Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра систем штучного інтелекту

Звіт

про виконання лабораторних та практичних робіт блоку № 5

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

з дисципліни: «Основи програмування»

до:

ВНС Лабораторної Роботи № 6
ВНС Лабораторної Роботи № 8
ВНС Лабораторної Роботи № 9
Алготестер Лабораторної Роботи №4
Алготестер Лабораторної Роботи №6
Практичних Робіт до блоку №5

Виконав:
Студент(ка) групи ШІ-13
Яцишин Роман Олегович

Львів – 2024

Тема :

Файли. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами. Створення й використання бібліотек.

Мета :

Навчитись працювати з файлами, вносити у нього зміни. Опрацювати деталі роботи з файлами, набір команд у бібліотеці. Навчитись створювати власні бібліотеки та доцільно їх використовувати.

Теоретичні відомості:

1. Вступ до Роботи з Файлами:
    - Основні операції з файлами: відкриття, читання, запис, закриття
    - Робота з файловими дескрипторами
2. Символи і Рядкові Змінні:
    - Робота з char та string: основні операції і методи
3. Текстові Файли:
    - Особливості читання та запису текстових файлів
    - Обробка рядків з файлу: getline, ignore, peek
    - Форматування тексту при записі: setw, setfill, setprecision
4. Бінарні Файли:
    - Вступ до бінарних файлів: відмінності від текстових, приклади (великі дані, ігрові ресурси, зображення)
    - Читання та запис бінарних даних
    - Робота з позиціонуванням у файлі: seekg, seekp
    - Серіалізація об'єктів у бінарний формат
5. Стандартна бібліотека та робота з файлами:
    - Огляд стандартної бібліотеки для роботи з файлами
    - Потоки вводу/виводу: ifstream, ofstream, fstream
6. Створення й використання бібліотек:
    - Вступ до створення власних бібліотек у C++
    - Правила розбиття коду на header-и(.h) та source(.cpp) файли
    - Статичні проти динамічних бібліотек: переваги та використання
    - Інтерфейси бібліотек: створення, документування, версіонування

Виконання роботи:

1. Опрацювання завдання та вимог до програм та середовища:

Програмний код №1

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. Виконати ввід рядка, використовуючи функцію gets(s) і здійснити обробку рядка у відповідності зі своїм варіантом.

4. Надрукувати всі слова, які співпадають з її першим словом.

Програмний код №2

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.

4. Структура "Людина":
   - прізвище, ім'я, по батькові;
   - домашня адреса;
   - номер телефону;
   - вік.

   Знищити усі елементи із заданим віком, додати елемент після елемента із заданим номером.

Програмний код №3

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

Виконати завдання.

4.

1) Скопіювати з файлу F1 у файл F2 рядки, починаючи з 4.
2) Підрахувати кількість символів в останньому слові F2.

Програмний код №4

Вам дано масив, який складається з N додатніх цілих чисел.

Ваше завдання - розділити його на три частини, по остачі від ділення на 3, по зростанню остачі (тобто спочатку йдуть числа, у яких остача 0, далі числа з остачею 1 і тоді нарешті числа з остачею 2).

Далі необхідно ті елементи, остача від ділення на 3 яких парна посортувати по зростанню, а ті, у яких остача 1 - по спаданню.

Після цього видаліть усі дублікати з масиву.

Виведіть результуючий масив.

Вхідні дані

У першому рядку N - кількість чисел.

У другому рядку N чисел a_i - елементи масиву.

Вихідні дані

У першому рядку M - кількість чисел у масиву

У другому рядку M посоротованих за умовою чисел.

Програмний код №4

Вам дано масив a з N цілих чисел.
Спочатку видаліть масиву a усі елементи що повторюються, наприклад масив [1, 3, 3, 4] має перетворитися у [1, 3, 4].
Після цього оберніть посортовану версію масиву a на K, тобто при $K = 3$ масив [1, 2, 3, 4, 5, 6, 7] перетвориться на [4, 5, 6, 7, 1, 2, 3].
Виведіть результат.

Вхідні дані

У першому рядку цілі числа N та K

У другому рядку N цілих чисел - елементи масиву a

Вихідні дані

У першому рядку ціле число N - розмір множини a
У наступному рядку N цілих чисел - множина a

Програмний код №5

У Клінта в черговий раз виключилось світло і йому немає чим зайнятися. Так як навіть це не заставить його подивитися збережені відео про програмування на ютубі - він вирішив придумати свою гру на основі судоку.

Гра виглядає так:

Є поле розміром $N \times N$, в якому частина клітинок заповнена цифрами, а частина клітинок пусті (позначаються нулем). Також у нього є Q пар координат X та Y.

Завданням гри є написати до кожної координати скільки чисел туди можна вписати (якщо вона пуста) і які це числа (обов'язково в посортовані по зростанню!). В клітинку можна вписати лише ті числа, які не зустрічаються в рядку та стовбці, які перетинаються у цій клітинці.

Під час гри поле не міняється!

Також необовязково, щоб це було валідне судоку! Якщо є клітинка, в яку не можна вписати ніяку цифру - виведіть 0.

Також допускаються рядки та стовпці, в яких цифра записана кілька разів.

Програмний код №6

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.

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.

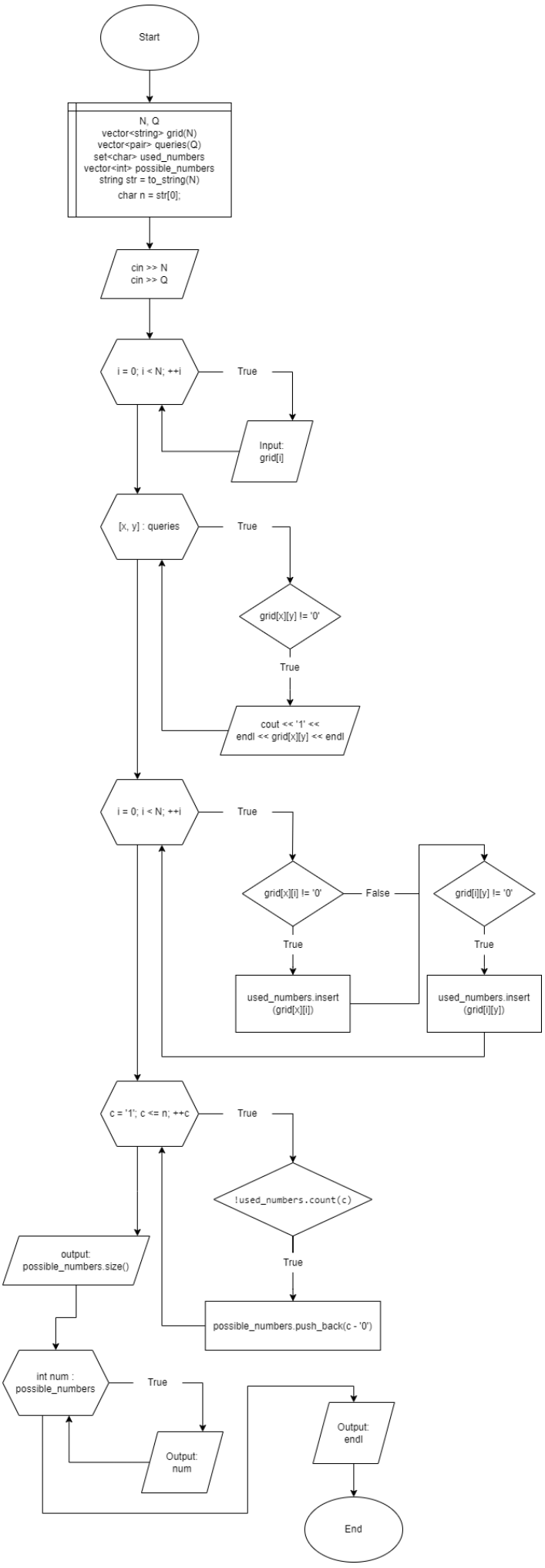
Програмний код №7

Після завершення основного туру олімпіади з програмування Зеник отримав невеличкий клаптик паперу, на якому було надруковане число x — кількість балів, що набрав Зеник. Зауважте, що згідно з кращими традиціями олімпіади з програмування, кількість балів Зеника не може бути нульовою чи від'ємною.

Помітивши не дуже щасливе обличчя Зеника, Марічка нагадала йому про щасливі цифри. Як ви вже напевно знаєте, щасливими вважають цифри 4 та 7. Марічка запевнила Зеника, що найкращим є не найбільший результат, а той, десятковий запис якого містить найбільше щасливих цифр.

Вам необхідно допомогти юному учаснику олімпіади з програмування та порахувати кількість щасливих цифр у його результаті.

2. Дизайн:


Start
N, Q
vector<string> grid(N)
vector<pair> queries(Q)
set<char> used_numbers
vector<int> possible_numbers
string str = to_string(N)
char n = str[0];
cin >> N
cin >> Q
i = 0; i < N; ++i
True
Input:
grid[i]
[x, y] : queries
True
grid[x][y] != '0'
True
cout << '1' <<
endl << grid[x][y] << endl
i = 0; i < N; ++i
True
grid[x][i] != '0'
False
grid[i][y] != '0'
True
True
used_numbers.insert
(grid[x][i])
used_numbers.insert
(grid[i][y])
c = '1'; c <= n; ++c
True
!used_numbers.count(c)
True
output:
possible_numbers.size()
possible_numbers.push_back(c - '0')
int num :
possible_numbers
True
Output:
num
Output:
endl
End

3. Код програми:

1) Algotester Lab 4 Variant 2

```cpp
#include <iostream>
#include <sstream>
#include <algorithm>
#include <vector>

using namespace std;

int partition(vector<int> &vec, int low, int high) {

    int pivot = vec[high];
    int i = (low - 1);

    for (int j = low; j <= high - 1; j++) {
        if (vec[j] <= pivot) {
            i++;
            int temp = vec[i];
            vec[i] = vec[j];
            vec[j] = temp;
        }
    }
    int temp2 = vec[i + 1];
    vec[i + 1] = vec[high];
    vec[high] = temp2;
    return (i + 1);
}

void quickSort(vector<int> &vec, int low, int high) {

    if (low < high) {
        int pi = partition(vec, low, high);
        quickSort(vec, low, pi - 1);
        quickSort(vec, pi + 1, high);
    }
}

void removeDuplicates(std::vector<int>& vec) {
    int size = vec.size();
```

```
    for (int i = 0; i < size; i++) {
        for (int j = i + 1; j < size; j++) {
            if (vec[i] == vec[j]) {
                for (int k = j; k < size - 1; k++) {
                    vec[k] = vec[k + 1];
                }
                size--;
                j--;
            }
        }
    }

    vec.resize(size);
}


int main(){
    int element = 0;
    int N = 0;
    int K = 0;
    cin >> N >> K;

    vector<int> arr;

    for (int c = 0; c < N; c++) {
        cin >> element;
        arr.push_back(element);
    }

    removeDuplicates(arr);
    quickSort(arr, 0, arr.size() - 1);

    vector<int> temp(arr.begin(), arr.begin() + K);

    for (int i = 0; i < K; i++) {
        arr[arr.size() - K + i] = temp[i];
    }

    int M = arr.size();
    cout << M << endl;
    for (int num : arr) {
        cout << num << " ";
    }

    return 0;
}
```

2) Self Practice task

```cpp
#include <iostream>
#include <string>

using namespace std;

int main() {

    string x;
    cin >> x;

    int count = 0;

    for (char digit : x) {
        if (digit == '4' || digit == '7') {
            count++;
        }
    }

    cout << count << endl;

    return 0;
}
```

3) Class practice task

```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <cstring>

using namespace std;

enum FileOpResult {Success, Failure};
FileOpResult write_to_file(char *name, char *content){
    ofstream outFile(name, ios::app);
    if (outFile.is_open()) {
        outFile << content;
        outFile.close();
    } else {
        return Failure;
    }

    outFile.close();

    return Success;
}

FileOpResult copy_file(char *file_from, char *file_to){
    ifstream inFile(file_from, ios::app);
    ofstream outFile(file_to, ios::app);

    if (inFile.is_open())
    {
        if (outFile.is_open())
        {
            string content;
            while (getline(inFile, content)) {
                outFile << content << endl;
            }
        } else {
            cout << "file_to.txt is not found!";
            return Failure;
        }
```

```
    } else {
        cout << "file_from.txt is not found!";
        return Failure;
    }

    inFile.close();
    outFile.close();

    return Success;
}

int main(){
    string file_from = "file_from.txt";
    string file_to = "file_to.txt";
    string content;
    getline(cin, content);

    char* file_from_char = new char[file_from.length() + 1];
    strcpy(file_from_char, file_from.c_str());

    char* file_to_char = new char[file_to.length() + 1];
    strcpy(file_to_char, file_to.c_str());

    char* content_char = new char[content.length() + 1];
    strcpy(content_char, content.c_str());


    write_to_file(file_from_char, content_char);
    copy_file(file_from_char, file_to_char);

    delete[] file_from_char;
    delete[] file_to_char;
    delete[] content_char;
}
```

4) VNS Lab 6

```cpp
// Надрукувати всі слова, які співпадають з її першим словом.

#include <iostream>
#include <sstream>
#include <string>
#include <vector>

using namespace std;

int main() {
    string input;
    getline(cin, input);
    char delimiter = ' ';
    istringstream iss(input);
    string word;
    vector<string> words;
    vector<int> idxs;

    while (getline(iss, word, delimiter)) {
        words.push_back(word);
    }

    int idx = 0;
    for (string w_now : words)
    {
        if ((w_now == words[0]) && (idx != 0))
        {
            cout << w_now << endl;
        }
        ++idx;
    }

    return 0;
}
```

5) VNS Lab 8

```cpp
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
#include <vector>

using namespace std;

struct Human {
    string firstname;
    string lastname;
    string middlename;
    string address;
    string phonenumber;
    string age;
};
void writeRecord(ofstream &outFile, const Human &person) {
    size_t length;

    // Write `firstname`
    length = person.firstname.size();
    outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
    outFile.write(person.firstname.c_str(), length);

    // Write `lastname`
    length = person.lastname.size();
    outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
    outFile.write(person.lastname.c_str(), length);

    // Write `middlename`
    length = person.middlename.size();
    outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
    outFile.write(person.middlename.c_str(), length);

    // Write `address`
    length = person.address.size();
    outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
    outFile.write(person.address.c_str(), length);

    // Write `phonenumber`
    length = person.phonenumber.size();
    outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
    outFile.write(person.phonenumber.c_str(), length);

    // Write `age`
    length = person.age.size();
    outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
    outFile.write(person.age.c_str(), length);
}
```

```cpp
void readRecord(ifstream &inFile, Human &person) {
    size_t length;

    // Read `firstname`
    inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
    person.firstname.resize(length);
    inFile.read(&person.firstname[0], length);

    // Read `lastname`
    inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
    person.lastname.resize(length);
    inFile.read(&person.lastname[0], length);

    // Read `middlename`
    inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
    person.middlename.resize(length);
    inFile.read(&person.middlename[0], length);

    // Read `address`
    inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
    person.address.resize(length);
    inFile.read(&person.address[0], length);

    // Read `phonenumber`
    inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
    person.phonenumber.resize(length);
    inFile.read(&person.phonenumber[0], length);

    // Read `age`
    inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
    person.age.resize(length);
    inFile.read(&person.age[0], length);
}
```

```cpp
void displayRecord(const Human &person) {
    cout << "-------------------------" << endl;
    cout << "First Name : " << person.firstname << endl;
    cout << "Last Name  : " << person.lastname << endl;
    cout << "Middle Name: " << person.middlename << endl;
    cout << "Address    : " << person.address << endl;
    cout << "Phone No.  : " << person.phonenumber << endl;
    cout << "Age        : " << person.age << endl;
    cout << "-------------------------" << endl;
}


int main(){
    int idx;
    cout << "index of record, after which to put new record: ";
    cin >> idx;

    vector<Human> records;
    Human person;
    // Human people[3] = {
    //     {"Roman", "Yatsyshyn", "Olehowych", "city Lviv", "+38666", "18"},
    //     {"Vitsliy", "Portnikov", "Batkowych", "city Kyiv", "+38666069", },
    //     {"Orest", "Melnyk", "Mykhaylowych", "city NONe", " ", "60"}
    // };

    Human newRecord = {"Olersandr", "Matrunych", "", "village Domazhyr", "+38096969696", "17"};

    // ofstream outFile("humans.bin", ios::binary);
    // if (!outFile) {
    //     cerr << "Failed to open file for writing." << endl;
    //     return 1;
    // }

    // // Write each record to the file
    // for (const Human &person : people) {
    //     writeRecord(outFile, person);
    // }
```

```
    // outFile.close();
    // cout << "Records written successfully to 'humans.dat'." << endl;

    ifstream file("humans.bin", ios::binary);

    if (file){
        while (file.peek() != EOF) { // Check for end of file
            readRecord(file, person);
            displayRecord(person);
            records.push_back(person);
        }
    }
    else {
        cerr << "Failed to open file for writing." << endl;
    }

    file.close();

    records.insert(records.begin() + idx + 1, newRecord);

    ofstream tempFile("temp.bin", ios::binary);
    if (!tempFile) {
        cerr << "Failed to open temporary file for writing." << endl;
        return 0;
    }

    for (const Human &record : records) {
        if (record.age.empty() != true) {
            writeRecord(tempFile, record);
        }
    }
    tempFile.close();

    string filename = "humans.bin";
    string tempfilename = "temp.bin";
```

```
    if (remove(filename.c_str()) != 0) {
        cerr << "Error deleting original file." << endl;
        return 0;
    }
    if (rename(tempfilename.c_str(), filename.c_str()) != 0) {
        cerr << "Error renaming temporary file." << endl;
        return 0;
    }

    cout << "Record added successfully." << endl;

    return 0;
}
```

6) VNS Lab 9

```cpp
// 1) Скопіювати з файлу F1 у файл F2 рядки, починаючи з 4.
// 2) Підрахувати кількість символів в останньому слові F2.

#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
#include <vector>

using namespace std;

int main(){
    string filename_f1 = "F1.txt";
    string filename_f2 = "F2.txt";
    string arr[13] = {
        "- .... .", "--- -. .-.. -.--", ".-- .- -.--", "- ---",
        "-.. ---", "--. .-. . .- -", ".-- --- .-. -.-",
        ".. ...", "- ---", ".--- --- ...- .", ".-- .... .- -",
        "-.-- --- ..-", "-.. ---"
    };
    size_t nmbr_of_symbols = 0;
    int idx_last = 10;
    int idx_target = 4;
    int idx_now = 0;
    string line;
    ifstream file1(filename_f1, ios::app);
    ifstream file2(filename_f2, ios::app);
    // ofstream file1(filename_f1, ios::app);
    // ofstream file2(filename_f2, ios::app);

```

```
    if ((file1.is_open()) && (file2.is_open()))
    {
        // for (int i = 0; i < 13; i++)
        // {
        //     file1 << arr[i] << endl;
        // }
        // while (getline(file1, line))
        // {
        //     ++idx_now;
        //     if (idx_now >= idx_target)
        //     {
        //         file2 << line << endl;
        //     }
        // }
        while (getline(file2, line))
        {
            ++idx_now;
            if (idx_now == idx_last)
            {
                // cout << line << endl;
                nmbr_of_symbols = line.size();
                break;
            }
        }
        cout << nmbr_of_symbols << endl;

    } else {
        cerr << "Error opening file for writing!\n";
    }
    file1.close();
    file2.close();
}

```

7) Algotester Lab 4

```cpp
#include <iostream>
#include <sstream>
#include <algorithm>
#include <vector>

using namespace std;

int main(){
    int element = 0;
    int n = 0;
    cin >> n;

    vector<int> arr;
    vector<int> mod0, mod1, mod2;

    for (int c = 0; c < n; c++) {
        cin >> element;
        arr.push_back(element);
    }

    for (int num : arr) {
        if (num % 3 == 0) mod0.push_back(num);
        else if (num % 3 == 1) mod1.push_back(num);
        else mod2.push_back(num);
    }

    // Сортування
    sort(mod0.begin(), mod0.end());
    sort(mod1.begin(), mod1.end(), greater<int>());
    sort(mod2.begin(), mod2.end());

    // Об'єднання частин
    vector<int> result;
    result.insert(result.end(), mod0.begin(), mod0.end());
    result.insert(result.end(), mod1.begin(), mod1.end());
    result.insert(result.end(), mod2.begin(), mod2.end());

    vector<int>::iterator it;
    it = unique(result.begin(), result.end());

    int M = distance(result.begin(), it);
    result.resize(distance(result.begin(), it));

    cout << M << endl;
    for (int num : result) {
        cout << num << " ";
    }

    return 0;
}
```

8) Algotester Lab 6

```cpp
#include <iostream>
#include <vector>
#include <set>
#include <string>
#include <algorithm>

using namespace std;

int main(){
    int N = 0;
    cin >> N;

    vector<string> grid(N);
    for (int i = 0; i < N; ++i) {
        cin >> grid[i];
    }

    int Q;
    cin >> Q;

    vector<pair<int, int>> queries(Q);
    for (int i = 0; i < Q; ++i) {
        cin >> queries[i].first >> queries[i].second;
        --queries[i].first;
        --queries[i].second;
    }

    for (const auto& [x, y] : queries) {
        // cout << grid[x][y] << endl;
        if (grid[x][y] != '0') {
            cout << '1' << endl << grid[x][y] << endl;
            continue;
```

```
        }

        set<char> used_numbers;
        for (int i = 0; i < N; ++i) {
            if (grid[x][i] != '0') {
                used_numbers.insert(grid[x][i]);
            }
            if (grid[i][y] != '0') {
                used_numbers.insert(grid[i][y]);
            }
        }

        vector<int> possible_numbers;

        string str = to_string(N);
        char n = str[0];
        for (char c = '1'; c <= n; ++c) {
            if (!used_numbers.count(c)) {
                possible_numbers.push_back(c - '0');
            }
        }

        cout << possible_numbers.size() << endl;
        for (int num : possible_numbers) {
            cout << num << " ";
        }
        cout << endl;
    }

    return 0;
}
```

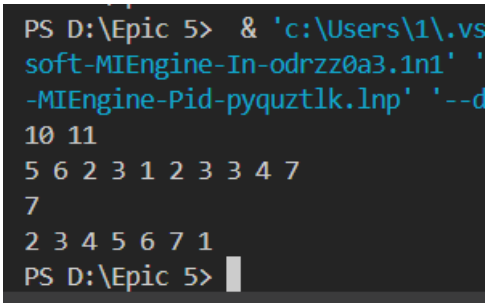
5. Результати виконання завдань, тестування та фактично затрачений час:

1) Algotester Lab 4

```
PS D:\Epic 5>  & 'c:\Users\
soft-MIEngine-In-0yr1zfal.t
-MIEngine-Pid-o2ruu5i3.vc5'
10
1 33 4 8 6 5 2 7 5 0
9
0 6 33 7 4 1 2 5 8
PS D:\Epic 5>
```

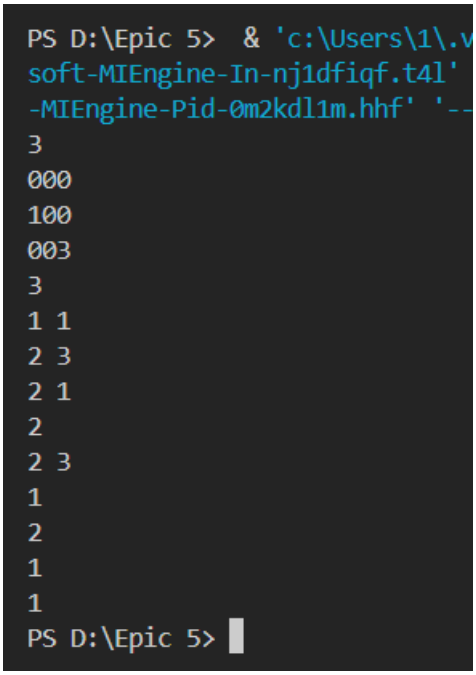
Фактично затрачений час: 20 хв

2) Algotester Lab 4 variant 2



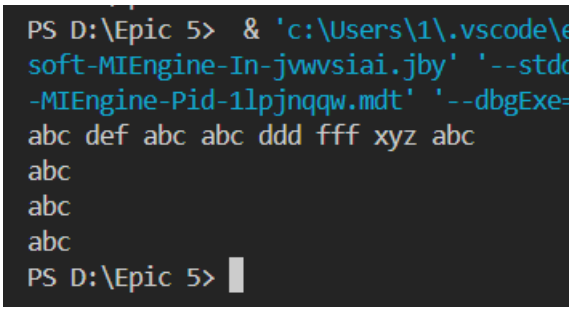
Фактично затрачений час: 20 хв

3) Algotester Lab 6



Фактично затрачений час: 40 хв

4) VNS Lab 6

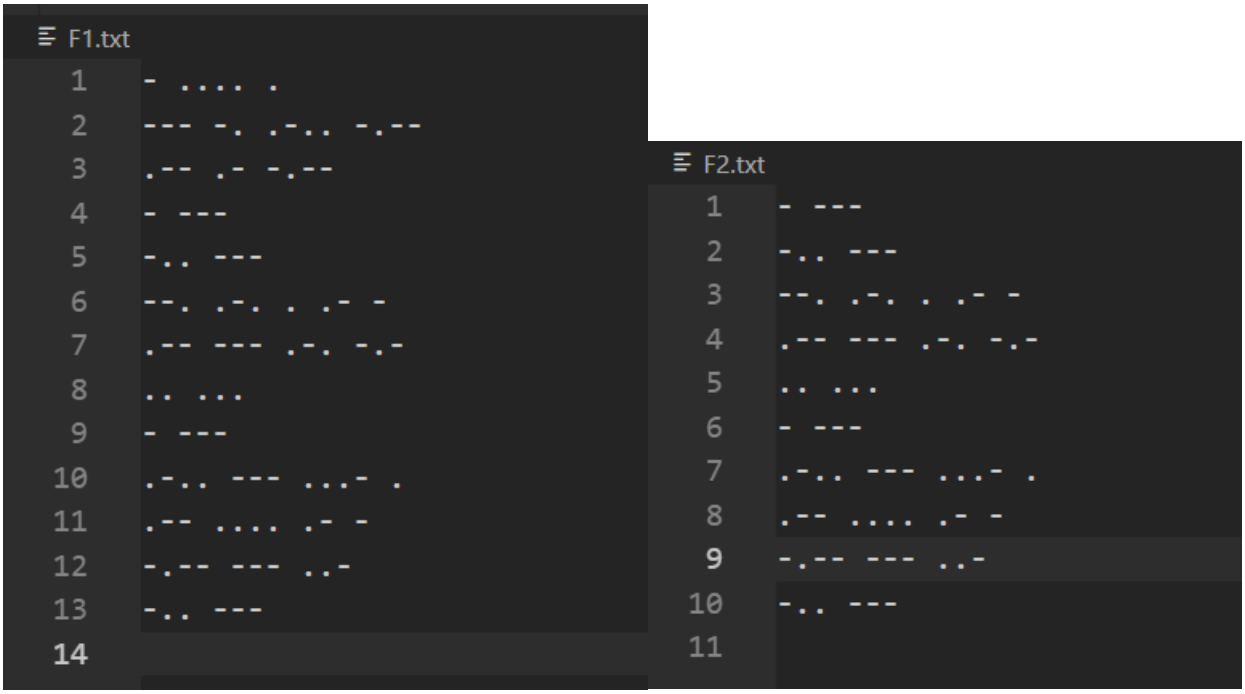


Фактично затрачений час: 30 хв

5) VNS Lab 8

Фактично затрачений час: 2.5 год.

6) VNS Lab 9



Фактично затрачений час: 1.2 год.

7) Class practice task

```
PS D:\Epic 5>  & 'c:\Users\
soft-MIEngine-In-buekyadg.z
-MIEngine-Pid-4tvd1fzb.e3d'
Hello, World!
PS D:\Epic 5>
```

file_from.txt

```
1   Hello, World!
```

file_to.txt

```
1   Hello, World!
2
```

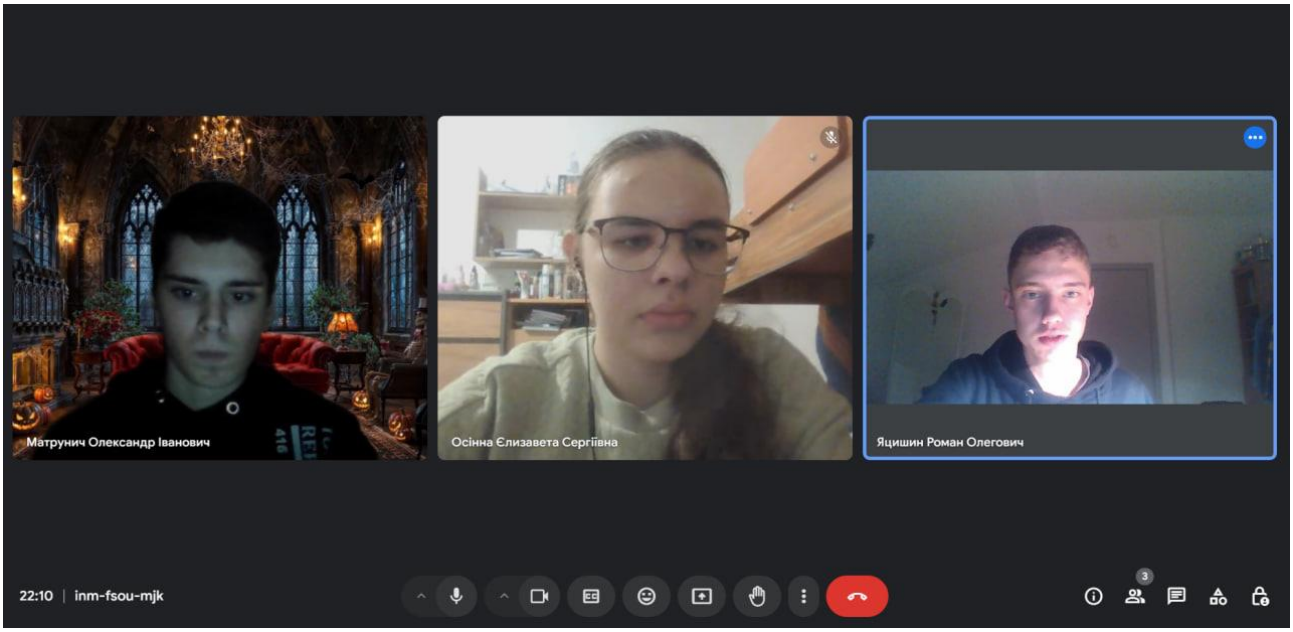
Фактично затрачений час: 90 хв.

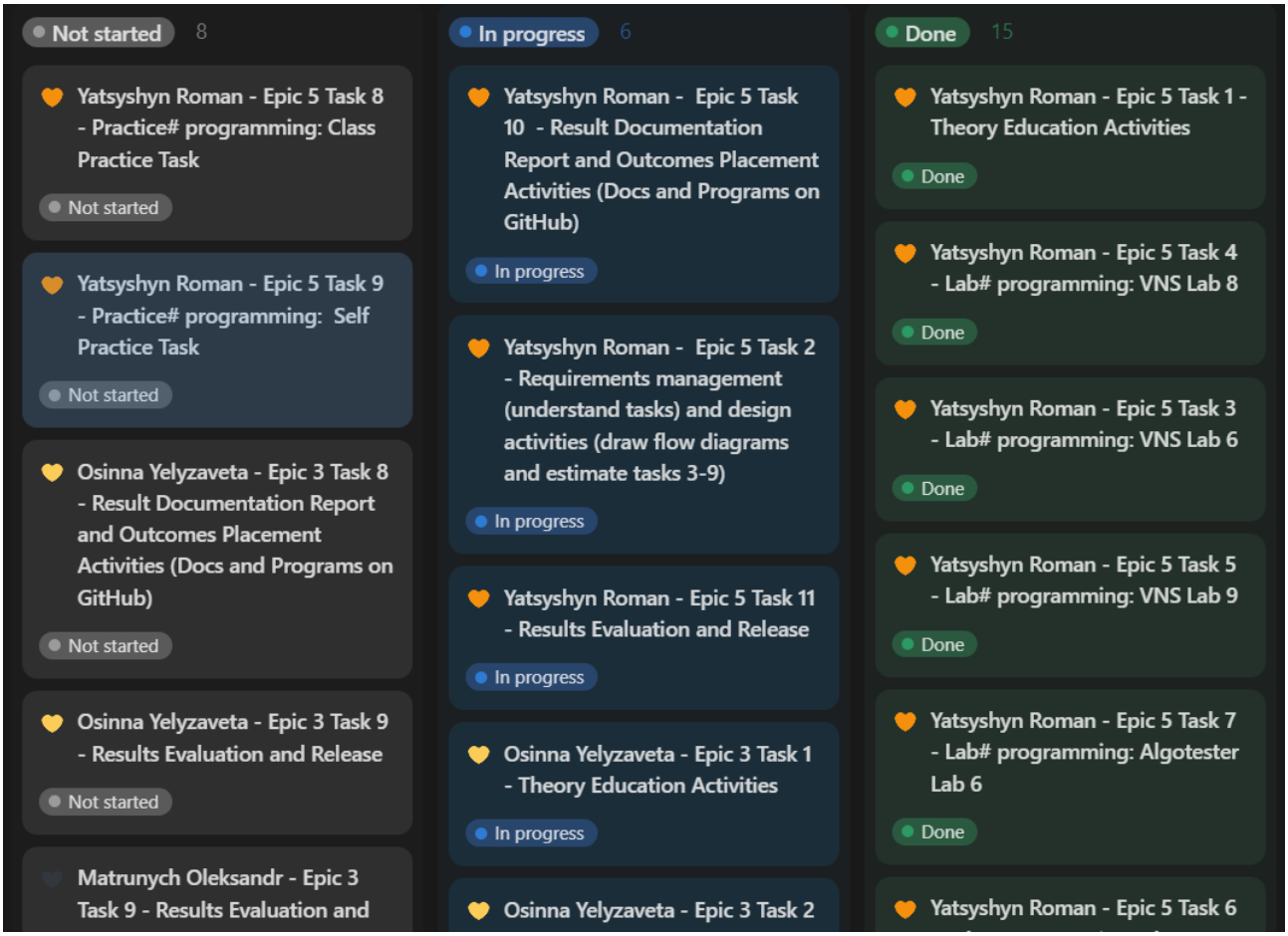
8) Self practice task

```
PS D:\Epic 5>  & 'c:\Users\
soft-MIEngine-In-clnvpnms.j
-MIEngine-Pid-zf5mvwj0.20g'
54656416321351231
2
PS D:\Epic 5>
```

Фактично затрачений час: 10 хв.

6. Кооперація з командою:





Висновок:

У ході лабораторної роботи було здійснено роботу з текстовими та бінарними файлами, опрацьовано символи та рядкові змінні, засвоєно методи читання, запису та модифікації файлів за допомогою стандартної бібліотеки C++ та здійснено виконання завдання з використанням бібліотек та з власною реалізацією.