5, 6, 7] перетвориться на [4, 5, 6, 7, 1, 2, 3]. Виведіть результат.

Вимоги: Реалізувати алгоритм двома способами:

1) З використанням засобів STL (std::unique, std::sort, std::rotate)
2) Зі своєю реалізацією

Завдання №5

Назва: Algotester Lab 4 Variant 3 (1, 2)

Опис: Вам дано масив, який складається з N додатних цілих чисел. Ваше завдання - розділити його на три частини, по остачі від ділення на 3, по зростанню остачі (тобто спочатку йдуть числа, у яких остача 0, далі числа з остачею 1 і тоді нарешті числа з остачею 2). Далі необхідно ті елементи, остача від ділення на 3 яких парна посортувати по зростанню, а ті, у яких остача 1 - по спаданню. Після цього видаліть усі дублікати з масиву. Виведіть результуючий масив.

Вимоги: Реалізувати алгоритм двома способами:

1) З використанням засобів STL (власноруч написаний компаратор або std::partition + std::sort + std::unique).

2) Зі своєю реалізацією. Алгоритм сортування можна вибрати будь який, окрім сортування бульбашкою і має працювати за N*logN часу.

Завдання №6

Назва: Algotester Lab 6 Variant 3

Опис: У Клінта в черговий раз виключилось світло і йому немає чим зайнятися. Так як навіть це не заставить його подивитися збережені відео про програмування на YouTube, він вирішив придумати свою гру на основі судоку.
Гра виглядає так: Є поле розміром N×N, в якому частина клітинок заповнена цифрами, а частина клітинок порожні (позначаються нулем). Також у нього є Q пар координат X та Y.
Завданням гри є написати до кожної координати скільки чисел туди можна вписати (якщо вона пуста) і які це числа (обов'язково впорядковані по зростанню). В клітинку можна вписати лише ті числа, які не зустрічаються в рядку та стовпці, які перетинаються у цій клітинці.
Поле не змінюється!
Якщо є клітинка, в яку не можна вписати жодну цифру — виведіть 0.
Також допускаються рядки та стовпці, в яких цифра записана кілька разів.

Завдання №7

Назва: Practice work 1

Опис:

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.

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.

Завдання №8

Назва: Self practice work

Опис: У вас є шахова дошка розміром 8×8 і дуже багато фігур. Кожна клітинка може мати одне із таких значень:

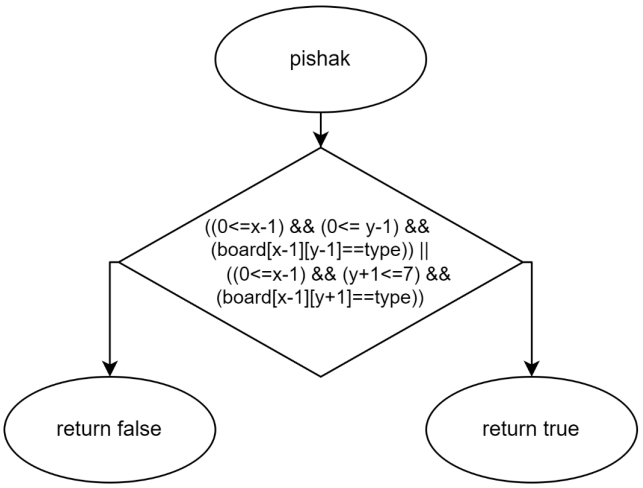
- Пуста клітинка: O
- Пішак: P
- Тура: R
- Кінь: N
- Слон: B
- Король: K
- Королева: Q

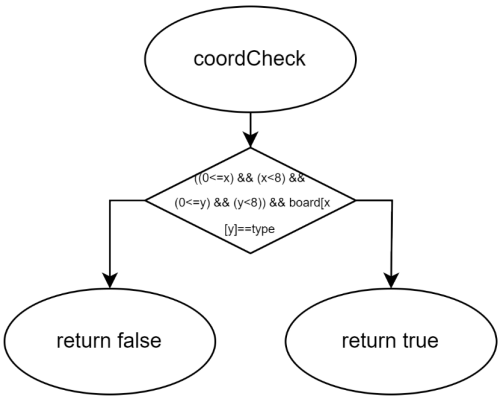
Вам дають позиції фігур на дошці (всі фігури одного кольору, кількість королів може бути > 1). Далі йдуть **Q** запитів з координатами клітинки x,y{x, y}x,y. На кожен запит ви маєте вивести стрічку sis_isi — посортовані за алфавітом букви фігур, які атакують цю клітинку (пішаки атакують вниз). У випадку, якщо на клітинці стоїть якась фігура — виведіть символ **X**. випадку, якщо

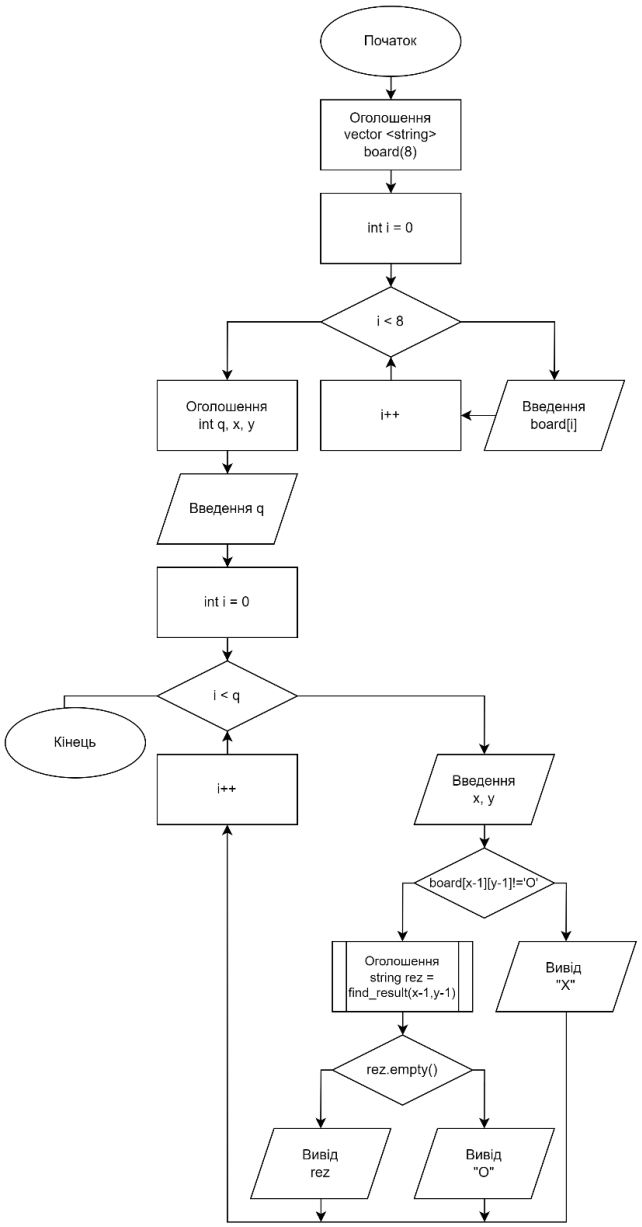
клітинку не атакують — виведіть **O**. Наявність фігури у певній клітинці не блокує атаку для іншої фігури. Тобто, якщо між турою та клітинкою стоїть інша фігура — вважається, що тура атакує цю клітинку.

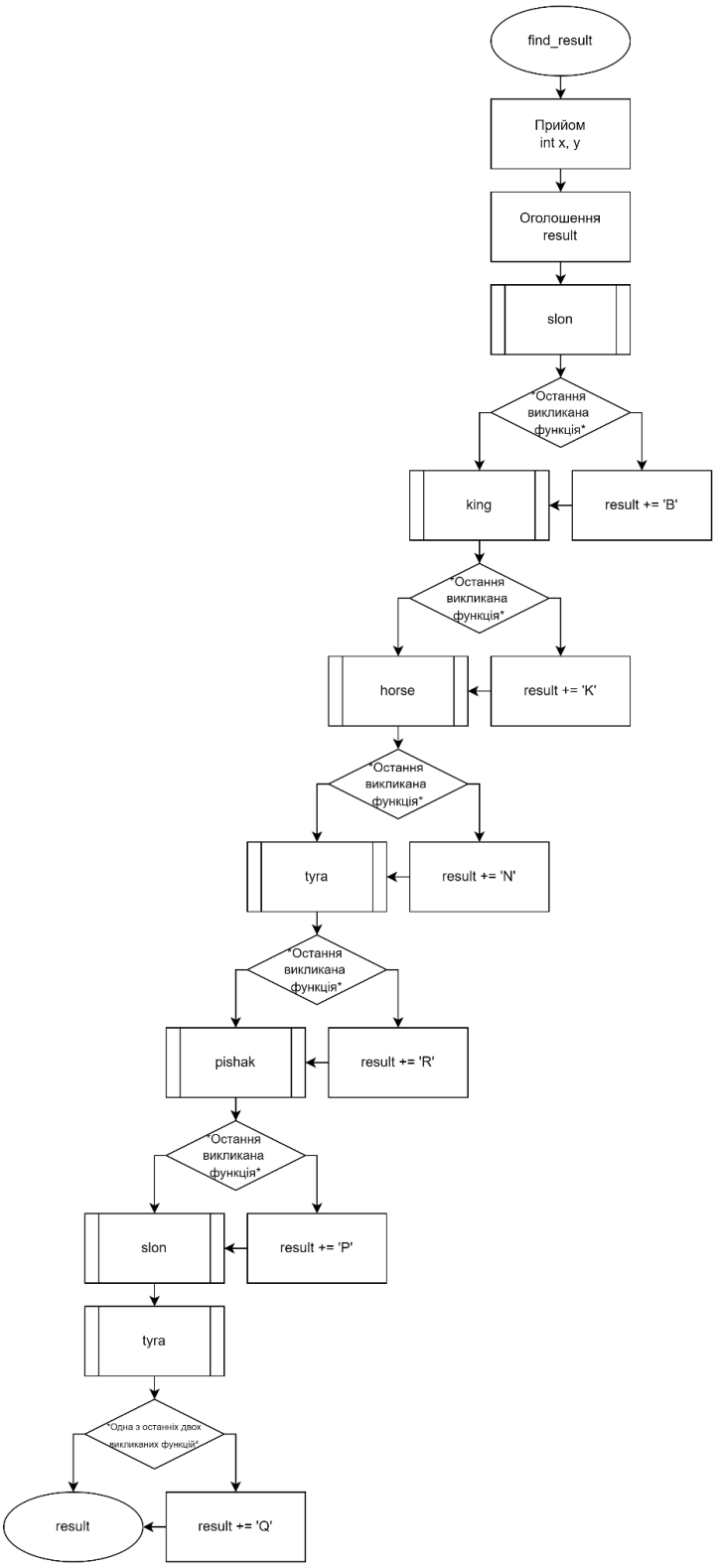
Частина 2

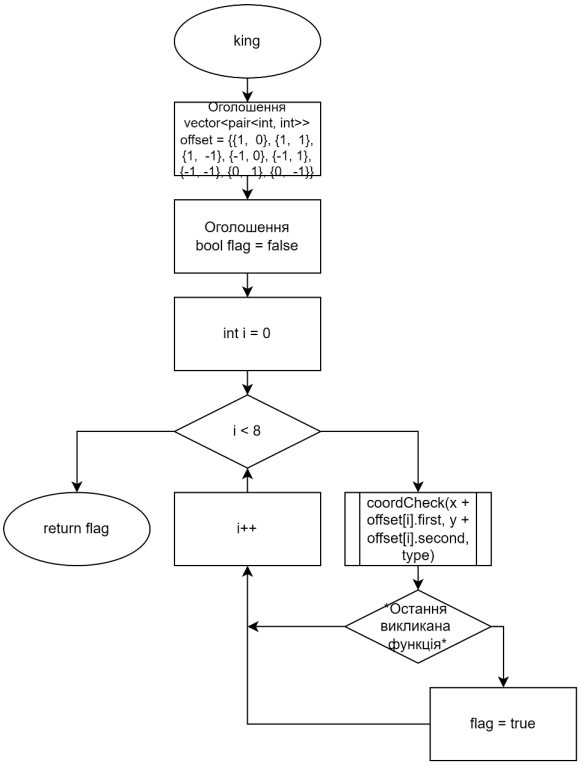
Завдання №7

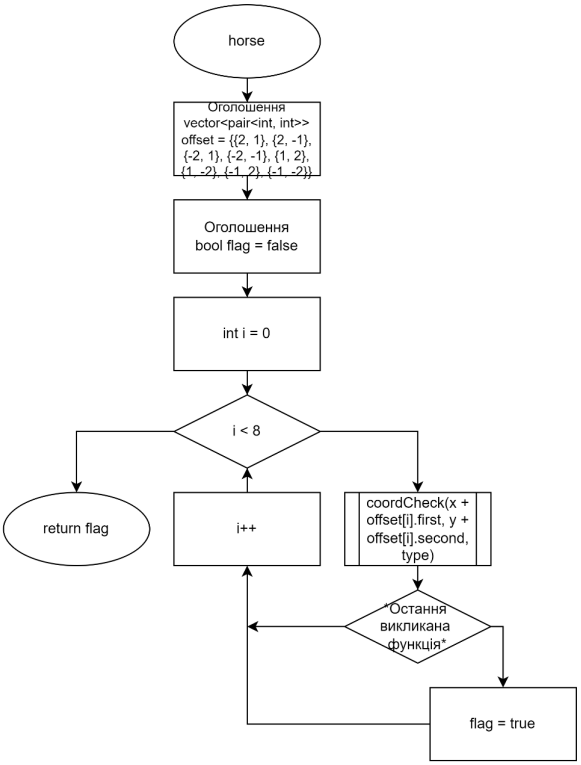


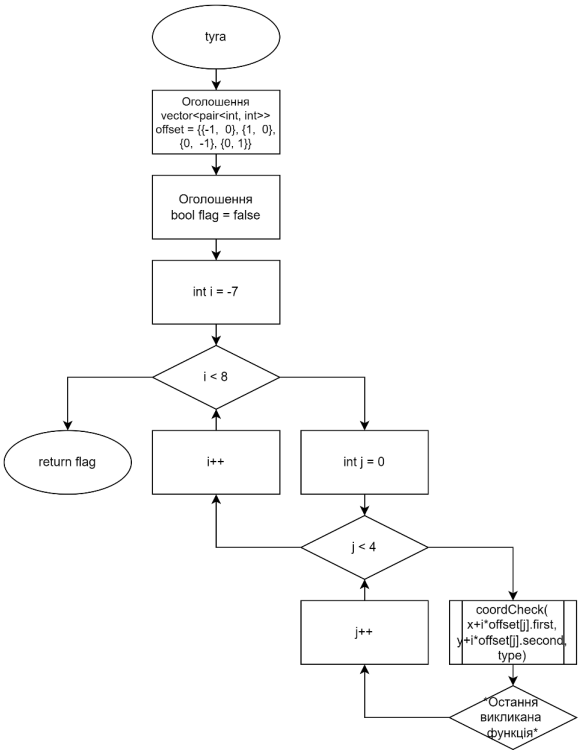


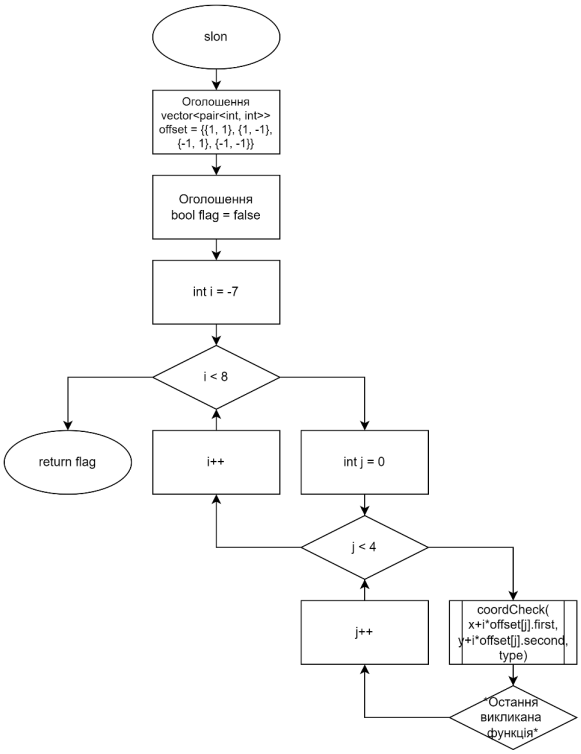

Початок
Оголошення
vector <string>
board(8)
int i = 0
i < 8
Оголошення
int q, x, y
i++
Введення
board[i]
Введення q
int i = 0
i < q
Кінець
i++
Введення
x, y
board[x-1][y-1]!='O'
Оголошення
string rez =
find_result(x-1,y-1)
Вивід
"X"
rez.empty()
Вивід
rez
Вивід
"O"


find_result
Прийом
int x, y
Оголошення
result
slon
*Остання
викликана
функція*
king
result += 'B'
*Остання
викликана
функція*
horse
result += 'K'
*Остання
викликана
функція*
tyra
result += 'N'
*Остання
викликана
функція*
pishak
result += 'R'
*Остання
викликана
функція*
slon
result += 'P'
tyra
*Одна з останніх двох
викликаних функцій*
result
result += 'Q'


king
Оголошення
vector<pair<int, int>>
offset = {{1, 0}, {1, 1},
{1, -1}, {-1, 0}, {-1, 1},
{-1, -1}, {0, 1}, {0, -1}}
Оголошення
bool flag = false
int i = 0
i < 8
return flag
i++
coordCheck(x +
offset[i].first, y +
offset[i].second,
type)
*Остання
викликана
функція*
flag = true

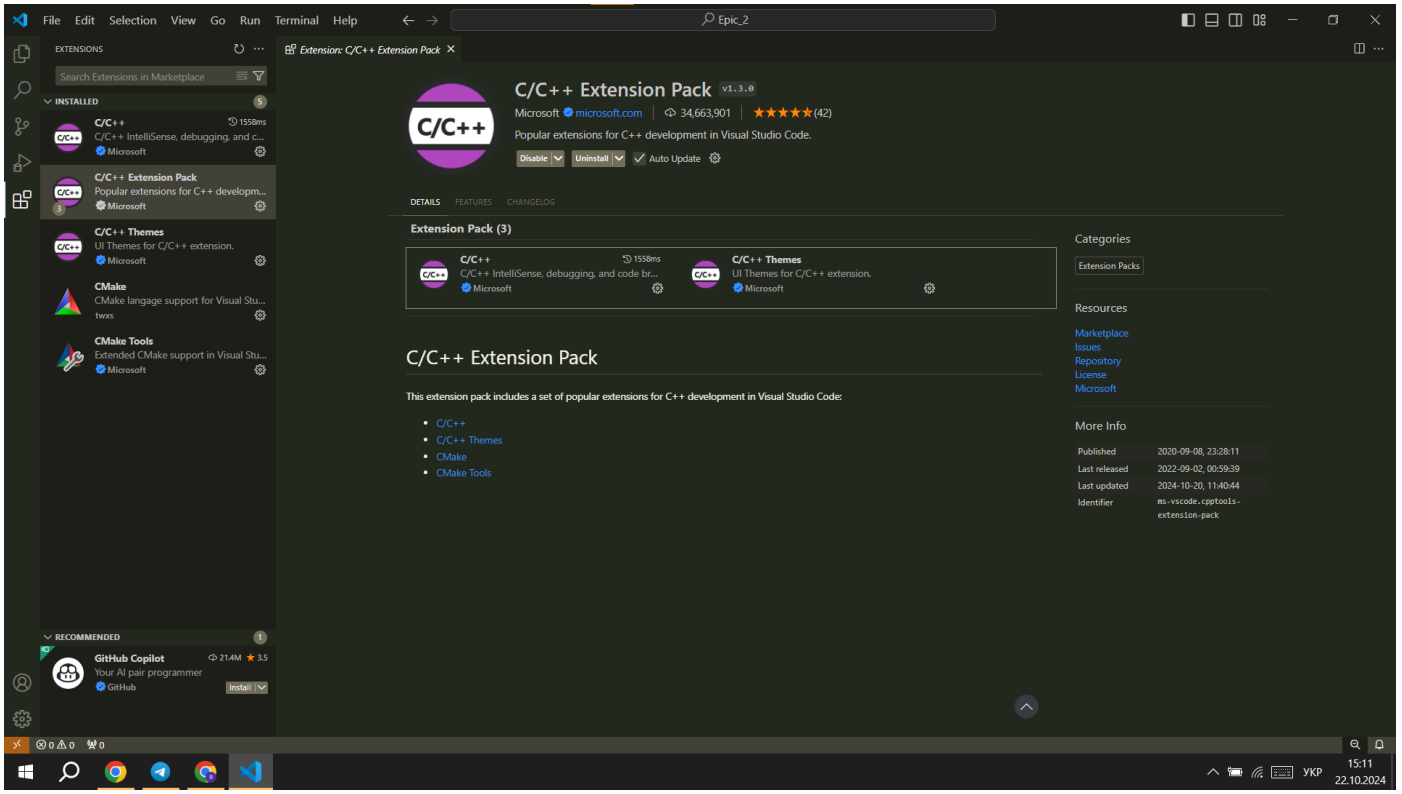

horse
Оголошення
vector<pair<int, int>>
offset = {{2, 1}, {2, -1},
{-2, 1}, {-2, -1}, {1, 2},
{1, -2}, {-1, 2}, {-1, -2}}
Оголошення
bool flag = false
int i = 0
i < 8
return flag
i++
coordCheck(x +
offset[i].first, y +
offset[i].second,
type)
*Остання
викликана
функція*
flag = true


tyra
Оголошення
vector<pair<int, int>>
offset = {{-1, 0}, {1, 0},
{0, -1}, {0, 1}}
Оголошення
bool flag = false
int i = -7
i < 8
return flag
i++
int j = 0
j < 4
j++
coordCheck(
x+i*offset[j].first,
y+i*offset[j].second,
type)
*Остання
викликана
функція*


slon
Оголошення
vector<pair<int, int>>
offset = {{1, 1}, {1, -1},
{-1, 1}, {-1, -1}}
Оголошення
bool flag = false
int i = -7
i < 8
return flag
i++
int j = 0
j < 4
j++
coordCheck(
x+i*offset[j].first,
y+i*offset[j].second,
type)
*Остання
викликана
функція*

Частина 3

Для виконання роботи використовується середовище **Visual Studio Code** зі встановленим розширенням **C/C++ Extension Pack.**



Використані бібліотеки:

- #include <stdio.h>
- #include <string.h>
- #include <iostream>
- #include <fstream>
- #include <vector>
- #include <random>
- #include <algorithm>
- #include <string>

Частина 4

Завдання №1

```c
#include <stdio.h>
#include <string.h>

int main() {
    char txt[255];

    fgets(txt, 255, stdin); // gets(txt)

    txt[strcspn(txt, "\n")] = '\0'; // для випадку з gets(txt) не потрібне

    int stop = strlen(txt);
    int size = strlen(txt);

    for (int i = 0; i != stop; i++) {
        if (txt[i] >= '0' && txt[i] <= '9') {
            int value = txt[i];
            for (int j = i; j < size; j++) {
                txt[j] = txt[j + 1];
            }
            txt[size - 1] = value;
            stop--;
            --i;
        }
    }

    for (int i = 0; i < size; i++) printf ("%c ", txt[i]);

    return 0;
}
```

Завдання №2

```cpp
#include <iostream>
#include <fstream>
#include <vector>
using namespace std;

struct Human {
    string fullName;
    int birthYear;
    int growthSm;
    int weightKg;
};

Human formElement();
void showElement();
void appendElement();
void deleteElement();

int main() {
    int choice;

    cout << "Available operations:\n";
    cout << "1. View all records\n";
    cout << "2. Add a record after the person with the specified surname\n";
    cout << "3. Delete records with the specified height and weight\n\n";
    cout << "What would you like to do?: ";
    cin >> choice;

    switch (choice) {
        case 1:
            showElement();
            break;
        case 2:
            appendElement();
            break;
        case 3:
            deleteElement();
            break;
        default:
            cout << "Invalid choice";
    }

    return 0;
}
```

```
45  Human formElement() {
46      Human createHuman;
47
48      cout << "Enter full name (Last Name, First Name, Patronymic): ";
49      cin.ignore();
50      getline(cin, createHuman.fullName);
51
52      cout << "Enter year of birth: ";
53      cin >> createHuman.birthYear;
54      cout << "Enter height (cm): ";
55      cin >> createHuman.growthSm;
56      cout << "Enter weight (kg): ";
57      cin >> createHuman.weightKg;
58
59      return createHuman;
60  }
61
62  void showElement() {
63      ifstream file("Humans.txt");
64      if (!file) {
65          cout << "File not found" << endl;
66          return;
67      }
68
69      string line;
70      while (getline(file, line)) {
71          cout << line << endl;
72      }
73
74      file.close();
75  }
76
```

```
77  void appendElement() {
78      string surname;
79      cout << "Enter the last name of the person after whom a new record will be added: ";
80      cin >> surname;
81
82      ifstream file("Humans.txt");
83      if (!file) {
84          cout << "File not found" << endl;
85          return;
86      }
87
88      vector<string> lines;
89      string line;
90      while (getline(file, line)) {
91          lines.push_back(line);
92      }
93
94      file.clear();
95      file.close();
96
97      Human newHuman = formElement();
98
99      for (int i = 0; i < lines.size(); i += 5) {
100         if (lines[i].find(surname) != string::npos) {
101             lines.insert(lines.begin() + i + 4, "");
102             lines.insert(lines.begin() + i + 5, newHuman.fullName);
103             lines.insert(lines.begin() + i + 6, to_string(newHuman.birthYear));
104             lines.insert(lines.begin() + i + 7, to_string(newHuman.growthSm));
105             lines.insert(lines.begin() + i + 8, to_string(newHuman.weightKg));
106             break;
107         }
108     }
109
110     ofstream file1("Humans.txt");
111
112     if (!file1) {
113         cout << "File not found" << endl;
114         return;
115     }
116
117     for (int i = 0; i < lines.size(); i++) file1 << lines[i] << endl;
118
119     file.close();
120 }
121
```

```
122  void deleteElement() {
123      string height, weight;
124      cout << "Enter height of the person to delete: ";
125      cin >> height;
126      cout << "Enter weight of the person to delete: ";
127      cin >> weight;
128
129      ifstream file("Humans.txt");
130      if (!file) {
131          cout << "File not found" << endl;
132          return;
133      }
134
135      vector<string> lines;
136      string line;
137      while (getline(file, line)) {
138          lines.push_back(line);
139      }
140
141      file.clear();
142      file.close();
143
144      for (int i = 0; i < lines.size(); i += 1) cout << lines[i] << " ";
145
146      for (int i = 3; i < lines.size(); i += 5) {
147          if (lines[i] == weight && lines[i - 1] == height) {
148              lines.erase(lines.begin() + i + 1);
149              lines.erase(lines.begin() + i);
150              lines.erase(lines.begin() + i - 1);
151              lines.erase(lines.begin() + i - 2);
152              lines.erase(lines.begin() + i - 3);
153
154          }
155      }
156
157      ofstream file1("Humans.txt");
158
159      if (!file1) {
160          cout << "File not found" << endl;
161          return;
162      }
163
164      for (int i = 0; i < lines.size(); i++) file1 << lines[i] << endl;
165
166      file.close();
167  }
```

Завдання №3

```cpp
#include <iostream>
#include <fstream>
#include <vector>
#include <random>

using namespace std;

void UpdateFile();

int main() {
    string answer;

    cout << "Update file?: ";
    cin >> answer;
    if (answer == "Yes" ) {
        UpdateFile();
        cout << "File updated\n";
    } else cout << "File not updated\n";

    int startCopyLine;
    cout << "Enter the line from which copying will begin: ";
    cin >> startCopyLine;

    ifstream fileOne("F1.txt");
    ofstream fileTwo("F2.txt");

    vector<string> vecOne;
    vector<string> vecTwo;

    string line;
    while (getline(fileOne, line)) vecOne.push_back(line);

    for (int i = startCopyLine - 1; i < startCopyLine + 4; i++) {
        if (i + 1 > vecOne.size()) break;

        vecTwo.push_back(vecOne[i]);
    }

    for (int i = 0; i < vecTwo.size(); i++) fileTwo << vecTwo[i] << endl;

    fileOne.close();
    fileTwo.close();

    cout << "File copied\n";

```

```
    int vowelsNum;
    for (int i = 0; i < vecTwo.size(); i++) {
        for (char j : vecTwo[i]) {
            if (j == 'a' || j == 'e' || j == 'i' || j == 'o' || j == 'u' || j == 'y') vowelsNum++;
        }
    }

    cout << "Number of vowels in F2: " << vowelsNum;

    return 0;
}

void UpdateFile() {
    ofstream file("F1.txt");

    vector<string> data;
    srand(time(nullptr));

    for (int i = 0; i < 10; ++i) {
        string str;
        for (int j = 0; j < 8; ++j) {
            char randomChar = 'a' + rand() % 26;
            str += randomChar;
        }
        data.push_back(str);
    }

    for (int i = 0; i < data.size(); i++) file << data[i] << endl;

    file.close();
}
```

Завдання №4.1

```cpp
#include <iostream>
#include <algorithm>
#include <vector>

using namespace std;

int main() {
    int N, K;
    cin >> N >> K;

    vector<int> vec(N);
    for (int i = 0; i < N; i++) cin >> vec[i];

    sort(vec.begin(), vec.end());
    auto dub = unique(vec.begin(), vec.end());
    vec.erase(dub, vec.end());

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

    cout << vec.size() << endl;
    for (int i = 0; i < vec.size(); i++) cout << vec[i] << " ";

    return 0;
}
```

Завдання №4.2

```cpp
#include <iostream>
#include <vector>

using namespace std;

int main() {
    int N, K;
    cin >> N >> K;

    vector<int> vec(N);
    for (int i = 0; i < N; i++) cin >> vec[i];

    for (int i = 0; i < N - 1; i++) {
        for (int j = i + 1; j < N; j++) {
            if (vec[i] > vec[j]) {
                int temp = vec[i];
                vec[i] = vec[j];
                vec[j] = temp;
            }
        }
    }

    for (int i = 0; i < N - 1;) {
        if (vec[i] == vec[i + 1]) {
            vec.erase(vec.begin() + i + 1);
            N--;
        } else i++;
    }


    while (K != 0) {
        vec.push_back(vec[0]);
        vec.erase(vec.begin());
        K--;
    }

    cout << vec.size() << endl;
    for (int i = 0; i < vec.size(); i++) cout << vec[i] << " ";

    return 0;
}
```

Завдання №5.1

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int main() {
    int N;
    cin >> N;

    vector<int> vec(N);
    for (int i = 0; i < N; ++i) cin >> vec[i];

    auto p1 = partition(vec.begin(), vec.end(), [](int x) {return x % 3 == 0;});
    auto p2 = partition(p1, vec.end(), [](int x) {return x % 3 == 1;});

    sort(vec.begin(), p1);
    sort(p1, p2, greater<int>());
    sort(p2, vec.end());

    vec.erase(unique(vec.begin(), vec.end()), vec.end());

    cout << vec.size() << endl;
    for (int x : vec) cout << x << " ";

    return 0;
}
```

Завдання №5.2

```cpp
#include <iostream>
#include <vector>

using namespace std;

void selectSort(vector<int> &vec, int p1, int p2, bool reverse);

int main() {
    int N;
    cin >> N;

    vector<int> vec(N);
    for (int i = 0; i < N; i++) cin >> vec[i];

    int p1 = 0, p2 = vec.size();

    for (int i = 0; i < p2;) {
        if (vec[i] % 3 == 0) {
            int element = vec[i];
            vec.erase(vec.begin() + i);
            vec.insert(vec.begin(), element);
            p1++;
            i++;
        } else if (vec[i] % 3 == 2) {
            int element = vec[i];
            vec.erase(vec.begin() + i);
            vec.insert(vec.end(), element);
            p2--;
        } else i++;
    }

    selectSort(vec, 0, p1, false);
    selectSort(vec, p1, p2, true);
    selectSort(vec, p2, vec.size(), false);

```

```
    for (int i = 0; i < N - 1;) {
        if (vec[i] == vec[i + 1]) {
            vec.erase(vec.begin() + i + 1);
            N--;
        } else i++;
    }

    cout << vec.size() << endl;
    for (int i = 0; i < N; i++) cout << vec[i] << " ";

    return 0;
}

void selectSort(vector<int> &vec, int p1, int p2, bool reverse) {
    for (int i = p1; i < p2; i++) {
        for (int j = i + 1; j < p2; j++) {
            if (!reverse) {
                if (vec[i] > vec[j]) {
                    int temp = vec[i];
                    vec[i] = vec[j];
                    vec[j] = temp;
                }
            } else {
                if (vec[i] < vec[j]) {
                    int temp = vec[i];
                    vec[i] = vec[j];
                    vec[j] = temp;
                }
            }
        }
    }
}
```

Завдання №6

```cpp
#include <iostream>
#include <vector>
using namespace std;

vector<int> possibleNumbers(vector<vector<int>>& matrix, int x, int y, int N);

int main() {
    int N;
    cin >> N;

    vector<vector<int>> matrix(N, vector<int>(N));
    for (int i = 0; i < N; ++i) {
        string row;
        cin >> row;
        for (int j = 0; j < N; ++j) {
            matrix[i][j] = row[j] - '0';
        }
    }

    int Q;
    cin >> Q;

    vector<pair<int, int>> coord(Q);
    for (int i = 0; i < Q; i++) {
        cin >> coord[i].first >> coord[i].second;
        --coord[i].first;
        --coord[i].second;
    }

    for (int i = 0; i < Q; i++) {
        vector<int> possible = possibleNumbers(matrix, coord[i].first, coord[i].second, N);
        if (!possible.empty()) {
            cout << possible.size() << endl;
            for (int num : possible) {
                cout << num << " ";
            }
        } else {
            cout << 0;
        }
        cout << endl << endl;
    }

    return 0;
}

```

```cpp
vector<int> possibleNumbers(vector<vector<int>>& matrix, int x, int y, int N) {
    if (matrix[x][y] != 0) {
        return {matrix[x][y]};
    }

    vector<int> possible(N);
    for (int i = 0; i < N; ++i) {
        possible[i] = i + 1;
    }

    for (int i = 0; i < N; ++i) {
        if (matrix[x][i] != 0) {
            for (int j = 0; j < possible.size(); j++) {
                if (matrix[x][i] == possible[j]) possible.erase(possible.begin()  + j);
            }
        }

        if (matrix[i][y] != 0) {
            for (int j = 0; j < possible.size(); j++) {
                if (matrix[i][y] == possible[j]) possible.erase(possible.begin()  + j);
            }
        }
    }

    return vector<int>(possible.begin(), possible.end());
}
```

Завдання №7

```cpp
#include <iostream>
#include <fstream>
#include <string>

using namespace std;


enum FileOpResult {
    Success,
    Failure
};


FileOpResult write_to_file(const string& name, const string& content);
FileOpResult copy_file(const char* file_from, const char* file_to);


```

```cpp
int main() {
    string choice;

    cout << "Available operations:\n";
    cout << "1. Create or overwrite a file: \n";
    cout << "2. Copy the contents of one file to another\n";
    cout << "What do you want to do?: ";
    cin >> choice;
    cin.ignore();

    if (choice == "1") {
        string file_name, content;

        cout << "Enter file name: ";
        getline(cin, file_name);

        cout << "Enter content to write: ";
        getline(cin, content);

        FileOpResult result = write_to_file(file_name, content);

        if (result == Success)  cout << "File written successfully";
        else cout << "Failed to write file";

    } else if (choice == "2") {
        char source_file[256];
        char target_file[256];

        cout << "Enter source file name: ";
        cin.getline(source_file, sizeof(source_file));

        cout << "Enter target file name: ";
        cin.getline(target_file, sizeof(target_file));

        FileOpResult result = copy_file(source_file, target_file);

        if (result == Success) {
            cout << "File copied successfully";
        } else {
            cout << "Failed to copy file";
        }

        } else cout << "Wrong input";

    return 0;
}
```

```
66  FileOpResult write_to_file(const string &name, const string &content) {
67      if (name.empty()) {
68          cerr << "Error: File name is empty." << endl;
69          return Failure;
70      }
71
72      try {
73          ofstream file(name, ios::binary);
74
75          if (!file.is_open()) {
76              cerr << "Error: Unable to open file '" << name << "'." << endl;
77              return Failure;
78          }
79
80          file << content;
81
82          if (file.fail()) {
83              cerr << "Error: Failed to write to file '" << name << "'." << endl;
84              return Failure;
85          }
86
87          file.close();
88          return Success;
89      } catch (const exception &e) {
90          cerr << "Exception: " << e.what() << endl;
91          return Failure;
92      }
93  }
94
95
```

```cpp
FileOpResult copy_file(const char *file_from, const char *file_to) {
    bool fileExist;

    ifstream fileChekOne(file_from);
    fileExist = fileChekOne.is_open();
    fileChekOne.close();
    if (!fileExist) {
        cout << "Sourse file doesn't exist\n";
        return Failure;
    }

    ifstream fileChekTwo(file_to);
    fileExist = fileChekTwo.is_open();
    fileChekTwo.close();
    if (!fileExist) {
        cout << "Target file doesn't exist\n";
        return Failure;
    }
    try {
        ifstream input_file(file_from, ios::binary);
        ofstream output_file(file_to, ios::binary);

        const size_t buffer_size = 4096;
        char buffer[buffer_size];
        while (input_file.read(buffer, buffer_size)) {
            output_file.write(buffer, input_file.gcount());
        }

        if (input_file.eof()) {
            output_file.write(buffer, input_file.gcount());
        } else {
            cerr << "Error: Failed during file read/write operations\n";
            return Failure;
        }

        if (output_file.fail()) {
            cerr << "Error: Failed to write to target file\n";
            return Failure;
        }
        return Success;
    } catch (const exception &e) {
        cerr << "Exception: " << e.what() << endl;
        return Failure;
    }
}
```

Завдання №8

```cpp
#include <iostream>
#include <vector>

using namespace std;
vector<string> board(8);

bool king(int x, int y, char type);
bool tyra(int x, int y, char type);
bool slon(int x, int y, char type);
bool horse(int x, int y, char type);
bool pishak(int x, int y, char type);
bool coordCheck(int x, int y, char type);
string find_result(int x, int y);

int main() {

    for (int i = 0; i < 8; i++) {
        cin >> board[i];
    }

    int q, x, y;
    cin >> q;
    for (int i = 0; i < q; i++) {
        cin >> x >> y;
        if(board[x-1][y-1]!='0') cout << 'X' << endl;
        else {
            string rez = find_result(x - 1, y - 1);
            if (rez.empty()) cout << '0' << endl;
            else cout << rez << endl;
        }
    }
}

bool coordCheck(int x, int y, char type) {
    if (((0 <= x) && (x < 8) && (0 <= y) && (y < 8)) && board[x][y] == type)
        return true;
    else
        return false;
}

```

```cpp
bool king(int x, int y, char type) {
    vector<pair<int, int>> offset = {{1,  0}, {1,  1}, {1,  -1}, {-1, 0}, {-1, 1}, {-1, -1}, {0,  1}, {0,  -1}};
    bool flag = false;
    for (int i = 0; i < 8; i++)
        if (coordCheck(x + offset[i].first, y + offset[i].second, type))
            flag = true;
    return flag;
}

bool tyra(int x, int y, char type) {
    vector<pair<int, int>> offset = {{-1, 0}, {1, 0}, {0, -1}, {0, 1}};
    bool flag = false;
    for (int i = -7; i < 8; i++) {
        for (int j = 0; j < 4; j++) {
            if (coordCheck(x + i * offset[j].first, y + i * offset[j].second, type))
                flag = true;
        }
    }
    return flag;
}

bool slon(int x, int y, char type) {
    vector<pair<int, int>> offset = {{1, 1}, {1, -1}, {-1, 1}, {-1, -1}};
    bool flag = false;
    for (int i = -7; i < 8; i++) {
        for (int j = 0; j < 4; j++) {
            if (coordCheck(x + i * offset[j].first, y + i * offset[j].second, type))
                flag = true;
        }
    }
    return flag;
}

bool horse(int x, int y, char type) {
    bool flag = false;
    vector<pair<int, int>> offset = {{2, 1}, {2, -1}, {-2, 1}, {-2, -1}, {1, 2}, {1, -2}, {-1, 2}, {-1, -2}};
    for (int i = 0; i < 8; i++) {
        if (coordCheck(x + offset[i].first, y + offset[i].second, type))
            flag = true;
    }
    return (flag);
}


bool pishak(int x, int y, char type) {
    if (((0 <= x - 1) && (0 <= y - 1) && (board[x - 1][y - 1] == type)) ||
        ((0 <= x - 1) && (y + 1 <= 7) && (board[x - 1][y + 1] == type)))
        return true;
    else return false;
}

string find_result(int x, int y) {
    string result;

    if (slon(x, y, 'B')) result += "B";
    if (king(x, y, 'K')) result += "K";
    if (horse(x, y, 'N')) result += "N";
    if (tyra(x, y,'R')) result += "R";
    if (pishak(x, y, 'P')) result += "P";
    if (slon(x, y,'Q') || tyra(x, y,'Q')) result += "Q";

    return result;
}
```

Частина 5

Завдання №1

```
d5d6f4fd8f6g4e3688d75f
d d f f d f g e d f 5 6 4 8 6 4 3_6 8 8 7 5
```

Орієнтовний час виконання: 1 год

Фактично затрачений час: 1 год

Завдання №2

```
Available operations:
1. View all records
2. Add a record after the person with the specified surname
3. Delete records with the specified height and weight

What would you like to do?: 1
Hnatiuk Yaroslav Bohdanovich
2007
177
60

Boba Aboba Abobovuch
2024
100
3

Lisov Ivan Oleksandrovich
2007
188
80

Black John Batkovich
1996
192
83
```

```
Available operations:
1. View all records
2. Add a record after the person with the specified surname
3. Delete records with the specified height and weight

What would you like to do?: 2
Enter the last name of the person after whom a new record will be added: Boba
Enter full name (Last Name, First Name, Patronymic): Petrenko Mykola Svaniv
Enter year of birth: 2001
Enter height (cm): 179
Enter weight (kg): 90
```

```
Hnatiuk Yaroslav Bohdanovich
2007
177
60

Boba Aboba Abobovuch
2024
100
3

Petrenko Mykola Svaniv
2001
179
90

Lisov Ivan Oleksandrovich
2007
188
80

Black John Batkovich
1996
192
83
```

Орієнтовний час виконання: 2 год

Фактично затрачений час: 3 год

Завдання №3

```
Update file?: Yes
File updated
Enter the line from which copying will begin: 5
File copied
Number of vowels in F2: 14
```

F1:

```
 1   kxohbist
 2   krylihvw
 3   dswfeytf
 4   iufjlteo
 5   yruzdrln
 6   nemhpigg
 7   ocunpiyg
 8   bblduoxa
 9   abrftyze
10   qcfqeakc
```

F2:

```
1   yruzdrln
2   nemhpigg
3   ocunpiyg
4   bblduoxa
5   abrftyze
```

Орієнтовний час виконання: 2 год

Фактично затрачений час: 2 год

Завдання №4.1

```
10 4
4 9 6 2 6 4 9 6 3 1
6
6 9 1 2 3 4
```

Орієнтовний час виконання: 30 хв

Фактично затрачений час: 30 хв

Завдання №4.2

```
10 4
4 9 6 2 6 4 9 6 3 1
6
6 9 1 2 3 4
```

Орієнтовний час виконання: 1 год

Фактично затрачений час: 1 год

Завдання №5.1

```
10
5 2 7 9 6 3 4 8 6 2
8
3 6 9 7 4 2 5 8
```

Орієнтовний час виконання: 30 хв

Фактично затрачений час: 30 хв

Завдання №5.2

```
10
5 2 7 9 6 3 4 8 6 2
8
3 6 9 7 4 2 5 8
```

Орієнтовний час виконання: 1 год

Фактично затрачений час: 1 год

Завдання №6

```
3
000
100
003
3
1 1
2 3
2 1
2
2 3

1
2

1
1
```

Орієнтовний час виконання: 1 год

Фактично затрачений час: 1.5 год

Завдання №7

```
Available operations:
1. Create or overwrite a file:
2. Copy the contents of one file to another
What do you want to do?: 1
Enter file name: Boooom
Enter content to write: Bim Bam Boom Boom
File written successfully

Available operations:
1. Create or overwrite a file:
2. Copy the contents of one file to another
What do you want to do?: 2
Enter source file name: Cooool
Enter target file name: Boooom
File copied successfully
```

Орієнтовний час виконання: 3 год

Фактично затрачений час: 5 год

Завдання №8

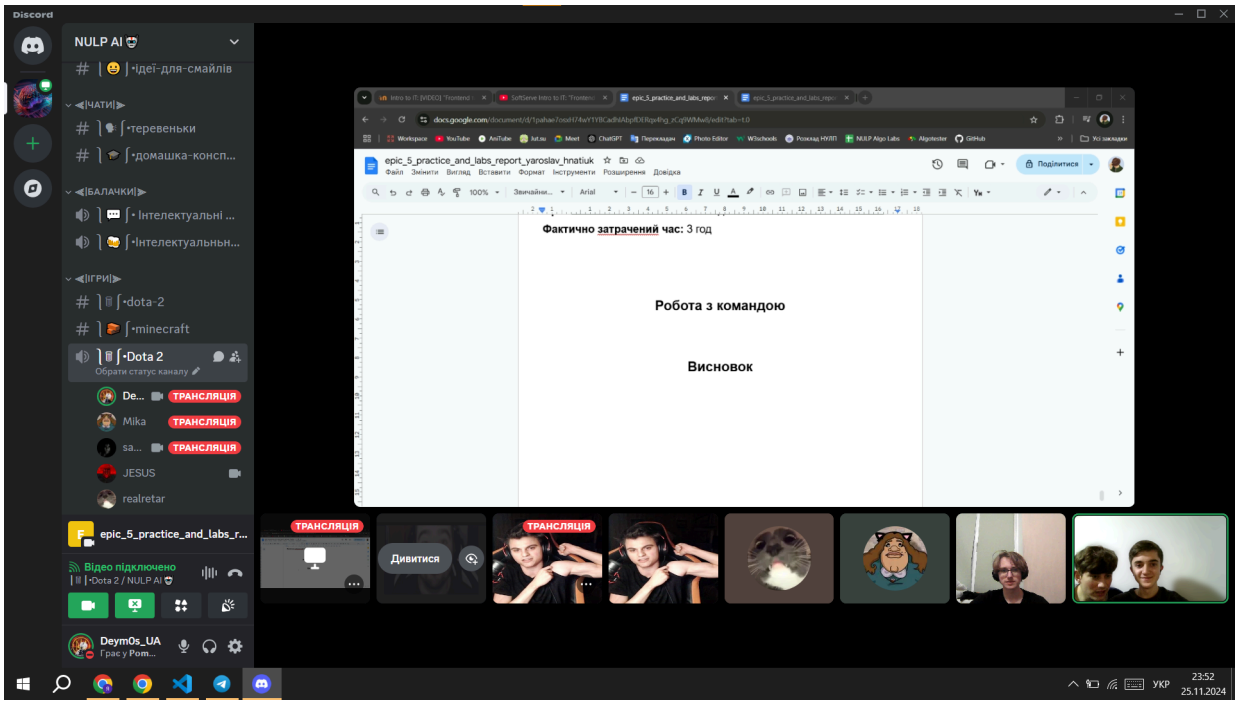
```
OOOOOOOO
ORQOOOOO
OONOOOOO
OOOOPOOO
OOOOOOOO
OOOOOOOO
KOQOOOOO
OOOOOOOR

7
8 1
KR
1 2
NR
5 4
NP
5 1
Q
6 2
KRQ
8 4
RQ
6 7
O
```

Орієнтовний час виконання: 1 год

Фактично затрачений час: 3 год

Робота з командою



Висновок

Я ознайомився з основами роботи з текстовими та бінарними файлами, стандартними бібліотеками та методами їх використання. Навчився працювати з символами, рядковими змінними й файлами, а також створювати та застосовувати власні бібліотеки для оптимізації коду.