Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

A blue and white logo

Description automatically generated

**Звіт**

**про виконання лабораторних та практичних робіт блоку № 5**

На тему:  «Файли. Системи числення. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами. Створення й використання бібліотек.»

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

до:

ВНС Лабораторної Роботи № 6

ВНС Лабораторної Роботи № 8

ВНС Лабораторної Роботи № 9

Алготестер Лабораторної Роботи №4

Алготестер Лабораторної Роботи №6

Практичних Робіт №5

**Виконав(ла):**

Студент(ка) групи ШІ-13

Поліщук Марина Олександрівна

# **Тема роботи:**

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

# **Мета роботи:**

Навчитися працювати з файлами, системами числення, бінарними файлами, символами і рядковими змінними, текстовими файлами.

**Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

-Тема №1: Ввід і вивід в С

-Тема №2: Системи числення

-Тема №3: Бінарні файли

-Тема №4: Стандартна бібліотека та деталі/методи роботи з файлами

1. Індивідуальний план опрацювання теорії:

* Тема №1: Ввід і вивід в С
  + Джерела Інформації
    - лекції, практичні заняття
    - <https://t.me/c/1980904348/1/299?single>
  + Що опрацьовано:

У С відсутні засоби вводу-виводу. Всі операції вводу-виводу

реалізуються за допомогою функцій, які перебувають у бібліотеці С.

На рівні **потокового вводу-виводу** обмін даними відбувається побайтно,

тобто за одне звертання до пристрою (файлу) відбувається зчитування або

запис фіксованої порції даних (512 чи 1024 байти). При вводі з диска або при зчитуванні з файлу дані поміщаються в буфер ОС, потім побайтно або порціямимпередаються в програмі користувача. При виводі у файл дані накопичуються в буфері, а при заповненні буфера записуються у вигляді єдиного блоку на диск.

Буфери ОС реалізуються у вигляді ділянок основної пам'яті. Функції бібліотеки С, що підтримують обмін, з даними на рівні потоку дозволяють обробляти дані різних розмірів і форматів.

**Потік** - це файл разом з наданими засобами буферизації. При роботі з

потоком можна:

1) Відкривати й закривати потоки ( зв'язувати вказівники на потік з

конкретними файлами);

2) вводити і виводити рядок, символ, форматовані дані, порцію даних довільної довжини;

3) аналізувати помилки вводу-виводу й досягнення кінця файлу;

4) управляти буферизацією потоку й розміром буфера;

5) отримувати й встановлювати вказівник поточної позиції у файлі;

Функції бібліотеки вводу-виводу перебувають у заголовковому файлі <stdio.h>.

При відкритті потоку, повертається вказівник на потік, тобто на об'єкт типу FILE.

#include <stdio.h>;

. . . . . . . .

FILE \*fp;

. . . . . . . . . . ..

fp= fopen( ”t.txt”, ”r”;

де fopen(<ім'я\_файлу>,<режим\_відкриття>) - функція для ініціалізації файлу.

Існують такі режими для відкриття файлу:

”w” - відкрити файл для запису, якщо файл існує, то він стирається;

”r” - відкрити файл для читання;

”a” - відкрити файл для додавання, якщо файл існує, то він не стирається і

можна писати в кінець файлу;

”w+” - відкрити файл для запису й виправлення, якщо файл існує, то він

стирається, а далі можна і читати, і писати, розміри файлу можна збільшувати;

”r+” - відкрити файл для читання й запису, але збільшити розмір файлу не

можна;

”a+” - відкрити файл для додавання, тобто можна й читати й писати, у тому

числі й у кінець файлу.

Потік можна відкрити в текстовому (t) або двійковому (b) режимі. За

замовчуванням - текстовий режим. У явному виді режим вказується в такий

спосіб: ”r+b” або ”rb” - двійковий (бінарний) режим.

Для **блокового вводу і виводу** використовуються функції:

1) int fread( void \*ptr, int size, int n, FILE \*fp) , де

void \*ptr - вказівник на область пам'яті, у якій розташовуються зчитувані з

файлу дані;

int size - розмір одного зчитуваного елемента;

int n - кількість зчитуваних елементів;

FILE \*fp - вказівник на файл, з якого відбувається зчитування.

У випадку успішного зчитування інформації функція повертає число

прочитаних елементів (а не байтів), інакше повертає EOF.

2) int fwrite( void \*ptr, int size, int n, FILE \*fp) , де

void \*ptr - вказівник на область пам'яті, у якій розташовуються записувані у

файл дані;

int size - розмір одного записуваного елемента;

int n - кількість записуваних елементів;

FILE \*fp - вказівник на файл, у який відбувається запис.

У випадку успішного запису інформації функція повертає число записаних

елементів, інакше повертає EOF.

Статус: Ознайомлена частково

* + - Початок опрацювання теми: 10.12.23
    - Звершення опрацювання теми: 13.12.23
* Тема №2: Системи числення
  + Джерела Інформації:

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

* + Що опрацьовано:
    - 2-кова система числення
    - 8-кова система числення
    - 10-кова система числення
    - 16-кова система числення
  + Статус: Ознайомлена
  + Початок опрацювання теми: 11.12.2023
  + Звершення опрацювання теми: 11.12.2023
* Тема №3: бінарні файли
  + Джерела Інформації:
  + <https://chat.openai.com/c/08b611a2-4fbb-4d72-88c7-54179be72a6e>
  + Що опрацьовано:
* Бінарні файли є способом збереження даних в комп'ютерному середовищі у вигляді бінарного коду, що складається з нулів і одиниць. У контексті програмування на C++, робота з бінарними файлами дозволяє зчитувати та записувати дані у їхньому найбільш ефективному, компактному та оптимізованому представленні.
* Основні операції з бінарними файлами включають в себе відкриття файлу, зчитування та запис даних у бінарному вигляді. Це відрізняється від текстових файлів, де дані представлені у читабельному для людини вигляді.
* Збереження інформації в бінарному форматі дозволяє ефективно працювати з великими обсягами даних та забезпечує більш швидкий доступ до них. При цьому важливо правильно визначити структуру даних, яку ви збираєтеся зберігати в бінарний файл, і обробляти дані з урахуванням цієї структури. У C++ використовуються функції, такі як **fstream**, для роботи з бінарними файлами.
  + Статус: Ознайомлена
  + Початок опрацювання теми: 11.12.2023
  + Звершення опрацювання теми: 11.12.2023
* Тема №4 Стандартна бібліотека та деталі/методи роботи з файлами
  + Джерела Інформації:
    - https://youtu.be/SSNJ7alki-E?si=X2CZZNdVzQFsHeEC
    - лекційні, практичні заняття
  + Що опрацьовано:

Ofstream(output file stream)- запис у файл, ifstream(input file stream)- читання файлу

. Статус: Ознайомлена

* + Початок опрацювання теми: 10.12.23
  + Звершення опрацювання теми: 14.12.23

# **Виконання роботи:**

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

Завдання №1 Опрацювання теорії

* Деталі завдання: опрацювання різноманітних матеріалів, аналіз та дослідження відео та статей на задані теми. Вивчення найважливіших моментів.

Завдання № 2 VNS lab 6 task 1

* Варіант завдання: 21
* Деталі завдання: Знищити всі парні слова у речені.
* Важливі деталі для врахування в імплементації програми: здійснено використовуючи функцію gets(s)

Завдання № 3 VNS lab 8 task 1

* Варіант завдання: 21
* Деталі завдання:

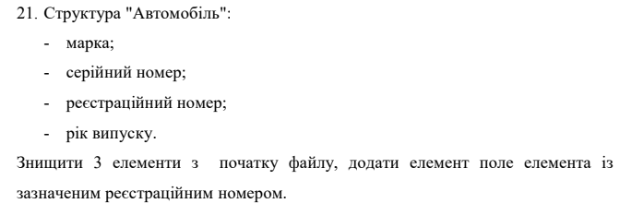


Рисунок 1 деталі завдання

* Важливі деталі для врахування в імплементації програми: розуміння як працюють структури (з цим складно)

Завдання № 4 VNS lab 9 task 1

* Варіант завдання: 21
* Деталі завдання:

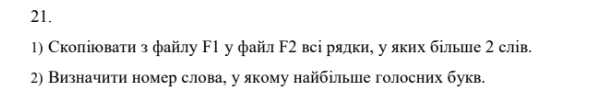


Рисунок 2 деталі завдання

* Важливі деталі для врахування в імплементації програми: розуміння як працювати з файлами
* Завдання №5 Algotester Lab 4v2
* Варіант завдання: 2
* Деталі завдання:

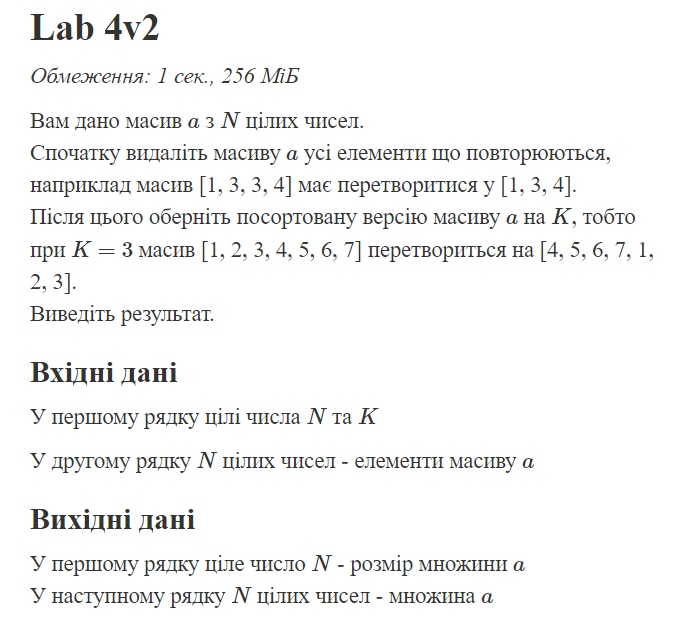


Рисунок 2 деталі завдання

* Важливі деталі для врахування в імплементації програми: написано код зі свою реалізацією без прописаних функцій.

Завдання №6 Algotester Lab 6v2

* Варіант завдання: 2
* Деталі завдання:

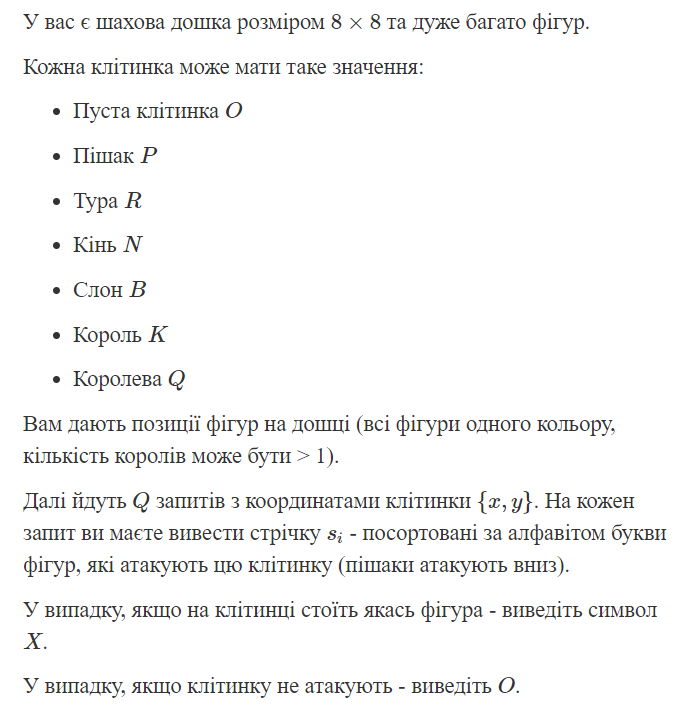


Рисунок 3 деталі завдання

* Важливі деталі для врахування в імплементації програми: використано багато того, чого я раніше не знала☺

Завдання №7 Class Practice Task 1

* Деталі завдання:

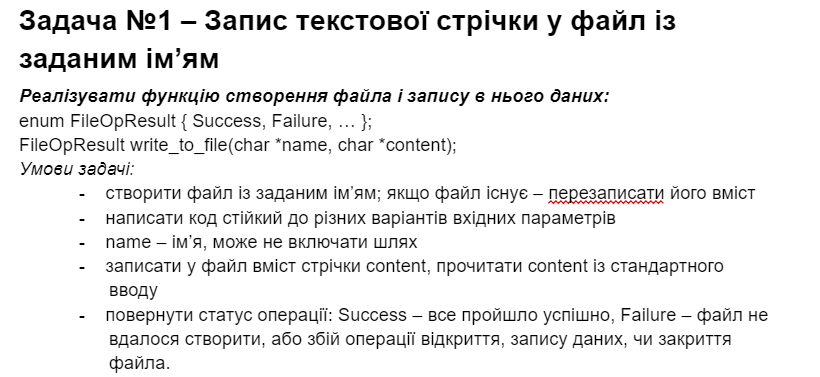


Рисунок 3 деталі завдання

Завдання №8 Class Practice Task 2

* Деталі завдання:

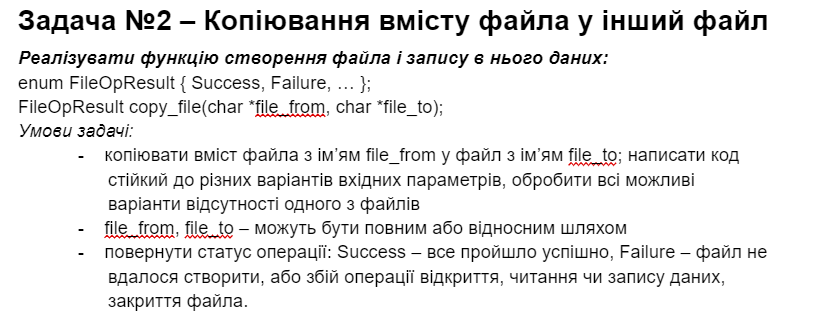


Рисунок 4 деталі завдання

Важливі деталі для врахування в імплементації програми:

Завдання №9 Self Practice Task

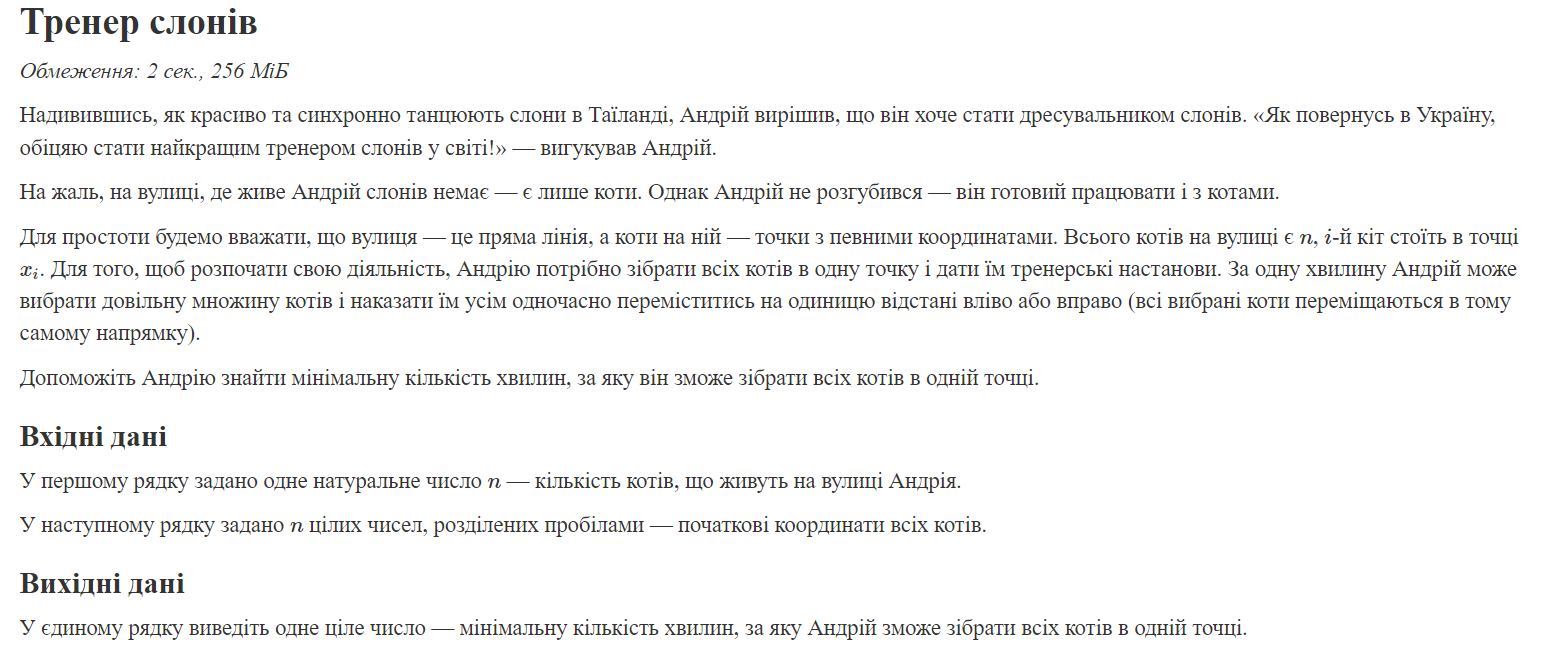


Рисунок 5 деталі завдання

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

Програма №1 VNS lab 8 task 1

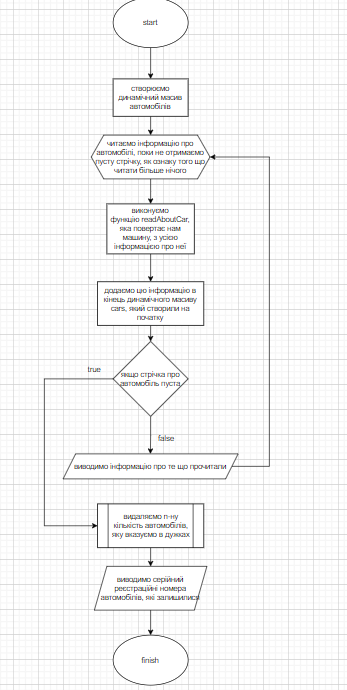
* Блок-схема
* 

Рисунок 6 VNS lab 8 task 1

* Планований час на реалізацію: 2 години

## **3. Конфігурація середовища до виконання завдань:**

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

Завдання № 1 VNS lab 6 task 1

Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <cstring>

using namespace std;

int main() {

    char s[255];

    cin.getline(s, sizeof(s)); // cin.getline(s, sizeof(s)); отримує рядок введення включаючи пробіли та зберігає його в масив s.

    string str;

    // strtok - функція для розбиття на слова на основі роздільника (в нашому випадку пробіл)

    char \*word = strtok(s, " ");

    int wordCount = 0;

    while (word != NULL) {

        wordCount++;

        //якщо false, то додаємо до рядка стр слово і пробіл щоб розділити ті слова

        if (wordCount % 2 == 1) {

            str += word;

            str += " ";

        }

        // переміщення вказівника на першу букву наступного слова

        word = strtok(NULL, " ");

    }

    cout << "Resulting sentence after removing even words: " << str << endl;

    return 0;

}ai\_13/maryna\_polishchuk/Epic 5- Maryna Polishchuk/vns\_lab\_6\_task\_1\_maryna\_polischuk.cpp

Завдання № 2 VNS lab 8 task 1

## Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <fstream>

#include <string>

#include <vector>

#include <iostream>

using namespace std;

struct Car {

  string mark, serialNum, regNum,  graduationYear;

};

void createFile()

{

    ofstream fout("dataStorage.dat", ios::binary);

    if (fout.is\_open()) {

        fout.close();

        cout << "file was created>" << endl;

    } else

        cout << "err of creating file>" << endl;

    cout << endl;

}

    //зчитування записаних данних про автомобілі

Car readfile(int index, int carsCount)

{

    ifstream fin("dataStorage.dat");

    string line;

        //тут пропускаємо index машин, щоб прочитати інфу про потрібну

    for (int i = 0; i < index \* 4; i++)

        getline(fin, line);

    Car output;

    for (int i = 0; getline(fin, line); i++)

    {

        switch (i)

        {

        case 0:

            output.mark = line;

            break;

        case 1:

            output.serialNum = line;

            break;

        case 2:

            output.regNum = line;

            break;

        case 3:

            output.graduationYear = line;

            break;

        }

    }

    return output;

}

    //додавання нового автомобіля до бінарного файлу

void writeToFile(Car car, int carCount)

{

    ofstream fout("dataStorage.dat", ios::binary | ios::app); //app -> відкриття файлу з кінця для запису в нього якихось данних

    if (fout.is\_open())

    {

        fout << car.mark << endl;

        fout << car.serialNum << endl;

        fout << car.regNum << endl;

        fout << car.graduationYear << endl;

        cout << "car added to file!" << endl;

    } else

        cout << "Error while try to tead file!" << endl;

  fout.close();

}

Car addCar(int carCount)

{

    Car input;

    cout << "Car mark: "; cin >> input.mark;

    cout << "Car serialNum: "; cin >> input.serialNum;

    cout << "Car regNum: "; cin >> input.regNum;

    cout << "Car graduationYear: "; cin >> input.graduationYear;

    cout << "car added... ";

    writeToFile(input, carCount);

    return input;

}

void deleteInitialCar(vector<Car> &cars, int index)

{

    ifstream fin("dataStorage.dat");

    ofstream fout("temp.dat");

    if (!fin.is\_open() || !fout.is\_open()) {

        cerr << "Error while try to open file..." << endl;

        return;

    }

    string line;

    vector<Car> tempVect;

    int curLine = 0;

    for (int i = 0; i + index < cars.size()-1; i++)

        tempVect.push\_back(cars[i+index]);

    cars = tempVect;

    while (getline(fin, line))

    {

        curLine++;

            //тут ми перезаписуємо файл

            //основна ідея в тому що ми ігоруємо ті чотири рядки, де

            //запиcана інф-ія про автомобіль,

            //кількість, скільки треба видалити автомобілів (index)

            //та кількість рядків з інформацією

        if(curLine <= index \* 4)

            continue;

        fout << line << endl;

    }

    fin.close();

    fout.close();

        //перезаписуємо файл, в якому зберігається інфа про автомобілі

    remove("dataStorage.dat");

    rename("temp.dat", "dataStorage.dat");

    cout << "Car deleted successfully" << endl;

}

int main()

{

    vector<Car> cars;

    int carsCount = 5;

    //createFile();

        //зчитуємо з файлу інфу про існуючі машини

    cout << "all info about cars: " << endl;

    for (int i = 0; true; i++)

    {

        cars.push\_back(readfile(i, carsCount));

        string temp = cars[i].mark; //для дебагу

        if(temp == "")

            break;

        cout << "Car " << i+1 << ": " << endl;

        cout << "   mark: " << cars[i].mark << endl;

        cout << "   serialNum: " << cars[i].serialNum << endl;

        cout << "   regNum: " << cars[i].regNum << endl;

        cout << "   graduationYear: " << cars[i].graduationYear << endl;

    }

        //видаляємо три перших автомобіля

    deleteInitialCar(cars, 3);

    if(cars.size() == 0)

        cout << "there are no cars";

    else

    {

        cout << "lefted cars: " << endl;

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

            cout << "   regNum: " << cars[i].regNum << endl;

    }

    return 0;

}

## ai\_13/maryna\_polishchuk/Epic 5- Maryna Polishchuk/vns\_lab\_8\_task\_1\_maryna\_polischuk.cpp

Завдання № 3 VNS lab 9 task 1

Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

int CheckHowManyVowel(string str, int startPos, int endPos)

{

    int vowelCount = 0;

    // Проходження через символи в заданому діапазоні

    for (int i = startPos; i <= endPos; ++i) {

        char currentChar = tolower(str[i]); // Переведення в нижній регістр для порівняння з голосними

        // Перевірка, чи є поточний символ голосним

        if (currentChar == 'a' || currentChar == 'e' || currentChar == 'i' || currentChar == 'o' || currentChar == 'u')

            ++vowelCount;

    }

    return vowelCount;

}

bool CheckIfMoreThan2Words(string line, string &largestVowelWord, int &maxVovels)

{

    line += ' ';

    int result = 0, temp = 0;

    bool isWordStarted = false;

    int wordStartIndex;

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

    {

        if(line[i] != ' ' && !isWordStarted)

        {

            temp = i;

            wordStartIndex = i;

            isWordStarted = true;

        }

        if(line[i] == ' ' && isWordStarted)

        {

            result++;

            isWordStarted = false;

            int num = CheckHowManyVowel(line, wordStartIndex, i);

            if(num > maxVovels)

            {

                maxVovels = num;

                largestVowelWord = "";

                for (int j = 0; j < i - wordStartIndex; j++)

                    largestVowelWord += line[wordStartIndex+j];

            }

        }

    }

    return result > 2;

}

int main ()

{

    ifstream var1("file1.dat");

    ofstream output("file2.dat");

    string str, largestVowelWord = "";

    int maxVovels = 0;

    while(getline(var1, str))

    {

        if (CheckIfMoreThan2Words(str, largestVowelWord, maxVovels))

            output << str << endl;

    }

    cout << "largestVowelWord: " << largestVowelWord;

    return 0;

}

## **ai\_13/maryna\_polishchuk/Epic 5- Maryna Polishchuk/vns\_lab\_9\_task\_1\_maryna\_polischuk.cpp**

## Завдання № 4 Algotester Lab 4v2

## Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

//

#include <iostream>

#include <vector>

using namespace std;

int main ()

{

    int N, K;

    cin >> N >> K;

    int a[N];

    for (int i=0; i< N; i++){

        cin >> a[i];

    }

    for (int i= 0; i< N; i++)

    {

        for (int  j=0; j< N-1-i; j++)

        {

            if (a[j]>a[j+1])

            {

                int temp=a[j];

                a[j]=a[j+1];

                a[j+1]= temp;

            }

        }

    }

    vector<int> c;

    c.push\_back(a[0]);

    for (int i = 1; i < N; i++)

    {

        if(a[i] != a[i-1])

            c.push\_back(a[i]);

    }

    cout << c.size() << endl;

    int result[c.size()];

        //перевірка різних значень K

    if(K > c.size())

        K = K % c.size();

    else if (K == N)

    {

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

            cout << c[i] << " ";

        return 0;

    }

    for (int i = 0, j = 0; i < c.size(); i++)

    {

        if(K > 0) {// проходимось по першим K-им елементам масиву і записуємо їх у кінець

            K--;

            result[c.size()-1 - K] = c[i];

            }

        else // записуємо всі елементи в початок

            result[j++] = c[i];

    }

        //вивід данних

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

        cout << result[i] << " ";

    return 0;

}

ai\_13/maryna\_polishchuk/Epic 5- Maryna Polishchuk/algotester\_lab\_4v2\_maryna\_polishchuk.cpp

Завдання № 5 Algotester Lab 6v2

Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

    //слон

void bishop(int &y, int &x, char desk[8][8], vector<char> &result)

//амперант тут використовуємо щоб не створювати копій х і у, а працювати напряму

//деск ми не використовуємо посилання, бо це масив

{

    int dir1[4][2] = {{1, 1}, {-1, -1}, {1, -1}, {-1, 1}};

    for (int i = 0; i < 4; ++i)

    {

        int dy = dir1[i][0];

        int dx = dir1[i][1];

        for (int j = 1; y + j \* dy >= 0 && y + j \* dy < 8 && x + j \* dx >= 0 && x + j \* dx < 8; ++j)

        {

            if (desk[y + j \* dy][x + j \* dx] == 'B')

            {

                result.push\_back('B');

                return;

            }

        }

    }

}

void king(int &y, int &x, char desk[8][8], vector<char> &result)

{

        //король

    int dir2[8][2] = {{-1, 0}, {-1, 1}, {0, 1}, {1, 1}, {1, 0}, {1, -1}, {0, -1}, {-1, -1}};

    for (int i = 0; i < 8; ++i)

    {

        int ny = y + dir2[i][0];

        int nx = x + dir2[i][1];

        if (ny >= 0 && ny < 8 && nx >= 0 && nx < 8 && desk[ny][nx] == 'K')

        {

            result.push\_back('K');

            break;

        }

    }

}

void queen(int &y, int &x, char desk[8][8], vector<char> &result)

{

    //королева

    int dir4[4][2] = {{1, 1}, {1, -1}, {-1, -1}, {-1, 1}};

    for (int i = 0; i < 4; i++)

    {

        int dy = dir4[i][0];

        int dx = dir4[i][1];

        for (int j = 1; y + j \* dy >= 0 && y + j \* dy < 8 && x + j \* dx >= 0 && x + j \* dx < 8; j++)

        {

            if (desk[y + j \* dy][x + j \* dx] == 'Q')

            {

                result.push\_back('Q');

                return;

            }

        }

    }

    for (int j = 0; j < 8; j++)

    {

        if ((y >= 0 && y < 8 && desk[y][j] == 'Q') || (j >= 0 && j < 8 && desk[j][x] == 'Q'))

        {

            result.push\_back('Q');

            return;

        }

    }

}

void rook(int &y, int &x, char desk[8][8], vector<char> &result)

{

    //тура

    int dir5[4][2] = {{0, 1}, {1, 0}, {0, -1}, {-1, 0}};

    for (int i = 0; i < 4; ++i)

    {

        int dy = dir5[i][0];

        int dx = dir5[i][1];

        for (int j = 1; y + j \* dy >= 0 && y + j \* dy < 8 && x + j \* dx >= 0 && x + j \* dx < 8; ++j)

        {

            if (desk[y + j \* dy][x + j \* dx] == 'R')

            {

                result.push\_back('R');

                return;

            }

        }

    }

}

void CheckFigureOnSquare(int y, int x, char desk[8][8], vector<char> &result)

{

    bishop(y, x, desk, result);

    king(y, x, desk, result);

    //кінь

    int dir3[8][2] = {{-2, 1}, {-1, 2}, {1, 2}, {2, 1}, {2, -1}, {1, -2}, {-1, -2}, {-2, -1}};

    for (int i = 0; i < 8; ++i)

    {

        int ny = y + dir3[i][0];

        int nx = x + dir3[i][1];

        if (ny >= 0 && ny < 8 && nx >= 0 && nx < 8 && desk[ny][nx] == 'N')

        {

            result.push\_back('N');

            break;

        }

    }

    //пішак

    if((y - 1 >= 0 && x + 1 < 8 && desk[y - 1][x + 1] == 'P') ||

        (y - 1 >= 0 && x - 1 >= 0 && desk[y - 1][x - 1] == 'P'))

        result.push\_back('P');

    queen(y, x, desk, result);

    rook(y, x, desk, result);

}

int main()

{

    char desk[8][8];

    //вводимо елементи

    for (int i = 0; i < 8; i++)

    {

        for (int j = 0; j < 8; j++)

            cin >> desk[i][j];

    }

    int q;

    cin >> q;

    int inputs[q][2];

    for (int i = 0; i < q; i++)

    {

        for (int j = 0; j < 2; j++)

            cin >> inputs[i][j];

    }

        //сам код

    for (int i = 0; i < q; i++)

    {

        int y = inputs[i][0] - 1;

        int x = inputs[i][1] - 1;

        vector<char> result;

        //перевіряємо чи є в клітинці якась фігура

        if (desk[y][x] != 'O')

        {

            cout << 'X' << endl;

            continue;

        } else

        {

            CheckFigureOnSquare(y, x, desk, result);

            //якщо ми не додали жодної літери у відповідь (тобто клітинку ніхто не атакує)

            if (result.empty())

                cout << "O";

            else

            {

                for (char a : result)

                    cout << a;

            }

            cout << endl;

        }

    }

    return 0;

}

## ai\_13/maryna\_polishchuk/Epic 5- Maryna Polishchuk/algotester\_lab\_6v2\_maryna\_polishchuk.cpp

Завдання № 6 Class Practice Task 1Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <fstream>

#include <iostream>

#include <string>

using namespace std;

enum FileOpResult {Success, Fail};

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

    ofstream file(name); // Use ofstream for writing to a file

    if (!file.is\_open()) {

        return Fail; // If file creation fails, return Failure

    }

    file << content;

    if (file.fail()) {

        cout << "Failed to write text ";

        file.close();

        return Fail;

    }

    cout << "File was opened successfully";

    file.close();

    return Success;

}

int main() {

    string content, name = "MyName";

    cout << "write some text: ";

    getline(cin, content);

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

    if (result == Success) {

        cout << "Content written to file successfully!" << endl;

    } else {

        cout << "Failed to write to file." << endl;

    }

    return 0;

}

ai\_13/maryna\_polishchuk/Epic 5- Maryna Polishchuk/class\_ practice\_task\_1\_epic\_5\_polishchuk\_maryna.cpp

Завдання № 7 Class Practice Task 2

Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

int main (){

    ofstream var1("file\_from");

    for(int i = 1; i < 3; i++)

    {

        string userwrite;

        cout << "Enter your text";

        getline(cin, userwrite);

        var1 << userwrite << endl;

    }

    var1.close(); // <- обов'язково закривати файл, щоб потім мати можливість його нормально відкрити заново

    ifstream var3("file\_from");

    if (var3.is\_open())

    {

        string str;

        //за допомогою getline відкриваємо файл var1,

        //та записуємо зміст першої стрічки в str

        getline(var3, str);

        while(getline(var3, str))

        {

            cout << str << endl;

        }

        var3.close();

        }

            string line;

    ifstream var4("file\_from");

    ofstream var2("file\_to");

    for(int i = 1; getline(var4, line); i++)

    {

        var2 << line << endl;

    }

    var4.close(); var2.close();

    remove("nameOfFile.dat");

    rename("temp.dat", "nameOfFile.dat");

    return 0;

}

ai\_13/maryna\_polishchuk/Epic 5- Maryna Polishchuk/class\_ practice\_task\_2\_epic\_5\_polishchuk\_maryna.cpp

Завдання № 8 self practice

Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

//тренер слонів 0114

# include <iostream>

#include <algorithm>

using namespace std;

int main (){

    int n;

    cin >> n;

    long xy[n];

        for (int i=0; i<n; i++){

        cin >> xy[i];

    }

    sort(xy, xy + n);

    cout << xy[n-1]-xy[0];

return 0;

}

ai\_13/maryna\_polishchuk/Epic 5- Maryna Polishchuk/self\_practice\_work\_algotester\_task\_0114\_maryna\_polishchuk.cpp

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

Завдання № 1 VNS lab 6 task 1

Деталі по виконанню і тестуванню програми

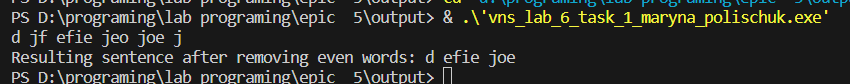


Рисунок 7 VNS lab 6 task 1

Час затрачений на виконання завдання: 6 годин

Завдання № 2 VNS lab 8 task 1

Деталі по виконанню і тестуванню програми

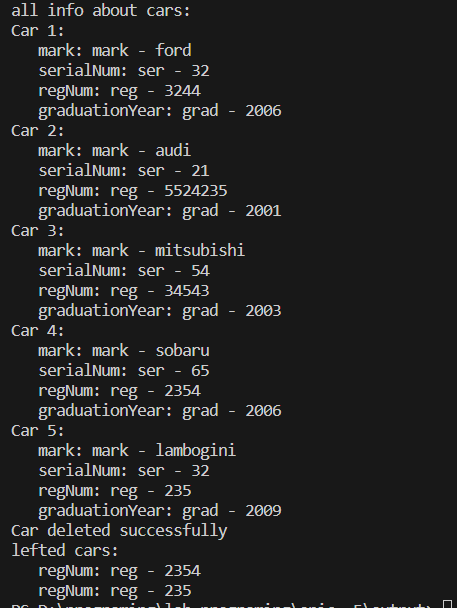
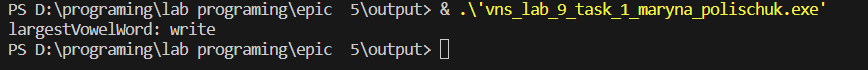


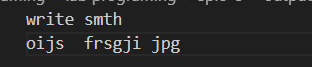
Рисунок 8 VNS lab 8 task 1

Час затрачений на виконання завдання: 10 годин

Завдання № 3 VNS lab 9 task 1

Деталі по виконанню і тестуванню програми





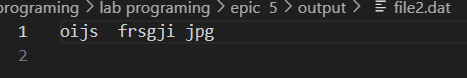


Рисунок 9 VNS lab 9 task 1

Час затрачений на виконання завдання: 7 годин

Завдання № 4 Algotester 4v2

Деталі по виконанню і тестуванню програми

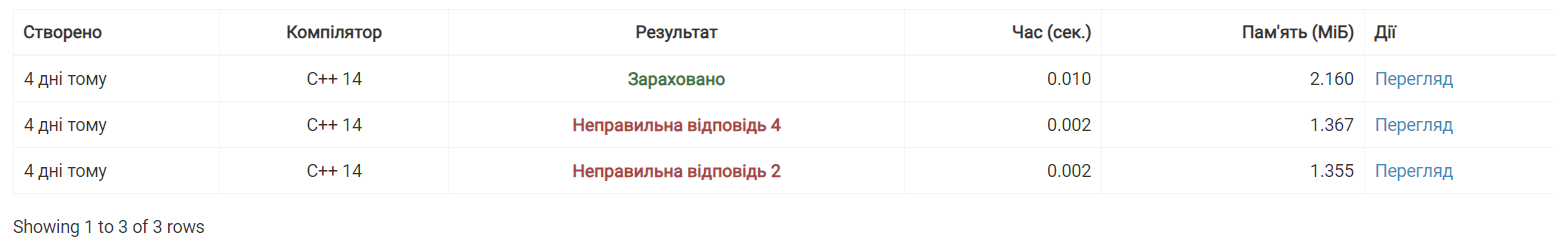


Рисунок 10алго прийняв

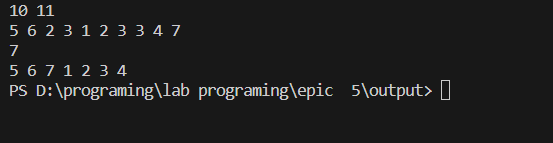


Рисунок 11 тест

Завдання № 5 Algotester 6v2

Деталі по виконанню і тестуванню програми

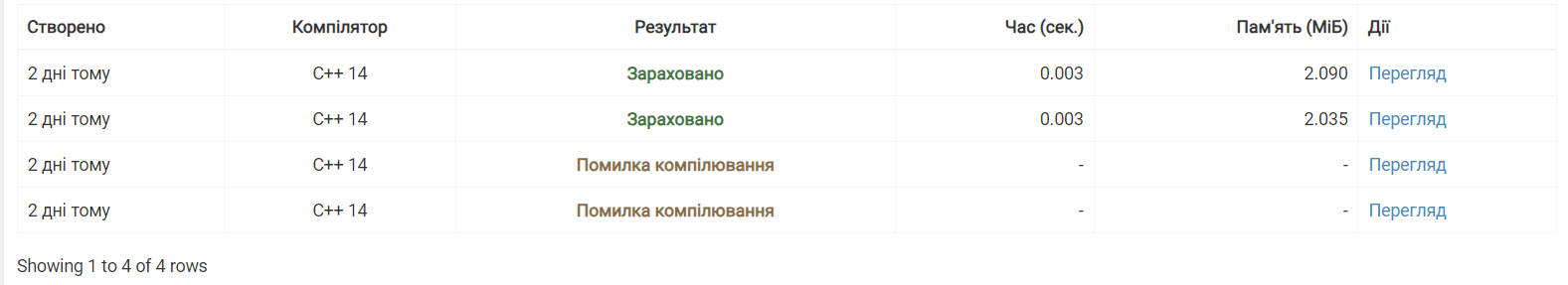


Рисунок 12 алго прийняв

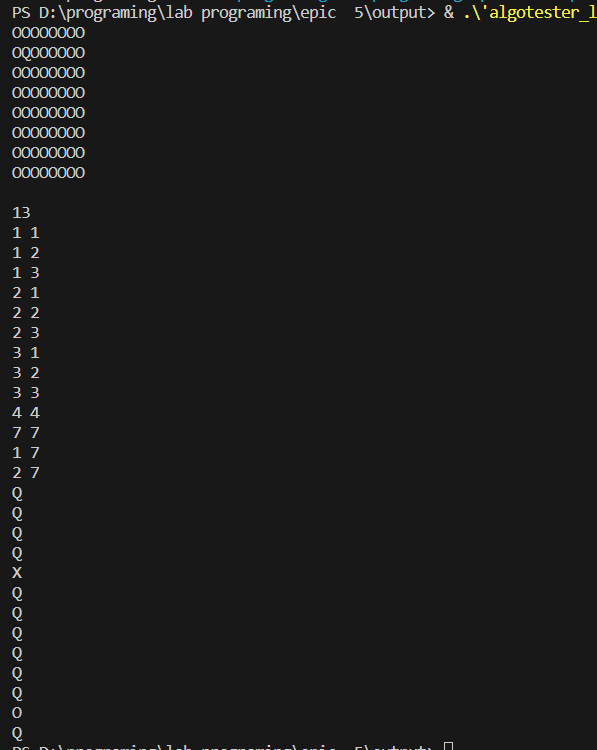
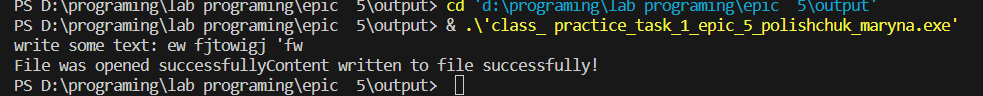


Рисунок 13тест

Завдання № 6 Class Practice Task

Деталі по виконанню і тестуванню програми



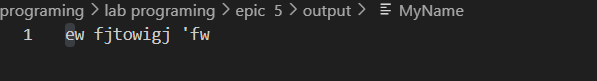


Рисунок 14 тест

Час затрачений на виконання завдання: 8 годин

Завдання № 7 Class Practice Task 2

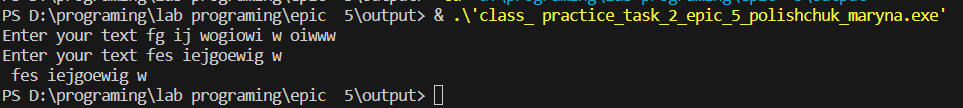


Рисунок 15 тест

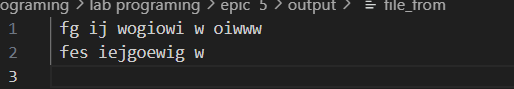


Рисунок 16 файл 1

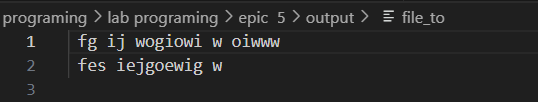


Рисунок 17 файл 2

Деталі по виконанню і тестуванню програми

Завдання №5 Self Practice Task

Деталі по виконанню і тестуванню програми

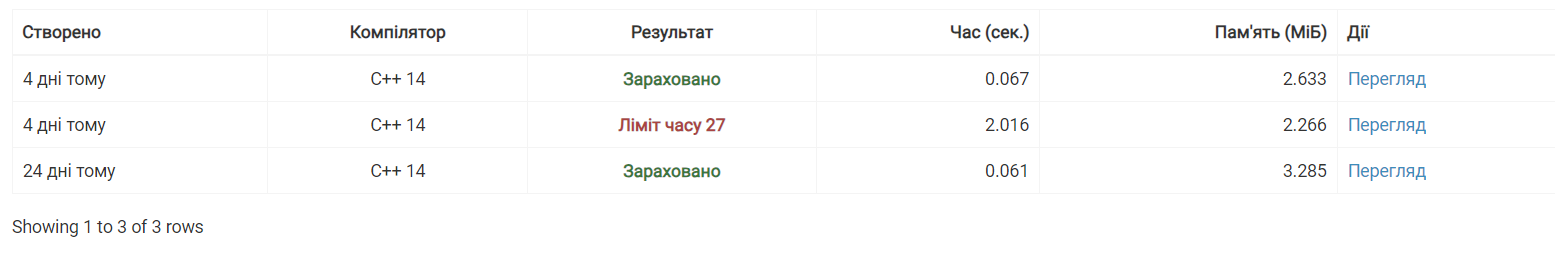


Рисунок 18 алго прийняв

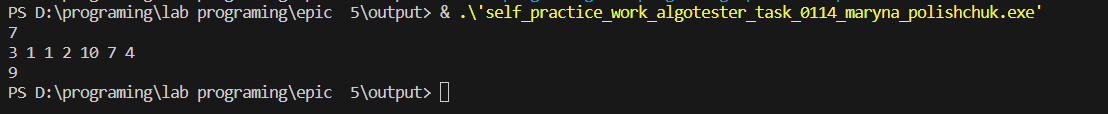


Рисунок 19 тест

Час затрачений на виконання завдання: 5 годин

# **Висновки:**

Навчилися працювати з файлами, системами числення, бінарними файлами, символами і рядковими змінними, текстовими файлами.