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

Сенів Роман

# **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 :

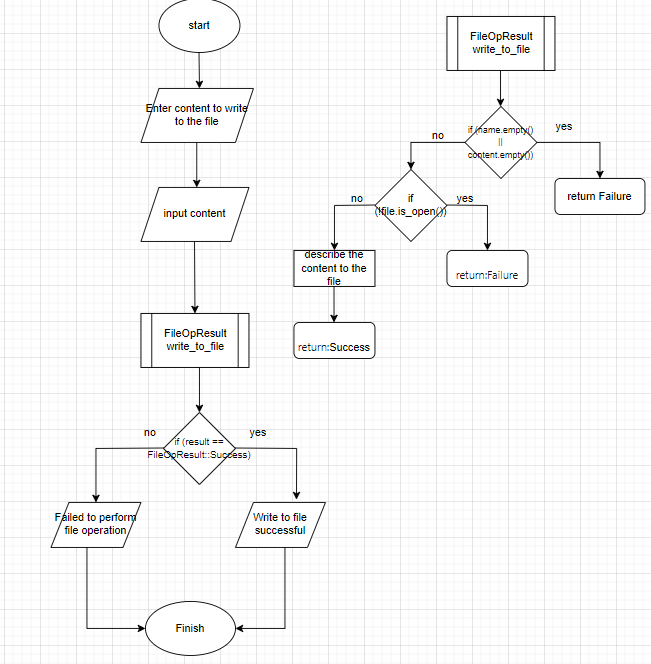
1) Реалізувати функцію створення файла і запису в нього даних

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 <cstring>

using namespace std;

struct Person {

    char surname[50];

    char name[50];

    char patronymic[50];

    int birthYear;

    double height;

    double weight;

};

void printBinaryFileContents(const char\* filename) {

    ifstream inFile(filename, ios::binary);

    if (!inFile.is\_open()) {

        cerr << "Unable to open file: " << filename << endl;

        return;

    }

    Person person;

    while (inFile.read(reinterpret\_cast<char\*>(&person), sizeof(person))) {

        cout << "Surname: " << person.surname << "\n";

        cout << "Name: " << person.name << "\n";

        cout << "Patronymic: " << person.patronymic << "\n";

        cout << "Birth Year: " << person.birthYear << "\n";

        cout << "Height: " << person.height << "\n";

        cout << "Weight: " << person.weight << "\n\n";

    }

    inFile.close();

}

void deleteByHeightAndWeight(const char\* filename, double heightToDelete, double weightToDelete) {

    ifstream inFile(filename, ios::binary);

    ofstream outFile("temp.bin", ios::binary);

    if (!inFile.is\_open() || !outFile.is\_open()) {

        cerr << "Unable to open file for deletion." << endl;

        return;

    }

    Person person;

    while (inFile.read(reinterpret\_cast<char\*>(&person), sizeof(person))) {

        if (person.height != heightToDelete || person.weight != weightToDelete) {

            outFile.write(reinterpret\_cast<char\*>(&person), sizeof(person));

        }

    }

    inFile.close();

    outFile.close();

    remove(filename);

    rename("temp.bin", filename);

}

void addAfterSurname(const char\* filename, const char\* surnameToFind, const Person& newPerson) {

    ifstream inFile(filename, ios::binary);

    ofstream outFile("temp.bin", ios::binary);

    if (!inFile.is\_open() || !outFile.is\_open()) {

        cerr << "Unable to open file for addition." << endl;

        return;

    }

    Person person;

    while (inFile.read(reinterpret\_cast<char\*>(&person), sizeof(person))) {

        outFile.write(reinterpret\_cast<char\*>(&person), sizeof(person));

        if (strcmp(person.surname, surnameToFind) == 0) {

            outFile.write(reinterpret\_cast<const char\*>(&newPerson), sizeof(newPerson));

        }

    }

    inFile.close();

    outFile.close();

    remove(filename);

    rename("temp.bin", filename);

}

int main() {

    ofstream createFile("people\_data.bin", ios::binary);

    Person person1;

    cout << "Enter surname: ";

    cin >> person1.surname;

    cout << "Enter name: ";

    cin >> person1.name;

    cout << "Enter patronymic: ";

    cin >> person1.patronymic;

    cout << "Enter birth year: ";

    cin >> person1.birthYear;

    cout << "Enter height: ";

    cin >> person1.height;

    cout << "Enter weight: ";

    cin >> person1.weight;

    createFile.write(reinterpret\_cast<const char\*>(&person1), sizeof(person1));

    createFile.close();

    cout << "Initial contents of binary file:\n";

    printBinaryFileContents("people\_data.bin");

    deleteByHeightAndWeight("people\_data.bin", 175.5, 80.0);

    cout << "\nContents after deletion:\n";

    printBinaryFileContents("people\_data.bin");

    Person newPerson;

    cout << "Enter new person details:\n";

    cout << "Enter surname to find: ";

    cin >> newPerson.surname;

    cout << "Enter name: ";

    cin >> newPerson.name;

    cout << "Enter patronymic: ";

    cin >> newPerson.patronymic;

    cout << "Enter birth year: ";

    cin >> newPerson.birthYear;

    cout << "Enter height: ";

    cin >> newPerson.height;

    cout << "Enter weight: ";

    cin >> newPerson.weight;

    addAfterSurname("people\_data.bin", "Johnson", newPerson);

    cout << "\nContents after addition:\n";

    printBinaryFileContents("people\_data.bin");

    return 0;

}

**Завдання №2 VNS Lab 9**

#include <iostream>

#include <fstream>

#include <cctype>

using namespace std;

int main() {

    ofstream f1("F1.txt");

    if (!f1.is\_open()) {

        cout << "Failed to open file F1.txt" << endl;

        return 1;

    }

    f1 << "Line 1: Hello world!" << endl;

    f1 << "Line 2: How are you?" << endl;

    f1 << "Line 3: This is a text file." << endl;

    f1 << "Line 4: Programming is interesting." << endl;

    f1 << "Line 5: Just another line." << endl;

    f1 << "Line 6: Another one." << endl;

    f1 << "Line 7: Some more information." << endl;

    f1 << "Line 8: Programming in C++." << endl;

    f1 << "Line 9: Learning about files." << endl;

    f1 << "Line 10: The last line." << endl;

    f1.close();

    ifstream f1Read("F1.txt");

    if (!f1Read.is\_open()) {

        cout << "Failed to open file F1.txt for reading" << endl;

        return 1;

    }

    ofstream f2("F2.txt");

    if (!f2.is\_open()) {

        cout << "Failed to open file F2.txt" << endl;

        return 1;

    }

    int K = 3;

    string line;

    int currentLine = 0;

    while (getline(f1Read, line)) {

        currentLine++;

        if (currentLine >= K && currentLine < K + 5) {

            f2 << line << endl;

        }

    }

    f1Read.close();

    f2.close();

    ifstream f2Read("F2.txt");

    if (!f2Read.is\_open()) {

        cout << "Failed to open file F2.txt for reading" << endl;

        return 1;

    }

    int vowelCount = 0;

    while (getline(f2Read, line)) {

        for (char ch : line) {

            if (tolower(ch) == 'a' || tolower(ch) == 'e' || tolower(ch) == 'i' || tolower(ch) == 'o' || tolower(ch) == 'u') {

                vowelCount++;

            }

        }

    }

    f2Read.close();

    cout << "Number of vowels in file F2: " << vowelCount << endl;

    return 0;

}

**Завдання №3 Algotester Lab 4**

#include <iostream>

#include <vector>

#include <algorithm>

#include <set>

using namespace std;

void printVector(const vector<int>& vec) {

    cout << vec.size() << endl;

    for (int i : vec) {

        cout << i << " ";

    }

    cout << endl << endl;

}

int main() {

    int N, M, n;

    vector<int> ar1;

    vector<int> ar2;

    cin >> N;

    for (int i = 0; i < N; i++) {

        cin >> n;

        ar1.push\_back(n);

    }

    cin >> M;

    for (int i = 0; i < M; i++) {

        cin >> n;

        ar2.push\_back(n);

    }

    sort(ar1.begin(), ar1.end());

    sort(ar2.begin(), ar2.end());

    vector<int> dif1;

    set\_difference(ar1.begin(), ar1.end(), ar2.begin(), ar2.end(), back\_inserter(dif1));

    printVector(dif1);

    vector<int> dif2;

    set\_difference(ar2.begin(), ar2.end(), ar1.begin(), ar1.end(), back\_inserter(dif2));

    printVector(dif2);

    vector<int> in;

    set\_intersection(ar1.begin(), ar1.end(), ar2.begin(), ar2.end(), back\_inserter(in));

    printVector(in);

    vector<int> uni;

    set\_union(ar1.begin(), ar1.end(), ar2.begin(), ar2.end(), back\_inserter(uni));

    printVector(uni);

    vector<int> sym\_dif;

    set\_symmetric\_difference(ar1.begin(), ar1.end(), ar2.begin(), ar2.end(), back\_inserter(sym\_dif));

    printVector(sym\_dif);

    return 0;

}

}

**Завдання №4 VNS Lab 6**

#include <iostream>

#include <vector>

using namespace std;

int main() {

    string inputString;

    cout << "Введіть рядок: ";

    getline(cin, inputString);

    vector<char> result;

    for (char ch : inputString) {

        if (isalpha(ch)) {

            result.push\_back(ch);

        }

    }

    for (char ch : inputString) {

        if (isdigit(ch)) {

            result.push\_back(ch);

        }

    }

    cout << "Відформатований рядок: ";

    for (char ch : result) {

        cout << ch;

    }

    return 0;

}

**Завдання №5 Algotester Lab 6**

#include <iostream>

#include <vector>

#include <algorithm>

#include <set>

#include <unordered\_map>

using namespace std;

int main() {

    int N, K;

    cin >> N >> K;

    unordered\_map<string, int> wordCount;

    vector<string> words;

    for (int i = 0; i < N; ++i) {

        string word;

        cin >> word;

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

        wordCount[word]++;

        words.push\_back(word);

    }

    sort(words.begin(), words.end());

    string repChars;

    for (vector<string>::iterator i = words.begin(); i != words.end(); i++) {

        int counter = 1;

        if (\*i == "") {

            continue;

        }

        for (vector<string>::iterator j = i + 1; j != words.end(); j++) {

            if (i != j && \*i == \*j) {

                counter++;

                \*j = "";

            } else if (i != j && \*j != \*(j - 1)) {

                break;

            }

        }

        if (counter >= K) {

            repChars.append(\*i);

        }

    }

    sort(repChars.begin(), repChars.end(), greater<char>());

    vector<char> uniqueChars;

    for (size\_t i = 0; i < repChars.length(); i++) {

        if (repChars[i] != repChars[i + 1]) {

            uniqueChars.push\_back(repChars[i]);

        }

    }

    if (uniqueChars.empty()) {

        cout << "Empty!";

    } else {

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

        for (char &ch : uniqueChars) {

            cout << ch << " ";

        }

    }

    return 0;

}

**Завдання №6 Class Practice Task**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

enum class FileOpResult { Success, Failure };

FileOpResult write\_to\_file(const string& name, const string& content) {

    if (name.empty() || content.empty()) {

        return FileOpResult::Failure;

    }

    ofstream file(name, ios::out);

    if (!file.is\_open()) {

        return FileOpResult::Failure;

    }

    file << content;

    return FileOpResult::Success;

}

int main() {

    string filename = "classpractice.txt";

    string content;

    cout << "Enter content to write to the file: ";

    getline(cin, content);

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

    if (result == FileOpResult::Success) {

        cout << "Write to file successful." << endl;

    } else {

        cout << "Failed to perform file operation." << endl;

    }

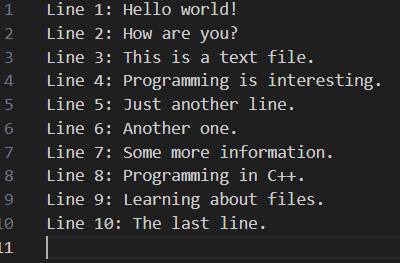
    return 0;

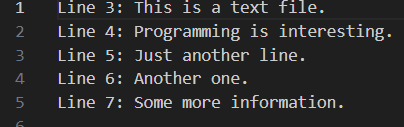
}

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

**Завдання №1** **VNS Lab 9**

****

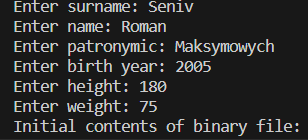




**Завдання №3 Algotester Lab 4**



**Завдання №4 VNS Lab 6**

****

**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.