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

Виконав(ла):
Студентка групи ШІ-11
Ільящук Марта Тарасівна

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: Навчитись працювати з файлами, символами, рядковими змінними, бібліотеками.

Теоретичні відомості:

1. Теоретичні відомості з переліком важливих тем:
    - Тема №1. Вступ до Роботи з Файлами
    - Тема №2. Символи і Рядкові Змінні
    - Тема №3. Текстові Файли
    - Тема №4. Бінарні Файли
    - Тема №5. Стандартна бібліотека та робота з файлами
    - Тема №6. Створення й використання бібліотек

2. Індивідуальний план опрацювання теорії:
    - Тема №1. Вступ до Роботи з Файлами
      Джерела інформації: Basics of File Handling in C - GeeksforGeeks
      Що опрацьовано: Основні операції з файлами, перевірка стану файлу

    - Тема №2. Символи і Рядкові Змінні
      Джерела інформації: C Strings, C String Functions
      Що опрацьовано: Робота з char та string

    - Тема №3. Текстові Файли
      Джерела інформації: C++ File handling: Create text file and write text, C Files I/O: Opening, Reading, Writing and Closing a file
      Що опрацьовано: Особливості читання та запису текстових файлів, обробка рядків

    - Тема №4. Бінарні Файли
      Джерела інформації: Reading and writing binary file in C/C++, How to store records in binary files in C++
      Що опрацьовано: Робота з бінарними файлами, відмінності від текстових

    - Тема №5. Стандартна бібліотека та робота з файлами
      Джерела інформації: C++ fstream Library (File Streams) Reference
      Що опрацьовано: Стандартна бібліотека для роботи з файлами, потоки вводу/виводу: ifstream, ofstream, fstream

- Тема №6. Створення й використання бібліотек
  Джерела інформації: How to write your own header file in C? - GeeksforGeeks
  Що опрацьовано: Вступ до створення власних бібліотек у C++

Виконання роботи:

1. Опрацювання завдань та вимог до середовища:

Завдання №1. VNS Lab 6 Variant 8

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:
Перетворити рядок так, щоб всі слова в ньому стали ідентифікаторами, слова, які складаються тільки із цифр - знищити.

Завдання №2. VNS Lab 8 Variant 8

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.

8. Структура "Покупець":
   - прізвище, ім'я, по батькові;
   - домашня адреса;
   - номер телефону;
   - номер кредитної картки.

Знищити 3 елементи з початку файлу, додати 3 елементи в кінець файлу.

Завдання №3. VNS Lab 9 Variant 8

Створити текстовий файл F1 не менше, ніж з 10 рядків і записати в нього інформацію:
1) Скопіювати з файлу F1 у файл F2 всі рядки, які не містять цифри.
2) Підрахувати кількість рядків, які починаються на букву «А» у файлі F2.

Завдання №4. Algotester Lab 4.1 Variant 3
(std::partition + std::sort + std::unique)
Завдання №5. Algotester Lab 4.2 Variant 3
(Зі своєю реалізацією)

Вам дано масив, який складається з N додатніх 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.

Вхідні дані

У першому рядку N - кількість чисел.

У другому рядку N чисел a_i - елементи масиву.

Вихідні дані

У першому рядку M - кількість чисел у масиву

У другому рядку M посоротованих за умовою чисел.

Обмеження

$1 \le N \le 10^3$

$0 \le a_i \le 10^3$

Приклади

Вхідні дані (*stdin*)	Вихідні дані (*stdout*)
10 1 33 4 8 6 5 2 7 5 0	9 0 6 33 7 4 1 2 5 8

Завдання №6. Algotester Lab 6 Variant 2

Lab 6v2

Обмеження: 2 сек., 256 МіБ

У вас є шахова дошка розміром 8×8 та дуже багато фігур.

Кожна клітинка може мати таке значення:

- 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 $\{x, y\}$. На кожен запит ви маєте вивести стрічку s_i - посортовані за алфавітом букви фігур, які атакують цю клітинку (пішаки атакують вниз).

У випадку, якщо на клітинці стоїть якась фігура - виведіть символ X.

У 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.

Вхідні дані

У перших 8 рядках стрічка row_i - стан i-го рядка дошки.

У наступному рядку ціле число Q - кількість записів

У наступних Q рядках 2 цілих числа x та y - координати клітинки

Вихідні дані

Q разів відповідь у наступному форматі:

Строка $result$ - усі фігури, які атакують клітинку з запиту.

Завдання №7. Class Practice Task

Задача №1 – 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.

Мета задачі

Розуміння методів роботи з файлами: Робота з файлами є одним з базових навиків програмування. Реалізація функції створення та запису в файл допоможе освоїти практичні навики роботи з файлами з використанням стандартної бібліотеки C++. Для виконання завдання студент має навчитись використовувати методи відкриття файла, запису масиву даних у файл, закриття файла та обробки помилок чи станів операції на кожному з етапів.

Розвиток алгоритмічне мислення: Запис у файл включає набір операції, які якнайкраще вкладаються в концепцію алгоритма, як списка детальних кроків. Імплементація цієї функції наочно демонструє створення алгоритмів у програмуванні.

Освоїти навики роботи з текстовими стрічками: завдання допоможе освоїти роботу з С стрічка, які є масивами з нульовим символом в кінці. Типові концепції при роботі з С стрічками це арифметика вказівників, ітерація по стрічці, копіювання частини стрічки, розбиття на токени по заданому символу.

Розвинути навички розв'язувати задачі: Запис у файл може супроводжуватись набором станів (немає доступу на створення, недостатньо місця, ін.), які необхідно передбачити у алгоритмі. Аналіз цих станів дозволяє розвинути навик розв'язання інженерних задач у програмуванні.

Задача №2 – 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.

Мета задачі

Розуміння методів роботи з файлами: Робота з файлами є одним з базових навиків програмування. Реалізація функції копіювання вмісту файла допоможе освоїти практичні навики роботи з файлами з використанням стандартної бібліотеки C++. Для виконання завдання студент має навчитись використовувати методи відкриття файла, читання вмісту файла, запису масиву даних у файл, закриття файла та обробки помилок чи станів операції на кожному з етапів.

Розвиток алгоритмічне мислення: Читання та запис у файл включає набір операцій, які якнайкраще вкладаються в концепцію алгоритма, як списка детальних кроків. Імплементація цієї функції наочно демонструє створення алгоритмів у програмуванні.

Освоїти навики роботи з потоком даних: завдання допоможе освоїти роботу з потоками даних (концепція реалізована в STL як набір класів *stream* - fstream, stringstream, streambuf та ін.). Концепція потоку даних дозволяє абстрагувати роботу з джерелами та приймачами даних та писати з її допомогою високорівневий код.

Розвинути навички розв'язувати задачі: Операції читання з файла та запис у файл можуть супроводжуватись набором різних станів (немає доступу на читання чи створення, недостатньо місця, ін.), які необхідно передбачити у алгоритмі. Аналіз цих станів дозволяє розвинути навик розв'язання інженерних задач у програмуванні.

Завдання №8. Self Practice Task Algotester Lab 4.1 Variant 2

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.

Вхідні дані

У першому рядку цілі числа N та K
У другому рядку N цілих чисел - елементи масиву a

Вихідні дані

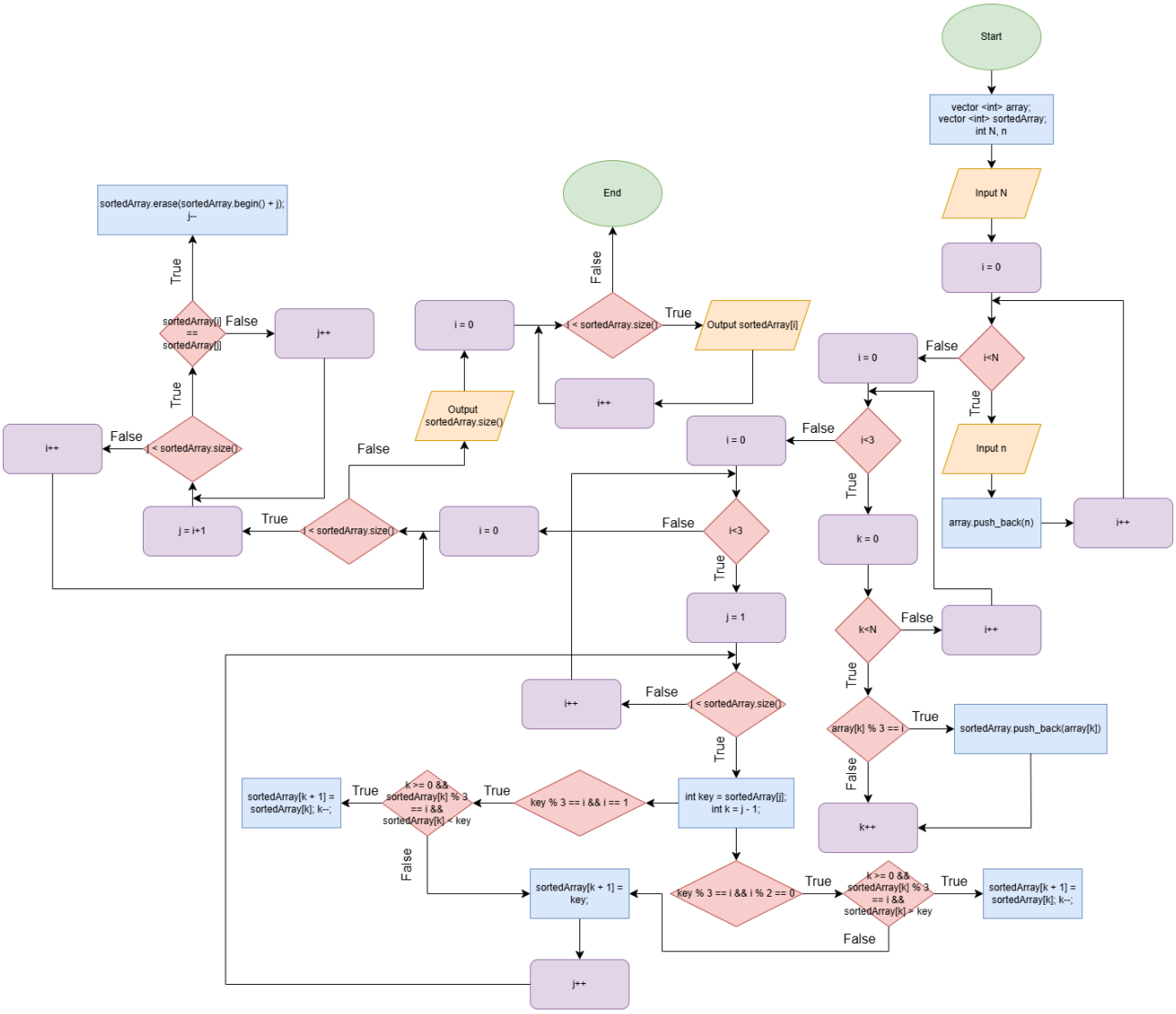
У першому рядку ціле число N - розмір множини a
У наступному рядку N цілих чисел - множина a

Обмеження

$1 \le N, K \le 1000$
$0 \le a_i \le 100$

2. Дизайн виконання завдань

Завдання №5. Algotester Lab 4.2 Variant 3



3. Код програм з посиланням на зовнішні ресурси

[Epic 5 - Marta Iliashchuk by martailiashchuk · Pull Request #593 · artificial-intelligence-department/ai_programming_playground_2024 · GitHub]

Завдання №1. VNS Lab 6 Variant 8

```cpp
#include <iostream>
#include <string.h>
using namespace std;

int main(){
    string result;
    char str[256];
```

```
    cout<<"Enter a string: ";
    gets(str);

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

    while (token != nullptr){
        bool isNumeric = true;
        for (int i = 0; token[i] != '\0'; i++){
            if (!isdigit(token[i])){
                isNumeric = false;
                result+=token;
                result+="_";
                break;
            }
        }
        token = strtok(nullptr, " ");
    }

    result.pop_back();

    cout<<"Changed string: "<<result<<endl;

    return 0;
}
```

Завдання №2. VNS Lab 8 Variant 8

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

struct Buyer{
    char fullName[66];
    char homeAddress[66];
    int mobilePhoneNumber;
    int creditCardNumber;
};

void createFile(const char *filename){
    FILE* f;
    f = fopen(filename, "wb");

    if(f==NULL){
        cout<<"Error opening the file";
        exit(1);
    }

    int number;
    cout<<"Enter number of buyers: ";
    cin>>number;

    Buyer b;
```

```
    for (int i=0; i<number; i++){
        cout<<"Enter fullname: ";
        cin>>b.fullName;

        cout<<"Enter address: ";
        cin>>b.homeAddress;

        cout<<"Enter phone number: ";
        cin>>b.mobilePhoneNumber;

        cout<<"Enter credit card number: ";
        cin>>b.creditCardNumber;

        fwrite(&b, sizeof(Buyer), 1, f);

        if(ferror(f)){
            cout<<"Error writing data to the file";
            exit(2);
        }
    }
    fclose(f);
}

void printFile(const char *filename){
    FILE* f;
    f = fopen(filename, "rb");

    if(f==NULL){
        cout<<"Error opening the file";
        exit(1);
    }

    Buyer b;

    cout<<endl<< "File: "<<endl;

    while (fread(&b, sizeof(Buyer), 1, f)) {
        cout<<"Fullname: "<<b.fullName<<endl;
        cout<< "Home address: "<<b.homeAddress<<endl;
        cout<< "Phone number: " <<b.mobilePhoneNumber<<endl;
        cout<<"Credit card number: "<<b.creditCardNumber<<endl;
        }
    fclose(f);
}

void deleteBuyerData(const char *filename){
    FILE* f;
    f = fopen(filename, "rb");

    if(f==NULL){
        cout<<"Error opening the file";
        exit(1);
    }
```

```
    vector<Buyer> buyers;
    Buyer b;
    int count = 0;

    while (fread(&b, sizeof(Buyer), 1, f)){
        if(count>=3){
            buyers.push_back(b);
        }
        count++;
    }
    fclose(f);

    f = fopen(filename, "wb");

    if(f==NULL){
        cout<<"Error opening the file";
        exit(1);
    }

    for (const auto& buyer : buyers) {
        fwrite(&buyer, sizeof(Buyer), 1, f);
    }

    if(ferror(f)){
        cout<<"Error writing data to the file";
        exit(2);
    }
    fclose(f);
}

void addBuyerData(const char *filename){
    FILE* f;
    f = fopen(filename, "rb");

    if(f==NULL){
        cout<<"Error opening the file";
        exit(1);
    }

    vector<Buyer> buyers;
    Buyer b;
    int count = 0;

    while (fread(&b, sizeof(Buyer), 1, f)){
        buyers.push_back(b);
    }

    for (int i=0; i<3; i++){
        cout<<"Enter fullname: ";
        cin>>b.fullName;

        cout<<"Enter address: ";
```

```
        cin>>b.homeAddress;

        cout<<"Enter phone number: ";
        cin>>b.mobilePhoneNumber;

        cout<<"Enter credit card number: ";
        cin>>b.creditCardNumber;

        buyers.push_back(b);
    }
    fclose(f);

    f = fopen(filename, "wb");

    if(f==NULL){
        cout<<"Error opening the file";
        exit(1);
    }

    for (const auto& buyer : buyers) {
        fwrite(&buyer, sizeof(Buyer), 1, f);
    }

    if(ferror(f)){
        cout<<"Error writing data to the file";
        exit(2);
    }
    fclose(f);
}


int main(){
    const char* fileName = "buyerdata.dat";

    createFile(fileName);
    printFile(fileName);
    deleteBuyerData(fileName);
    printFile(fileName);
    addBuyerData(fileName);
    printFile(fileName);

    return 0;
}
```

Завдання №3. VNS Lab 9 Variant 8

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

int main() {
    string result;
    string input;
```

```
    vector<string> lines;
    string A = "A";
    int count = 0;

    FILE* first = fopen("F1.txt", "w");
    FILE* second = fopen("F2.txt", "w");

    if (first == nullptr || second == nullptr) {
        cerr << "Error opening file" << endl;
        exit(1);
    }

    cout << "Enter 10 strings:" << endl;

    for (int i = 0; i < 10; i++) {
        cout << i + 1 << ") ";
        getline(cin, input);

        fputs(input.c_str(), first);
        fputs("\n", first);

        string noDigits;
        for (char wordInLine : input) {
            if (!isdigit(wordInLine)) {
                noDigits += wordInLine;
            }
        }

        lines.push_back(noDigits);
    }

    fclose(first);

    for (const string& line : lines) {
        fputs(line.c_str(), second);
        fputs("\n", second);
    }

    fclose(second);

    for (string line : lines){
        if(line[0] == A[0]){
            count++;
        }
    }

    cout << "Done" << endl;
    cout<<count;
    return 0;
}
```

Завдання №4 - 5. Algotester Lab 4 Variant 3

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

using namespace std;
int main(){
    int N;
    cin >> N;
    vector<int> a(N);
    for (int i = 0; i < N; ++i) cin >> a[i];

    set<int> unique_elements(a.begin(), a.end());
    vector<int> unique_array(unique_elements.begin(), unique_elements.end());

    vector<int> group0, group1, group2;
    for (int x : unique_array) {
        if (x % 3 == 0)
            group0.push_back(x);
        else if (x % 3 == 1)
            group1.push_back(x);
        else
            group2.push_back(x);
    }

    sort(group0.begin(), group0.end());
    sort(group1.rbegin(), group1.rend());
    sort(group2.begin(), group2.end());

    vector<int> result;
    result.insert(result.end(), group0.begin(), group0.end());
    result.insert(result.end(), group1.begin(), group1.end());
    result.insert(result.end(), group2.begin(), group2.end());

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

    return 0;
}
```

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

int main() {
    vector<int> array;
    vector<int> sortedArray;
    int N, n;
    cin >> N;
```

```
    for (int i = 0; i < N; i++) {
        cin >> n;
        array.push_back(n);
    }

    for (int i = 0; i < 3; i++) {
        for (int k = 0; k < N; k++) {
            if (array[k] % 3 == i) {
                sortedArray.push_back(array[k]);
            }
        }
    }

    // Сортування вставками
    for (int i = 0; i < 3; i++) {
        for (int j = 1; j < sortedArray.size(); j++) {
            int key = sortedArray[j];
            int k = j - 1;

            if (key % 3 == i && i % 2 == 0) {
                while (k >= 0 && sortedArray[k] % 3 == i && sortedArray[k] > key) {
                    sortedArray[k + 1] = sortedArray[k];
                    k--;
                }
            }
            else if (key % 3 == i && i == 1) {
                while (k >= 0 && sortedArray[k] % 3 == i && sortedArray[k] < key) {
                    sortedArray[k + 1] = sortedArray[k];
                    k--;
                }
            }
            sortedArray[k + 1] = key;
        }
    }

    for (int i = 0; i < sortedArray.size(); i++) {
        for (int j = i + 1; j < sortedArray.size(); j++) {
            if (sortedArray[i] == sortedArray[j]) {
                sortedArray.erase(sortedArray.begin() + j);
                j--;
            }
        }
    }


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

    return 0;
}
```

Завдання №6. Algotester Lab 6 Variant 2

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

using namespace std;

bool movePiece(char element, int elementX, int elementY, int x, int y) {
    switch (element) {
        case 'R':
            return elementX == x || elementY == y;
        case 'B':
            return abs(elementX - x) == abs(elementY - y);
        case 'N':
            return (abs(elementX - x) == 2 && abs(elementY - y) == 1) ||
                   (abs(elementX - x) == 1 && abs(elementY - y) == 2);
        case 'P':
            return elementX + 1 == x && abs(elementY - y) == 1;
        case 'K':
            return abs(elementX - x) <= 1 && abs(elementY - y) <= 1;
        case 'Q':
            return (elementX == x || elementY == y) ||
                   (abs(elementX - x) == abs(elementY - y));
        default:
            return false;
    }
}

int main() {
    vector<string> rows(8);

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

    int Q;
    cin >> Q;
    vector<string> result(Q);

    for (int k = 0; k < Q; k++) {
        int x, y;
        cin >> x >> y;
        x--;
        y--;

        if (rows[x][y] != 'O') {
            result[k] = "X";
            continue;
        }
```

```
        set<char> pieces;

        for (int i = 0; i < 8; i++) {
            for (int j = 0; j < 8; j++) {
                char element = rows[i][j];
                if (element != 'O' && movePiece(element, i, j, x, y)) {
                    pieces.insert(element);
                }
            }
        }

        if (pieces.empty()) {
            result[k] = "O";
        } else {
            string attackPiece(pieces.begin(), pieces.end());
            sort(attackPiece.begin(), attackPiece.end());
            result[k] = attackPiece;
        }
    }

    for (const string& r : result) {
        cout << r << endl;
    }

    return 0;
}
```

Завдання №7. Class Practice Task

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

enum FileOpResult {Success, FailureCreate, FailureOpen, FailureWrite};

FileOpResult write_to_file(char *name, char *content){

    if (name == nullptr){
        return FailureCreate;
    }

    ofstream file(name);
    if (!file.is_open()) {
        return FailureOpen;
    }

    file<<content;

    if(file.fail()){
        return FailureWrite;
    }
```

```
    file.close();
    return Success;
}

int main(){
    char fileName[124];
    char content[777];

    cout<<"Enter filename: ";
    cin.getline(fileName, 100);

    cout<<"Enter content: ";
    cin.getline(content, 1000);

    FileOpResult result = write_to_file(fileName, content);

    if (result == Success) {
        cout << "Data successfully written to the file";
    }
    else if (result == FailureCreate) {
        cout << "Failed to create the file: missing filename";
    }
    else if (result == FailureOpen) {
        cout << "Error opening the file";
    }
    else if (result == FailureWrite) {
        cout << "Error writing data to the file";
    }

    return 0;
}
```

Завдання №8. Self Practice Task Algotester Lab 4.1 Variant 2

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

using namespace std;

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

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

    set<int> uniqueElements(a.begin(), a.end());
    vector<int> uniqueArray(uniqueElements.begin(), uniqueElements.end());
```

```
    sort(uniqueArray.begin(), uniqueArray.end());

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

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

    return 0;
}
```

4. Результати виконання завдань

Завдання №1. VNS Lab 6 Variant 8

```
Enter a string: dbdhsdn555dhk dkjflk555 jk55 555 44h
Changed string: dbdhsdn555dhk_dkjflk555_jk55_44h
```

Завдання №2. VNS Lab 8 Variant 8

```
Enter number of buyers: 3
Enter fullname: aaa
Enter address: bbb
Enter phone number: 111
Enter credit card number: 222
Enter fullname: fff
Enter address: gg
Enter phone number: 2223
Enter credit card number: 444
Enter fullname: jj
Enter address: sss
Enter phone number: 555
Enter credit card number: 77

File:
Fullname: aaa
Home address: bbb
Phone number: 111
Credit card number: 222
Fullname: fff
Home address: gg
Phone number: 2223
Credit card number: 444
Fullname: jj
Home address: sss
Phone number: 555
Credit card number: 77
```

Завдання №3. VNS Lab 9 Variant 8

```
Enter 10 strings:
1) fksss4446 778
2) kjfn,s45 kmld
3) d.d.d 5565
4) A,m,mf
5) kfjsfkn,
6) nd,sfm,A45
7) fdm,d,.33
8) d,fms,A
9) AAAAA
10) AA333
Done
3
```

F1.txt

```
1   fksss4446 778
2   kjfn,s45 kmld
3   d.d.d 5565
4   A,m,mf
5   kfjsfkn,
6   nd,sfm,A45
7   fdm,d,.33
8   d,fms,A
9   AAAAA
10  AA333
11
```

F2.txt

```
1   fksss
2   kjfn,s kmld
3   d.d.d
4   A,m,mf
5   kfjsfkn,
6   nd,sfm,A
7   fdm,d,.
8   d,fms,A
9   AAAAA
10  AA
11
```

Завдання №4 - 5. Algotester Lab 4 Variant 3

```
5
3
0
6
7
4
5
0 3 6 7 4
```

Завдання №6. Algotester Lab 6 Variant 2

```
OOOOOR
OOOOBO
OONOOO
POOOQO
OOKOOO
OOOOOO
OOOOOP
OOOOOO
2
1
6
4
1
X
X
```

Завдання №7. Class Practice Task

```
Enter filename: named
Enter content: content of named
Data successfully written to the file
```

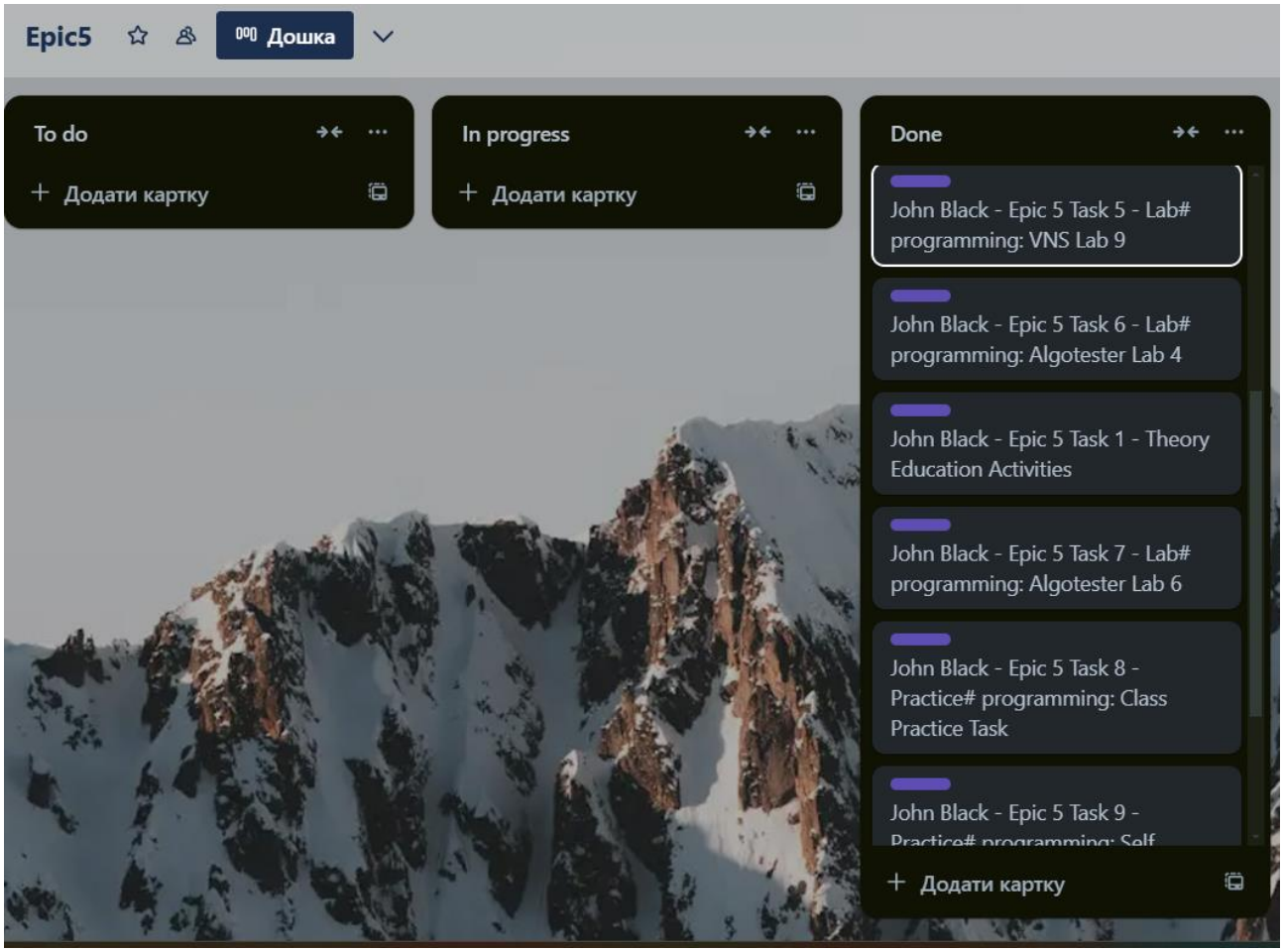
named

```
1    content of named
```

Завдання №8. Self Practice Task Algotester Lab 4.1 Variant 2

```
5
1
2
3
4
5
3
4
3 4 5 2
```

5. Кооперація з командою



Висновок: Під час виконання роботи я теоретично ознайомилась та на практиці закріпила знання про роботу з текстовими, бінарними файлами у мові C/C++.