Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Лабораторних та практичних робіт № 5**

***з дисципліни:*** «Мови та парадигми програмування»

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

***Виконав:***

Студент групи ШІ-13

Роман Літовінський

# **Тема роботи:**

Робота із двійковими файлами. Ввід-вивід рядків.

# **Мета роботи:**

Ознайомитись з організацією вводу-виводу структурованої інформації із зовнішніх носіїв. Навчитися працювати з текстовими файлами, ввід-вивід текстової інформації і її зберігання на зовнішніх носіях.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Потоковий ввід-вивід
* Тема №2: Бінарні файли
* Тема №3: Робота з файлами

1. Індивідуальний план опрацювання теорії:

* Тема №1: Потоковий ввід-вивід.
  + Джерела Інформації
    - <http://cpp.dp.ua/potokove-vvedennya-vyvedennya/>
    - <https://acode.com.ua/urok-215-potoky-vvodu-i-vyvodu/>
  + Що опрацьовано:
    - Ввід/вивід в C++
    - Стандартні потоки С++
  + Статус: Ознайомлений
  + Початок опрацювання теми: 09.12.23
  + Звершення опрацювання теми: 11.12.23
* Тема №2: Блоковий ввід-вивід
  + Джерела Інформації:
  + https://purecodecpp.com/uk/archives/2751Що опрацьовано:
    - Функції, що використовуються для блочного вводу та виводу
  + Статус: Ознайомлений
  + Початок опрацювання теми: 05.12.23
  + Звершення опрацювання теми: 10.11.23

* Тема №3: Робота з файлами
  + Джерела Інформації:
    - http://www.kytok.org.ua/post/vector-u-cplusplus
    - https://www.bestprog.net/uk/2021/10/12/c-the-vector-class-methods-that-define-and-modify-the-general-characteristics-of-array-ua/
    - https://acode.com.ua/urok-99-vvedennya-v-std-vector/
  + Статус: Ознайомлений
  + Початок опрацювання теми: 10.12.23
  + Звершення опрацювання теми: 12.12.23

**Виконання роботи:**

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

**Завдання №1 Опрацювання теорії**

* Деталі завдання:

Опрацювання різноманітних матеріалів, аналіз та дослідження відео, статей, книг на задані теми. Організація та структурування отриманих даних для можливості ефективніше засвоювати отримані знання на практиці. Вивчення найважливіших моментів.

**Завдання №2 Перегляд вимог та проектування**

* Деталі завдання

Проектування блок-схем в Draw.io.

**Завдання №3 VNS Lab 6**

* Варіант завдання: 23
* 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.

**Завдання №4 VNS Lab 8**

* Варіант завдання: 5
* Деталі завдання:

Сформувати двійковий файл із елементів, заданої у варіанті структури,

роздрукувати його вмістиме, виконати знищення й додавання елементів у

відповідності зі своїм варіантом.

**Завдання №5 VNS Lab 9**

1) Скопіювати з файлу F1 у файл F2 рядки, починаючи з K до K+5.

2) Підрахувати кількість голосних букв у файлі F2.

**Завдання №6 Algotester Lab 4**

* Варіант завдання: 1
* Деталі завдання:

Вам дано 2 цiлих чисел масиви, розмiром N та M.

Ваше завдання вивести:1. Рiзницю N-M; 2. Рiзницю M-N; 3. Їх перетин; 4. Їх обєднання; 5. Їх симетричну рiзницю.

- Важливі деталі для врахування при імплементації програми:

Для розв’язку необхідно використовувати засоби STL

**Завдання №7 Algotester Lab 6**

* Варіант завдання: 1
* Деталі завдання :

Вам дано N слiв та число K. Ваше завдання перечислити букви в словах, якi зустрiчаються в текстi бiльше-рiвне нiж K разiв

- Важливі деталі для врахування при імплементації програми:

Великi та маленькі букви вважаються однаковими, виводити необхiдно малi, посортованi вiд останньої до першої у алфавiтi.

**Завдання №8 Class Practice Task**

* Деталі завдання :

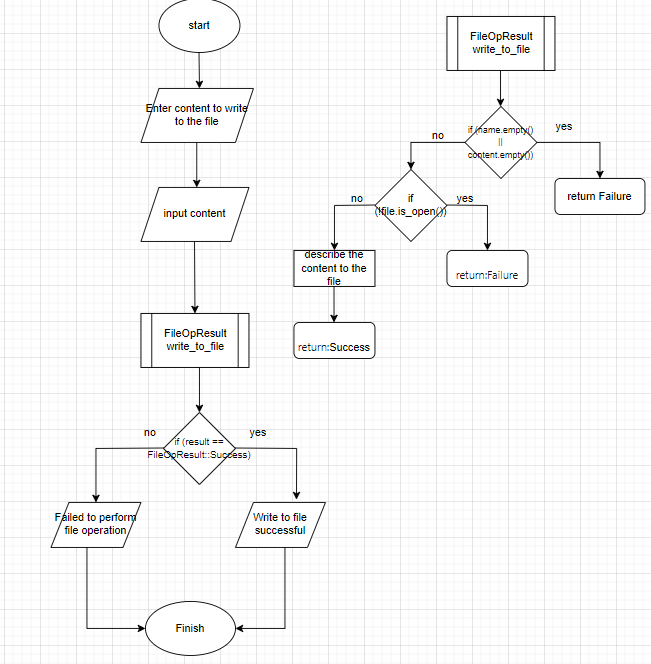
1) Реалізувати функцію створення файла і запису в нього даних

2) Реалізувати функцію створення файла і запису в нього даних

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

**Програма №1 VNS Lab 8**

* Блок-схема:



* Планований час на реалізацію: 2 години

**Програма №2 VNS Lab 9**

* Планований час на реалізацію: 3 години

**Програма №3 Algotester Lab 4**

* Планований час на реалізацію: 3 години
* Важливі деталі для врахування в імплементації: Для розв’язку необхідно використовувати засоби STL

**Програма №4 VNS Lab 6**

* Планований час на реалізацію: 5 годин

**Програма №5 Algotester Lab 6**

* Планований час на реалізацію: 12 годин

- Важливі деталі для врахування в імплементації:

Великi та маленькі букви вважаються однаковими, виводити необхiдно малi, посортованi вiд останньої до першої у алфавiтi.

**Програма №6 Class Practice Task**

* Планований час на реалізацію: 3 години

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

**Завдання №1 VNS Lab 8**

#include <iostream>

#include <fstream>

#include <vector>

#include <string>

#include <sstream>

#include <algorithm>

using namespace std;

struct Student {

    string lastName;

    string firstName;

    string middleName;

    string address;

    string group;

    int rating;

};

// Функція для зчитування студентів з файлу

vector<Student> readStudentsFromFile(const string& filename) {

    vector<Student> students;

    ifstream file(filename);

    if (file.is\_open()) {

        string line;

        while (getline(file, line)) {

            if (line.find("PIB: ") != string::npos) {

                Student tempStudent;

                size\_t pos = line.find("PIB: ") + 5;

                line = line.substr(pos);

                istringstream iss(line);

                iss >> tempStudent.lastName >> tempStudent.firstName >> tempStudent.middleName;

                if (getline(file, line)) {

                    if (line.find("Domashnya adresa: ") != string::npos) {

                        pos = line.find("Domashnya adresa: ") + 18;

                        tempStudent.address = line.substr(pos);

                    }

                }

                if (getline(file, line)) {

                    if (line.find("Group: ") != string::npos) {

                        pos = line.find("Group: ") + 7;

                        tempStudent.group = line.substr(pos);

                    }

                }

                if (getline(file, line)) {

                    if (line.find("Reityng: ") != string::npos) {

                        pos = line.find("Reityng: ") + 9;

                        tempStudent.rating = stoi(line.substr(pos));

                    }

                }

                students.push\_back(tempStudent);

            }

        }

        file.close();

    }

    return students;

}

// Функція для запису студентів у файл

void writeStudentsToFile(const string& filename, const vector<Student>& students) {

    ofstream file(filename);

    if (file.is\_open()) {

        for (const auto& student : students) {

            file << "PIB: " << student.lastName << " " << student.firstName << " " << student.middleName << endl;

            file << "Domashnya adresa: " << student.address << endl;

            file << "Group: " << student.group << endl;

            file << "Reityng: " << student.rating << endl;

        }

        file.close();

    } else {

        cout << "Unable to open file for writing." << endl;

    }

}

int main() {

    string filename = "students.txt";

    vector<Student> students = readStudentsFromFile(filename);

    if (students.empty()) {

        cout << "No data found in the file." << endl;

        return 1;

    }

    int minimumRating;

    cout << "Enter the minimum rating to delete students: ";

    cin >> minimumRating;

    // Видалення елементів з рейтингом менше заданого

    students.erase(

        remove\_if(students.begin(), students.end(), [minimumRating](const Student& s) { return s.rating < minimumRating; }),

        students.end()

    );

    Student newStudent;

    // Введення нового студента

    cout << "Enter the details of the new student:" << endl;

    cout << "Last Name: ";

    cin >> newStudent.lastName;

    cout << "First Name: ";

    cin >> newStudent.firstName;

    cout << "Middle Name: ";

    cin >> newStudent.middleName;

    cout << "Address: ";

    cin.ignore(); // Щоб зчитати рядок з пробілами

    getline(cin, newStudent.address);

    cout << "Group: ";

    getline(cin, newStudent.group);

    cout << "Rating: ";

    cin >> newStudent.rating;

    // Додавання нового студента в кінець списку

    students.push\_back(newStudent);

    // Запис оновлених даних у файл

    writeStudentsToFile(filename, students);

    cout << "Operation completed successfully." << endl;

    return 0;

}

**Завдання №2 VNS Lab 9**

#include <iostream>

#include <fstream>

#include <string>

#include <sstream>

int main() {

    std::ifstream fileInput("F1.txt"); // Відкриття файлу для читання

    std::ofstream fileOutput("F2.txt"); // Відкриття файлу для запису

    if (!fileInput.is\_open() || !fileOutput.is\_open()) { // Перевірка на успішне відкриття файлів

        std::cerr << "Unable to open file!\n"; // Повідомлення про помилку відкриття файлу

        return 1;

    }

    std::string line;

    int firstWordLength = 0;

    while (std::getline(fileInput, line)) { // Читання файлу рядок за рядком

        if (!line.empty() && line[0] != 'A') { // Перевірка на порожній рядок та перший символ не є 'A'

            fileOutput << line << '\n'; // Запис рядка у файл F2

            std::istringstream iss(line);

            std::string word;

            iss >> word; // Зчитуємо перше слово з рядка

            firstWordLength = word.length(); // Зберігаємо довжину першого слова

            break; // Зупиняємо цикл після першого запису

        }

    }

    fileInput.close(); // Закриття файлів

    fileOutput.close();

    std::cout << "Length of the first word in F2: " << firstWordLength << std::endl; // Виведення довжини першого слова у файлі F2

    return 0;

}

**Завдання №3 Algotester Lab 4**

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

vector<int> ost0; // Вектор для чисел, остаток від ділення на 3 яких дорівнює 0

vector<int> ost1; // Вектор для чисел, остаток від ділення на 3 яких дорівнює 1

vector<int> ost2; // Вектор для чисел, остаток від ділення на 3 яких дорівнює 2

vector<int> final; // Вектор для зберігання відсортованих та унікальних чисел

// Функція для сортування у зворотньому порядку

bool sortDescending(int a, int b);

int main() {

    int N;

    cin >> N;

    vector<int> chusla(N);

    for (int i = 0; i < N; i++) {

        cin >> chusla[i];

    }

    // Розділення чисел за остатками від ділення на 3

    for (int i = 0; i < N; i++) {

        if ((chusla[i] % 3) == 0) {

            ost0.push\_back(chusla[i]);

        } else if ((chusla[i] % 3) == 1) {

            ost1.push\_back(chusla[i]);

        } else if ((chusla[i] % 3) == 2) {

            ost2.push\_back(chusla[i]);

        }

    }

    // Сортування та видалення дублікатів для кожного вектора

    sort(ost0.begin(), ost0.end());

    ost0.erase(unique(ost0.begin(), ost0.end()), ost0.end());

    sort(ost1.begin(), ost1.end(), sortDescending);

    ost1.erase(unique(ost1.begin(), ost1.end()), ost1.end());

    sort(ost2.begin(), ost2.end());

    ost2.erase(unique(ost2.begin(), ost2.end()), ost2.end());

    // Додавання унікальних чисел у final з усіх векторів

    for (int i = 0; i < ost0.size(); i++) {

        final.push\_back(ost0.at(i));

    }

    for (int i = 0; i < ost1.size(); i++) {

        final.push\_back(ost1.at(i));

    }

    for (int i = 0; i < ost2.size(); i++) {

        final.push\_back(ost2.at(i));

    }

    // Виведення кількості унікальних чисел та їх значень

    cout << final.size() << endl;

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

        cout << final.at(i) << " ";

    }

    return 0;

}

// Функція для сортування у зворотньому порядку

bool sortDescending(int a, int b) {

    return a > b;

}

**Завдання №4 VNS Lab 6**

#include <iostream>

#include <string>

#include <algorithm>

int main() {

    std::string inputString;

    std::cout << "Введіть рядок: ";

    std::getline(std::cin, inputString); // Зчитування рядка з консолі

    // Фільтруємо цифри з рядка і зберігаємо їх у окремій стрічці

    std::string digits;

    for (char ch : inputString) {

        if (std::isdigit(ch)) {

            digits += ch; // Додаємо цифру до стрічки digits

        }

    }

    // Сортуємо цифри за спаданням

    std::sort(digits.rbegin(), digits.rend()); // Сортування у зворотньому порядку (спадаючому)

    // Замінюємо цифри у вихідному рядку відсортованими за спаданням

    size\_t digitIndex = 0;

    for (char &ch : inputString) {

        if (std::isdigit(ch)) {

            ch = digits[digitIndex++]; // Заміна цифри на відсортовану за спаданням

        }

    }

    std::cout << "Результат: " << inputString << std::endl; // Виведення результату

    return 0;

}

**Завдання №5 Algotester Lab 6**

#include <iostream>

#include <cstring>

#include <vector>

using namespace std;

// Структура для зберігання координат

struct Coord

{

    int x;

    int y;

};

int N; // Розмір судоку (NxN)

int Q; // Кількість запитів

vector<vector<int>> Sudoky; // Вектор для зберігання судоку

// Функція для визначення можливих відповідей для конкретної клітинки судоку

void possibilityAndAnswer(Coord c)

{

    vector<int> possibleAnswer; // Вектор для зберігання можливих відповідей

    // Перевірка, чи вже є значення в клітинці судоку

    if (Sudoky[c.y - 1][c.x - 1] != 0)

    {

        cout << 1 << "\n" << Sudoky[c.y - 1][c.x - 1] << "\n"; // Якщо є, вивести його

        return;

    }

    // Заповнення вектора можливих відповідей від 1 до N

    for (int i = 0; i < N; i++)

    {

        possibleAnswer.push\_back(i + 1);

    }

    // Перевірка чисел у рядку та стовпці для визначення можливих відповідей

    for (int i = 0; i < N; i++)

    {

        if (Sudoky[c.y - 1][i] != 0 && i != (c.x - 1))

        {

            int value = Sudoky[c.y - 1][i];

            possibleAnswer[value - 1] = 0;

        }

    }

    for (int j = 0; j < N; j++)

    {

        if (Sudoky[j][c.x - 1] != 0 && j != (c.y - 1))

        {

            int value = Sudoky[j][c.x - 1];

            possibleAnswer[value - 1] = 0;

        }

    }

    int answers = 0;

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

    {

        if (possibleAnswer[i] != 0)

        {

            answers++;

        }

    }

    // Виведення кількості та можливих відповідей

    cout << answers << endl;

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

    {

        if (possibleAnswer[i] != 0)

        {

            cout << possibleAnswer[i] << " ";

        }

    }

    cout << endl;

}

int main()

{

    cin >> N; // Введення розміру судоку

    string numbers;

    // Цикл для зчитування введення судоку

    for (int i = 0; i < N; i++)

    {

        vector<int> sudoky1;

        for (int j = 0; j < N; j++)

        {

            sudoky1.push\_back(0);

        }

        cin >> numbers;

        // Запис чисел судоку в вектор

        for (int i = 0; i < numbers.length(); i++)

        {

            char ch = numbers.c\_str()[i];

            int elem = atoi(&ch);

            sudoky1[i] = elem;

        }

        Sudoky.push\_back(sudoky1); // Додавання рядка судоку до вектора

    }

    cin >> Q; // Введення кількості запитів

    vector<Coord> coords;

    // Зчитування координат для кожного запиту

    for (int i = 0; i < Q; i++)

    {

        Coord c;

        cin >> c.y >> c.x;

        coords.push\_back(c);

    }

    // Виклик функції для кожного запиту

    for (int i = 0; i < Q; i++)

    {

        possibilityAndAnswer(coords[i]);

    }

    return 0;

}

**Завдання №6 Class Practice Task**

#include <iostream>

#include <fstream>

#include <ctime>

using namespace std;

int main() {

    time\_t t = time(0);    // Отримання поточного часу (часу з системного годинника)

    tm\* now = localtime(&t); // Конвертація часу в місцевий час

    char tme[20];    // Масив для зберігання форматованого часу

    strftime(tme, 20, "%Y-%m-%d %H:%M:%S", now); // Форматування часу в рядок за заданим шаблоном

    ofstream file("F1.txt");    // Відкриття файлу для запису часу

    if (file.is\_open()) { // Перевірка успішності відкриття файлу

        file << tme; // Запис часу у файл

        file.close(); // Закриття файлу після запису

        cout << "success" << endl; // Виведення повідомлення про успішність запису у файл

    } else {

        cout << "fail" << endl; // Виведення повідомлення про невдачу відкриття файлу для запису

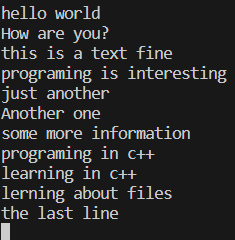
    }

    return 0;

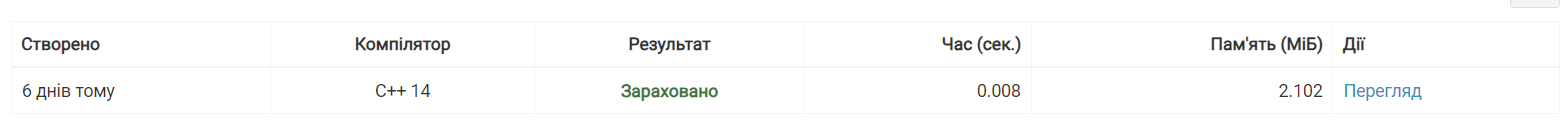
}

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

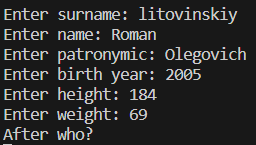
**Завдання №1** **VNS Lab 9**



**Завдання №3 Algotester Lab 4**



**Завдання №4 VNS Lab 8Lit**

****

**Завдання №6 Class Practice Task**

****

**Завдання №7 Algotester lab6**

****

# **Висновки:**

В ході виконання лабораторних та практичних завдань, що стосувалися взаємодії з двійковими файлами та операцій введення-виведення рядків, мені вдалося успішно ознайомитися з організацією обміну структурованою інформацією через зовнішні носії. Робота над цими етапами дозволила мені отримати практичні навички роботи з текстовими файлами, виконанням операцій введення-виведення текстової інформації, а також зберіганням даних на зовнішніх носіях. Цей досвід виявився надзвичайно корисним, допомагаючи зрозуміти процес зберігання та обробки інформації в різних форматах, зокрема, вивчити особливості роботи з двійковими файлами та текстовими рядками.