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 групи ШІ-12
Климишин Данило

Львів 2024

Тема роботи: Файлова система в C++. Робота з бінарними файлами та текстовими файлами, маніпуляції символами й рядковими змінними, як типу std::string, так і char*. Ознайомлення з можливостями стандартної бібліотеки C++ для роботи з файлами та створенням власних бібліотек для розширення функціональності.

Мета роботи: Опанувати практичні навички роботи з файлами в мові C++: створення, зчитування та запис даних у бінарні й текстові файли. Засвоїти принципи роботи з рядковими змінними різних типів (std::string і char*), вивчити використання стандартних методів та функцій для маніпуляцій з ними.

Джерела:

https://cplusplus.com/reference/cstdio

https://www.programiz.com/dsa/merge-sort

https://chatgpt.com

Виконання роботи:

TASK 2:

Requirements management (understand tasks) and design activities (draw flow diagrams and estimate tasks 3-9)

На виконання пішло 1.5 години.

TASK 3:

Lab# programming: VNS Lab 6

На виконання пішло 20 хв

```cpp
#include <stdio.h>
#include <iostream>
#include <string.h>


int findNumber(char *s){
    int number = 0;
    gets(s);
    char *temp;
    temp = strtok(s, " ");
    if(temp[0] == 'a')
        ++number;
    while(temp!=nullptr){
        temp = strtok(nullptr, " ");
        if(temp != nullptr && temp[0] == 'a')
            ++number;
    }
    return number;
}




int main(){


char sentence[255];
std::cout << "Enter your sentence: ";
int result = findNumber(sentence);

std::cout << "Your sentence contains " << result << " words that start with 'a'";


    return 0;
}
```

```
Enter your sentence: a rainy day in august
Your sentence contains 2 words that start with 'a'
```

Task 4:

Lab# programming: VNS Lab 8

На виконання пішло 3 години

```cpp
#include <iostream>
#include <fstream>
#include <cstdio>
#include <vector>
#include <cstring>
#include <limits>

struct CarOwner{

    char FirstNameSurname[50];
    int autoNumber;
    int licenseNumber;


};


void fillFile(int n){
    FILE* file = fopen("data.bin", "wb");
    if (!file) {
        std::cerr << "Error" << std::endl;
        return ;
    }


    CarOwner temp;
    bool check = false;
    for (int i = 0; i < n; ++i) {
        std::cout << i + 1<< " person:\n";
        std::cout << "Enter full name: ";
        std::cin.ignore();
        fgets(temp.FirstNameSurname, 50, stdin);

        size_t len = strlen(temp.FirstNameSurname);
        if (len > 0 && temp.FirstNameSurname[len - 1] == '\n') {
            temp.FirstNameSurname[len - 1] = '\0';
        }

        std::cout << "Enter auto number: ";
```

```
        std::cin >> temp.autoNumber;
        std::cout << "Enter license number: ";
        std::cin >> temp.licenseNumber;
        auto written = fwrite(&temp, sizeof(CarOwner), 1, file);
        if(written != 1)
            check = true;
    }
    if(check)
        std::cerr << "Error" << std::endl;
    else
        std::cout << "Your data has been saved" << std::endl;
    fclose(file);
    }




void readFile(){

    FILE* file = fopen("data.bin", "rb");
    if (!file) {
        std::cerr << "Error" << std::endl;
        return ;
    }

    std::cout << "\tYour current data: \n";
    CarOwner temp;
    while (fread(&temp, sizeof(CarOwner), 1, file) == 1){
        fputs(temp.FirstNameSurname, stdout);
        std::cout << " ";
        std::cout << temp.autoNumber << " ";
        std::cout << temp.licenseNumber << " \n";

    }
    fclose(file);
```

```
void deleteNumber(int n){

FILE *file_read = fopen("data.bin", "rb");
if(!file_read){
    std::cerr << "Error openning file.";
    return ;
}

CarOwner temp;

CarOwner *array = new CarOwner[n];

int i = 0;
while(fread(&temp, sizeof(CarOwner), 1, file_read) == 1){
    array[i] = temp;
    ++i;
}

fclose(file_read);

FILE *file_delete = fopen("data.bin", "wb");

if(!file_delete){
    std::cerr << "Error openning file.";
    delete[] array;
    return ;
}


int delete_number;
std::cout << "Enter auto number of person you want to delete: ";
std::cin >> delete_number;

bool check = false;
for(int i = 0; i < n; ++i){
    if(array[i].autoNumber == delete_number)
        continue;
    auto written = fwrite(&array[i], sizeof(CarOwner), 1, file_delete);
    if( written != 1)
```

```
        check = true;
}
    if(!check)
        std::cout << "Data has benn successfully deleted." << std::endl;
    else{
        std::cerr << "Error deleting data. " << std::endl;
        delete[] array;
        return ;
    }

fclose(file_delete);
delete[] array;

readFile();

}



void addTwoEl(int n){
    FILE *read_file = fopen("data.bin", "rb");

    struct CarOwner *temp_arr = new CarOwner[n + 2];

if(!read_file){
    std::cerr << "Error openning file.";
    delete[] temp_arr;
    return ;
}
    char surn[50];
    std::cout << "Enter surname: ";
    std::cin.ignore();
    fgets(surn, 50, stdin);
    int len = strlen(surn);

    if (len > 0 && surn[len - 1] == '\n') {
        surn[len - 1] = '\0';
```

```
    surn[len - 1] = '\0';
}

const char delimiters[] = " ,.";

CarOwner temp;
int i = 0;
while (fread(&temp, sizeof(CarOwner), 1, read_file) == 1){
    temp_arr[i] = temp;
    ++i;
}

fclose(read_file);

for (int i = 0; i < n + 1; i++) {
    if (strcmp(temp_arr[i].FirstNameSurname, surn) == 0) {

        for (int j = n ; j > i ; --j) {
            temp_arr[j+1] = temp_arr[j-1];
        }

        std::cout << "Enter first new full name: ";
        std::cin.ignore();
        char x[30];
        std::cin.getline(x, sizeof(char[30]));
        len = strlen(temp_arr[i].FirstNameSurname);
        if (len > 0 && temp_arr[i].FirstNameSurname[len - 1] == '\n') {
            temp_arr[i].FirstNameSurname[len - 1] = '\0';
        }
        std::cout << "Enter auto number: ";
        std::cin >> temp_arr[i].autoNumber;
        std::cout << "Enter license number: ";
        std::cin >> temp_arr[i].licenseNumber;

        std::cout << "Enter second new full name: ";
        std::cin.ignore();
        std::cin.getline(temp_arr[i+1].FirstNameSurname, sizeof(temp));
```

```
        len = strlen(temp_arr[i+1].FirstNameSurname);
        if (len > 0 && temp_arr[i+1].FirstNameSurname[len - 1] == '\n') {
            temp_arr[i+1].FirstNameSurname[len - 1] = '\0';
        }
        std::cout << "Enter auto number: ";
        std::cin >> temp_arr[i+1].autoNumber;
        std::cout << "Enter license number: ";
        std::cin >> temp_arr[i+1].licenseNumber;
        break;
    }
}


for(int i = 0; i < n + 1; ++i){
    if(temp_arr[i].autoNumber == 0)
        break;
    fputs(temp_arr[i].FirstNameSurname, stdout);
    std::cout << temp_arr[i].autoNumber << " ";
    std::cout << temp_arr[i].licenseNumber << " \n";
}

FILE *write_file = fopen("data.bin", "wb");

bool check = false;
for(int i = 0; i < n + 1; ++i){
    auto written = fwrite(&temp_arr[i], sizeof(CarOwner), 1, write_file);
    if(written == 0)
        check = true;
}
if(check){
    std::cerr << "Error during changing data." << std::endl;
    delete[] temp_arr;
    return ;
}
else
    std::cout << "Data has been changed." << std::endl;
fclose(write_file);
```

```
        return ;
    }
    else
        std::cout << "Data has been changed." << std::endl;
    fclose(write_file);
    delete[] temp_arr;

    readFile();

}




int main(){

int n;
std::cout << "How much elements you want to enter: ";
std::cin >> n;

fillFile(n);

deleteNumber(n);

addTwoEl(n);



    return 0;
}
```

```
How much elements you want to enter: 3
1 person:
Enter full name: Danylo
Enter auto number: 1
Enter license number: 2
2 person:
Enter full name: Ivan
Enter auto number: 3
Enter license number: 4
3 person:
Enter full name: Peter
Enter auto number: 5
Enter license number: 6
Your data has been saved
Enter auto number of person you want to delete: 5
Data has benn successfully deleted.
        Your current data:
Danylo 1 2
Ivan 3 4
Enter surname: Ivan
Enter first new full name: Olha
Enter auto number: 12
Enter license number: 13
Enter second new full name: George
Enter auto number: 16
Enter license number: 17
Data has been changed.
        Your current data:
Danylo 1 2
Olha 12 13
George 16 17
Ivan 3 4
```

Task5:

Lab# programming: VNS Lab 9

На виконання потратив годину

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

void fileFill(){
FILE *F1 = fopen("F1.txt", "w");
if(!F1){
    std::cerr << "Error openning file. ";
    return ;
}

char temp[100];
for(int i = 0; i < 10; ++i){
    std::cout << "Enter " << i+1 << " line: ";
    std::cin.getline(temp, sizeof(temp));
    if (fputs(temp, F1) == EOF || fputc('\n', F1) == EOF) {
        std::cerr << "Error saving data.\n";
        fclose(F1);
        return;
}
}

fclose(F1);
}
```

```cpp
void copy(){
FILE *F1 = fopen("F1.txt", "r");
FILE *F2 = fopen("F2.txt", "w");
if(!F1 || !F2){
    std::cerr << "Error openning files.";
    return ;
}

char temp[100];
const char delim[] = " .,!?";
for(int i = 0; i < 10; ++i){
    fgets(temp, sizeof(temp), F1);
    char *token = strtok(temp, delim);
    if(token != nullptr && strtok(nullptr, delim) == nullptr){
```

```
        char *token = strtok(temp, delim);
        if(token != nullptr && strtok(nullptr, delim) == nullptr){
            fputs(temp, F2);
            fputc('\n', F2);
        }
    }

    fclose(F1);
    fclose(F2);
}

int longestWord(){

FILE *F2 = fopen("F2.txt", "r");

char temp[100];

fgets(temp, sizeof(temp), F2);
int longest = strlen(temp);
int k = 0;
for(int i = 1; i < 10; ++i){
    fgets(temp, sizeof(temp), F2);
    if(strlen(temp) > longest){
        longest = strlen(temp);
        k = i;
    }
}
fclose(F2);
return k + 1;
}

int main(){

fileFill();
copy();
int result = longestWord();
std::cout  << "Longest word index: " <<result;
    return 0;
    }
```

```
Enter 1 line: i love cats
Enter 2 line: sun
Enter 3 line: flower
Enter 4 line: end
Longest word index: 3
```

TASK 6:

Lab# programming: Algotester Lab 4

На виконання потратив 2 години

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

void merge(std::vector<int> &v, int left, int mid, int right){

    int n1 = mid-left + 1;
    int n2 = right - mid;

    std::vector<int> L(n1), R(n2);

    for(int i = 0; i < n1; ++i)
        L[i] = v[left+i];
    for(int j = 0; j < n2; ++j)
        R[j] = v[mid+1+j];

    int i = 0, j = 0, k = left;
    while(i < n1 && j < n2){
        if(L[i] <= R[j]){
            v[k] = L[i];
            ++i;
        }
        else {
            v[k] = R[j];
            ++j;
        }
        ++k;
    }


    while(i < n1){
        v[k] = L[i];
        ++i;
        ++k;
    }
    while(j < n2){
        v[k] = R[j];
        ++j;
        ++k;
    }
}
```

```cpp
void mergesort(std::vector<int> &v, int left, int right){
    if(left >= right)
        return ;

    int mid = left + (right - left)/2;

    mergesort(v, left, mid);
    mergesort(v, mid+1, right);
    merge(v, left, mid, right);

}



void merge_rev(std::vector<int> &v, int left, int mid, int right){
    int n1 = mid-left + 1;
    int n2 = right - mid;

    std::vector<int> L(n1), R(n2);

    for(int i = 0; i < n1; ++i)
        L[i] = v[left+i];
    for(int j = 0; j < n2; ++j)
        R[j] = v[mid+1+j];

    int i = 0, j = 0, k = left;
    while(i < n1 && j < n2){
        if(L[i] >= R[j]){
            v[k] = L[i];
            ++i;
        }
        else {
            v[k] = R[j];
            ++j;
        }
        ++k;
    }


    while(i < n1){
        v[k] = L[i];
```

```
    while(i < n1){
        v[k] = L[i];
        ++i;
        ++k;
    }
    while(j < n2){
        v[k] = R[j];
        ++j;
        ++k;
    }

}

void mergesort_rev(std::vector<int> &v, int left, int right){
    if(left >= right)
        return ;

    int mid = left + (right - left)/2;

    mergesort_rev(v, left, mid);
    mergesort_rev(v, mid+1, right);
    merge_rev(v, left, mid, right);

}

int main(){

int N;
std::cin >> N;

std::vector<int> array;
std::vector<int> O1;
std::vector<int> O2;
std::vector<int> O3;

int temp;
for(int i = 0; i < N; ++i){
    std::cin >> temp;
    if((temp % 3) == 0)
```

```cpp
std::vector<int> array;
std::vector<int> O1;
std::vector<int> O2;
std::vector<int> O3;

int temp;
for(int i = 0; i < N; ++i){
    std::cin >> temp;
    if((temp % 3) == 0)
        O1.push_back(temp);
    else if ((temp % 3) == 1)
        O2.push_back(temp);
    else
        O3.push_back(temp);
}

mergesort(O1, 0, O1.size()-1);
mergesort(O3, 0, O3.size()-1);
mergesort_rev(O2, 0, O2.size()-1);


for(int a = 0; a < O1.size(); ++a)
    array.push_back(O1[a]);
for(int b = 0; b < O2.size(); ++b)
    array.push_back(O2[b]);
for(int c = 0; c < O3.size(); ++c)
    array.push_back(O3[c]);

for(int i = array.size()-1; i > 0; --i){
    if(array[i] == array[i-1]){
        array.erase(array.begin()+ i);
    }
}

std::cout << array.size() << std::endl;
for(auto el:array)
    std::cout << el << ' ';


    return 0;
}
```

```
7
2 7 8 9 10 2 1
6
9 10 7 1 2 8
```

2 варіант:

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

int main(){

int N;
std::cin >> N;

std::vector<int> array;
std::vector<int> O1;
std::vector<int> O2;
std::vector<int> O3;

int temp;
for(int i = 0; i < N; ++i){
    std::cin >> temp;
    if((temp % 3) == 0)
        O1.push_back(temp);
    else if ((temp % 3) == 1)
        O2.push_back(temp);
    else
        O3.push_back(temp);
}

std::sort(O1.begin(), O1.end());
std::sort(O3.begin(), O3.end());
std::sort(O2.begin(), O2.end(), std::greater{});

for(int a = 0; a < O1.size(); ++a)
    array.push_back(O1[a]);
for(int b = 0; b < O2.size(); ++b)
    array.push_back(O2[b]);
for(int c = 0; c < O3.size(); ++c)
    array.push_back(O3[c]);
array.erase(std::unique(array.begin(), array.end()), array.end());

std::cout << array.size() << std::endl;
for(auto el:array)
    std::cout << el << ' ';
    return 0;
}
```

```
7
1 8 7 2 9 3 4
7
3 9 7 4 1 2 8
```

Task 7:

Lab# programming: Algotester Lab 6

На виконання пішло 5 годин

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


enum figures{
    O,
    P,
    R,
    N,
    B,
    K,
    Q
};

void fillArray(figures array[8][8]){
    char temp;
    for(int i = 0; i < 8; ++i)
    for(int j = 0; j < 8; ++j){
        std::cin >> temp;
        switch(temp){
            case 'O': array[i][j] = figures::O; break;
            case 'P': array[i][j] = figures::P; break;
            case 'R': array[i][j] = figures::R; break;
            case 'N': array[i][j] = figures::N; break;
            case 'B': array[i][j] = figures::B; break;
            case 'K': array[i][j] = figures::K; break;
            case 'Q': array[i][j] = figures::Q; break;
            }
    }
}


bool pishak(figures array[8][8], int x, int y) {

    if (x - 1 >= 0 && y - 1 >= 0 && array[x - 1][y - 1] == figures::P){
        return true;
    }
    if (x - 1 >= 0 && y + 1 < 8 && array[x - 1][y + 1] == figures::P){
        return true;
        }
    return false;
}
```

```
bool tura(figures array[8][8], int x, int y){
    for(int j = 0; j < 8; ++j){
        if(array[x][j] == figures::R){
            return true;

        }
    }
    for(int i = 0 ; i < 8; ++i){
        if(array[i][y] == figures::R)
            return true;
    }
return false;
}

bool horse(figures array[8][8], int x, int y) {
    if (x + 2 < 8 && y + 1 < 8 && array[x+2][y+1] == figures::N)
        return true;
    if (x + 2 < 8 && y - 1 >= 0 && array[x+2][y-1] == figures::N)
        return true;
    if (x - 2 >= 0 && y + 1 < 8 && array[x-2][y+1] == figures::N)
        return true;
    if (x - 2 >= 0 && y - 1 >= 0 && array[x-2][y-1] == figures::N)
        return true;
    if (x - 1 >= 0 && y + 2 < 8 && array[x-1][y+2] == figures::N)
        return true;
    if (x - 1 >= 0 && y - 2 >= 0 && array[x-1][y-2] == figures::N)
        return true;
    if (x + 1 < 8 && y + 2 < 8 && array[x+1][y+2] == figures::N)
        return true;
    if (x + 1 < 8 && y - 2 >= 0 && array[x+1][y-2] == figures::N)
        return true;
    return false;
}
```

```
bool king(figures array[8][8], int x, int y) {
    if (x + 1 < 8 && y - 1 >= 0 && array[x+1][y-1] == figures::K) {
        return true;
    }
    if (x + 1 < 8 && y < 8 && array[x+1][y] == figures::K) {
        return true;
    }
    if (x + 1 < 8 && y + 1 < 8 && array[x+1][y+1] == figures::K) {
        return true;
    }
    if (x - 1 >= 0 && y + 1 < 8 && array[x-1][y+1] == figures::K) {
        return true;
    }
    if (x - 1 >= 0 && y < 8 && array[x-1][y] == figures::K) {
        return true;
    }
    if (x - 1 >= 0 && y - 1 >= 0 && array[x-1][y-1] == figures::K) {
        return true;
    }
    if (x < 8 && y + 1 < 8 && array[x][y+1] == figures::K) {
        return true;
    }
    if (x >= 0 && y - 1 >= 0 && array[x][y-1] == figures::K) {
        return true;
    }
    return false;
}
```

```
bool queen(figures array[8][8], int x, int y){
    for (int n = 1; n < 8; ++n) {
        if (x - n >= 0 && y - n >= 0 && array[x-n][y-n] == figures::Q)
            return true;
        if (x - n >= 0 && y + n < 8 && array[x-n][y+n] == figures::Q)
            return true;
        if (x + n < 8 && y - n >= 0 && array[x+n][y-n] == figures::Q)
            return true;
        if (x + n < 8 && y + n < 8 && array[x+n][y+n] == figures::Q)
            return true;
    }
    for(int j = 0; j < 8; ++j){
        if(array[x][j] == figures::Q){
            return true;

        }
    }
    for(int i = 0 ; i < 8; ++i){
        if(array[i][y] == figures::Q)
            return true;
    }
    return false;
}
```

```cpp
void func(figures array[8][8]){
    int Q, x, y;
    bool check;
    std::cin >> Q;
    std::vector<std::pair<int, int>> coord(Q);
    std::vector<char> results;
    for(int i = 0; i < Q; ++i){
        check = true;
    std::cout << '\n';
        std::cin >> coord[i].first;
        std::cin >> coord[i].second;

        coord[i].first -=1;
        coord[i].second -=1;

            if(array[coord[i].first][coord[i].second] != figures::0){
                results.push_back('X');
                check = false;
            }
            else{

            if(elephant(array, coord[i].first, coord[i].second)){
                results.push_back('B');
                check = false;
            }

            if(king(array, coord[i].first, coord[i].second)){
                results.push_back('K');
                check = false;
            }

            if(horse(array, coord[i].first, coord[i].second)){
                results.push_back('N');
                check = false;
            }

            if(pishak(array, coord[i].first, coord[i].second)){
            results.push_back('P');
            check = false;
            }
```

```
            if(queen(array, coord[i].first, coord[i].second)){
                results.push_back('Q');
                check = false;
            }

            if(tura(array, coord[i].first, coord[i].second)){
                results.push_back('R');
                check = false;
            }
            if(check)
                results.push_back('0');
        }
    for(auto &el : results){
            std::cout << el;
    }
    results.clear();
    }

}

int main(){

figures array[8][8];
fillArray(array);

func(array);

    return 0;
}
```

```
K0000000
00000000
00000000
00000000
00000000
00000000
00000000
00000000

5

1 1
X
1 2
K
2 1
K
2 2
K
3 1
0
```

Task 8 :

Practice# programming: Class Practice Task

На виконання потратив 1.5 години

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

enum FileOpResult{
    Success,
    Failure,
};

FileOpResult write_to_file(char *name, char *content){

    FILE *file = fopen(name, "w");
    if (!file) {
        std::cerr << "Error" << std::endl;
        return Failure;
    }

    auto written = fwrite(content, sizeof(char), strlen(content), file);
    if (written != strlen(content))
    return Failure;

    auto close = fclose(file);
    if(close != 0)
        return Failure;
    return Success;
}


FileOpResult copy_file(char *file_from, char *file_to){

    FILE *file1 = fopen(file_from, "r");
    if(!file1){
        std::cerr << "Error occured during openning \"file_from \"";
        return Failure;
    }

    FILE *file2 = fopen(file_to, "w");
    if(!file2){
        std::cerr << "Error occurred during openning \"file_to \"";
        return Failure;
    }
```

```
    }

    FILE *file2 = fopen(file_to, "w");
    if(!file2){
        std::cerr << "Error occurred during openning \"file_to \"";
        return Failure;
    }
    char temp[30];
    while(fread(temp, sizeof(char), 1, file1) == 1){
        auto written = fwrite(temp, sizeof(char), 1, file2);
        if(written != 1){
            std::cerr << "Error occured during copying data. ";
            return Failure;
        }
    }

    auto close1 = fclose(file1);
    if(close1 != 0){
        std::cerr << "Error occured during cloing \"file_from\"";
        return Failure;
    }
    auto close2 = fclose(file2);
    if(close2 != 0){
        std::cerr << "Error occured during cloing \"file_to\"";
        return Failure;
    }

    return Success;

}
```

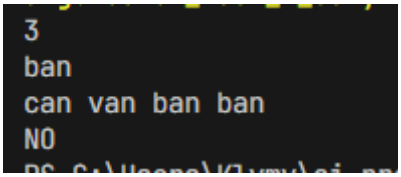
```cpp
int main(){

    char content[30], name[15];
    std::cout << "Enter the name of your file: ";
    fgets(name, 15, stdin);
    name[strcspn(name, "\n")] = '\0';

    std::cout << "Enter content for your file: ";
    std::cin.ignore();
    fgets(content, 30, stdin);
    content[strcspn(content, "\n")] = '\0';

    FileOpResult f1 = write_to_file(name, content);
    if(f1 == Success){
        std::cout << "Your data has benn successfuly saved. ";
    }
    else
        std::cout << "An error occured during saving your data. ";



    char name2[20];
    std::cout << "\nEnter the name of second file: ";
    fgets(name2, 20, stdin);
    name2[strcspn(name2, "\n")] = '\0';
    FileOpResult f2 = copy_file(name, name2);


    if(f2 == Success)
        std::cout << "Your data has been successfuly copied! ";

    return 0;
}
```

```
Enter the name of your file: garden.txt
Enter content for your file: apple, orange
Your data has benn successfuly saved.
Enter the name of second file: orchard.txt
Your data has been successfuly copied!
```

Task 9:

Epic 5 Task 9 - Practice# programming: Self Practice Task

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

int main() {
    int n;
    char x[11];
    scanf("%d", &n);
    scanf("%s", x);

    int count = 0;

    for (int i = 0; i < n; ++i) {
        char statement[11];
        scanf("%s", statement);
        if (strcmp(statement, x) == 0) {
            ++count;
        }
    }

    if (count == n) {
        printf("YES\n");
    } else {
        printf("NO\n");
    }

    return 0;
}
```



Робота з командою:
Обговорили проблематичні питання

Висновок: У ході роботи було вивчено основи роботи з файловою системою в C++: опрацьовано принципи обробки текстових і бінарних файлів, включаючи процеси запису, зчитування й редагування даних. Завдяки використанню різних типів рядкових змінних (std::string та char*) вдалося ознайомитися з різними підходами до зберігання й обробки текстових даних. Використання стандартної бібліотеки значно спростило роботу з файлами, дозволяючи зосередитися на вирішенні основних завдань.