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. Системи числення. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами»

***Виконав:***

студент групи ШІ-11

Ковалець Владислав Миколайович

# **Тема роботи:**

" Файли. Системи числення. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами"

# **Мета роботи:**

Дослідження основних концепцій і принципів роботи з файлами в С++. Основний акцент буде зроблено на роботі з бінарними та текстовими файлами, системами числення, а також вивченні стандартної бібліотеки для роботи з файлами.

Ключові завдання роботи включають:

1. **Робота з бінарними файлами:**
   * Вивчення принципів створення, запису і читання бінарних файлів.
2. **Символи та рядкові змінні:**
   * Вивчення методів роботи з рядковими змінними в с++.
3. **Робота з текстовими файлами:**
   * Дослідження методів читання та запису текстових файлів.
   * Розгляд форматування текстових файлів, обробка рядків.
4. **Стандартна бібліотека для роботи з файлами:**
   * Вивчення стандартних функцій та класів для роботи з файлами у C++

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Бінарні файли
* Тема №2: Стандартна бібліотека
* Тема №3: Системи числення

1. Індивідуальний план опрацювання теорії:

* Тема №1: Бінарні файли

Джерела Інформації:

* + - Відео:

https://www.youtube.com/watch?v=2lzVB8bkM8o

* + Що опрацьовано:
    - Поняття бінарних файлів і їх застосування(частково)
  + Статус: Ознайомлений
  + Початок опрацювання теми: 15.11.2023
  + Звершення опрацювання теми: 15.11.2023
* Тема №2: Стандартна бібліотека
  + Джерела Інформації:
    - Відео:

https://www.youtube.com/watch?v=m-WJikuZGuU&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=79

* + Що опрацьовано:
    - Ознайомлення з стандартною бібліотекою С++
  + Статус: Ознайомлений
  + Початок опрацювання теми: 15.11.2023
  + Звершення опрацювання теми:15.11.2023
* Тема №3: Системи числення
  + Джерела Інформації:
    - Відео

https://www.youtube.com/watch?v=c5FSYMpsLN0&list=PLiPRE8VmJzOpn6PzYf0higmCEyGzo2A5g&index=67

* + Що опрацьовано:
    - 10-кова система числення
    - 2-кова система числення
    - 8-кова система числення
    - 16-кова система числення
  + Статус: Ознайомлений
  + Початок опрацювання теми: 20.11.2023
  + Звершення опрацювання теми: 20.11.2023

# **Виконання роботи:**

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

Завдання № VNS lab6

* 2 варіант

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.

Надрукувати всі слова, які не містять голосних букв.

Завдання № VNS lab8

* 2 варіант

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 "Співробітник":

- прізвище, ім'я, по батькові;

- посада

- рік народження;

- заробітна плата.

Знищити елемент із зазначеним прізвищем, додати елемент після елемента

із зазначеним номером.

Завдання № VNS lab9

* 2 варіант

Створити текстовий файл F1 не менше, ніж з 10 рядків і записати в нього

інформацію

1) Скопіювати у файл F2 тільки ті рядки з F1, які починаються з букви «А».

2) Підрахувати кількість слів в F2.

Завдання № algotester lab4

* 1 варіант

Обмеження: 1 сек., 256 МіБ

Вам дано 2 цілих чисел масиви, розміром N та M

Ваше завдання вивести:

1. Різницю N-M

2. Різницю M-N

3. Їх перетин

4. Їх обєднання

5. Їх симетричну різницю

Вхідні дані

У першому рядку ціле число N - розмір масиву 1

У другому рядку N цілих чисел - елементи масиву 1

У третьому рядку ціле число M- розмір масиву 2

У четвертом рядку M цілих чисел - елементи масиву 2

Вихідні дані

Вивести результат виконання 5 вищезазначених операцій у форматі:

У першому рядку ціле число N- розмір множини

У наступному рядку N цілих чисел - посортована у порядку зростання множина

Обмеження

1≤N, M≤100

1≤n(i),m(i)≤100

Завдання № algotester lab6

* 1 варіант

Обмеження: 2 сек., 256 МіБ

Вам дано N слів та число K

Ваше завдання перечислити букви в словах, які зустрічаються в тексті більше-рівне ніж K разів (саме слово, не буква!).

Великі та маленькі букви вважаються однаковими, виводити необхідно малі, посортовані від останьої до першої у алфавіті. Букву потрібно виводити лише один раз.

У випадку якщо таких букв немає - вивести "Empty!".

Вхідні дані

Цілі числа N та K - загальна кількість слів та мінімальна кількість слів щоб враховувати букви цього слова в результаті.

N стрічок s

Вихідні дані

У першому рядку ціле число M - кількість унікальних букв

У другому рядку унікальні букви через пробіли

Обмеження

1≤K≤N≤10^5

1≤|s(i)|≤10

S(i) ∈a..Z

Завдання № practice5\_task1

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.

Завдання № practice5\_task2

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 № selfpractice

Петрик любить грати в ігри, де світ складається з кубів. Сьогодні він вирішив написати свій варіант такої гри. Петрик хоче, щоб у цій грі можна було програмувати нескладні механізми на основі електричного сигналу. Вважається, що сигнал має певну додатну силу, яка позначається цілим числом. Також, сигнал може бути відсутнім, що позначається як сигнал силою 0. Петрик додав в гру три блоки, які по-різному обробляють електричний сигнал:

Кабель отримує сигнал силою k і видає сигнал силою max(k−1,0)

Повторювач підсилює сигнал до рівня 15 при ненульовому вхідному сигналі, і не видає сигнал при його відсутності. Формально, нехай сила вхідного сигналу k. Тоді повторювач видає сигнал силою 15, якщо k≥1, і сигнал силою 0, якщо k=0

Змінювач не видає сигнал при ненульовому вхідному сигналі та видає сигнал силою 15 при відсутності вхідного сигналу. Формально, нехай сила вхідного сигналу k. Тоді змінювач видає сигнал силою 0, якщо k≥1, і сигнал силою 15, якщо k=0

.

Петрик задає вам електричну лінію, в якій кабелі, повторювачі і змінювачі позначені літерами k, p, z відповідно. На перший елемент лінії подано сигнал силою 15. Допоможіть Петрику знайти вихідний сигнал останнього блоку в лінії.

Вхідні дані

У першому рядку задано одне ціле число n— кількість елементів в електричній лінії.

У другому рядку задано рядок s— послідовність елементів в електричній лінії.

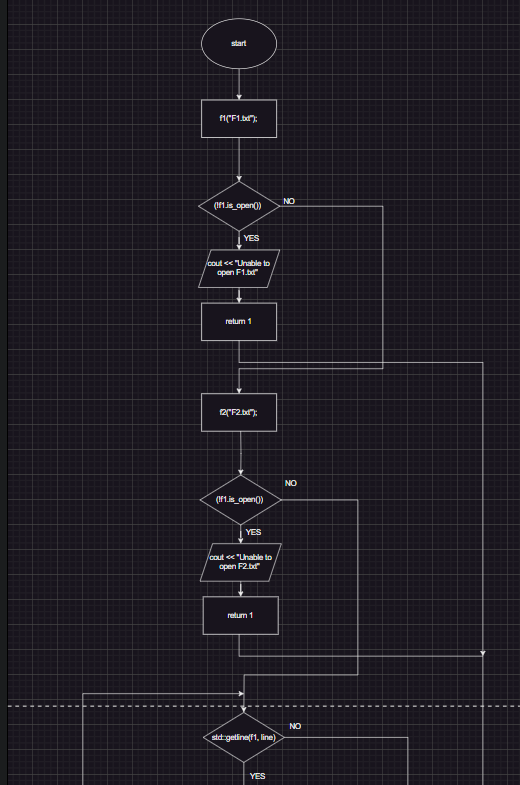
Вихідні дані

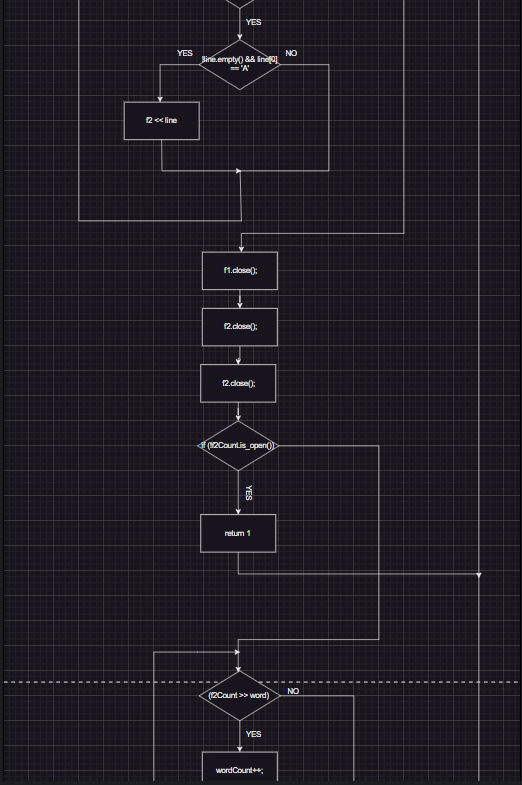
В єдиному рядку виведіть одне число, що позначає силу вихідного сигналу останнього елемента в лінії.

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

Програма № VNS lab9

* Блок-схема





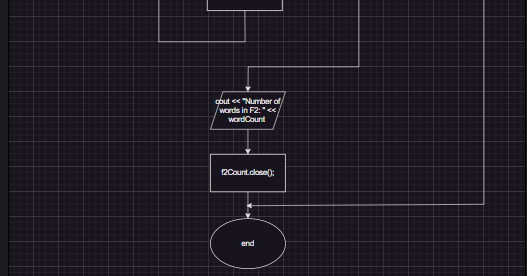


Рисунок 1 Блок-схема до завдання vns lab9

* Планований час на реалізацію

10 хвилин

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

**Завдання № vns lab6**

#include <iostream>

#include <string>

#include <sstream>

int main()

{

    std::string input;

    std::cout << "Enter a line of text: ";

    std::getline(std::cin, input);

    // використовується для розбиття введеного рядка на окремі слова які можна потім обробити окремо.

    std::istringstream iss(input); //  клас для створення лінії який можна використовувати для читання з рядка

    std::string token;

    std::cout << "Tokens without vowels in the input: ";

    while (iss >> token)

    // код який виконується для кожного слова в лінії 'iss'

    {

        bool hasVowel = false;

        for (char c : token)

        // код який виконується для кожного символу в змінній 'token'

        {

            if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' ||

                c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U')

            {

                hasVowel = true;

                break;

            }

        }

        if (!hasVowel)

        {

            std::cout << token << " ";

        }

    }

    std::cout << std::endl;

    return 0;

}

**Завдання № vns lab8**

#include <iostream>

#include <vector>

#include <algorithm>

// Структура для представлення інформації про співробітника

struct Employee

{

    std::string lastName;

    std::string firstName;

    std::string middleName;

    std::string position;

    int birthYear;

    double salary;

};

// Функція для видалення співробітника за прізвищем

void removeEmployeeByLastName(std::vector<Employee> &employees, const std::string &lastName)

{ // Використовуємо алгоритм erase-remove\_if для видалення співробітника за прізвищем

    employees.erase

        (std::remove\_if//переміщує в кінець потрібне прізвище

            (employees.begin(),

                employees.end(),

                    [lastName](const Employee &e) { return e.lastName == lastName;}

            ),

            employees.end()//видаляє його

        );

}

// Функція для додавання нового співробітника після вказаного індексу

void addEmployeeAfterIndex(std::vector<Employee> &employees, const Employee &newEmployee, size\_t index)

{

    if (index < employees.size()) // Перевірка, чи індекс в межах розміру вектора

    {

        employees.insert(employees.begin() + index + 1, newEmployee);// Вставка нового співробітника після вказаного індексу

    }

    else

    {

        // Якщо індекс за межами, додаємо нового співробітника в кінці

        employees.push\_back(newEmployee);

    }

}

int main()

{

     // Створення вектора для збереження співробітників

    std::vector<Employee> employees;

    // Додавання прикладових співробітників

    employees.push\_back({"Biden", "Joe", "Simson", "President", 1950,70000.0});

    employees.push\_back({"Son", "Alice", "Marie", "Engineer", 1990, 60000.0});

    employees.push\_back({"Smit", "Bob", "James", "Analyst", 1986, 60000.0});

     // Виведення початкового списку співробітників

    std::cout << "Початковий список співробітників:" << std::endl;

    for (const auto &employee : employees)

    {

        std::cout << employee.lastName << ", " << employee.firstName << " - " << employee.position << std::endl;

    }

    // Видалення співробітника за прізвищем

    removeEmployeeByLastName(employees, "Biden");

     // Виведення списку співробітників після видалення

    std::cout << "Список співробітників після видалення Biden:" << std::endl;

    for (const auto &employee : employees)

    {

        std::cout << employee.lastName << ", " << employee.firstName << " - " << employee.position << std::endl;

    }

    // Додавання нового співробітника після вказаного індексу

    Employee newEmployee{"Taylor", "Derden", "Grace", "Runner", 1985, 65000.0};

    addEmployeeAfterIndex(employees, newEmployee, 0);

    // Виведення остаточного списку співробітників

    std::cout << "Остаточний список співробітників:" << std::endl;

    for (const auto &employee : employees)

    {

        std::cout << employee.lastName << ", " << employee.firstName << " - " << employee.position << std::endl;

    }

    return 0;

}

**Завдання № vns lab9**

#include <iostream>

#include <fstream>

#include <sstream>

int main()

{

    // Відкриття файлу F1 для считання

    std::ifstream f1("F1.txt");

    // Перевірка чи файл F1 вдалося відкрити

    if (!f1.is\_open())

    {

        std::cout << "Unable to open F1.txt" << std::endl;

        return 1; // Помилка відкриття файлу

    }

    // Відкриття файлу F2 для запису

    std::ofstream f2("F2.txt");

    // Перевірка чи файл F2 вдалося відкрити

    if (!f2.is\_open())

    {

        std::cout << "Unable to open F2.txt" << std::endl;

        return 1; // Помилка відкриття файлу

    }

    std::string line;

    // Зчитування рядків з F1

    while (std::getline(f1, line))

    {

        // Перевірка, чи рядок починається з букви "A"

        if (!line.empty() && line[0] == 'A')

        {

            // Запис рядка у файл F2

            f2 << line << std::endl;

        }

    }

    // Закриття файлів

    f1.close();

    f2.close();

    // Відкриття файлу для підрахунку слів

    std::ifstream f2Count("F2.txt");

    // Перевірка, чи файл F2 вдалося відкрити

    if (!f2Count.is\_open())

    {

        std::cout << "Unable to open F2.txt for word count" << std::endl;

        return 1; // Помилка відкриття файлу

    }

    int wordCount = 0;

    std::string word;

    // Підрахунок слів у файлі F2

    while (f2Count >> word)

    {

        wordCount++;

    }

    // Виведення кількості слів

    std::cout << "Number of words in F2: " << wordCount << std::endl;

    f2Count.close();

    return 0;

}

**Завдання № algotester lab4**

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main()

{

    int N, M;

    // Зчитуємо розмір та елементи масиву 1

    cin >> N;

    vector<int> array1(N);

    for (int i = 0; i < N; i++)

    {

        cin >> array1[i];

    }

    // Зчитуємо розмір та елементи масиву 2

    cin >> M;

    vector<int> array2(M);

    for (int i = 0; i < M; i++)

    {

        cin >> array2[i];

    }

    // Сортуємо масиви

    sort(array1.begin(), array1.end());

    sort(array2.begin(), array2.end());

    // Різниця N-M

    vector<int> diffNM;

    set\_difference(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(diffNM));

    // Різниця M-N

    vector<int> diffMN;

    set\_difference(array2.begin(), array2.end(), array1.begin(), array1.end(), back\_inserter(diffMN));

    // Перетин

    vector<int> intersection;

    set\_intersection(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(intersection));

    // Об'єднання

    vector<int> unionSet;

    set\_union(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(unionSet));

    // Симетрична різниця

    vector<int> symmetricDiff;

    set\_symmetric\_difference(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(symmetricDiff));

    // Виведення результатів

    cout << diffNM.size() << endl;

    for (int value : diffNM)

    {

        cout << value << " ";

    }

    cout << endl;

    cout << diffMN.size() << endl;

    for (int value : diffMN)

    {

        cout << value << " ";

    }

    cout << endl;

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

    for (int value : intersection)

    {

        cout << value << " ";

    }

    cout << endl;

    cout << unionSet.size() << endl;

    for (int value : unionSet)

    {

        cout << value << " ";

    }

    cout << endl;

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

    for (int value : symmetricDiff)

    {

        cout << value << " ";

    }

    cout << endl;

    return 0;

}

**Завдання № algotester lab6**

#include <iostream>

#include <vector>

#include <map>

#include <unordered\_set>

#include <algorithm>

// Функція для перетворення рядка у нижній регістр

std::string to\_lower(std::string str)

{

    for (char& c : str)

    {

        c = tolower(c);

    }

    return str;

}

int main()

{

    int n, k;

    std::cin >> n >> k;

    std::vector<std::string> words(n);

    // Перший елемент пари - слово

    // Другий елемент пари - кількість входжень слова

    std::map<std::string, int> words\_count;

    // Неупорядкована множина для унікальних символів

    std::unordered\_set<char> charset;

    for (int i = 0; i < n; ++i)

    {

        std::cin >> words[i];

        // Нормалізуємо слово (верхній регістр також можливий)

        words[i] = to\_lower(words[i]);

        words\_count[words[i]]++;

    }

    // Проходимося по словах та додаємо символи унікальних слів в множину

    for (const auto& pair : words\_count)

    {

        if (pair.second >= k)

        {

            for (char c : pair.first)

            {

                charset.insert(c);

            }

        }

    }

    // Перетворюємо множину вектор та сортуємо в зворотньому порядку

    std::vector<char> chars(charset.begin(), charset.end());

    std::sort(chars.begin(), chars.end());

    std::reverse(chars.begin(), chars.end());

    if (chars.empty())

    {

        std::cout << "Empty!";

        return 0;

    }

    std::cout << chars.size() << std::endl;

    // Виводимо унікальні символи

    for (char c : chars)

    {

        std::cout << c << " ";

    }

    return 0;

}

**Завдання № practice5 task1**

#include <iostream>

#include <fstream>

#include <sstream>

enum FileOpResult { Success, Failure};

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

{

    std::ofstream outputFile(name);

    // Перевірка чи файл  вдалося відкрити

    if (!outputFile.is\_open())

    {

        std::cout << "Unable to open" << std::endl;

        return Failure;

    }

    // cтворює об'єкт для читання із стрічки content у форматі потоку вводу

    std::istringstream contentStream(content);

    std::string line;

    while (std::getline(contentStream, line))

    {

        if (!line.empty())

        {

            outputFile << line << std::endl;

        }

    }

    outputFile.close();

    return Success;

}

int main()

{

    std::string name;

    std::cout << "Enter name: ";

    std::getline(std::cin, name);

    std::string content;

    std::cout << "Enter a line of text: ";

    std::getline(std::cin, content);

    const char \*nameCStr = name.c\_str();

    const char \*contentCStr = content.c\_str();

    FileOpResult result = write\_to\_file(nameCStr, contentCStr);

    if (result == Success)

    {

        std::cout << "File was successful" << std::endl;

    }

    else

    {

        std::cout << "File was failed" << std::endl;

    }

}

**Завдання № practice5 task2**

#include <iostream>

#include <fstream>

#include <sstream>

enum FileOpResult { Success, Failure };

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

{

    std::ifstream outputFile(file\_from);

    if (!outputFile.is\_open())

    {

        std::cout << "Unable to open" << std::endl;

        return Failure;

    }

    std::ofstream inputFile(file\_to);

    if (!inputFile.is\_open())

    {

        std::cout << "Unable to open" << std::endl;

        return Failure;

    }

    std::string line;

    while (std::getline(outputFile, line))

    {

        // перевірка чи лінія не пуста

        if (!line.empty())

        {

            // Запис в лінії з файлу

            inputFile << line << std::endl;

        }

    }

    return Success;

}

int main()

{

    std::string file\_from;

    std::cout << "Enter name file\_from: ";

    std::getline(std::cin, file\_from);

    std::string file\_to;

    std::cout << "Enter name file\_to: ";

    std::getline(std::cin, file\_to);

    const char \*file\_fromCStr = file\_from.c\_str();

    const char \*file\_toCStr = file\_to.c\_str();

    FileOpResult result = copy\_file(file\_fromCStr, file\_toCStr);

    if (result == Success)

    {

        std::cout << "File copy was successful" << std::endl;

    }

    else

    {

        std::cout << "File copy failed" << std::endl;

    }

    return 0;

}

**Завдання № selfpractice**

#include <iostream>

#include <string>

int my\_max(int a, int b)

{

    return (a > b) ? a : b;

}

int main()

{

    int n;

    std::cin >> n;

    std::string s;

    std::cin >> s;

    int signal\_strength = 15;

    for (int i = 0; i < n; ++i)

    {

        char block = s[i];

        if (block == 'k')

        {

            signal\_strength = my\_max(signal\_strength - 1, 0);

        }

        else if (block == 'p')

        {

            signal\_strength = (signal\_strength >= 1) ? 15 : 0;

        }

        else if (block == 'z')

        {

            signal\_strength = (signal\_strength >= 1) ? 0 : 15;

        }

    }

    std::cout << signal\_strength << std::endl;

    return 0;

}

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

Завдання № vns lab6

Використовується стандартна бібліотека C++ для роботи зі введеним текстом та рядками.

Використовує цикли та умови для обробки символів у словах.

Виводить тільки ті слова, які не мають голосних літер.

Час затрачений на виконання завдання

30 хвилин

Завдання № vns lab8

Програма включає функції для видалення співробітника за прізвищем та додавання нового співробітника після вказаного індексу.

Час затрачений на виконання завдання

30 хвилин

Завдання № vns lab9

Використовується стандартна бібліотека C++ для роботи з файлами (ifstream та ofstream).

Проводиться перевірка наявності та відкриття файлів для читання та запису.

Здійснюється обробка рядків з файлу F1, вибір та запис певних рядків до файлу F2.

Виводиться кількість слів у файлі F2.

Час затрачений на виконання завдання

30 хвилин

Завдання № algotester lab4

Функції N\_M, sum, suma та rizni реалізують операції над множинами.

Використовуються тимчасові масиви для зберігання проміжних результатів та впорядкованого виведення.

Програма приймає вхідні дані для двох масивів, які представляють множини, та виконує операції над ними.

Час затрачений на виконання завдання

40 хвилин

Завдання № algotester lab6

Використовується стандартна бібліотека C++ для роботи зі стрічками, введення та виведення даних.

Використовується динамічний масив слів для зберігання введених слів.

Обчислюється кількість входжень кожного слова та знаходиться слово з найбільшою кількістю входжень.

Для цього слова обчислюється та виводиться кількість унікальних символів у порядку зменшення їхнього ASCII-коду.

Час затрачений на виконання завдання

40 хвилин

Завдання № practice5 task1

Функція write\_to\_file приймає ім'я файлу та рядок тексту, відкриває файл для запису та записує у файл введений текст.

Підпрограма main викликає функцію, передаючи їй ім'я файлу та текст з клавіатури.

Після виконання запису в файл, виводиться повідомлення про успішність чи невдачу операції запису.

Час затрачений на виконання завдання

40 хвилин

Завдання № practice5 task2

Функція copy\_file приймає два шляхи до файлів, відкриває їх для читання/запису та виконує копіювання вмісту.

Підпрограма main вводить імена вхідного та вихідного файлів з клавіатури та викликає функцію copy\_file.

Виводиться повідомлення про успішність чи невдачу операції копіювання файлу.

Час затрачений на виконання завдання

40 хвилин

Завдання № selfpractice

Програма робить обробку сигнальних блоків ('k', 'p', 'z'), які впливають на сигнальну силу. Результат виводиться в кінці, вказуючи, якою стала сигнальна сила після обробки усіх блоків.

Час затрачений на виконання завдання

40 хвилин

## **5. Кооперація з командою:**

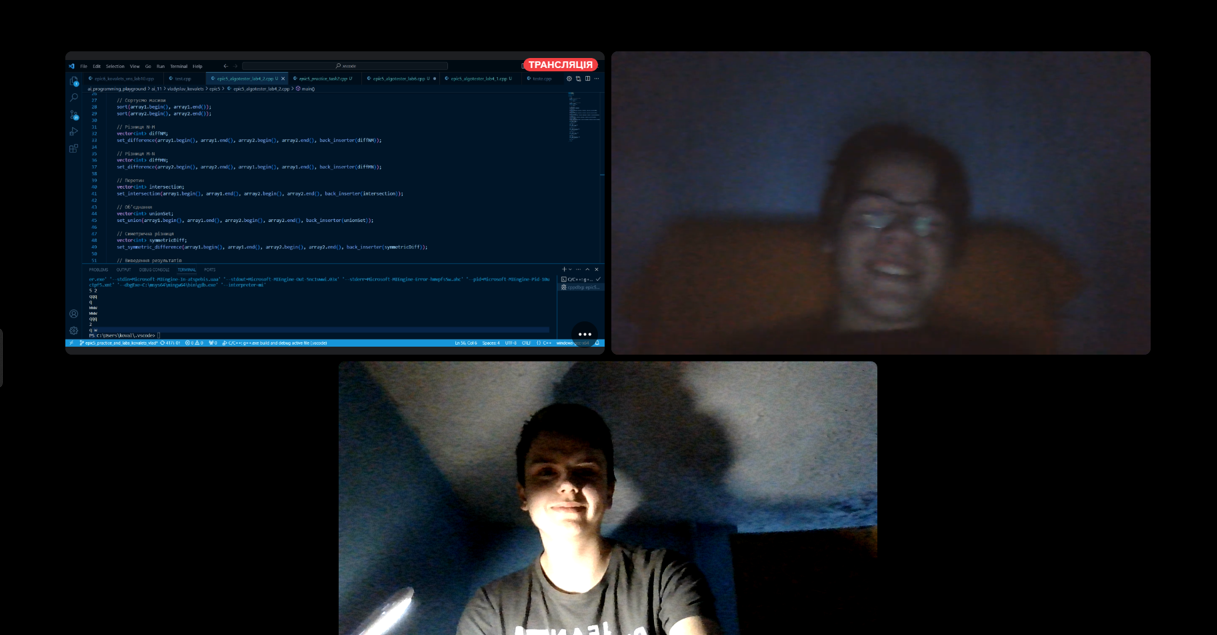


Рисунок 6 Скріншот з конференції 02.12.2023

## **Висновки:**

У ході вивчення теми "Файли. Системи числення. Бінарні файли. Символи і Рядкові Змінні та Текстові файли. Стандартна бібліотека та деталі/методи роботи з файлами" в мові програмування C++, було проведено дослідження основних концепцій та принципів роботи з файлами.

1. \*\*Робота з бінарними файлами:\*\*

- Були вивчені принципи створення, запису бінарних файлів.

2. \*\*Символи та рядкові змінні:\*\*

- Вивчено методи роботи з рядковими змінними в C++.

3. \*\*Робота з текстовими файлами:\*\*

- Досліджено методи читання та запису текстових файлів.

- Розглянуто форматування текстових файлів та обробку рядків.

4. \*\*Стандартна бібліотека для роботи з файлами:\*\*

- Ознайомлено зі стандартними функціями та класами для роботи з файлами в мові програмування C++.

- Практично використано ці функції для розв'язання конкретних завдань.