МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів
Кафедра систем управління літальних апаратів

Лабораторна робота № 8

з дисципліни «Алгоритмізація та програмування»
на тему «Реалізація алгоритмів сортування та робота з файлами на мові С ++»

ХАІ.301.174.312.4 ЛР

Виконав студент гр. _____312_____

_____Денис Муратов_____
(підпис, дата) (П.І.Б.)

Перевірив
_____ к.т.н., доц. Олена ГАВРИЛЕНКО
(підпис, дата) (П.І.Б.)

2023

МЕТА РОБОТИ

Ознайомитися з алгоритмами обробки масивів у мові С++. Навчитися реалізовувати алгоритми сортування та фільтрації, а також використовувати текстові файли для зчитування і збереження даних.

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Завдання 1. За допомогою текстового редактору створити текстовий файл «array_in_98.txt» з елементами вихідного масиву (n - номер варіанта). У програмі на С++ зчитати і перетворити цей масив відповідно до свого варіанту завдання(Дано цілочисельний масив розміру N. Видалити з масиву всі елементи, що зустрічаються менше трьох разів, і вивести розмір отриманого масиву та його вміст.), ім'я файлу і необхідні змінні ввести з консолі. Вивести результати у файл «array_out_98.txt».

Завдання 2. За допомогою текстового редактору створити текстовий файл «matr_in_33.txt» з елементами вихідного двовимірного масиву (n - номер варіанта). У програмі зчитати і обробити матрицю відповідно до свого варіанту завдання(Дана цілочисельна матриця розміру M × N. Знайти номер останнього з її стовпців, що містять однакову кількість додатних і від’ємних елементів (нульові елементи матриці не враховуються). Якщо таких стовпців немає, то вивести 0.), ім'я файлу і необхідні змінні ввести з консолі. Дописати результати в той же файл.

Завдання 3. Вивчити метод сортування відповідно до свого варіанту (див. табл. 1), проаналізувати його складність і продемонструвати на прикладі з 7-ми елементів (14. Метод сортування: Бульбашкова, Порядок: Зменшення, Тип елементів: Дійсний). Реалізувати у вигляді окремої функції алгоритм сортування елементів масиву. Зчитування і виведення відсортованого масиву організувати на файлах.

Завдання 4. Введення, виведення, обробку масивів реалізувати окремими функціями з параметрами.

ВИКОНАННЯ РОБОТИ

Завдання 1.

Розділ: array98

Вхідні дані:

- filename – ім’я вхідного файлу (string)
- arr[N] – масив цілих чисел, де N = 20 (int)
- n – фактична кількість елементів масиву (int, $1 \leq n \leq 20$)

Вихідні дані:

- arr[N] – фільтрований масив, що містить тільки ті елементи, які зустрічаються тричі і більше (int)
- new_n – нова кількість елементів у масиві (int)

Алгоритм зчитування:

- Відкрити текстовий файл на читання.
- Зчитати перше число — це розмір масиву n.
- Почергово зчитати n чисел і записати їх до масиву arr[].
- Закрити файл

Алгоритм запису:

- Відкрити інший текстовий файл на запис (наприклад, array_out_98.txt).
- Записати нову кількість елементів new_n.
- Вивести елементи масиву, що задовольняють умову (зустрічаються ≥ 3 разів).
- Закрити файл.

Лістинг коду вирішення задачі array98 наведено в дод. А (стор. 9)

Завдання 2.

Розділ: matrix, варіант 33

Вхідні дані:

- filename – ім’я вхідного файлу (string)
- matrix[M][N] – двовимірний масив цілих чисел, де M = 20, N = 20 (int)
- rows, cols – фактичні розміри масиву (int, $1 \leq rows, cols \leq 20$)

Вихідні дані:

- matrix[M][N] – матриця після видалення рядків з максимальним елементом (int)
- new_rows – нова кількість рядків після обробки (int)

Алгоритм зчитування:

- Відкрити файл matr_in_33.txt.
- Зчитати два перших числа — rows і cols.
- У циклі по рядках і стовпцях зчитати всі елементи у матрицю matrix[rows][cols].
- Закрити файл.

Алгоритм запису:

- Відкрити файл matr_in_33.txt в режимі дозапису (ios::app).
- Додати до нього результат обробки (номер останнього стовпця, що задовольняє умову).
- Закрити файл.

Лістинг коду вирішення задачі matrix33 наведено в дод. А (стор. 11)

Завдання 3.

Розділ: Сортування (метод бульбашки), Табл. 1, пункт 14

Вхідні дані:

- filename – ім’я вхідного файлу (string)
- arr[N] – масив дійсних чисел, N = 7 (float)

Вихідні дані:

- arr[N] – відсортований за спаданням масив (float)

Опис:

Масив до сортування:

5.1 2.3 7.4 1.0 4.8

Прохід 1:

5.1 < 7.4 → обмін

2.3 < 5.1 → обмін

1.0 < 2.3 → обмін

4.8 < 1.0 → ні

Після 1-го проходу:

7.4 5.1 2.3 1.0 4.8

І так далі…

Для зчитування та сортування побудовано діаграми активності (див. рис. Б.1, Б.2).

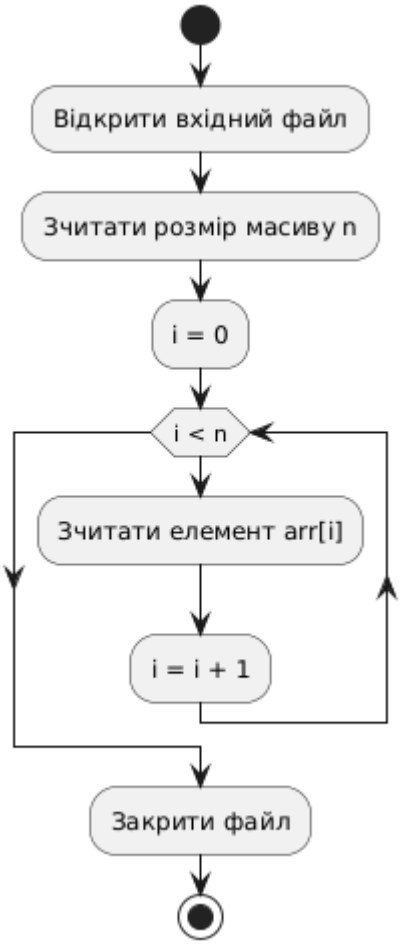

рис. Б.1 - Читання масиву з файлу

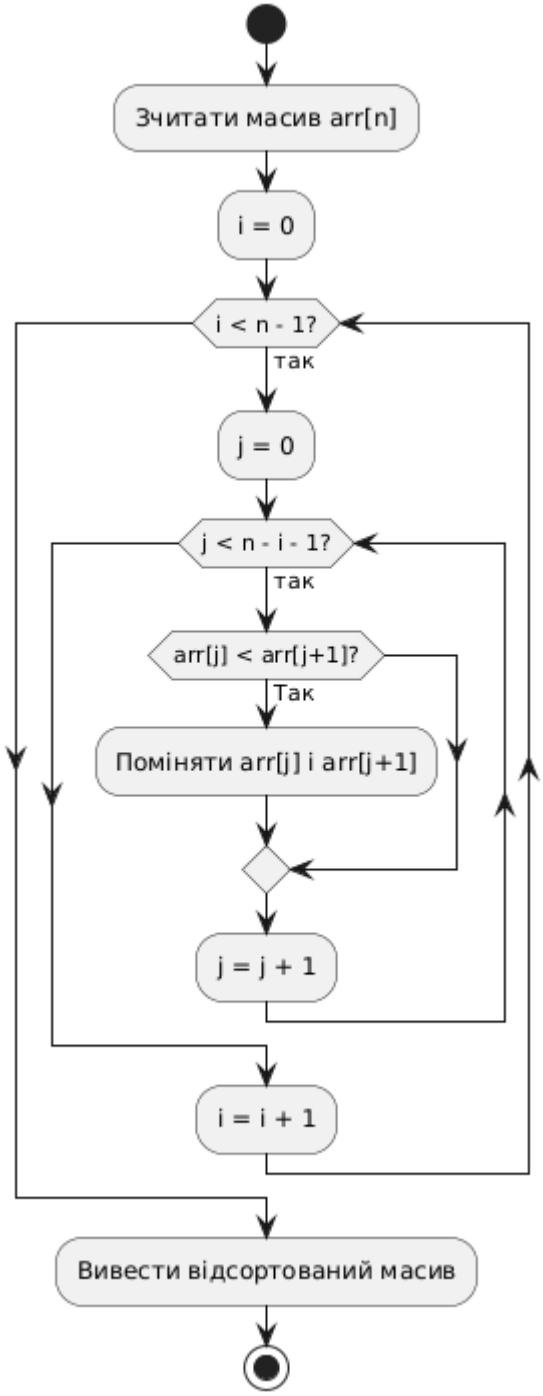

рис. Б.2 - Сортування масиву методом бульбашки за спаданням

Лістинг коду наведено у додатку А (стор. 9)

ВИСНОВКИ

Було вивчено роботу з одновимірними та двовимірними масивами. Закріплено навички використання текстових файлів у C++. Відпрацьовано сортування методом бульбашки та реалізацію фільтрації без допоміжних масивів. Отримано досвід у багатофайловій структурі проєкту.

ДОДАТОК А
Лістинг коду програми

main.cpp

```
#include <iostream>
#include <string>
#include "array_utils.h"
#include "matrix_utils.h"

using namespace std;

int main() {
    string filename;
    int choice;

    cout << "Оберіть завдання (1 - масив, 2 - матриця, 3 - сортування): ";
    cin >> choice;

    if (choice == 1) {
        cout << "Введіть ім'я вхідного файлу масиву: ";
        cin >> filename;
        process_array_task(filename);
    } else if (choice == 2) {
        cout << "Введіть ім'я вхідного файлу матриці: ";
        cin >> filename;
        process_matrix_task(filename);
    } else if (choice == 3) {
        cout << "Введіть ім'я вхідного файлу для сортування: ";
        cin >> filename;
        process_sorting_task(filename);
    } else {
        cout << "Невірний вибір." << endl;
    }

    return 0;
}
```

array_utils.h

```
#pragma once
#include <string>
using namespace std;

const int N = 20;

void get_mas(string filename, int mas[], int &n);
void show_mas(string filename, const int mas[], int n);
void bubble_sort_desc(string filename);
void process_array_task(string filename);
void process_sorting_task(string filename);
```

array_utils.cpp

```
#include <fstream>
#include <iostream>
#include <map>
#include "array_utils.h"

using namespace std;

void get_mas(string filename, int mas[], int &n) {
    ifstream fin(filename);
    if (!fin) {
        cout << "Помилка відкриття файлу!" << endl;
        return;
    }
    fin >> n;
    for (int i = 0; i < n; i++) {
        fin >> mas[i];
    }
    fin.close();
}

void show_mas(string filename, const int mas[], int n) {
    ofstream fout(filename);
    fout << n << endl;
    for (int i = 0; i < n; i++) {
        fout << mas[i] << " ";
    }
    fout << endl;
    fout.close();
}

void process_array_task(string filename) {
    int mas[N], n;
    get_mas(filename, mas, n);

    map<int, int> count;
    for (int i = 0; i < n; i++) {
        count[mas[i]]++;
    }

    int new_n = 0;
    for (int i = 0; i < n; i++) {
        if (count[mas[i]] >= 3) {
            mas[new_n++] = mas[i];
        }
    }

    cout << "Новий розмір масиву: " << new_n << endl;
    for (int i = 0; i < new_n; i++) {
```

```
        cout << mas[i] << " ";
    }
    cout << endl;

    show_mas("array_out_98.txt", mas, new_n);
}

void process_sorting_task(string filename) {
    double mas[7];
    ifstream fin(filename);
    if (!fin) {
        cout << "Помилка відкриття файлу!" << endl;
        return;
    }
    for (int i = 0; i < 7; i++) {
        fin >> mas[i];
    }
    fin.close();

    // Сортування бульбашкою за спаданням
    for (int i = 0; i < 6; i++) {
        for (int j = 0; j < 6 - i; j++) {
            if (mas[j] < mas[j + 1]) {
                swap(mas[j], mas[j + 1]);
            }
        }
    }

    ofstream fout("sorted_out.txt");
    fout << "Відсортований масив: ";
    for (int i = 0; i < 7; i++) {
        fout << mas[i] << " ";
    }
    fout << endl;
    fout.close();
}
```

matrix_utils.h

```
#pragma once
#include <string>
using namespace std;

const int M = 20;

void get_matrix(string filename, int matrix[M][N], int &rows, int &cols);
void show_matrix(string filename, const int matrix[M][N], int rows, int cols);
void process_matrix_task(string filename);
```

matrix_utils.cpp

```
#include <fstream>
#include <iostream>
#include "matrix_utils.h"

using namespace std;

void get_matrix(string filename, int matrix[M][N], int &rows, int &cols) {
    ifstream fin(filename);
    if (!fin) {
        cout << "Помилка відкриття файлу!" << endl;
        return;
    }
    fin >> rows >> cols;
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            fin >> matrix[i][j];
        }
    }
    fin.close();
}

void show_matrix(string filename, const int matrix[M][N], int rows, int cols) {
    ofstream fout(filename, ios::app);
    fout << "\nМатриця:\n";
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            fout << matrix[i][j] << " ";
        }
        fout << endl;
    }
    fout.close();
}

void process_matrix_task(string filename) {
    int matrix[M][N], rows, cols;
    get_matrix(filename, matrix, rows, cols);

    int result = 0;
    for (int j = 0; j < cols; j++) {
        int pos = 0, neg = 0;
        for (int i = 0; i < rows; i++) {
            if (matrix[i][j] > 0) pos++;
            else if (matrix[i][j] < 0) neg++;
        }
        if (pos == neg && pos != 0) {
            result = j + 1; // останній підходящий стовпець (нумерація з 1)
        }
    }
```

```
    ofstream fout(filename, ios::app);
    fout << "\nРезультат: ";
    fout << (result == 0 ? "0" : to_string(result)) << endl;
    fout.close();
}
```