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 групи ШІ-11

Бубельник Юрій 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, записувати, приєднувати та читати у різних форматах. Використати їх для практичних застосувань

Теоретичні відомості:

1) Вивчив/знав:
    1. Файли та робота з ними
    2. Бінарний та текстовий запис даних у файли
    3. Створення бібліотек
2) Джерела:

    Всю інформацію до теоретичних відомостей я отримав на лекційних/практичних парах. Додатково використовував сайт https://acode.com.ua/ та https://www.youtube.com/@BloganProgramming

Виконання роботи:

1) Опрацювання завдання та вимог до програми та середовища

Завдання №1 Епік 5 - Практичне завдання

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
- 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, запису даних, чи закриття файла.

Завдання №2 внс лаб 6 завдання 1

Завдання:

Надрукувати найдовше й найкоротше слово в цьому рядку.

Завдання №3 внс лаб 8 завдання 1

Завдання:

Структура "Абітурієнт":

- прізвище, ім'я, по батькові;

- рік народження;

- оцінки вступних іспитів (3);

- середній бал атестата.

Знищити елемент із зазначеним номером, додати елемент після елемента із

зазначеним прізвищем.

Завдання №4 внс лаб 9 завдання 1

Завдання:

1) Скопіювати у файл F2 тільки парні рядки з F1.

2) Підрахувати розмір файлів F1 й F2 (у байтах).

Завдання №5 алготестер лаб 4v2

Завдання:

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

[4, 5, 6, 7, 1, 2, 3].
Виведіть результат.

Пам’ятайте, ви маєте написати 2 варіанти розвязку, один з використанням засобів STL (std::unique, std::sort, std::rotate), інший зі своєю реалізацією.

Завдання №6 алготестер лаб 6v2

Завдання:

У вас є шахова дошка розміром 8×88×8 та дуже багато фігур.

Кожна клітинка може мати таке значення:

Пуста клітинка OO

Пішак PP

Тура RR

Кінь NN

Слон BB

Король KK

Королева QQ

Вам дають позиції фігур на дошці (всі фігури одного кольору, кількість королів може бути > 1).

Далі йдуть QQ запитів з координатами клітинки {x,y}{x,y}. На кожен запит ви маєте вивести стрічку sisi - посортовані за алфавітом букви фігур, які атакують цю клітинку (пішаки атакують вниз).

У випадку, якщо на клітинці стоїть якась фігура - виведіть символ XX.

У випадку, якщо клітинку не атакують - виведіть OO.

Наявніть фігури у певній клітинці не блокує атаку для іншої фігури. Тобто якщо між турою та клітинкою стоїть інша фігура - вважається що тура атакує цю клітинку.

Завдання №7 self practice

Завдання:

Додатково попрацювати з файлами((((.

Код програм з посиланням на зовнішні ресурси:

Завдання №1:

[Class Practice Task]

self practice work algotester task 1 yurii bubelnyk.cpp | practice_work_task_1_yurii_bubelnyk.cpp | algotester

C: > Users > Admin > Desktop > EPIC (1-6) > epic 5 > Code > practice_work_task_1_yurii_bubelnyk.cpp > ...

```cpp
#include <iostream>
#include <cstdio>
using namespace std;

enum FileOpResult { Success, Failure };

FileOpResult write_to_file(const char* name, const char* content) {
    FILE* file = fopen(name, "w");

    if (file == nullptr) {
        return Failure;
    }

    if (fputs(content, file) == EOF) {
        fclose(file);
        return Failure;
    }

    fclose(file);
    return Success;
}
FileOpResult copy_file(const char* file_from, const char* file_to) {

    FILE* file1 = fopen(file_from, "r");
    if (file1 == nullptr) {
        return Failure;
    }

    FILE* file2 = fopen(file_to, "w");
    if (file2 == nullptr) {
        fclose(file1);
        return Failure;
    }

    char content[256];

    while (fgets(content, sizeof(content), file1) != nullptr) {
        if (fputs(content, file2) == EOF) {
            fclose(file1);
            fclose(file2);
            return Failure;
        }
    }

    fclose(file1);
    fclose(file2);
    return Success;
}
int main() {
    const char* filename1 = "file_from.txt";
    const char* filename2 = "file_to.txt";

    char content[256];

    cout << "Enter content to write to the file: ";
    cin.getline(content, sizeof(content));

    FileOpResult result = write_to_file(filename1, content);
    if (result == Success) {
        cout << "File successfully written to " << filename1 << endl;
    } else {
        cout << "Failed to write to " << filename1 << endl;
    }

    result = copy_file(filename1, filename2);
    if (result == Success) {
        cout << "File successfully copied to " << filename2 << endl;
    } else {
        cout << "Failed to copy file to " << filename2 << endl;
    }

    return 0;
}
```

Результат виконання завдань

```
PS C:\Users\Admin> cd "c:\Users\Admin\Desktop\EPIC (1-6
Enter content to write to the file: Hello World
File successfully written to file_from.txt
File successfully copied to file_to.txt
```

Завдання №2:

[внс лаб 6 завдання 1]

```c
#include <stdio.h>
#include <string.h>

void Find(char s[]);

int main() {
    char s[256];

    fgets(s, sizeof(s), stdin);  // Зчитуємо рядок

    Find(s);  // Виводимо зчитаний рядок

    return 0;
}

void Find(char s[]){
    char* token = strtok(s, " ");
    char largestWord[256] = "";
    char smallestWord[256] = "";

    int largestLength = 0;
    int smallestLength = strlen(token);

    while( token != NULL){
        int tokenLength = strlen(token);
        printf("%s\n", token);

        if(tokenLength > largestLength){
            largestLength = tokenLength;
            strcpy(largestWord, token);
        }

        if(tokenLength < smallestLength){
            smallestLength = tokenLength;
            strcpy(smallestWord, token);
        }

        token = strtok(NULL, " ");
    }
    printf("%s\n", largestWord);
    printf("%s\n", smallestWord);

}
```

Результат виконання завдань

```
PS C:\Users\Admin> cd "c:\Users\Admin\Desktop\EPIC (1-6
Hello World hi and    bye
Hello
World
hi
and
bye

Hello
hi
```

Завдання №3:

[внс лаб 8 завдання 1]

C: > Users > Admin > Desktop > EPIC (1-6) > epic 5 > Code > vns_lab_8_task_2_variant_1_yurii_bubelnyk.cpp > ...

```cpp
#include <iostream>
#include <cstring>
#include <string>
#include <iomanip>
#include <vector>
using namespace std;

struct Abiturient {
    string firstName;
    string lastName;
    string patronymic;
    int birthYear;
    vector<int> examScores;
    double averageGrade;
};

double calculateAverage(const vector<int>& grades) {
    int sum = 0;
    for (int grade : grades) {
        sum += grade;
    }
    return static_cast<double>(sum) / grades.size();
}
void addStudent(const char* filename) {
    Abiturient student;
    cout << "Enter student's first name: ";
    cin >> student.firstName;
    cout << "Enter student's last name: ";
    cin >> student.lastName;
    cout << "Enter student patronymic: ";
    cin >> student.patronymic;
    cout << "Enter birthYear: ";
    cin >> student.birthYear;
    cout << "Enter grades (separated by space, end with -1): ";
    int grade;
    while (std::cin >> grade && grade != -1) {
        student.examScores.push_back(grade);
    }

    FILE* file = fopen(filename, "a");
    if (file) {
        fprintf(file, "%s %s %s %d", student.firstName.c_str(), student.lastName.c_str(),
                student.patronymic.c_str(), student.birthYear);
        for (int g : student.examScores) {
            fprintf(file, " %d", g);
        }
        fprintf(file, "\n");
        fclose(file);
        cout << "Student added successfully!\n";
    } else {
        cerr << "Error opening file for writing!\n";
    }
}

void searchStudent(const char* filename, const string& lastName) {
    FILE* file = fopen(filename, "r");
    if (file) {
        char firstName[50], lastNameFromFile[50], patronymic[50];
        int birthYear;
        bool found = false;
```

```
            int grade;
            while (fscanf(file, "%d", &grade) == 1) {
                student.examScores.push_back(grade);
            }


            if (student.lastName == lastName) {
                cout << "Found student: " << student.firstName << " " << student.lastName << endl;
                cout << "Average grade: " << fixed << setprecision(2) << calculateAverage(student.examScores) << endl;
                found = true;
                break;
            }
        }
        fclose(file);
        if (!found) {
            cout << "No student found with the last name: " << lastName << endl;
        }
    } else {
        cerr << "Error opening file for reading!\n";
    }
}

void deleteStudentByLastName(const char* filename, const string& lastName) {
    FILE* file = fopen(filename, "r");
    if (!file) {
        cerr << "Error opening file for reading!\n";
        return;
    }

    FILE* tempFile = fopen("temp.txt", "w");
    if (!tempFile) {
        cerr << "Error opening temporary file for writing!\n";
        fclose(file);
        return;
    }

    char firstName[50], lastNameFromFile[50], patronymic[50];
    int birthYear;
    vector<int> grades;
    bool found = false;

    while (fscanf(file, "%s %s %s %d", firstName, lastNameFromFile, patronymic, &birthYear) == 4) {
        grades.clear();
        int grade;
        while (fscanf(file, "%d", &grade) == 1) {
            grades.push_back(grade);
        }

        if (lastName == lastNameFromFile) {
            found = true;
            continue;
        }

        fprintf(tempFile, "%s %s %s %d", firstName, lastNameFromFile, patronymic, birthYear);
        for (int g : grades) {
            fprintf(tempFile, " %d", g);
        }
        fprintf(tempFile, "\n");
    }

    fclose(file);
    fclose(tempFile);
```

```
    if (found) {
        remove(filename);
        rename("temp.txt", filename);
        cout << "Student with last name \"" << lastName << "\" has been deleted.\n";
    } else {
        remove("temp.txt");
        cout << "No student found with the last name \"" << lastName << "\".\n";
    }
}

int main(){

    const char* filename = "Data.txt";
    int choice;
    string lastName;


    do {
        cout << "1. Add student\n2. Search student by last name\n3. Delete Student\n4. Exit\n";
        cin >> choice;

        switch (choice) {
            case 1:
                addStudent(filename);
                break;
            case 2:
                cout << "Enter last name to search: ";
                cin >> lastName;
                searchStudent(filename, lastName);
                break;
            case 3:
                cout << "Enter last name to delete: ";
                cin >> lastName;
                deleteStudentByLastName(filename, lastName);
            case 4:
                cout << "Exiting...\n";
                break;
            default:
                cerr << "Invalid choice!\n";
        }
    } while (choice != 4);


    return 0;
}
```

Результат виконання завдань

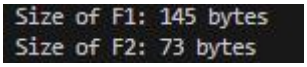
```
PS C:\Users\Admin> cd "c:\Users\Admin\Desktop\EPIC (1-6)\epic 5\
1. Add student
2. Search student by last name
3. Delete Student
4. Exit
1
Enter student's first name: Yura
Enter student's last name: Bubelnyk
Enter student patronymic: Olegovych
Enter birthYear: 2007
Enter grades (separated by space, end with -1): 3 4 5 -1
Student added successfully!
1. Add student
2. Search student by last name
3. Delete Student
4. Exit
2
Enter last name to search: Bubelnyk
Found student: Yura Bubelnyk
Average grade: 4.00
1. Add student
2. Search student by last name
3. Delete Student
4. Exit
```

Завдання №4:

[внс лаб 9 завдання 1]

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

using namespace std;
void CreateFile(const char* filename);
void CreateSecondFile(const char* filename2, const char* filename1);
long getFileSize(const char* filename);
int main(){

    const char* filename1 = "1vns_lab_8.txt";
    const char* filename2 = "2vns_lab_8.txt";

    CreateFile(filename1);
    CreateSecondFile(filename2, filename1);
    long sizeF1 = getFileSize(filename1);
    long sizeF2 = getFileSize(filename2);

    cout << "Size of F1: " << sizeF1 << " bytes\n";
    cout << "Size of F2: " << sizeF2 << " bytes\n";

    return 0;
}

void CreateFile(const char* filename)
{
    FILE* file1 = nullptr;
    file1 = fopen(filename, "w");
    if(file1 == nullptr)
    {
        cout << "Error";
        return;
    }

    const char* strings[10] = {
        "First string",
        "Second string",
        "Third string",
        "Fourth string",
        "Fifth string",
        "Sixth string",
        "Seventh string",
        "Eighth string",
        "Ninth string",
        "Tenth string"
    };

    for (int i = 0; i < 10; i++) {
        fputs(strings[i], file1);
        fputs("\n", file1);
    }

    fclose(file1);
}

void CreateSecondFile(const char* filename2, const char* filename1)
{
    FILE* file1;
    file1 = fopen(filename1, "r");
    if(file1 == nullptr)
    {
        cout << "Error";
        return;
    }

    FILE* file2 = fopen(filename2, "w");
    if (file2 == nullptr) {
        cout << "Error opening file F2 for writing.\n";
        fclose(file1);
        return;
    }
    char line[256];
    int index = 1;
    while(fgets(line, sizeof(line), file1) != nullptr)
    {
        if(index % 2 ==0)
        {
            fputs(line, file2);
        }
        index++;
    }
    fclose(file1);
    fclose(file2);

}

long getFileSize(const char* filename) {
    FILE* file = fopen(filename, "r");
    if (file == nullptr) {
        cout << "Error opening file for size calculation.\n";
        return -1;
    }

    fseek(file, 0, SEEK_END);

    long size = ftell(file);

    fclose(file);

    return size;
}
```

Результат виконання завдань



Завдання №5:

[алготестер лаб 4v2]

Результат виконання завдань

17 hours ago	C++ 23	Accepted	0.003	1.473	View

Завдання №6:

[алготестер лаб 6v2]

Результат виконання завдань

Created	Compiler	Result	Time (sec.)	Memory (MiB)	Actions
an hour ago	C++ 23	Accepted	0.003	1.375	View

Завдання №7:

[Self practice]

C: > Users > Admin > Desktop > EPIC (1-6) > epic 5 > Code > self_practice_work_algotester_task_1_yurii_bubelnyk.cpp > RenumberNotes(const char *)

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

using namespace std;

void AddNote(const char* filename);
void ShowNote(const char* filename);
void DeleteNote(const char* filename, int index);
void RenumberNotes(const char* filename);

int main() {
    const char* filename = "Note.txt";
    int choice, index;
    do {
        cout << "1. Add note\n2. Show note\n3. Delete note\n4. Exit\n";
        cin >> choice;
        cin.ignore();

        switch (choice) {
            case 1:
                AddNote(filename);
                break;
            case 2:
                ShowNote(filename);
                break;
            case 3:
                cout << "Enter index to be deleted: ";
                cin >> index;
                DeleteNote(filename, index);
                break;
            case 4:
                break;
            default:
                cerr << "Invalid choice!" << endl;
        }

    } while (choice != 4);

    return 0;
}

void AddNote(const char* filename) {
    FILE* file = fopen(filename, "a");
    if (!file) {
        cerr << "Failed to open Note\n";
        return;
    }

    string content;
    cout << "Enter text: ";
    getline(cin, content);


    fprintf(file, "%s\n", content.c_str());
    cout << "Text written to the Note\n";
    fclose(file);

```

```
    RenumberNotes(filename);
}

void ShowNote(const char* filename) {
    FILE* file = fopen(filename, "r");
    if (!file) {
        cerr << "Failed to open Note\n";
        return;
    }

    char line[256];
    cout << "Notes:\n";
    while (fgets(line, sizeof(line), file)) {
        cout << line;
    }

    fclose(file);
    RenumberNotes(filename);
}

void DeleteNote(const char* filename, int index) {
    FILE* file = fopen(filename, "r");
    if (!file) {
        cerr << "Error opening file for reading!\n";
        return;
    }

    const char* filename2 = "temp.txt";
    FILE* tempFile = fopen(filename2, "w");
    if (!tempFile) {
        cerr << "Error opening temporary file for writing!\n";
        fclose(file);
        return;
    }

    int currentLine = 1;
    char line[256];

    while (fgets(line, sizeof(line), file)) {
        if (currentLine != index) {
            fputs(line, tempFile);
        }
        currentLine++;
    }

    fclose(file);
    fclose(tempFile);

    remove(filename);
    rename(filename2, filename);

    cout << "Note " << index << " deleted successfully.\n";

    RenumberNotes(filename);
}
```

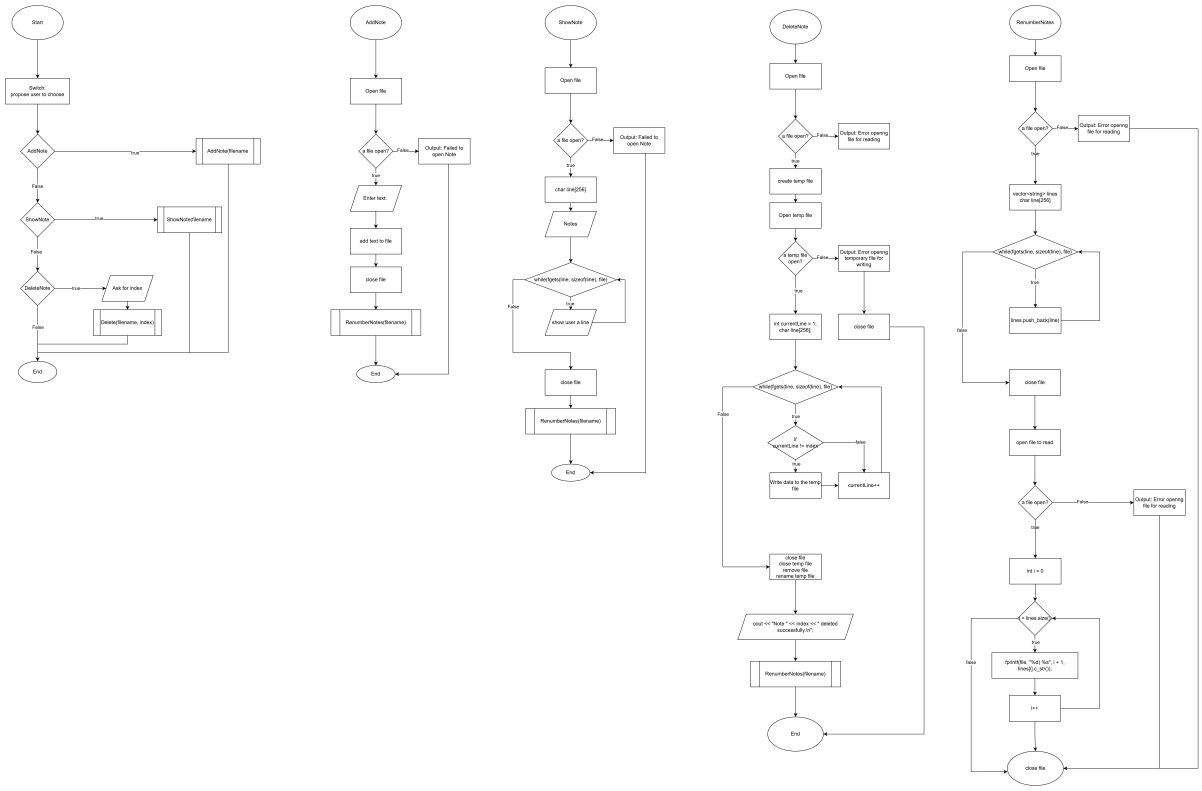
```
void RenumberNotes(const char* filename) {
    FILE* file = fopen(filename, "r");
    if (!file) {
        cerr << "Error opening file for reading!\n";
        return;
    }

    vector<string> lines;
    char line[256];

    while (fgets(line, sizeof(line), file)) {
        lines.push_back(line);
    }
    fclose(file);

    // Перезаписуємо файл з новими номерами
    file = fopen(filename, "w");
    if (!file) {
        cerr << "Error opening file for writing!\n";
        return;
    }
    for (int i = 0; i < lines.size(); i++) {
        fprintf(file, "%d) %s", i + 1, lines[i].c_str());
    }

    fclose(file);
}
```

Результат виконання завдань

```
1. Add note
2. Show note
3. Delete note
4. Exit
2
Notes:
1) Yura
1. Add note
2. Show note
3. Delete note
4. Exit
3
Enter index to be deleted: 1
Note 1 deleted successfully.
1. Add note
2. Show note
3. Delete note
4. Exit
2
Notes:
1. Add note
2. Show note
3. Delete note
4. Exit
```

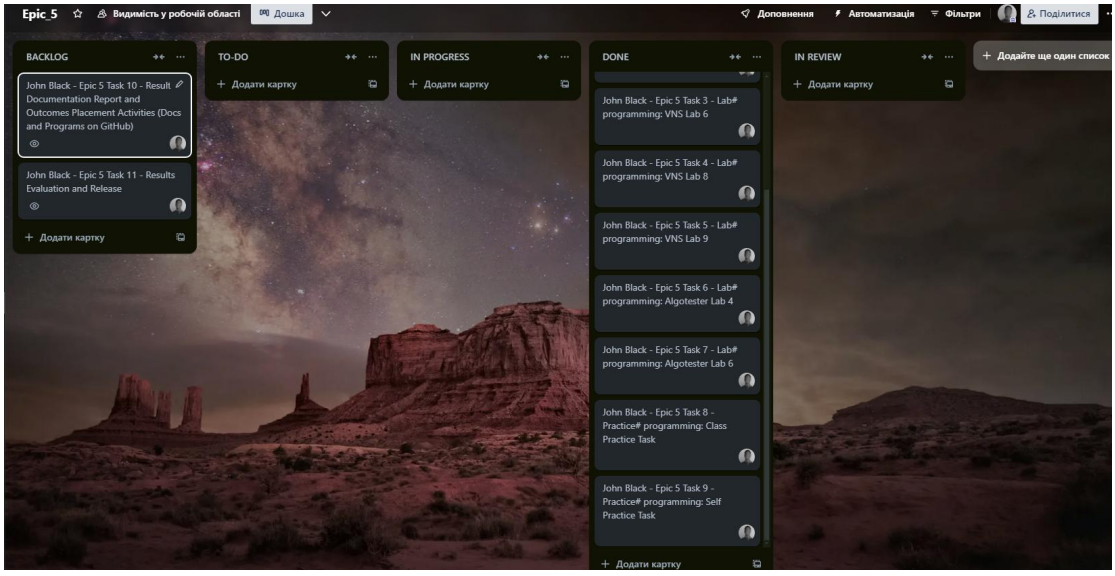
Дизайн

Self practice



Робота з командою:

Налаштували Trello для Epic 5:



Висновки:

Отже, я старався зрозуміти що як працювати з файлами, створював та добавляв вміст за допомого команд C/C++, також зрозумів для чого розширення у файлах та нащо вони впливають.

Посилання на pull request