Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Лабораторних та практичних робіт №5**

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

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

***Виконав:***

студент групи ШІ-14

Чумаченко Дем’ян Сергійович

# **Тема роботи:**

Бінарні та текстові файли. Робота з файлами у С++. Символи та рядкові змінні. Зчитування з файлу та запис у файл. Створення й використання бібліотек

# **Мета роботи:**

Ознайомитися з файлами та з деталями роботи з ними у С++. Опрацювати зчитування та запис у файл. Ознайомитися з символами та рядковими змінними

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Файли. Бінарні та текстові файли. Використання файлів. Бібліотеки.
* Тема №2: Робота з файлами у С++.
* Тема №3: Символи та рядкові змінні.
* Тема №4: Зчитування та запис у файл.

1. Індивідуальний план опрацювання теорії:

* Тема №1: Файли. Бінарні та текстові файли. Використання файлів.
  + Джерела Інформації:
    - Стаття.

<https://acode.com.ua/statychni-i-dynamichni-biblioteky/>

<https://www.bestprog.net/uk/2020/05/14/c-examples-of-working-with-text-files-modification-of-files-sort-data-in-files-convert-file-data-to-list-ua/#q02>

<http://cpp.dp.ua/osnovni-funktsiyi-obrobky-symvolnyh-typiv/>

<https://acode.com.ua/urok-82-ryadky-c-style/>

<https://www.bestprog.net/uk/2019/09/11/examples-of-using-c-tools-for-working-with-files-ua/>

<https://www.w3schools.com/cpp/cpp_files.asp>

* + Що опрацьовано:
    - Бінарні та текстові файли
    - Робота з файлами за допомогою fstream
    - Відкриття файлів у різних режимах
    - Рядки та символи
    - Основні функції обробки символьних типів та рядків
    - Стандартна бібліотека С++
    - функція getline
  + Статус: Частково ознайомлений
  + Початок опрацювання теми: 25.11.2023
  + Звершення опрацювання теми: 11.12.2023

# **Виконання роботи:**

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

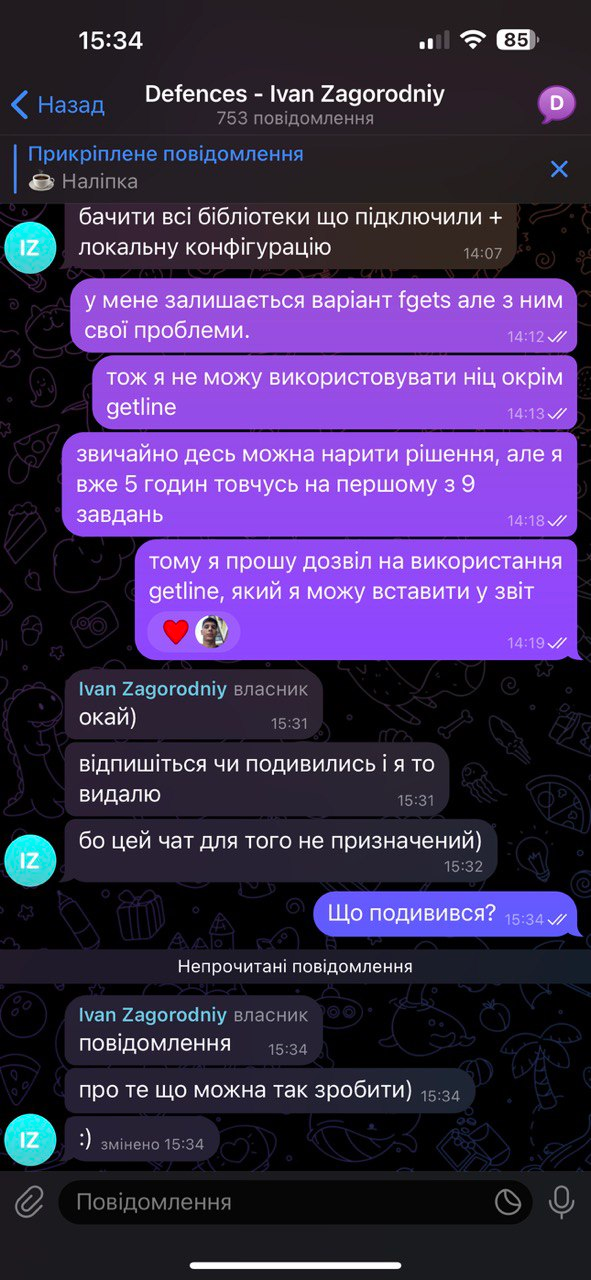
Завдання №1 VNS Lab 6 - Task 1

* Варіант 15
* Деталі завдання:

Визначити яке слово зустрічається в рядку найчастіше.

* Важливі деталі для врахування в імплементації програми

Використовував функцію strtok для розбиття рядка на токени.



*Додаткове Зображення дозвіл на getline*

Завдання №2 VNS Lab 8 - Task 1

* Варіант 15
* Деталі завдання

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.

Структура "Автомобіль":

- марка;

- рік випуску;

- ціна;

- кольори.

Знищити всі елементи, у яких рік випуску меншее заданого, додати елемент на

початок файлу.

* Важливі деталі для врахування в імплементації програми

Використання функцій блокового вводу/виводу read/write, для запису і зчитування структури.

Завдання №3 VNS Lab 9 - Task 1

* Варіант 15
* Деталі завдання

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

інформацію

1) Скопіювати з файлу F1 у файл F2 всі рядки, що закінчуються на букву «А» і

розташовані між рядками з номерами N1 й N2.

2) Визначити номер того рядка, у якій найбільше букв «А», файлу F2.

* Важливі деталі для врахування в імплементації програми

Використав стріми

Завдання №4 Algotester Lab 4

* Варіант 3
* Деталі завдання

Вам дано масив, який складається з N додатнiх цiлих чисел.

Ваше завдання - роздiлити його на три частини, по остачi вiд дiлення на 3, по зростанню

остачi (тобто спочатку йдуть числа, у яких остача 0, далi числа з остачею 1 i тодi нарештi числа

з остачею 2).

Далi необхiдно тi елементи, остача вiд дiлення на 3 яких парна посортувати по зростанню, а

тi, у яких остача 1 - по спаданню.

Пiсля цього видалiть усi дублiкати з масиву.

Виведiть результуючий масив.

* Важливі деталі для врахування в імплементації програми

Спочатку розв’язав без засобів stl, використовуючи двовимірний масив, а потім розв’язав іншим способ із використанням засобів stl;

Завдання №5 Algotester Lab 6

* Варіант 3
* Деталі завдання

У Клiнта в черговий раз виключилось свiтло i йому немає чим зайнятися. Так як навiть це не

заставить його подивитися збереженi вiдео про програмування на ютубi - вiн вирiшив придумати

свою гру на основi судоку.

Гра виглядає так:

Є поле розмiром N × N, в якому частина клiтинок заповнена цифрами, а частина клiтинок

пустi (позначаються нулем). Також у нього є Q пар координат X та Y .

Завданням гри є написати до кожної координати скiльки чисел туди можна вписати (якщо

вона пуста) i якi це числа (обов’язково в посортованi по зростанню!). В клiтинку можна вписати

лише тi числа, якi не зустрiчаються в рядку та стовбцi, якi перетинаються у цiй клiтинцi.

Пiд час гри поле не мiняється!

Також необовязково, щоб це було валiдне судоку! Якщо є клiтинка, в яку не можна вписати

нiяку цифру - виведiть 0.

Також допускаються рядки та стовпцi, в яких цифра записана кiлька разiв.

* Важливі деталі для врахування в імплементації програми

Використав структуру і вектор структур.

Завдання №6 Class Practice Work

* Без варіанту
* Деталі завдання

1)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.

2)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.

* Важливі деталі для врахування в імплементації програми

В першому завданні відкриваю файл зі знищенням попереднього вмісту та запису нового. В одному із завдань перевіряю існування файлу за введеною назвою

Завдання №7 Self Practice Work

* Задача з алготестеру - більше блоків
* Деталі завдання

Петрик любить грати в ігри, де світ складається з кубів. Сьогодні він вирішив написати свій варіант такої гри.

Петрик хоче, щоб у цій грі можна було програмувати нескладні механізми на основі електричного сигналу. Вважається, що сигнал має певну додатну силу, яка позначається цілим числом. Також, сигнал може бути відсутнім, що позначається як сигнал силою 0. Петрик додав в гру три блоки, які по-різному обробляють електричний сигнал:

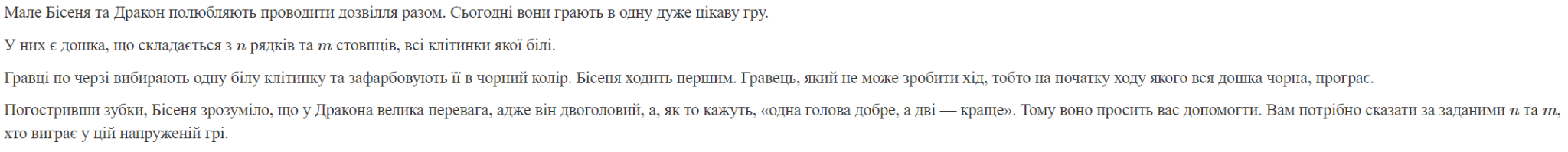
* **Кабель** отримує сигнал силою k і видає сигнал силою max(k−1,0).
* **Повторювач** підсилює сигнал до рівня 15 при ненульовому вхідному сигналі, і не видає сигнал при його відсутності. Формально, нехай сила вхідного сигналу k. Тоді повторювач видає сигнал силою 15, якщо k≥1, і сигнал силою 0, якщо k=0.
* **Змінювач** не видає сигнал при ненульовому вхідному сигналі та видає сигнал силою 15 при відсутності вхідного сигналу. Формально, нехай сила вхідного сигналу k. Тоді змінювач видає сигнал силою 0, якщо k≥1, і сигнал силою 15, якщо k=0.

Петрик задає вам електричну лінію, в якій кабелі, повторювачі і змінювачі позначені літерами k, p, z відповідно. На перший елемент лінії подано сигнал силою 15. Допоможіть Петрику знайти вихідний сигнал останнього блоку в лінії.

* Важливі деталі для врахування в імплементації програми
* працював зі строкою

Завдання №8 Self Practice Work

* Задача з алготестеру - цікава гра
* Деталі завдання

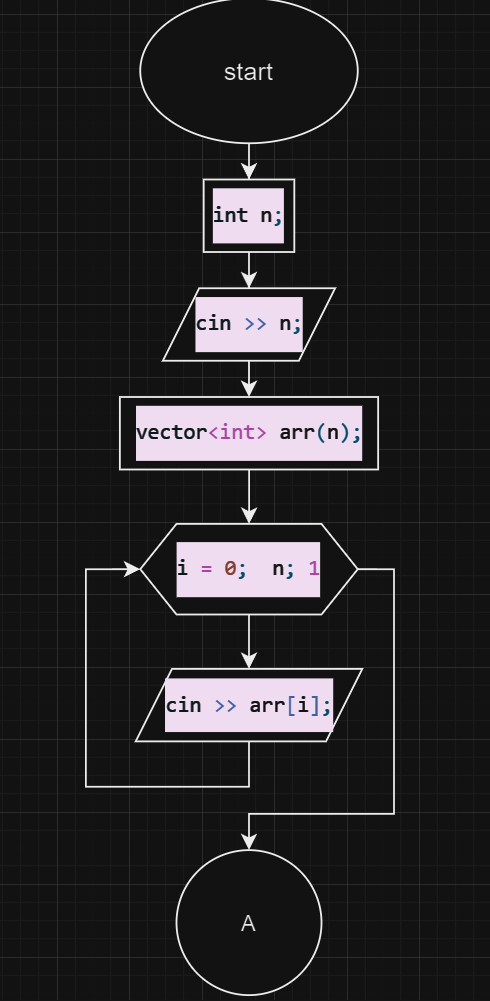


* Важливі деталі для врахування в імплементації програми

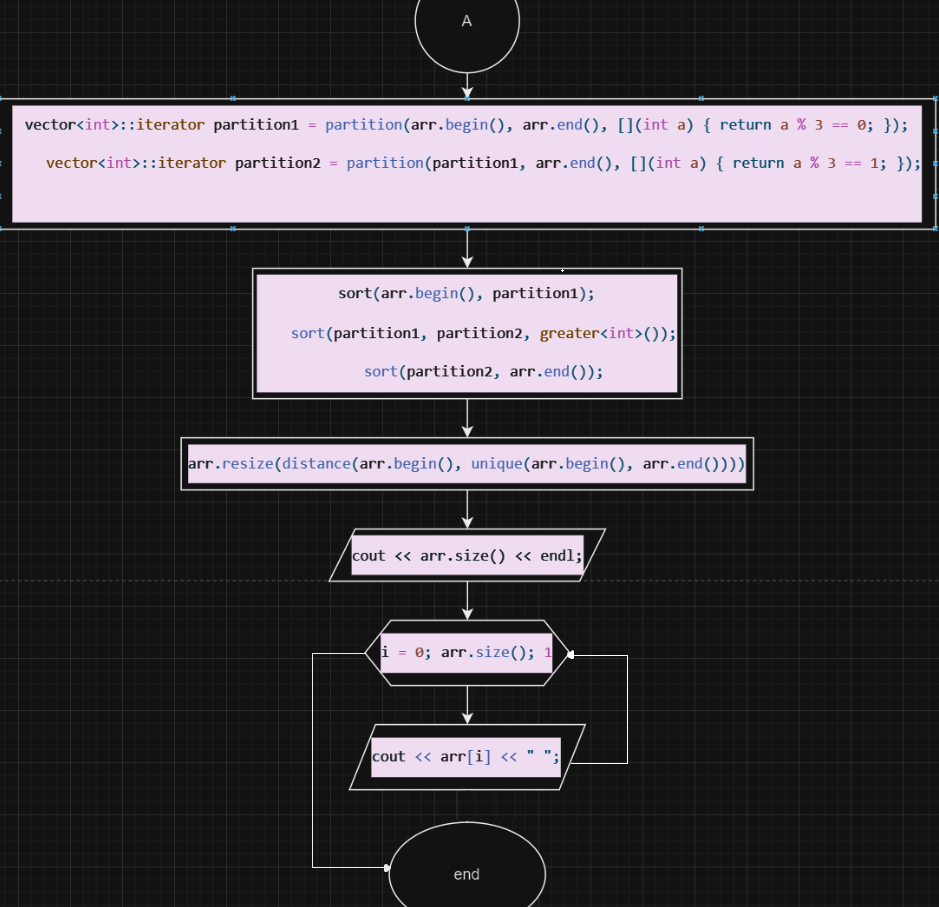
Використав

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

Програма №4 algotester lab 4v2



*Зображення №1. Блок-схема до програми №4 algo Lab 4v2*

**

*Зображення №2. Блок-схема до програми №4 algo Lab 4v2*

* Планований час на реалізацію: година

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

Використано налаштування з попередніх лабораторних робіт.

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

Посилання на пул-реквест: <https://github.com/artificial-intelligence-department/ai_programming_playground/pull/802>

Завдання №1 VNS Lab 6 - Task 1

#include <iostream>

#include <cstring>

#include <map>

using namespace std;

int main() {

char str[256];

char \*word = nullptr;

int max = 0;

map<string, int> wordCounts;

cin.getline(str, 256, '.');

char \*token = strtok(str, " ");

while (token != nullptr) {

wordCounts[token]++;

if (wordCounts[token] > max) {

max = wordCounts[token];

word = token;

}

token = strtok(nullptr, " ");

}

cout << word;

return 0;

}

Завдання №2 VNS Lab 8 - Task 1

#include <iostream>

#include <cstdio>

#include <vector>

#include <cstring>

using namespace std;

struct CAR {

char brand[20];

int year;

int price;

char colors[50];

};

void FileOutput(const char \*filename) {

FILE\* file = fopen(filename, "rb");

if (!file) {

cerr << "Error opening file: " << filename << endl;

return;

}

CAR carData;

while (fread(&carData, sizeof(carData), 1, file)) {

cout << "Brand: " << carData.brand << " Year: " << carData.year

<< " Price: " << carData.price << " Colors: " << carData.colors << endl;

}

fclose(file);

}

void addCar(const char \*filename, const CAR& car) {

vector<CAR> carVector;

FILE\* file = fopen(filename, "rb");

if (file) {

CAR currentCar;

while (fread(&currentCar, sizeof(currentCar), 1, file)) {

carVector.push\_back(currentCar);

}

fclose(file);

}

carVector.insert(carVector.begin(), car);

file = fopen(filename, "wb");

if (file) {

for (const auto& currentCar : carVector) {

fwrite(&currentCar, sizeof(currentCar), 1, file);

}

fclose(file);

} else {

cout << "Error opening file for writing: " << filename << endl;

return;

}

}

void deleteCar(const char \*filename, const int& conditionYear) {

FILE\* inputFile = fopen(filename, "rb");

if (!inputFile) {

cout << "Error opening file for reading: " << filename << endl;

return;

}

vector<CAR> carVector;

CAR currentCar;

while (fread(&currentCar, sizeof(currentCar), 1, inputFile)) {

if(currentCar.year >= conditionYear) {

carVector.push\_back(currentCar);

}

}

fclose(inputFile);

FILE\* outputFile = fopen(filename, "wb");

if (!outputFile) {

cout << "Error opening file for writing: " << filename << endl;

return;

}

for (const auto& currentCar : carVector) {

fwrite(&currentCar, sizeof(currentCar), 1, outputFile);

}

fclose(outputFile);

}

int main() {

const char \*filename = "cars.bin";

int yearToDelete;

CAR newCar1 = {"Toyota", 1999, 30000, "Blue"};

CAR newCar2 = {"Ford", 2005, 25000, "Red"};

CAR newCar3 = {"Honda", 2023, 35000, "Silver"};

addCar(filename, newCar1);

addCar(filename, newCar2);

addCar(filename, newCar3);

cout << "Starting file contents:\n";

FileOutput(filename);

cout << "Delete cars older than (write year of release): \n";

cin >> yearToDelete;

deleteCar(filename, yearToDelete);

cout << "\nFile contents after deleting cars with the year under " << yearToDelete << ":\n";

FileOutput(filename);

CAR newCar4 = {"Chevrolet", 2017, 40000, "Black"};

addCar(filename, newCar4);

cout << "\nFile contents after adding a new car:\n";

FileOutput(filename);

return 0;

}

Завдання №3 VNS Lab 9 - Task 1

#include <iostream>

#include <algorithm>

#include <fstream>

#include <string>

#include <vector>

using namespace std;

int main() {

int N,N1,N2;

vector<string> myVector;

int maxLine = 0;

int maxCount = 0;

string line, line2;

int currentLine = 0;

int currentLine2 = 0;

cin >> N;

for(int i = 0; i < 10; i++){

string a;

cin >> a;

myVector.push\_back(a);

}

cout << endl;

cin >> N1 >> N2;

ofstream fileF1("F1.txt");

for (int i = 0; i < N; ++i) {

fileF1 << myVector[i] << endl;

}

fileF1.close();

ifstream fileF2("F1.txt");

ofstream fileF3("F2.txt");

while (getline(fileF2, line)) {

++currentLine;

if (currentLine >= N1 && currentLine <= N2 && line.back() == 'A') {

fileF3 << line << '\n';

}

}

fileF2.close();

fileF3.close();

ifstream fileF4("F2.txt");

while (getline(fileF4, line)) {

++currentLine2;

int Count = count(line.begin(), line.end(), 'A');

if (Count > maxCount) {

maxCount = Count;

maxLine = currentLine2;

}

}

fileF4.close();

cout << "The line in F2 with the most 'A' characters is: " << maxLine << '\n';

return 0;

}

Завдання №4 Algotester Lab 4

way 1

#include <iostream>

#include <vector>

using namespace std;

void merge(vector<int>& arr, int left, int mid, int right, bool reverse) {

int n1 = mid - left + 1;

int n2 = right - mid;

vector<int> Left\_arr(n1), Right\_arr(n2);

for (int i = 0; i < n1; i++)

Left\_arr[i] = arr[left + i];

for (int j = 0; j < n2; j++)

Right\_arr[j] = arr[mid + 1 + j];

int i = 0, j = 0, k = left;

while (i < n1 && j < n2) {

if ((!reverse && Left\_arr[i] <= Right\_arr[j]) || (reverse && Left\_arr[i] >= Right\_arr[j])) {

arr[k] = Left\_arr[i];

i++;

} else {

arr[k] = Right\_arr[j];

j++;

}

k++;

}

while (i < n1) {

arr[k] = Left\_arr[i];

i++;

k++;

}

while (j < n2) {

arr[k] = Right\_arr[j];

j++;

k++;

}

}

void mergeSort(vector<int>& arr, int left, int right, bool reverse) {

if (left < right) {

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

mergeSort(arr, left, mid, reverse);

mergeSort(arr, mid + 1, right, reverse);

merge(arr, left, mid, right, reverse);

}

}

vector<int> mergeArrays(vector<int> arr1, vector<int> arr2, vector<int>& arr3){

vector<int> result;

for (int num : arr1) {

result.push\_back(num);

}

for (int num : arr2) {

result.push\_back(num);

}

for (int num : arr3) {

result.push\_back(num);

}

return result;

}

int removeDuplicates(std::vector<int>& arr) {

int n = arr.size();

for (int i = 0; i < n; ++i) {

for (int j = i + 1; j < n; ) {

if (arr[i] == arr[j]) {

for (int k = j; k < n - 1; ++k) {

arr[k] = arr[k + 1];

}

--n;

} else {

++j;

}

}

}

arr.resize(n);

return n;

}

int sorting(vector<int>& arr){

std::vector<int> arr0, arr1, arr2;

for (int num : arr) {

int remainder = num % 3;

switch (remainder)

{

case 0:

arr0.push\_back(num);

break;

case 1:

arr1.push\_back(num);

break;

case 2:

arr2.push\_back(num);

break;

default:

break;

}

}

mergeSort(arr0, 0, arr0.size() - 1, false);

mergeSort(arr1, 0, arr1.size() - 1, true);

mergeSort(arr2, 0, arr2.size() - 1, false);

arr = mergeArrays(arr0, arr1, arr2);

int n = removeDuplicates(arr);

return n;

}

int main(){

int N;

vector<int> input;

cin >> N;

for(int i = 0; i < N; i++){

int a;

cin >> a;

input.push\_back(a);

}

int n = sorting(input);

cout << n << endl;

for(int i = 0; i < n; i++){

cout << input[i] << " ";

}

return 0;

}

way 2

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

int n;

cin >> n;

vector<int> arr(n);

for (int i = 0; i < n; ++i) {

cin >> arr[i];

}

vector<int>::iterator partition1 = partition(arr.begin(), arr.end(), [](int a) { return a % 3 == 0; });

vector<int>::iterator partition2 = partition(partition1, arr.end(), [](int a) { return a % 3 == 1; });

sort(arr.begin(), partition1);

sort(partition1, partition2, greater<int>());

sort(partition2, arr.end());

arr.resize(distance(arr.begin(), unique(arr.begin(), arr.end())));

cout << arr.size() << endl;

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

cout << arr[i] << " ";

}

return 0;

}

Завдання №5 Algotester Lab 6

#include <iostream>

#include <algorithm>

#include <vector>

using namespace std;

struct Pair{

int X;

int Y;

};

vector<int> possibleValues(vector<vector<int>> game, int row, int col){

int N = game.size();

vector<int> possible;

for (int value = 1; value <= N; ++value) {

possible.push\_back(value);

}

for (int j = 0; j < N; ++j) {

auto it = find(possible.begin(), possible.end(), game[row][j]);

if (it != possible.end()) {

possible.erase(it);

}

}

for (int i = 0; i < N; ++i) {

auto it = find(possible.begin(), possible.end(), game[i][col]);

if (it != possible.end()) {

possible.erase(it);

}

}

return possible;

}

int main(){

int N;

int Q;

cin >> N;

vector<vector<int>> game(N, vector<int>(N, 0));

vector<int> num;

vector<Pair> pairs;

for(int i = 0; i < N; i++){

int a;

cin >> a;

num.push\_back(a);

}

cin >> Q;

for(int i = 0; i < Q; i++){

int x, y;

cin >> x >> y;

pairs.push\_back({x, y});

}

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

int number = num[i];

for (int j = N - 1; j >= 0; --j) {

game[i][j] = number % 10;

number /= 10;

}

}

for(int i = 0; i < Q; i++){

if(game[pairs[i].X - 1][pairs[i].Y - 1] != 0){

cout << 1 << endl;

cout << game[pairs[i].X - 1][pairs[i].Y - 1];

cout << endl;

cout << endl;

}

else{

vector<int> possibleValuesVector = possibleValues(game, pairs[i].X - 1, pairs[i].Y - 1);

cout << possibleValuesVector.size() << endl;

for(int j = 0; j < possibleValuesVector.size(); j++){

cout << possibleValuesVector[j] << " ";

}

cout << endl;

cout << endl;

}

}

return 0;

}

Завдання №6 ClassPractice Work

Task 1

#include <iostream>

#include <string>

#include <fstream>

#include <cstring>

using namespace std;

enum FileOpResult{

Success,

Failure

};

FileOpResult write\_to\_file(const char \*name, char \*content){

ofstream F(name);

if(F.is\_open()){

cout << "opened successfully!" << endl;

F << content;

F.close();

if(!F.is\_open()){

cout << "closed successfully!" << endl;

return Success;

}

else{

cout << "problem in closing!" << endl;

return Failure;

}

}

else{

cout << "problem in opening!" << endl;

return Failure;

}

}

int main(){

const char \*name = "text.txt";

char content[200];

char answer[5];

do{

cout << "input: ";

cin.ignore();

cin.getline(content, sizeof(content));

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

if(result == Success){

ifstream file1(name);

string line;

while(getline(file1, line)){

cout << line << endl;

}

file1.close();

cout << "successfully printed!" << endl;

cout << "do u want to continue? ";

cin >> answer;

}

else{

cout << "do u want to try one more time? ";

cin >> answer;

}

}while(strcmp(answer, "yes") == 0);

cout << "bye!" << endl;

return 0;

}

Task 2

#include <iostream>

#include <string>

#include <fstream>

#include <cstring>

using namespace std;

enum FileOpResult{

Success,

Failure

};

FileOpResult copy\_file(char file\_from[], char file\_to[]){

ifstream sourceFile(file\_from);

ofstream destinationFile(file\_to);

if(sourceFile.is\_open() && destinationFile.is\_open()){

cout << "files opened successfully!" << endl;

string line;

while(getline(sourceFile, line)){

destinationFile << line << endl;

}

cout << "file copied successfully!" << endl;

sourceFile.close();

destinationFile.close();

if(!sourceFile.is\_open() && !destinationFile.is\_open()){

cout << "files closed successfully!" << endl;

return Success;

}

else{

cout << "problem in closing files!" << endl;

return Failure;

}

}

else if (!sourceFile) {

cout << "Failed to open source file." << endl;

destinationFile.close();

return Failure;

}

else if (!destinationFile) {

cout << "Failed to open destination file." << endl;

sourceFile.close();

return Failure;

}

return Failure;

}

int main(){

char file\_from[20];

char file\_to[20];

char content[200];

char answer[5];

do{

cout << "file from: ";

cin.getline(file\_from, sizeof(file\_from));

cout << "file to: ";

cin.getline(file\_to, sizeof(file\_to));

FileOpResult copyResult = copy\_file(file\_from, file\_to);

if(copyResult == Success){

ifstream copiedFile(file\_to);

string line;

while(getline(copiedFile, line)){

cout << line << endl;

}

copiedFile.close();

cout << "file printed successfully!" << endl;

cout << "want to try more? " << endl;

cin >> answer;

}

else{

cout << "want to try more? " << endl;

cin >> answer;

}

cin.ignore();

}while(strcmp(answer, "yes") == 0);

cout << "bye!" << endl;

return 0;

}

Завдання №7 Self Practice Work

#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

int main(){

int source = 15;

int n;

string input;

cin >> n;

cin >> input;

for(int i = 0; i < n; i++){

if(input[i] == 'k'){

source = max(source - 1, 0);

}

else if(input[i] == 'p'){

if(source != 0){

source = 15;

}

else{

source = 0;

}

}

else if(input[i] == 'z'){

if(source == 0){

source = 15;

}

else if(source != 0){

source = 0;

}

}

}

cout << source;

return 0;

}

Завдання №8 Self Practice Work

#include <iostream>

int main() {

int n, m;

std::cin >> n >> m;

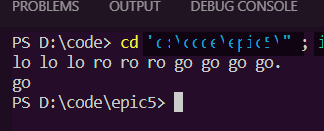
std::cout << ((n\*m)%2 == 0 ? "Dragon" : "Imp");

return 0;

}

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

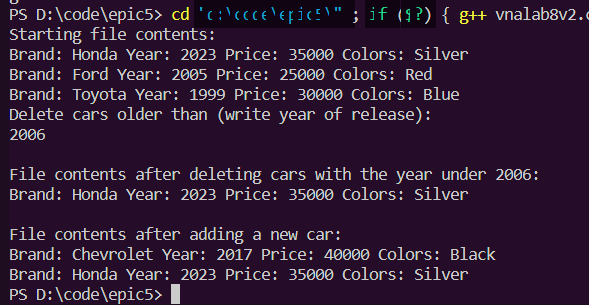
Завдання №1 VNS Lab 6 - Task 1



*Зображення №3. Результати VNS Lab 6 Task 1*

Час затрачений на виконання завдання: 5 год

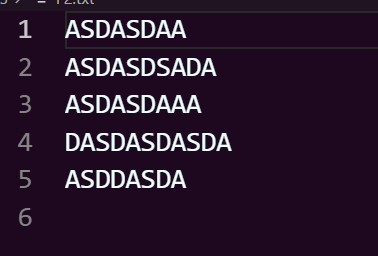
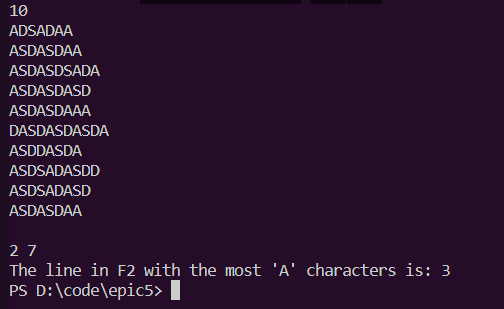
Завдання №2 VNS Lab 8 - Task 1



*Зображення №4. Результати VNS Lab 8 Task 1*

Час затрачений на виконання завдання: 6 год

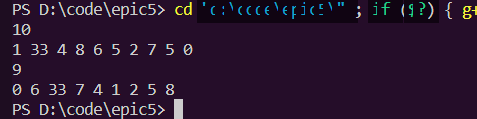
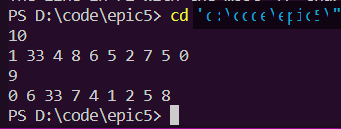
Завдання №3 VNS Lab 9 - Task 1



*Зображення №5, 6. Результати VNS Lab 9 Task 1*

Час затрачений на виконання завдання: 4 год

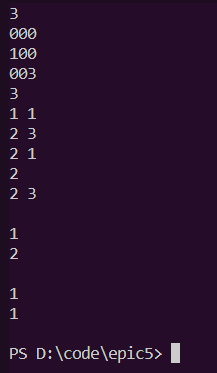
Завдання №4 Algotester Lab 4



*Зображення №7,8. Результати Algotester Lab 4 і Algotester Lab 4 в2*

Час затрачений на виконання завдання: 5 год

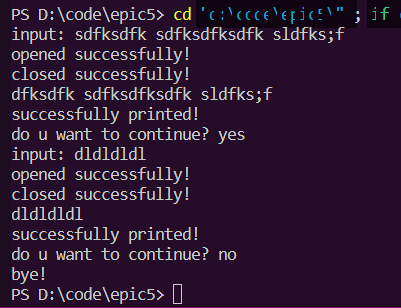
Завдання №5 Algotester Lab 6



*Зображення №9. Результати Algotester Lab 6*

Час затрачений на виконання завдання: 4 год

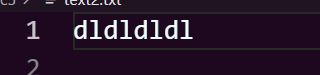
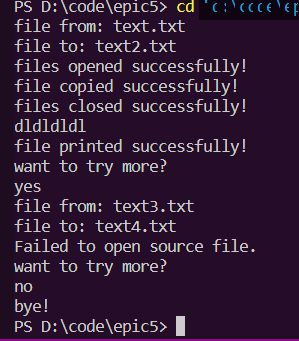
Завдання №6 ClassPractice Work







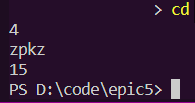
*Зображення №10, 11, 12. Результати Class Practice Work*



*Зображення №13, 14. Результати Class Practice Work*

Час затрачений на виконання завдання: 3 год

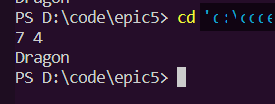
Завдання №7 Self Practice Work



*Зображення №15. Результати №7 Self Practice Work*

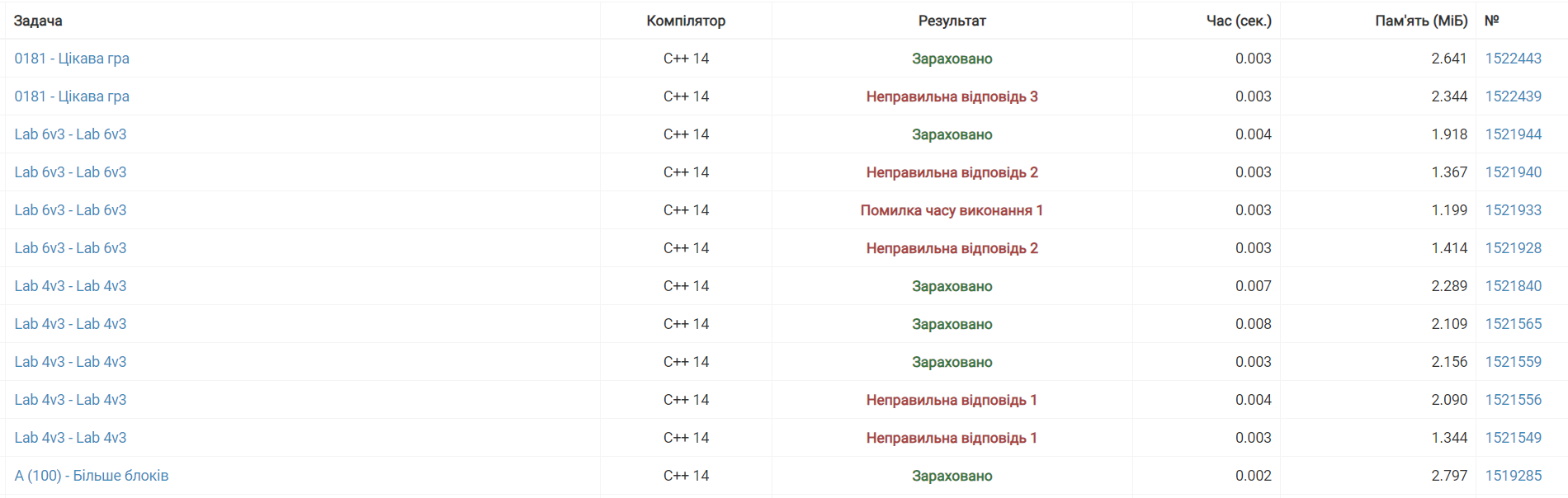
Час затрачений на виконання завдання: 1 год

Завдання №8 Self Practice Work



*Зображення №16. Результати №8 Self Practice Work*

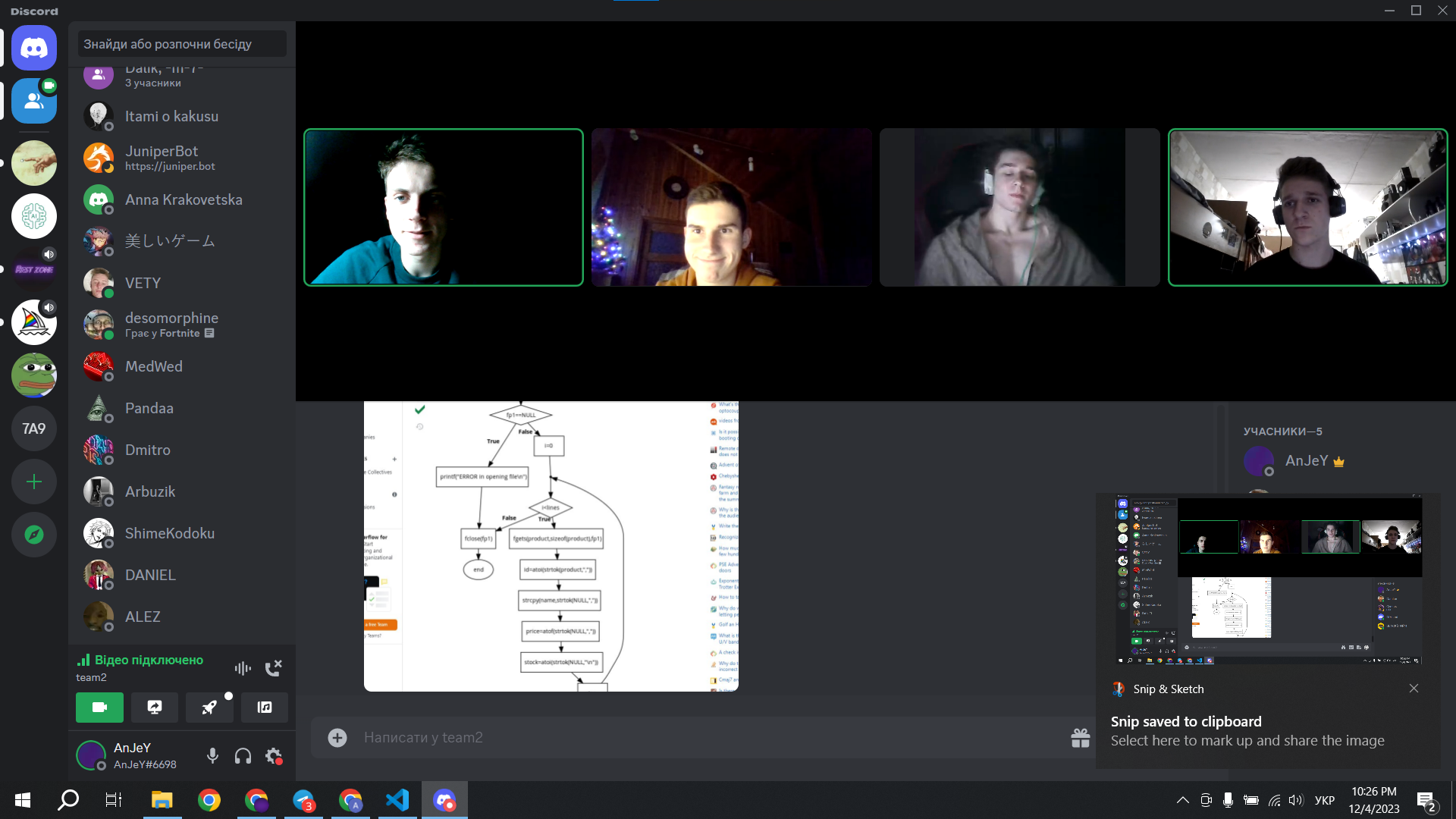
Час затрачений на виконання завдання: 10 хв



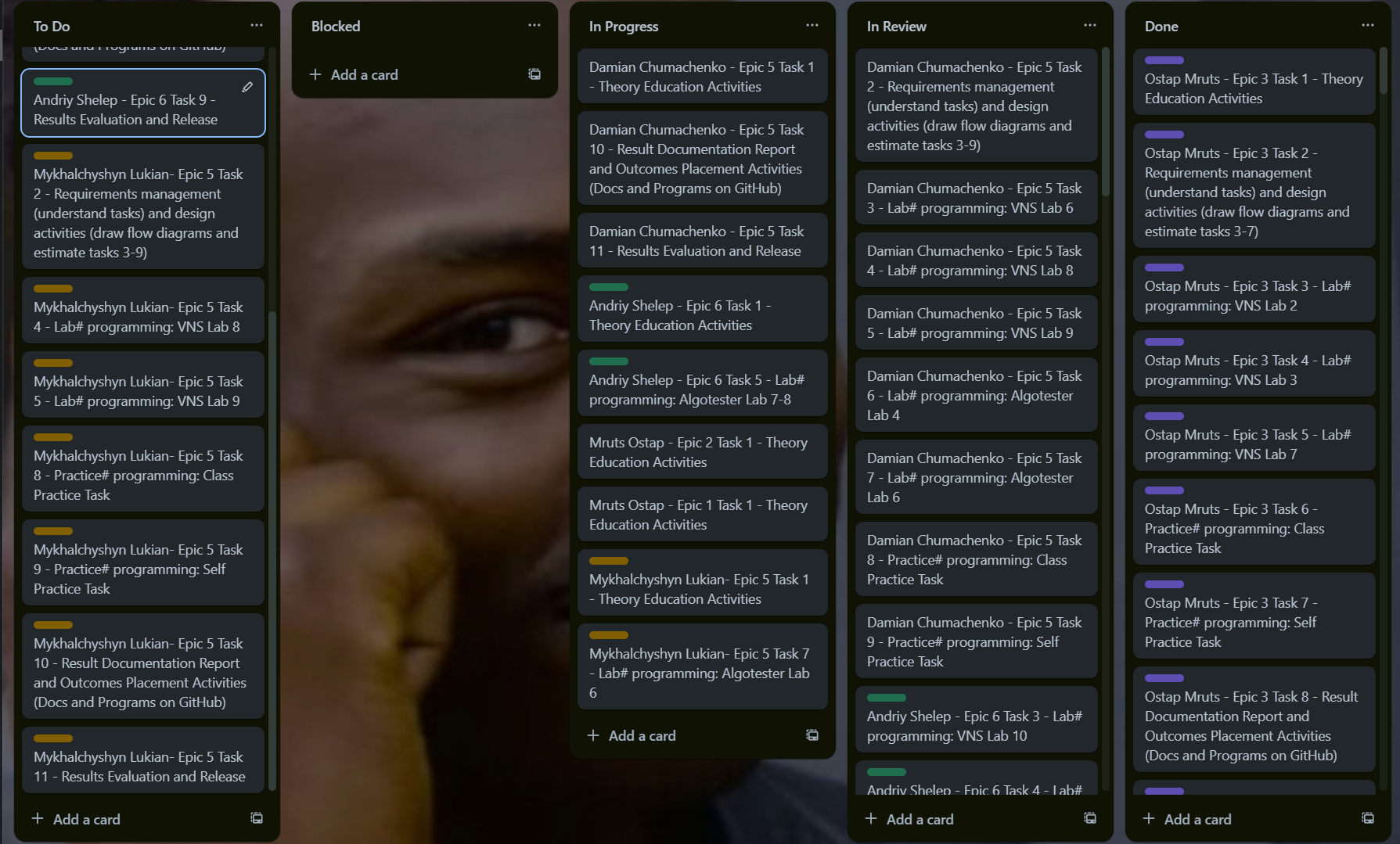
*Зображення №17. Результати algotested*

## **6. Кооперація з командою:**

* Скрін з 1-ї зустрічі по обговоренню задач Епіку та Скрін прогресу по Трелло



*Зображення №18. Кооперація з командою*



*Зображення №19. Кооперація з командою*

# **Висновки:**

Ознайомився з основними методами та деталями роботи з файлами, запису у файл і зчитування з нього у С++. Застосував ці знання, виконуючи завдання лабораторних робіт з ВНС та практичного завдання. Також опрацював рядкові змінні і символи використовував їх у завданнях з алготестера та у лабораторних з ВНС.