Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

A blue and white logo

Description automatically generated

**Звіт**

**Звіт**

**про виконання лабораторних та практичних робіт блоку № 5**

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

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

до:

ВНС Лабораторної Роботи № 6

ВНС Лабораторної Роботи № 8

ВНС Лабораторної Роботи № 9

Алготестер Лабораторної Роботи №4

Алготестер Лабораторної Роботи №6

Практичних Робіт №5

**Виконав:**

Студент групи ШІ-11

Стожук Роман Ігорович

# **Тема роботи:**

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

# **Мета роботи:**

# Навчитися працювати з файлами та бібліотеками

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Робота з файлами
* Тема №2: Системи числення
* Тема №3: Стандартна бібліотека
* Тема №3: Створення й використання бібліотек

1. Індивідуальний план опрацювання теорії:

* Тема №1: Робота з файлами
  + Джерела Інформації
    - <https://acode.com.ua/urok-220-bazovyj-fajlovyj-vvid-i-vyvid/>
    - <https://acode.com.ua/urok-221-randomnyj-fajlovyj-vvid-i-vyvid/>
  + Що опрацьовано:
    - Створення, знищення, переіменування файлів, запис та зчитування інформації
  + Статус: Ознайомлений
  + Початок опрацювання теми: 01.12.2023
  + Звершення опрацювання теми: 02.12.2023
* Тема №2: Системи числення
  + Джерела Інформації:
    - <https://acode.com.ua/urok-47-konvertatsiya-chysel-z-dvijkovoyi-systemy-chyslennya-v-desyatkovu-i-navpaky/>
  + Що опрацьовано:
    - Десяткову, двійкову, вісімкову та шіснадцяткову системи числення
  + Статус: Ознайомлений
  + Початок опрацювання теми: 02.12.2023
  + Звершення опрацювання теми: 03.12.2023
* Тема №3: Стандартна бібліотека
  + Джерела Інформації
    - <https://acode.com.ua/urok-204-standartna-biblioteka-shabloniv-stl/>
    - <https://acode.com.ua/urok-205-kontejnery-stl/>
    - <https://acode.com.ua/urok-206-iteratory-stl/>
    - <https://acode.com.ua/urok-207-algorytmy-stl/>
  + Що опрацьовано:
    - Базові контейнери та алгоритми стандартної бібліотеки
  + Статус: Ознайомлений
  + Початок опрацювання теми: 03.12.2023
  + Звершення опрацювання теми: 05.12.2023
* Тема №4: Створення та використання бібліотек
  + Джерела Інформації:
    - <https://acode.com.ua/urok-81-bagatovymirni-masyvy/>
  + Що опрацьовано:
    - Статичні та динамічні бібліотеки
  + Статус: Ознайомлений
  + Початок опрацювання теми: 07.12.2023
  + Звершення опрацювання теми: 07.12.2023

# **Виконання роботи:**

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

Завдання №3 VNS Lab 6

* Варіант завдання: 6
* 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.
* Перетворити рядок так, щоб всі букви в ньому були відсортовані за зростанням.

Завдання №4 VNS Lab 8

* Варіант завдання: 6
* 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 "Школяр":
  + прізвище, ім'я, по батькові;
  + клас;
  + номер телефону;
  + оцінки по предметах (математика, фізика, українська мова, література).
* Знищити всі елементи, у яких є 2 хоча б з одного предмету, додати елемент у початок файлу.

Завдання №5 VNS Lab 9

* Варіант завдання: 6

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

# 1) Скопіювати з файлу F1 у файл F2 рядки, починаючи з N до K. 2) Підрахувати кількість приголосних букв у файлі F2.

Завдання №6 Algotester Lab 4

* Варіант завдання: 2
* 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].

# 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
* Обмеження:

1≤N,K≤1000  
0≤a≤100

Завдання №7 Algotester Lab 6

* Варіант завдання: 2
* У вас є шахова дошка розміром 8×88×8 та дуже багато фігур. Кожна клітинка може мати таке значення:
* Пуста клітинка O
* Пішак P
* Тура R
* Кінь N
* Слон B
* Король K
* Королева Q

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

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

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

У випадку, якщо клітинку не атакують - виведіть O.

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.

# Вхідні дані

У перших 8 рядках стрічка row - стан i-го рядка дошки.

У наступному рядку ціле число Q - кількість записів

У наступних Q рядках 2 цілих числа x та y - координати клітинки

# Вихідні дані

Q разів відповідь у наступному форматі:

Строка result - усі фігури, які атакують клітинку з запиту.

# Обмеження:

|rowi|=N

rowi∈{O,P,R,N,B,K,Q}

1≤Q≤64

1≤x,y≤8

Завдання №8 Class Practice Task

* 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.

Завдання №9.1 Self Practice Task 0213

* З Усі ви знаєте, що студентам часто доручають надзвичайно важливу роботу, яку просто нікому виконати. Оте приберіть, оце зберіть, туди сходіть, звідти прийдіть...

От і цього разу студентів факультету прикладної математики попросили порахувати кількість дерев у парку, що біля гуртожитку.

Студенти, котрі щойно відсвяткували День програміста, з радістю погодились виконати це цікаве завдання.

Кожен порахував кількість дерев кожного виду у парку.

Біда в тому, що молодь діяла не надто узгоджено, тому різні студенти могли врахувати в свою статистику ті самі дерева. Звісно ж, дехто вважав зайвим рахувати всі дерева, і тому врахував лише деякі. Але ми впевнені, що усі студенти безпомилково визначають вид дерева та завжди пам’ятають, котрі дерева вони вже врахували.

Відомо, що завдання виконували n студентів, і кожен із них зібрав дані про m видів дерев.

Вам потрібно допомогти адміністрації гуртожитку — напишіть програму, котра визначить мінімальну можливу кількість дерев в парку.

# Вхідні дані

У першому рядку задано цілі числа n і m — кількість студентів та кількість видів дерев відповідно.

У наступних n рядках задано результати, котрі записав один із студентів.

Результати, записані кожним із студентів — m цілих невід’ємних чисел, i-e з яких відповідає кількості дерев іі-го виду, які знайшов у парку даний студент.

# Вихідні дані

У єдиному рядку виведіть одне ціле число — мінімальну можливу кількість дерев у парку.

# Обмеження:

0≤n,m≤77

0≤numberij<7474

Завдання №9.2 Self Practice Task 0754

* Зеник і Марічка приїхали до Антарктиди подивитися на пінгвінів. Мабуть, не знали, що у Львові вони теж є. Усього є n пінгвінів, причому i-ий на початку розміщений у точці (xi,yi) на площині. Ціль пінгвінів — зустрітись разом у певній точці. На жаль, пінгвіни не дуже класно вміють ходити, тому за одну секунду з точки (x,y) пінгвін може перейти хіба що в (x+1,y) або (x,y+1). Ба більше, у будь-який момент часу рухатися може лише один пінгвін.

За який мінімальний час всі пінгвіни можуть зустрітися в певній точці?

# Вхідні дані

У першому рядку задано одне ціле число n — кількість пінгвінів.

У наступних n рядках задано по два цілих числа xi, yi — початкові координати пінгвінів.

# Вихідні дані

У єдиному рядку виведіть одне ціле число — відповідь на задачу.

# Обмеження:

1≤n≤

0≤xi,yi≤

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

Програма №4 VNS Lab 8

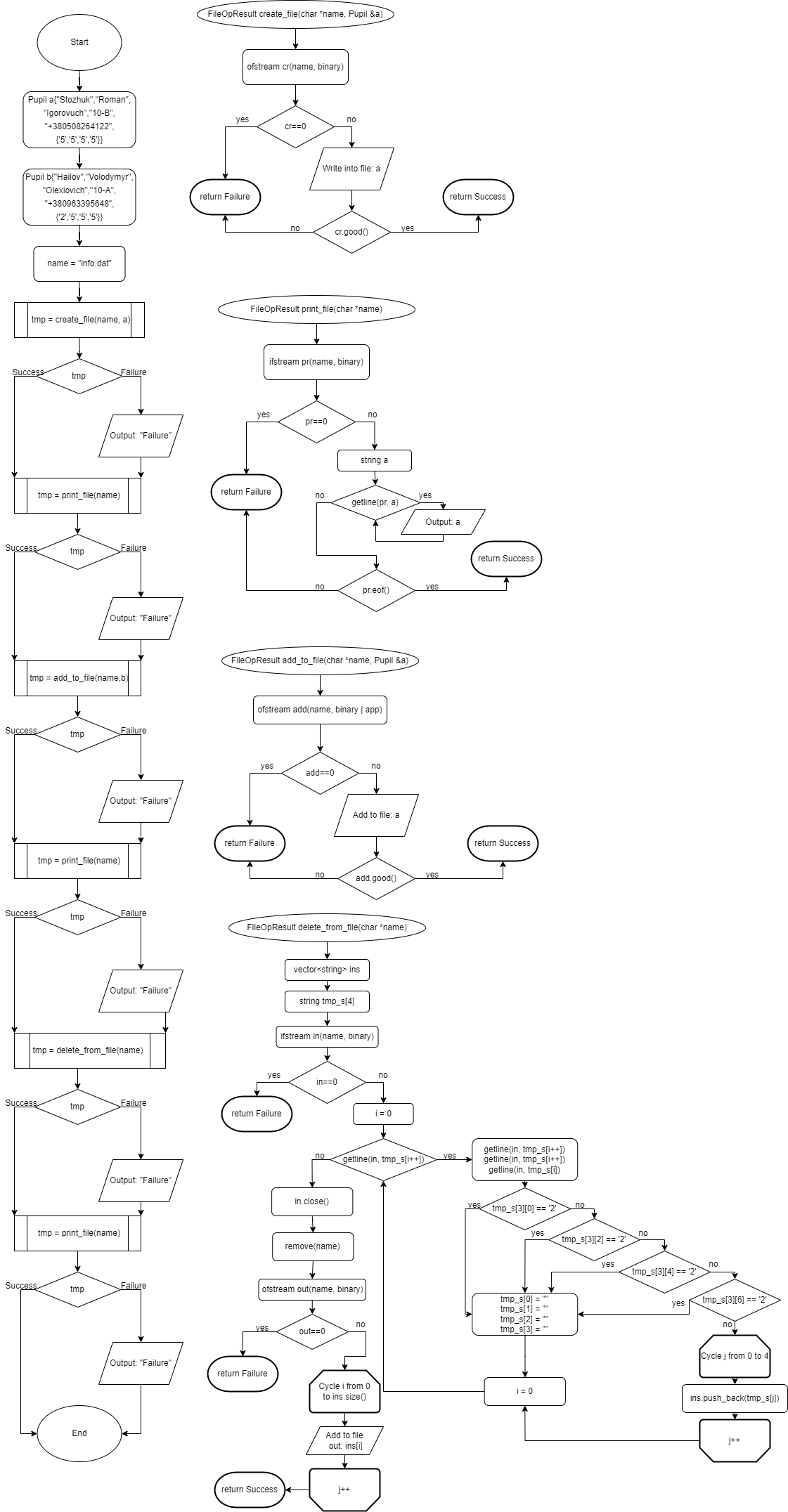
* 

Рисунок 1Блок схема для програми №4

* Планований час на реалізацію: 2год

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

## Змін у конфігурацію середовища не вносив.

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

Завдання №3 VNS Lab 6

#include <stdio.h>

#include <string.h>

int main() {

    char s[256], pr[256];

    gets\_s(s, 255);

    int letters[52] = {0};

    int abc=0;

    while (s[abc] != '.') {

        for (int j = 0; j < 26; j++) {

            if (s[abc] == (char)(j+65)) {letters[j]++;}

            if (s[abc] == (char)(j+97)) {letters[j+26]++;}

        }

        abc++;

    }

    int counter = 0;

    for (int i = 0; i<26; i++) {

        while (letters[i] !=0) {

            pr[counter]=(char)(i+65);

            --letters[i];

            ++counter;

        }

        while (letters[i+26]!=0) {

            pr[counter]=(char)(i+97);

            --letters[i+26];

            ++counter;

        }

    }

    pr[counter] = '.';

    pr[counter+1] = '\0';

    puts(pr);

    return 0;

}

<https://github.com/artificial-intelligence-department/ai_programming_playground/pull/727/files#diff-7379df6f1ae551996c7bb9dbb28331f681c5a123fd0fb3a118240b0843ebb0a2>

Завдання №4 VNS Lab 8

#include <fstream>

#include <iostream>

#include <string>

#include <vector>

using namespace std;

enum FileOpResult {

    Success,

    Failure

};

struct Pupil {

    string last\_name;

    string first\_name;

    string surname;

    string clas;

    string phone\_num;

    char evaluations[4];

};

FileOpResult create\_file(char \*name, Pupil &a) {

    ofstream cr(name, ios::binary);

    if (!cr) {return Failure;}

    cr << a.last\_name  << ' ' << a.first\_name << ' '

    << a.surname << '\n' << a.clas << '\n' << a.phone\_num << '\n'

    << a.evaluations[0] << ' ' << a.evaluations[1] << ' '<< a.evaluations[2] << ' ' << a.evaluations[3] << '\n';

    if (cr.ios::good()) {return Success;}

    else {return Failure;}

}

FileOpResult print\_file(char \*name) {

    ifstream pr(name, ios::binary);

    if (!pr) {return Failure;}

    string a;

    while (getline(pr, a)) {

        cout << a << '\n';

    }

    if (pr.ios::eof()) {return Success;}

    else {return Failure;}

}

FileOpResult add\_to\_file(char \*name, Pupil &a) {

    ofstream add(name, ios::app | ios::binary);

    if (!add) {return Failure;}

    add << a.last\_name << ' ' << a.first\_name << ' '

    << a.surname << '\n' << a.clas << '\n' << a.phone\_num << '\n'

    << a.evaluations[0] << ' ' << a.evaluations[1] << ' '<< a.evaluations[2] << ' ' << a.evaluations[3] << '\n';

    if (add.ios::good()) {return Success;}

    else {return Failure;}

}

FileOpResult delete\_from\_file(char \*name) {

    vector<string> ins;

    string tmp\_s[4];

    ifstream in(name, ios::binary);

    if (!in) {return Failure;}

    short i = 0;

    while(getline(in, tmp\_s[i++])) {

        getline(in, tmp\_s[i++]);

        getline(in, tmp\_s[i++]);

        getline(in, tmp\_s[i]);

        if ((tmp\_s[3][0] == '2') || (tmp\_s[3][2] == '2') || (tmp\_s[3][4] == '2') || (tmp\_s[3][6] == '2')) {tmp\_s[0] = ""; tmp\_s[1] = ""; tmp\_s[2] = ""; tmp\_s[3] = "";}

        else {for(int j=0;j<4;j++) {ins.push\_back(tmp\_s[j]);}}

        i = 0;

    }

    in.close();

    remove(name);

    ofstream out(name, ios::binary);

    if (!out) {return Failure;}

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

        out << ins[i] << '\n';

    }

    return Success;

}

int main() {

    Pupil a{"Stozhuk","Roman","Igorovuch","10-B","+380508264122",{'5','5','5','5'}};

    Pupil b{"Hailov","Volodymyr","Olexiovich","10-A","+380963395648",{'2','5','5','5'}};

    char name[9] = "info.dat";

        if (create\_file(name, a)) { std::cout << "Failure\n"; }

        if (print\_file(name)) { std::cout << "Failure\n"; }

        if (add\_to\_file(name, b)) { std::cout << "Failure\n"; }

        if (print\_file(name)) { std::cout << "Failure\n"; }

        if (delete\_from\_file(name)) { std::cout << "Failure\n"; }

        if (print\_file(name)) { std::cout << "Failure\n"; }

    return 0;

}

<https://github.com/artificial-intelligence-department/ai_programming_playground/pull/727/files#diff-76c7be0c1a3f4a7f31d0ca916c6d2e1e2d91508e5190163f9284bdfea6f79418>

Завдання №5 VNS Lab 9

#include <fstream>

#include <iostream>

#include <string>

using namespace std;

int main() {

    char file\_from[9] = "test.txt", file\_to[11] = "copied.cpp";

    unsigned int n = 5, k = 9;

    ifstream in(file\_from, ios::binary);

    ofstream out(file\_to, ios::binary | ios::app);

    if ((!out) || (!in)) {cout << "Failure1\n";}

    string tmp;

    for (int i = 1; i < k; i++) {

        getline(in, tmp);

        if (i >= n) {out << tmp << '\n';}

    }

    if (out.ios::good() && in.ios::good()) {std::cout << "Success\n";}

    else {cout << "Failure2\n";}

    in.close(); out.close();

    ifstream new\_in(file\_to, ios::binary);

    if(!new\_in) {cout << "Failure3\n";}

    unsigned int cononsants = 0;

    string check;

    while(getline(new\_in, check)) {

        for (int i = 0; i<check.length(); i++) {

            if (check[i] == 'B' || check[i] == 'b' || check[i] == 'C' || check[i] == 'c' ||

                check[i] == 'D' || check[i] == 'd' || check[i] == 'F' || check[i] == 'f' ||

                check[i] == 'G' || check[i] == 'g' || check[i] == 'H' || check[i] == 'h' ||

                check[i] == 'J' || check[i] == 'j' || check[i] == 'K' || check[i] == 'k' ||

                check[i] == 'L' || check[i] == 'l' || check[i] == 'M' || check[i] == 'm' ||

                check[i] == 'N' || check[i] == 'n' || check[i] == 'P' || check[i] == 'p' ||

                check[i] == 'Q' || check[i] == 'q' || check[i] == 'R' || check[i] == 'r' ||

                check[i] == 'S' || check[i] == 's' || check[i] == 'T' || check[i] == 't' ||

                check[i] == 'V' || check[i] == 'v' || check[i] == 'W' || check[i] == 'w' ||

                check[i] == 'X' || check[i] == 'x' || check[i] == 'Z' || check[i] == 'z') ++cononsants;

        }

    }

    cout << cononsants;

    return 0;

}

<https://github.com/artificial-intelligence-department/ai_programming_playground/pull/727/files#diff-bb6676d6be8fcca0e5870139f9a360e923afa8e0828dfa39821f733fccd188d3>

Завдання №6 Algotester Lab 4

#include <iostream>

#include <algorithm>

#include <vector>

int main() {

    short n,k,mx,m\_index;

    std::cin >> n >> k;

    std::vector<short> a(n);

    for (short i = 0; i<n; i++) { std::cin >> a[i]; }

    std::sort(a.begin(), a.end());

    mx = a[n-1];

    std::unique(a.begin(), a.end());

    for (short i=0; i<n; i++) {if(a[i] == mx) {m\_index = i+1; break;}}

    a.resize(m\_index);

    for (short i = 0; i<k; i++) {

        std::rotate(a.begin(), a.begin() + 1, a.end());

    }

    std::cout << a.size() << std::endl;

    for (short i = 0; i<a.size(); i++) {std::cout << a[i] << ' ';}

    return 0;

}

<https://github.com/artificial-intelligence-department/ai_programming_playground/pull/727/files#diff-2a4cd8a6c605faa18b7ee0caef58b777f0ac676f6447aaf6f0c41b96dd1b7872>

Завдання №7 Algotester Lab 6

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

    //Input figures

    vector<vector<char>> figures(8);

    for (short i = 0; i<8; i++) {

        figures[i].resize(8);

        for (short j = 0; j<8; j++) { cin >> figures[i][j]; }

    }

    //Input requests

    short req\_quantity=0; cin >> req\_quantity;

    vector<short> req\_y(req\_quantity), req\_x(req\_quantity);

    for (short i = 0; i<req\_quantity; i++) {

        cin >> req\_y[i] >> req\_x[i];

        req\_y[i]--;

        req\_x[i]--;

    }

    vector<vector<char>> result(req\_quantity);

    //Searching answer

    for (short cur\_req = 0; cur\_req < req\_quantity; cur\_req++) {

        //Something is here

        if(figures[req\_y[cur\_req]][req\_x[cur\_req]] != 'O') {

            result[cur\_req].push\_back('X');

        }

        //Something attacks it

        else {

            short x = req\_x[cur\_req], y = req\_y[cur\_req];

            //kings & pawns

            if (0 <= x-1 && x-1 < 8 && 0<=y-1 && y-1 < 8) {

                if (figures[y-1][x-1] == 'K')

                result[cur\_req].push\_back('K');

                if (figures[y-1][x-1] == 'P')

                result[cur\_req].push\_back('P');

            }

            if (0<=y-1 && y-1 < 8) {

                if (figures[y-1][x] == 'K')

                result[cur\_req].push\_back('K');

            }

            if (0 <= x+1 && x+1 < 8 && 0<=y-1 && y-1 < 8) {

                if (figures[y-1][x+1] == 'K')

                result[cur\_req].push\_back('K');

                if (figures[y-1][x+1] == 'P')

                result[cur\_req].push\_back('P');

            }

            if (0 <= x+1 && x+1 < 8) {

                if (figures[y][x+1] == 'K')

                result[cur\_req].push\_back('K');

            }

            if (0 <= x+1 && x+1 < 8 && 0<=y+1 && y+1 < 8) {

                if (figures[y+1][x+1] == 'K')

                result[cur\_req].push\_back('K');

            }

            if (0<=y+1 && y+1 < 8) {

                if (figures[y+1][x] == 'K')

                result[cur\_req].push\_back('K');

            }

            if (0 <= x-1 && x-1 < 8 && 0<=y+1 && y+1 < 8) {

                if (figures[y+1][x-1] == 'K')

                result[cur\_req].push\_back('K');

            }

            if (0 <= x-1 && x-1 < 8) {

                if (figures[y][x-1] == 'K')

                result[cur\_req].push\_back('K');

            }

            //knights

            if (0 <= x-1 && x-1 < 8 && 0<=y-2 && y-2 < 8) {

                if (figures[y-2][x-1] == 'N')

                result[cur\_req].push\_back('N');

            }

            if (0 <= x+1 && x+1 < 8 && 0<=y-2 && y-2 < 8) {

                if (figures[y-2][x+1] == 'N')

                result[cur\_req].push\_back('N');

            }

            if (0 <= x+2 && x+2 < 8 && 0<=y-1 && y-1 < 8) {

                if (figures[y-1][x+2] == 'N')

                result[cur\_req].push\_back('N');

            }

            if (0 <= x+2 && x+2 < 8 && 0<=y+1 && y+1 < 8) {

                if (figures[y+1][x+2] == 'N')

                result[cur\_req].push\_back('N');

            }

            if (0 <= x+1 && x+1 < 8 && 0<=y+2 && y+2 < 8) {

                if (figures[y+2][x+1] == 'N')

                result[cur\_req].push\_back('N');

            }

            if (0 <= x-1 && x-1 < 8 && 0<=y+2 && y+2 < 8) {

                if (figures[y+2][x-1] == 'N')

                result[cur\_req].push\_back('N');

            }

            if (0 <= x-2 && x-2 < 8 && 0<=y+1 && y+1 < 8) {

                if (figures[y+1][x-2] == 'N')

                result[cur\_req].push\_back('N');

            }

            if (0 <= x-2 && x-2 < 8 && 0<=y-1 && y-1 < 8) {

                if (figures[y-1][x-2] == 'N')

                result[cur\_req].push\_back('N');

            }

            //rooks and first half of queens

            for (short i = 0; i<8; i++) {

                    if (figures[i][x] == 'R')

                    result[cur\_req].push\_back('R');

                    if (figures[i][x] == 'Q')

                    result[cur\_req].push\_back('Q');

            }

            for (short j = 0; j<8; j++) {

                    if (figures[y][j] == 'R')

                    result[cur\_req].push\_back('R');

                    if (figures[y][j] == 'Q')

                    result[cur\_req].push\_back('Q');

            }

            //bishops and second half of queens

            while (1) {

                if (0 <= --x && x < 8 && 0<=--y && y < 8) {

                    if (figures[y][x] == 'B')

                    result[cur\_req].push\_back('B');

                    if (figures[y][x] == 'Q')

                    result[cur\_req].push\_back('Q');

                }

                else {x = req\_x[cur\_req], y = req\_y[cur\_req]; break;}

            }

            while (1) {

                if (0 <= ++x && x < 8 && 0<=--y && y < 8) {

                    if (figures[y][x] == 'B')

                    result[cur\_req].push\_back('B');

                    if (figures[y][x] == 'Q')

                    result[cur\_req].push\_back('Q');

                }

                else {x = req\_x[cur\_req], y = req\_y[cur\_req]; break;}

            }

            while (1) {

                if (0 <= ++x && x < 8 && 0<=++y && y < 8) {

                    if (figures[y][x] == 'B')

                    result[cur\_req].push\_back('B');

                    if (figures[y][x] == 'Q')

                    result[cur\_req].push\_back('Q');

                }

                else {x = req\_x[cur\_req], y = req\_y[cur\_req]; break;}

            }

            while (1) {

                if (0 <= --x && x < 8 && 0<=++y && y < 8) {

                    if (figures[y][x] == 'B')

                    result[cur\_req].push\_back('B');

                    if (figures[y][x] == 'Q')

                    result[cur\_req].push\_back('Q');

                }

                else {x = req\_x[cur\_req], y = req\_y[cur\_req]; break;}

            }

            if(result[cur\_req].empty()) {result[cur\_req].push\_back('O'); }

            else {

                sort(result[cur\_req].begin(), result[cur\_req].end());

                for (short i = 0; i+1<result[cur\_req].size(); i++){

                    if (result[cur\_req][i] != result[cur\_req][i+1]) {cout << result[cur\_req][i]; }

                }

            }

        }

        cout << result[cur\_req][result[cur\_req].size()-1] << '\n';

    }

    return 0;

}

<https://github.com/artificial-intelligence-department/ai_programming_playground/pull/727/files#diff-7b24024bad94515296b4117ec623ed58bf65a0cf09fbb465f7da076839a8f048>

Завдання №8 Class Practice Task

#include <fstream>

#include <iostream>

#include <vector>

#include <string>

using namespace std;

enum FileOpResult {

    Success,

    Failure

};

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

    ofstream out(name, ios::binary);

    if (!out) {out.close(); return Failure;}

    out << content;

    if (out.ios::good()) {out.close(); return Success;}

    else {out.close(); return Failure;}

}

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

    ifstream in(file\_from, ios::binary);

    ofstream out(file\_to, ios::binary);

    if ((!out) || (!in)) {return Failure;}

    string tmp;

    while (getline(in, tmp)) {out << tmp << '\n';}

    if (out.ios::good() && in.ios::eof()) {return Success;}

    else {return Failure;}

}

int main() {

    char name[9] = "test.txt", c[] = "copied.cpp";

    vector<char> content;

    char temp;

    temp = cin.get();

    content.push\_back(temp);

    int i=1;

    while(1) {

        temp = cin.get();

        content.push\_back(temp);

        if ((content[i] == '\n')&&(content[i-1] == '\n')) {break;}

        ++i;

    }

    char \*to\_write = new char[content.size() + 1];

    for (i=0; i<content.size(); i++) {to\_write[i] = content[i]; }

    to\_write[i] = '\0';

    if (write\_to\_file(name, to\_write)) { std::cout << "Failure"; }

    else {std::cout << "Success"; }

    if (copy\_file(name, c)) { std::cout << "Failure"; }

    else {std::cout << "Success"; }

    return 0;

}

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/727/files#diff-295cfcd8c2985410a90ed88292ab576e9d4bf9135a31b4f504b4dbe8d03d3513

Завдання №9.1 Self Practice Task 0213

#include <iostream>

#include <vector>

int main() {

    short n, m;

    std::cin >> n >> m;

    if (!n || !m) {std::cout << "0";}

    else {

        std::vector<short> number, max;

        max.resize(m), number.resize(m);

        max = {0};

        for (short a = 0; a<n; a++) {

            for (short i = 0; i<m; i++) {

                std::cin >> number[i];

                if (max[i] < number[i]) max[i] = number[i]; }

        }

        int sum = 0;

        for (short i = 0; i < m; i++) {sum += max[i];}

        std::cout << sum;

    }

    return 0;

}

<https://github.com/artificial-intelligence-department/ai_programming_playground/pull/727/files#diff-61d1d06581403dcf36abd709229666e83dffe00686d541b19eae39b4b2e7e7c6>

Завдання №9.2 Self Practice Task 0754

#include <iostream>

#include <vector>

#include <algorithm>

int main() {

    long time = 0, n;

    long mx\_x = 0, mx\_y = 0;

    std::cin >> n;

    std::vector<long> x(n), y(n);

    for (int i = 0; i < n; i++) { std::cin >> x[i] >> y[i]; }

    mx\_x = \*max\_element(begin(x), end(x));

    mx\_y = \*max\_element(begin(y), end(y));

    for (int i = 0; i<n; i++) {

        time += mx\_x - x[i];

        time += mx\_y - y[i];

    }

    std::cout << time;

    return 0;

}

<https://github.com/artificial-intelligence-department/ai_programming_playground/pull/727/files#diff-7ad84f6ac3f38463bf949baac7295d6fd17242f4439b41009c89562dbf11b592>

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

Завдання №3 VNS Lab 6

Ввід: Built like a spreadsheet project tables give you a live canvas to filter sort and.

Вивід: aaaaaaaBbccddeeeeeeeeefghiiiiijklllllnnoooopprrrrssssstttttttuuvvvy.

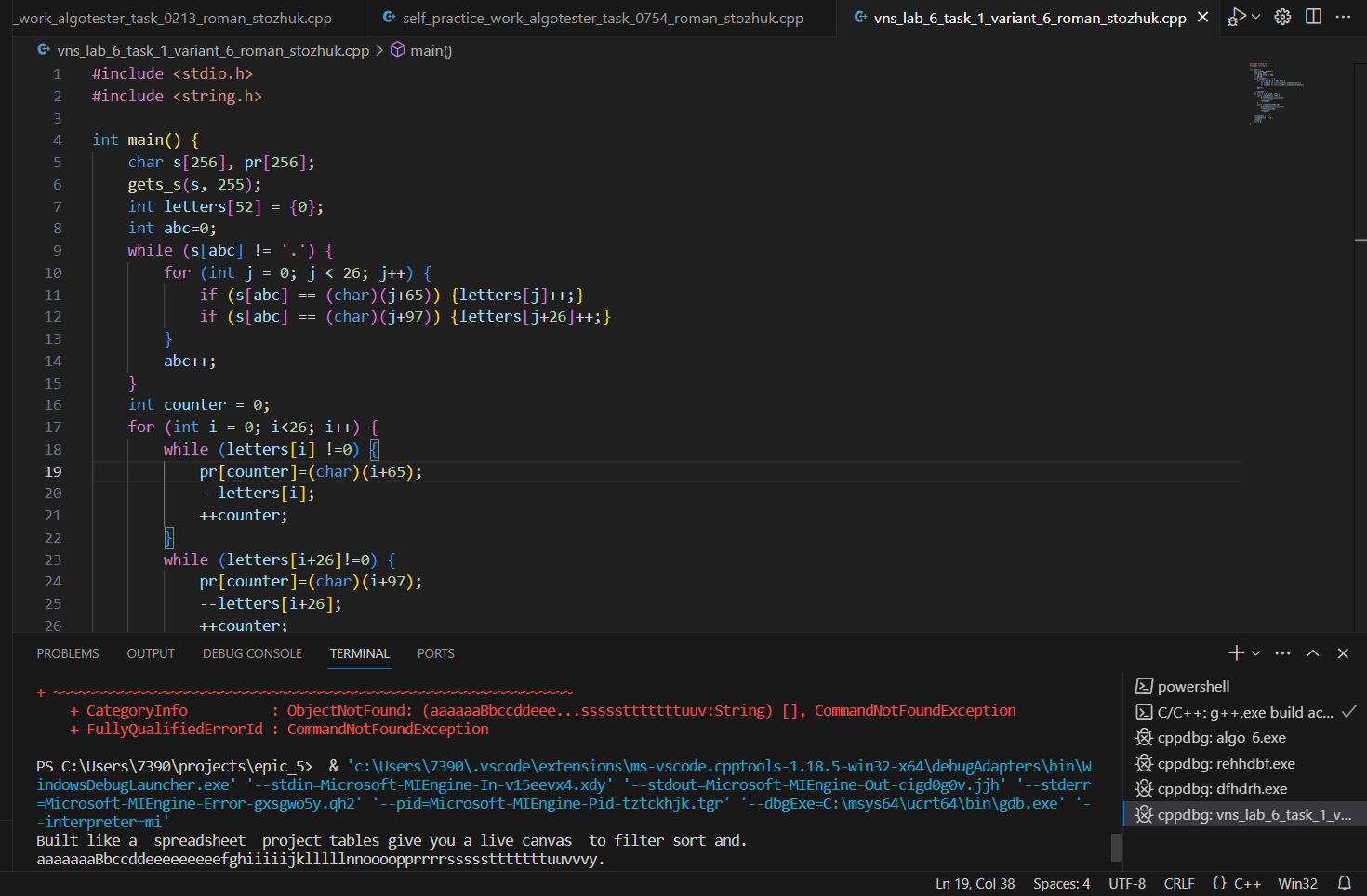
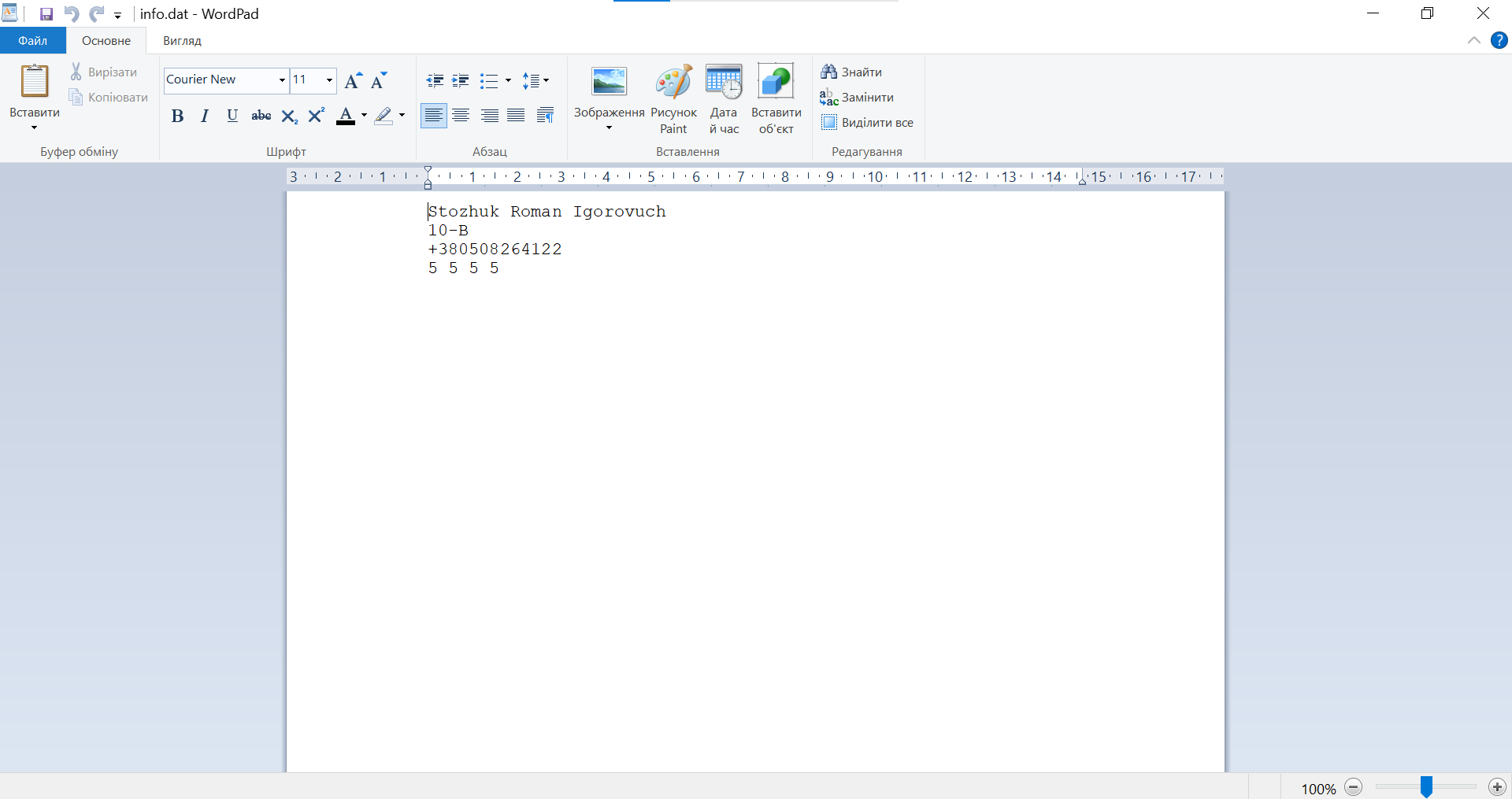


Рисунок 2Результат виконання завдання №3

Час затрачений на виконання завдання: 30хв

Завдання №4 VNS Lab 8



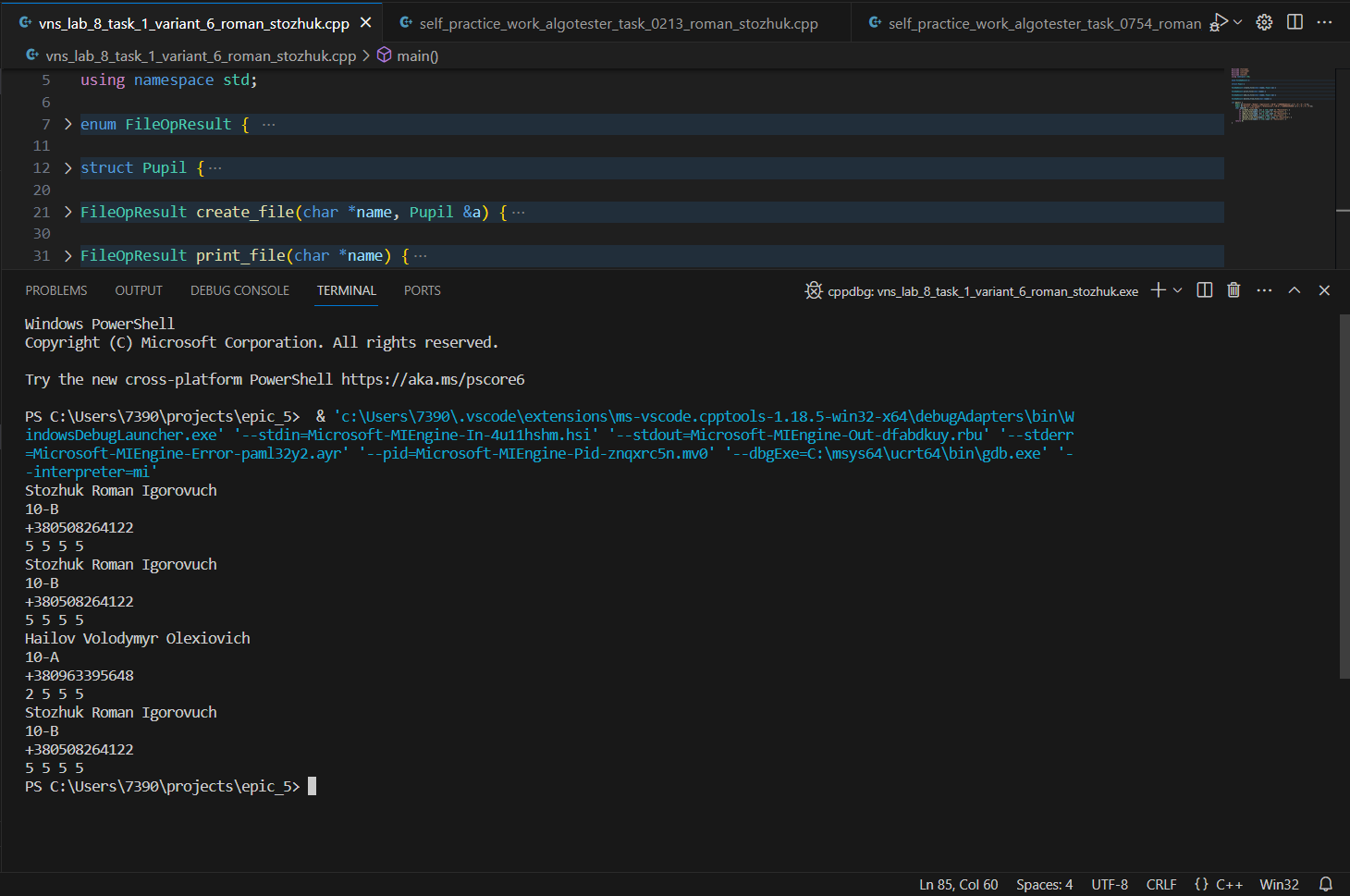
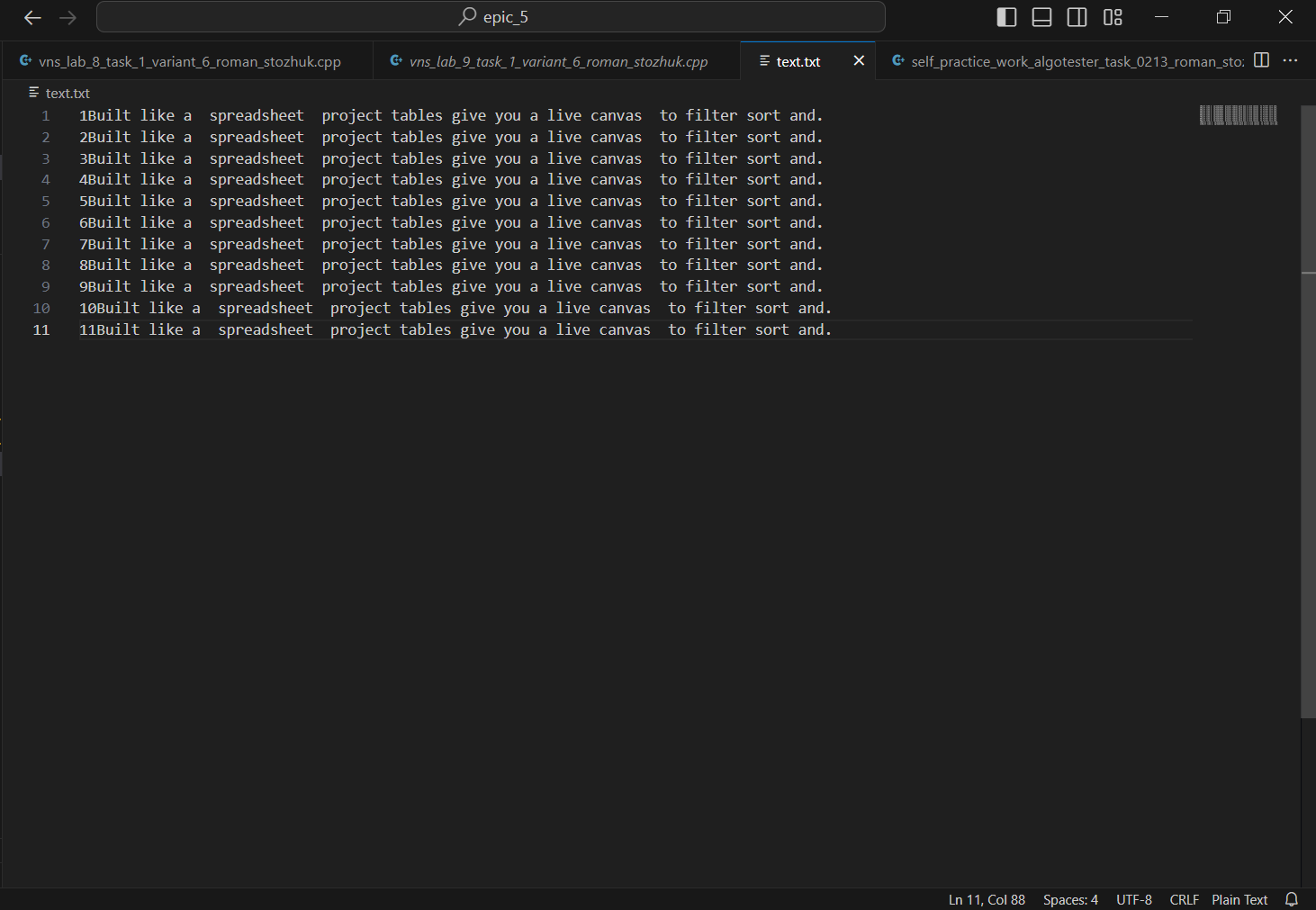
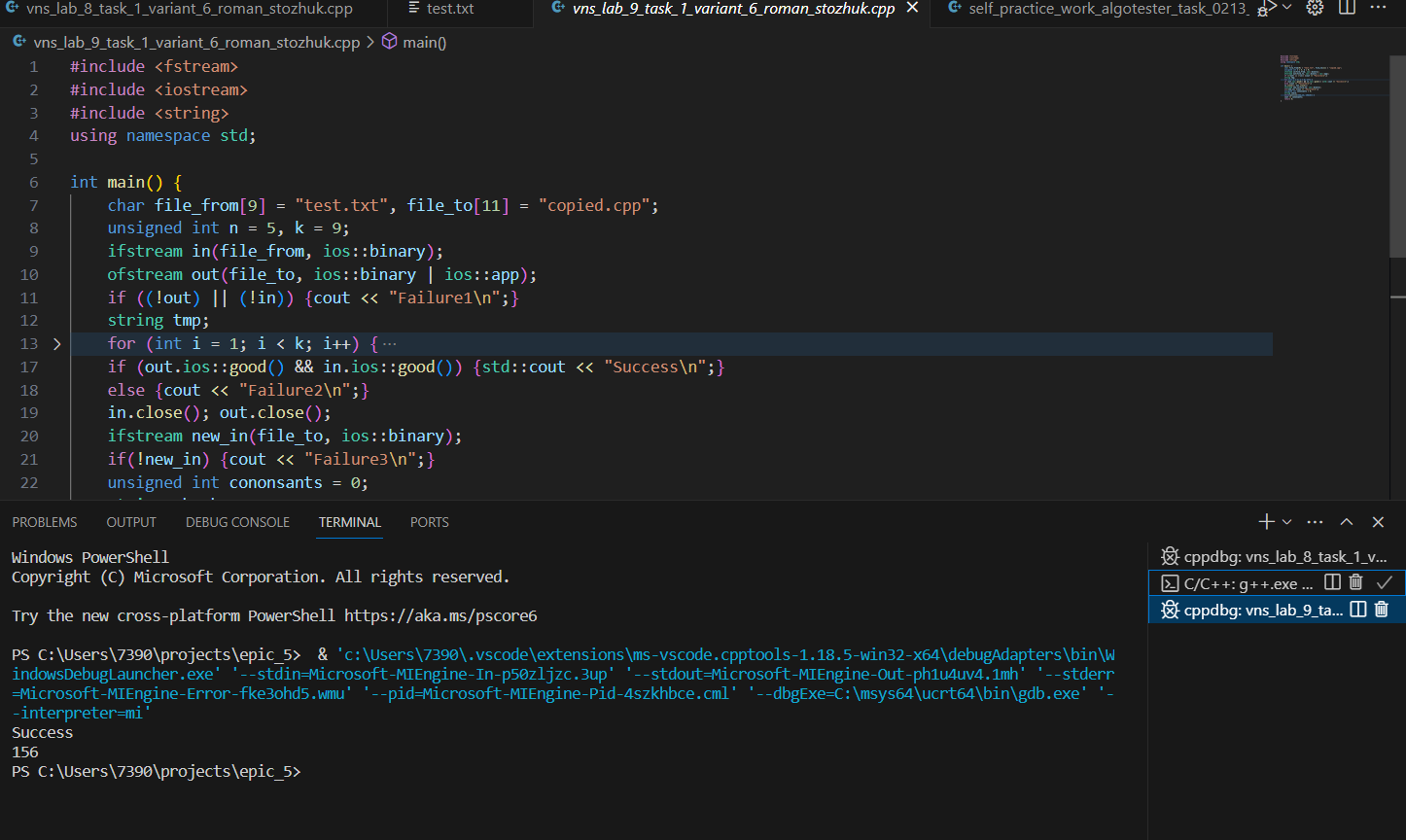


Рисунок 3Результат виконання завдання №4

Час затрачений на виконання завдання: 2год30хв

Завдання №5 VNS Lab 9





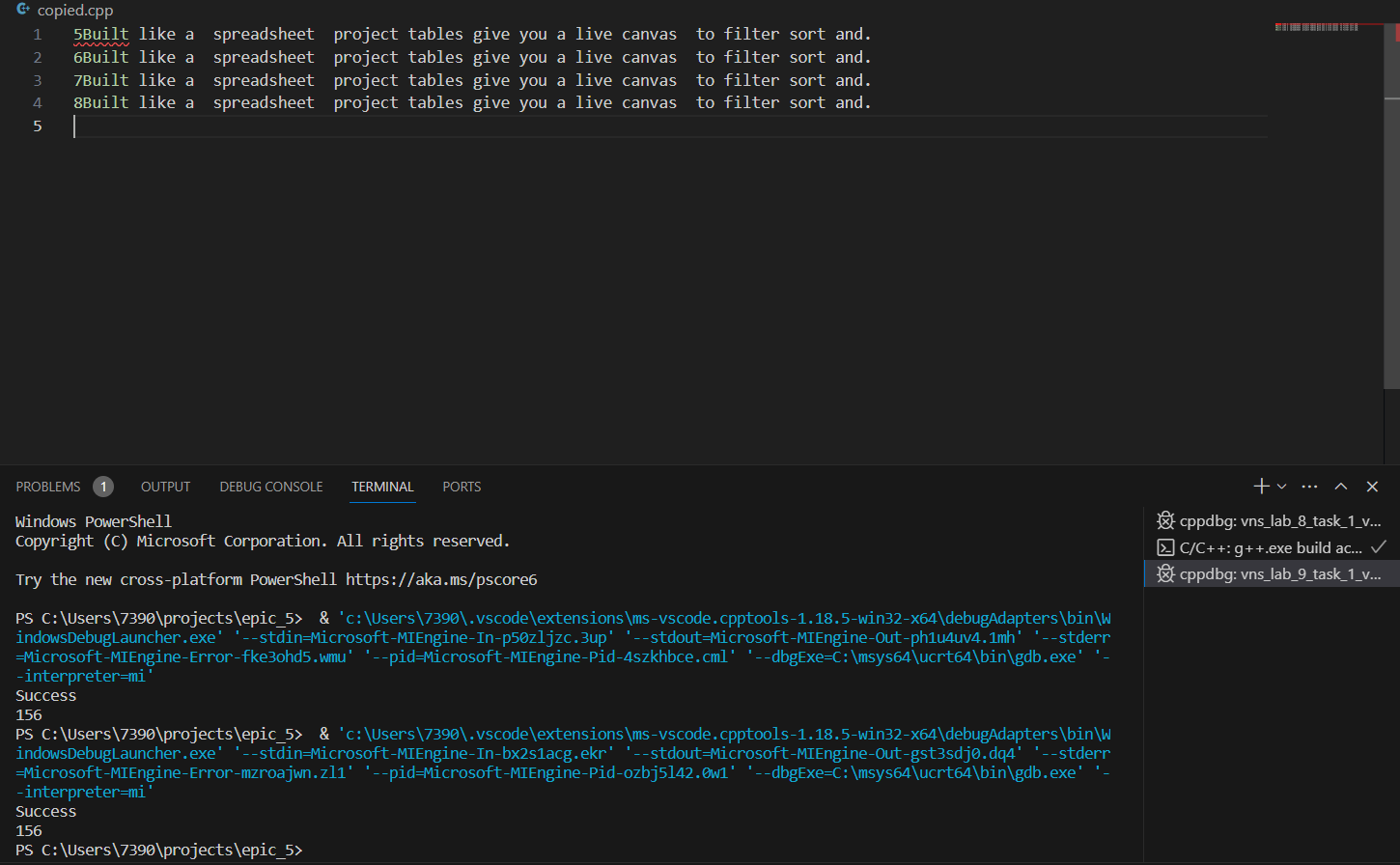


Рисунок 4Результат виконання завдання №5

Час затрачений на виконання завдання: 30хв

Завдання №6 Algotester Lab 4

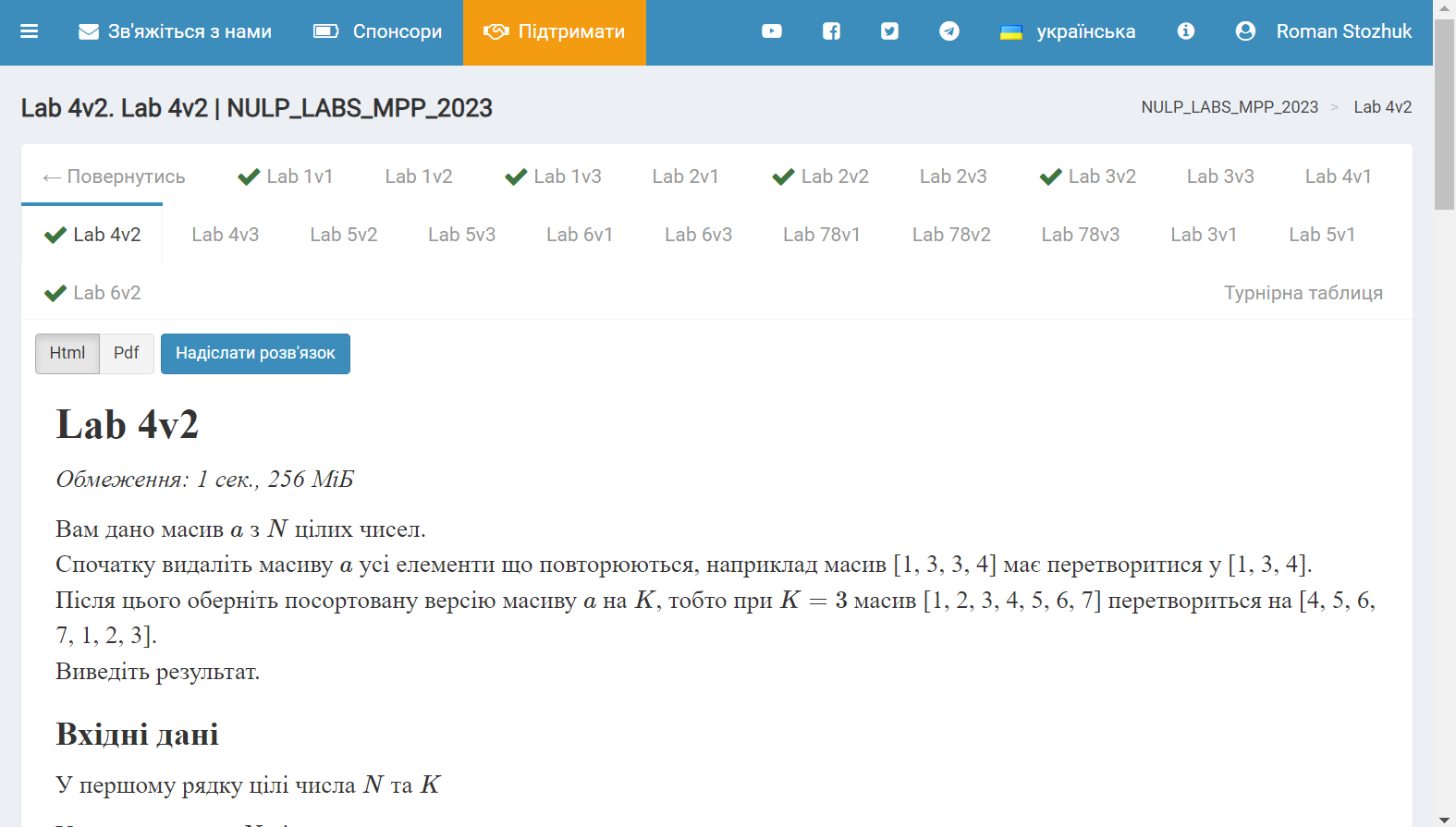


Рисунок 5Результат виконання завдання №6

Час затрачений на виконання завдання: 25хв

Завдання №7 Algotester Lab 6

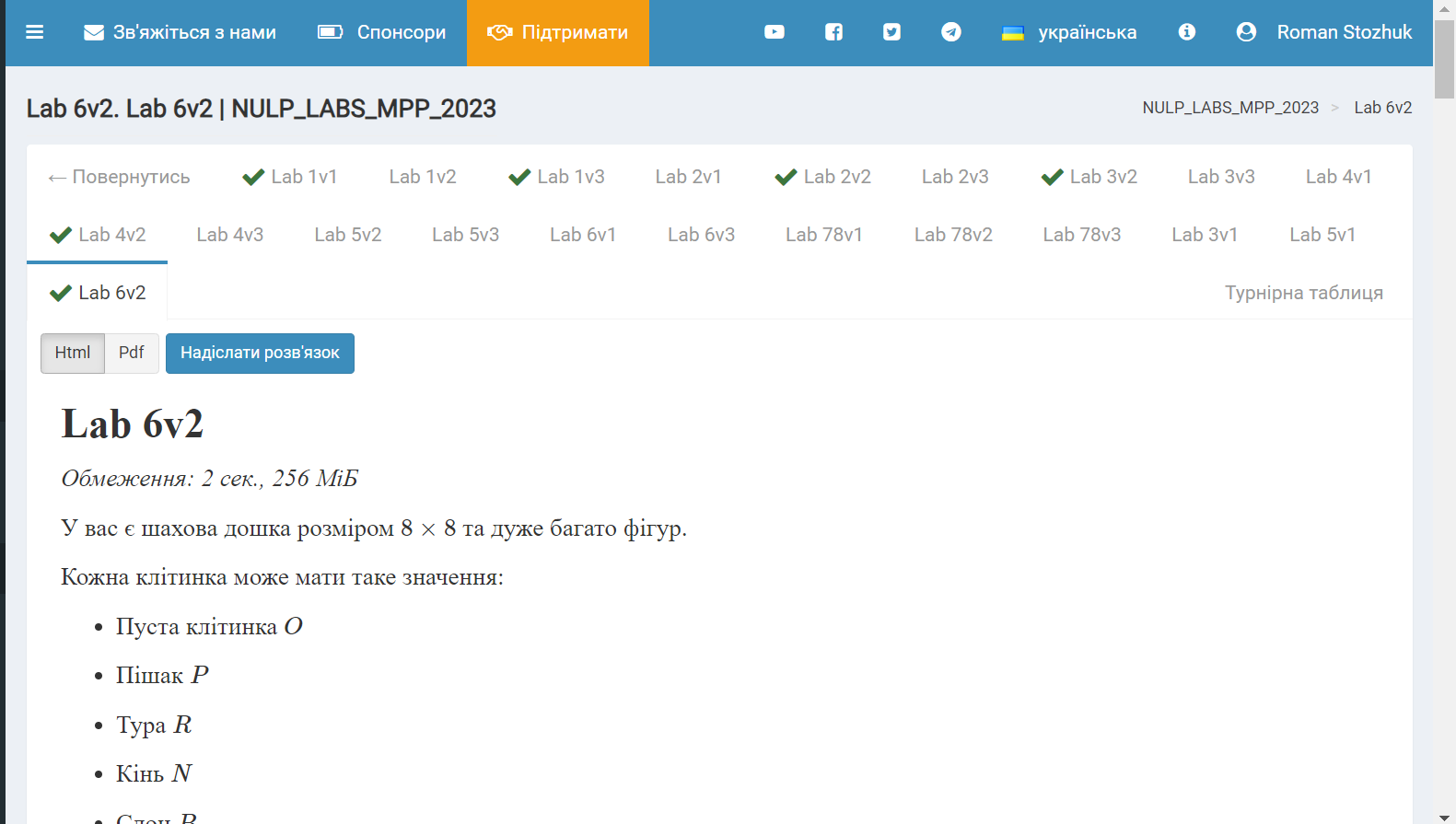
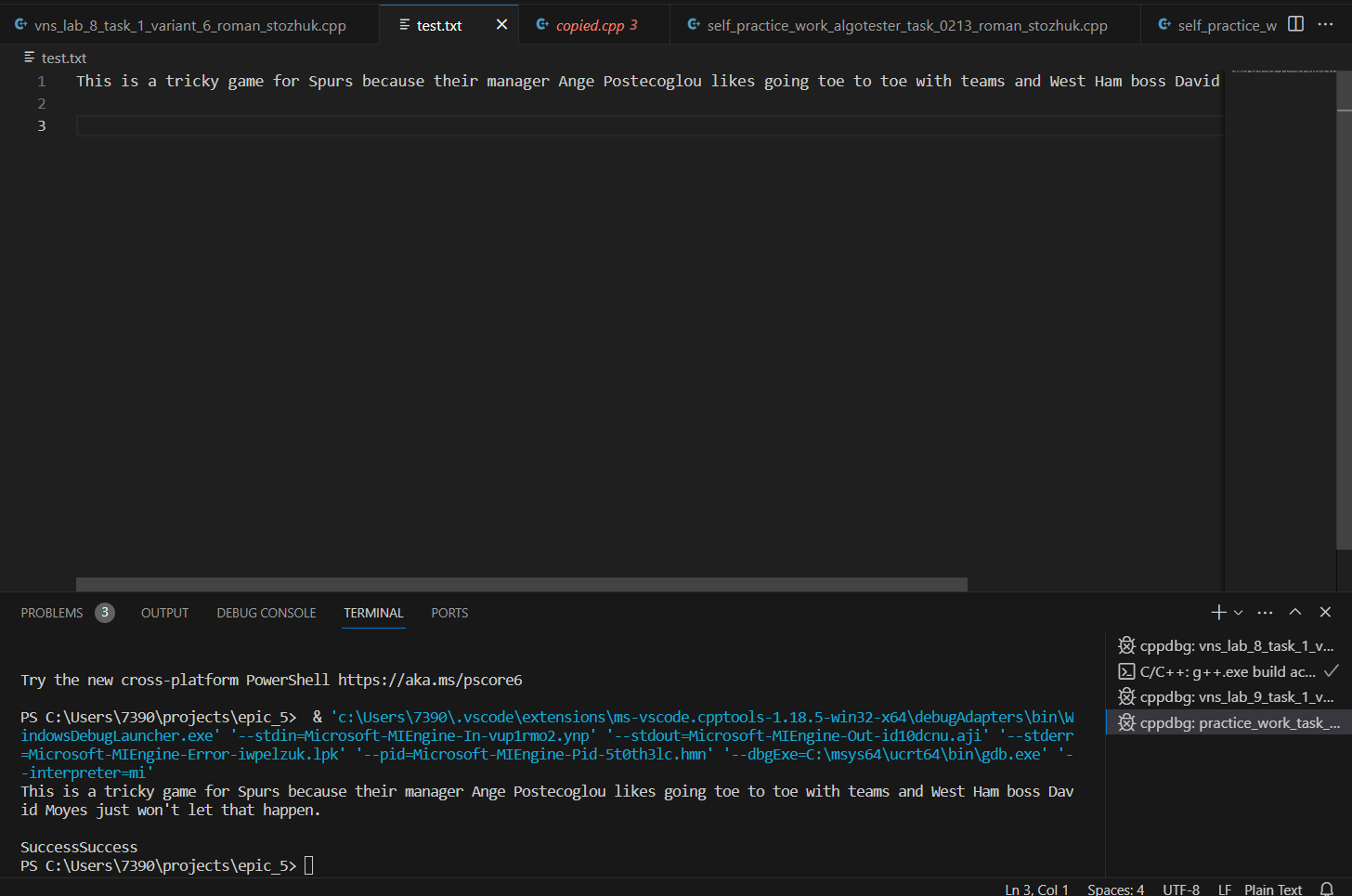


Рисунок 6Результат виконання завдання №7

Час затрачений на виконання завдання: 3год

Завдання №8 Class Practice Task



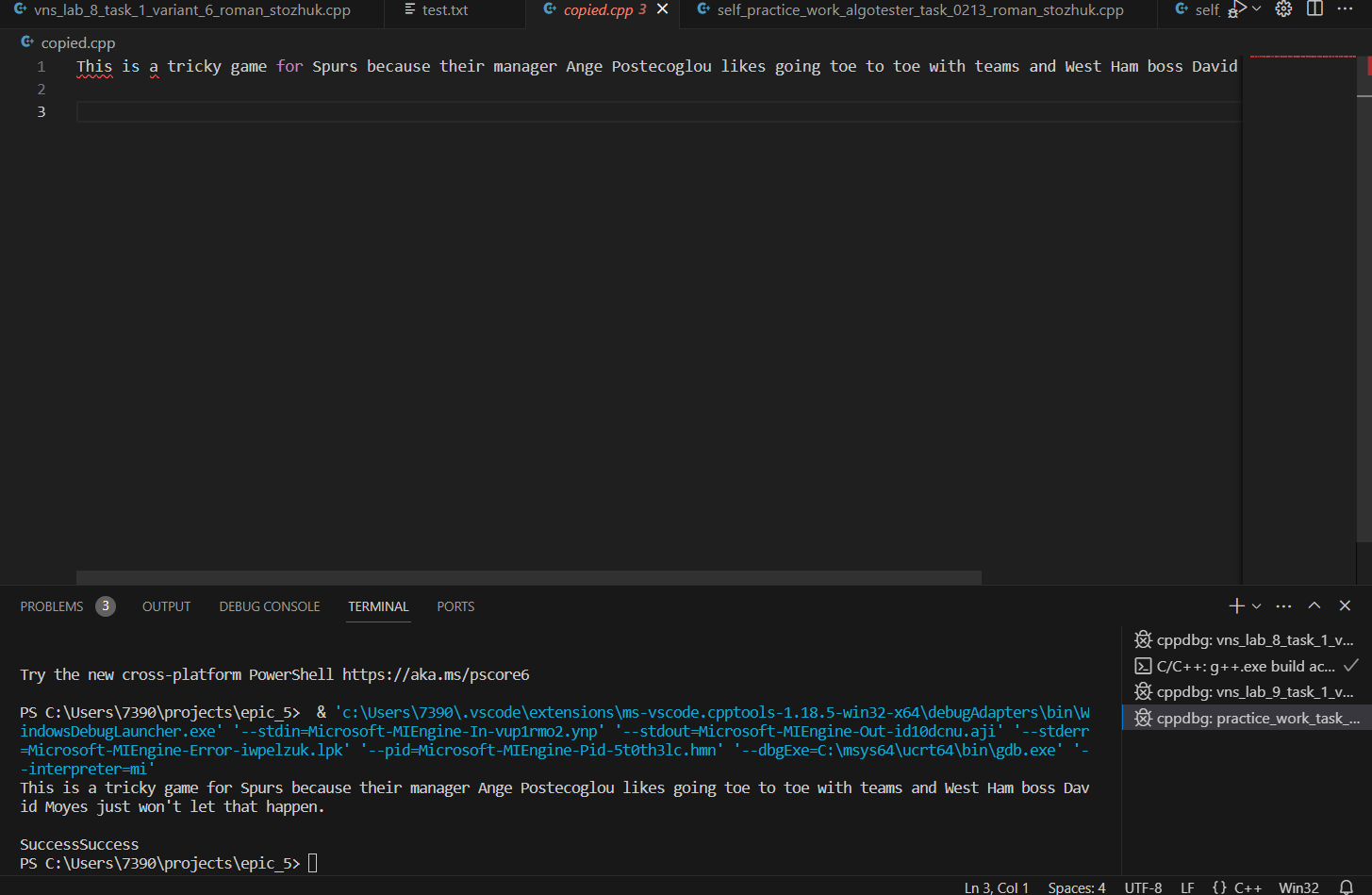


Рисунок 7Результат виконання завдання №8.1

Час затрачений на виконання завдання: 100хв

Завдання №9.1 Self Practice Task 0213

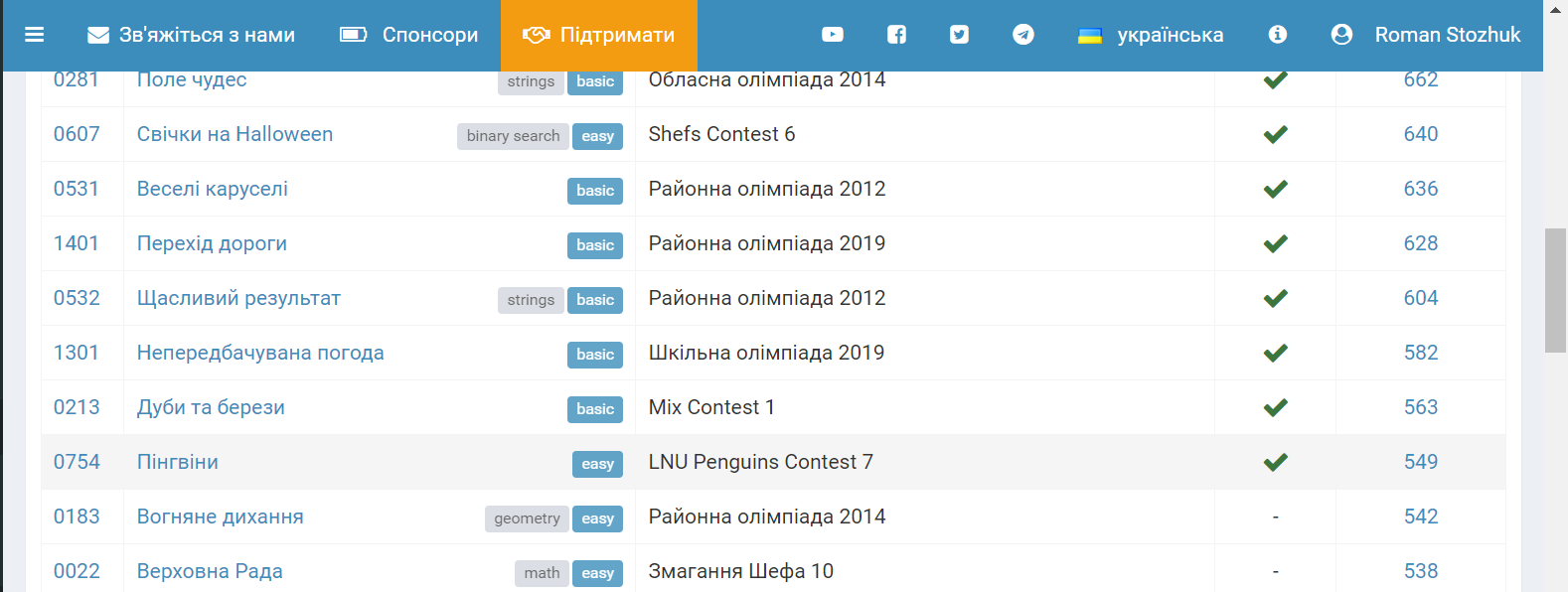
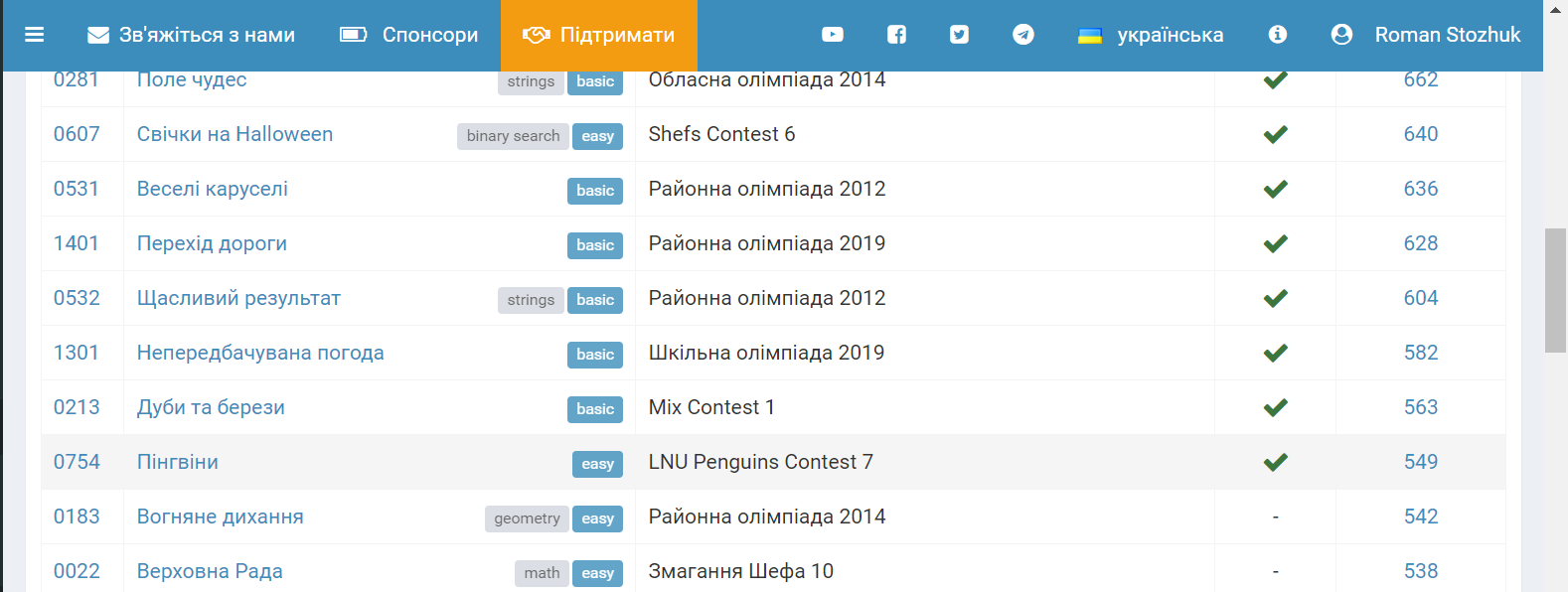


Рисунок 8Результат виконання завдання №8.2

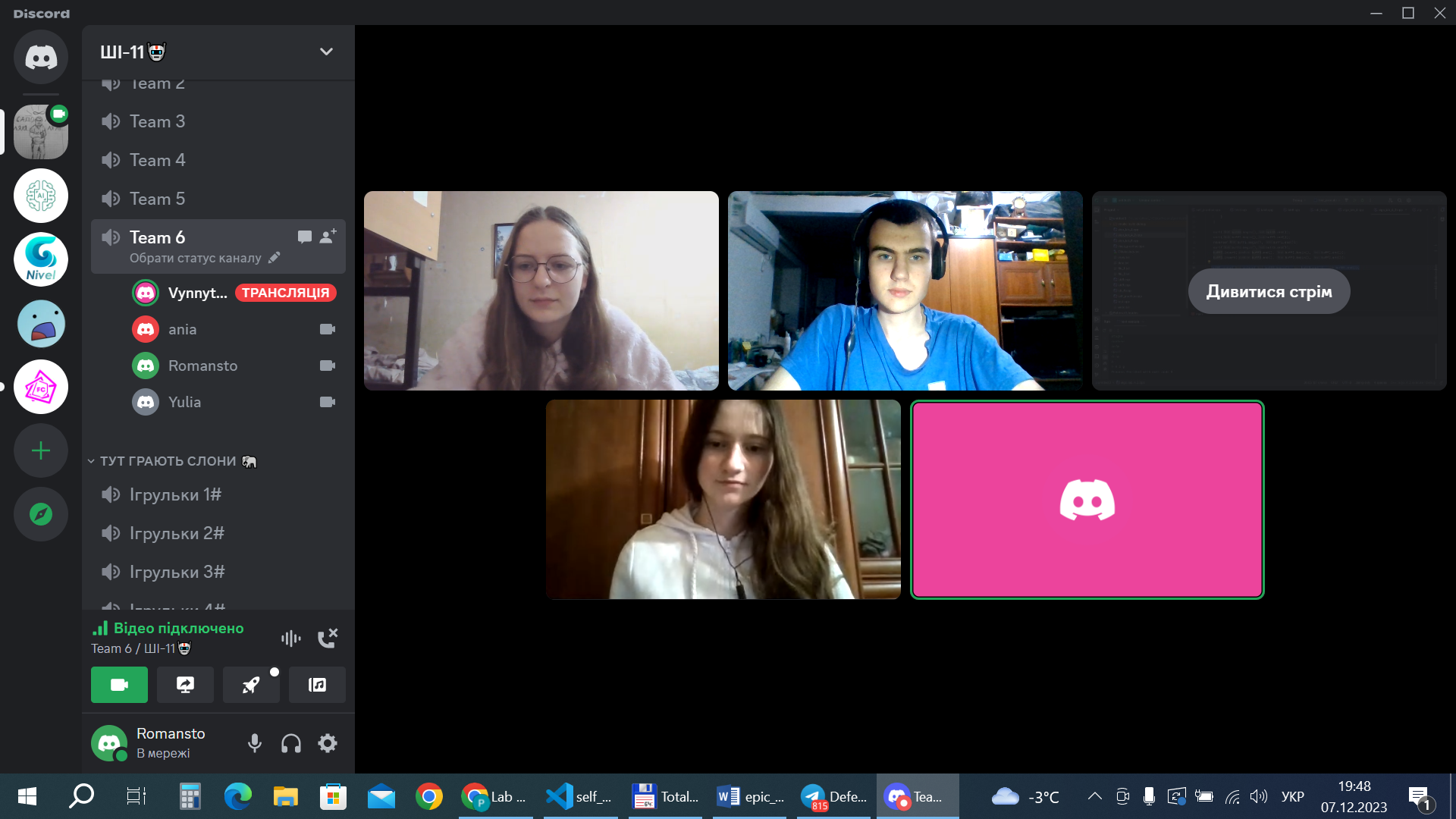
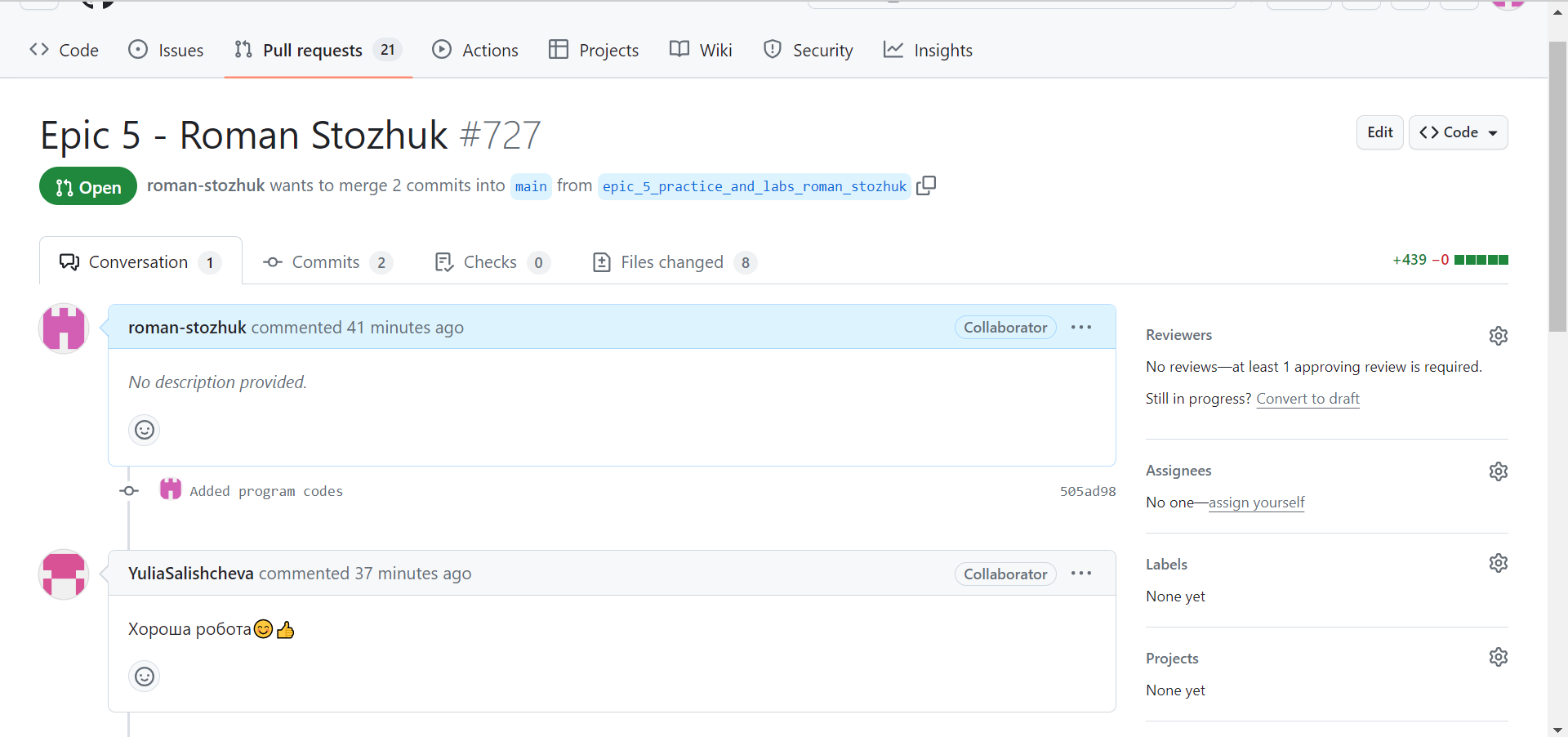
Час затрачений на виконання завдання: 60хв

Завдання №9.2 Self Practice Task 0754



Час затрачений на виконання завдання: 60хв

## **6. Кооперація з командою:**

* 
* 

# **Висновки:**

Протягом п’ятого епіку я навчився працювати з файлами та бібліотеками