Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**VNS Lab 6, VNS Lab 8, VNS Lab9, Algotester Lab4, Algotester Lab6 та Practice Task #5**

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

***з розділу***: “Epic 5. 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

Шевченко Данііл Вікторович

# **Тема роботи:**

Робота з різновидами стрічок, такими як строки C-Style та std::string, методи обробки та перетворення стрічок, вводу стрічок з клавіатури. Робота з файлами, створення, відкриття та перегляд файлів, імпортування в файл та експортування з файлу, копіювання даних з файлу в файл.

# **Мета роботи:**

Ознайомитись з двома видами строк у мові С++: строками С-style(масиви типу сhar) та std::string, зрозуміти їх відмінності та спільні моменти, навчитись працювати з ввідом стрічок. Ознайомитись з методами обробки, перегляду та редагування файлів

Завдання:

* Daniil Shevchenko - Epic 5 Task 1 - Theory Education Activities
* Daniil Shevchenko - Epic 5 Task 2 - Requirements management (understand tasks) and design activities (draw flow diagrams and estimate tasks 3-9)
* Daniil Shevchenko - Epic 5 Task 3 - Lab# programming: VNS Lab 6
* Daniil Shevchenko - Epic 5 Task 4 - Lab# programming: VNS Lab 8
* Daniil Shevchenko - Epic 5 Task 5 - Lab# programming: VNS Lab 9
* Daniil Shevchenko - Epic 5 Task 6 - Lab# programming: Algotester Lab 4
* Daniil Shevchebnko - Epic 5 Task 7 - Lab# programming: Algotester Lab 6
* Daniil Shevchenko - Epic 5 Task 8 - Practice# programming: Class Practice Task
* Daniil Shevchenko - Epic 5 Task 9  - Practice# programming:  Self Practice Task
* Daniil Shevchenko - Epic 5 Task 10  - Result Documentation Report and Outcomes Placement Activities (Docs and Programs on GitHub)
* Daniil Shevchenko - Epic 5 Task 11 - Results Evaluation and Release

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Строки C-Style
* Тема №2: Строки std::string
* Тема №3: Обробка вводу стрічок, функція getline
* Тема №4: Бібліотека fstream та використання її для роботи з файлами
* Тема №5: Інші бібліотеки файлової обробки

1. Індивідуальний план опрацювання теорії:

* Тема №1: Строки C-Style
* Джерела Інформації
  + - Практика в НУ “ЛП”
    - Курси https://acode.com.ua/
  + Що опрацьовано:
    - Ініціалізація строк типу C-Style
    - Усвідомлення структуру строк C-Style у вигляді масиву елементів типу char
  + Статус: Ознайомлений
  + Початок опрацювання теми: 08.11.23
  + Звершення опрацювання теми: 26.11.23
* Тема №2: Строки std::string
  + Джерела Інформації
    - Практика в НУ “ЛП”
    - Курси https://acode.com.ua/
  + Що опрацьовано:
    - Бібліотека <string> та її застосування
    - Різниця між строками C-Style та std::string
    - Методи обробки std::string
  + Статус: Ознайомлений
  + Початок опрацювання теми: 08.11.23
  + Звершення опрацювання теми: 26.11.23

Тема №3: Обробка вводу стрічок, функція getline

* Джерела Інформації
  + - Практика в НУ “ЛП”
    - Курси https://acode.com.ua/
  + Що опрацьовано:
    - Застосування функції getline
    - Відмінності getline від cin
    - Використання функції cin.ignore() в парі з cin та getline
  + Статус: Ознайомлений
  + Початок опрацювання теми: 08.11.23
* Звершення опрацювання теми: 26.11.23
* Тема №4: Бібліотека fstream та використання її для роботи з файлами
* Джерела Інформації

Практика в НУ “ЛП”

* + - Курси https://acode.com.ua/
  + Що опрацьовано:
    - Використання бібліотеки fstream для запису у файл
    - Використання бібліотеки fstream для перегляду та імпорту з файлу
    - Поняття про різні режими перегляду файлів
  + Статус: Ознайомлений
  + Початок опрацювання теми: 08.11.23
* Звершення опрацювання теми: 26.11.23
* Тема №5: Інші бібліотеки файлової обробки
* Джерела Інформації
  + - Практика в НУ “ЛП”
    - Курси https://acode.com.ua/
  + Що опрацьовано:
    - Алгоритми обробки файлів мови С, що були актуальні в колишньому
  + Статус: Ознайомлений
  + Початок опрацювання теми: 08.11.23
* Звершення опрацювання теми: 26.11.23

# **Виконання роботи:**

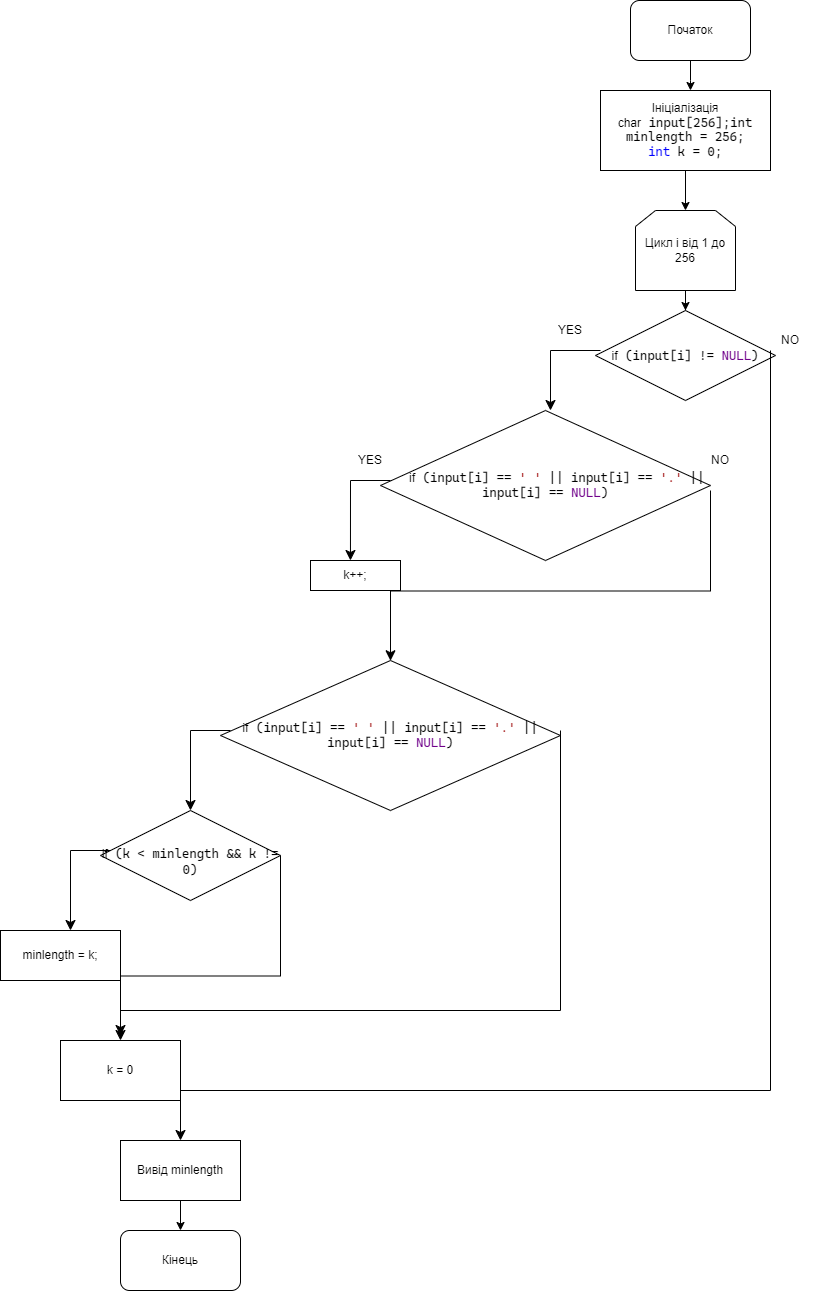
**VNS LAB 6**

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

Завдання №1 VNS Lab6

Варіант 23

23. Для рядка знайти довжину найкоротшого слова.**2. Дизайн програми**



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

#include <iostream>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

using namespace std;

int main()

{

char input[256];

gets\_s(input);

int minlength = 256;

int k = 0;

for (int i = 0; i < 256; i++)

{

if (input[i] != NULL)

{

if (input[i] != ' ' && input[i] != '.' && input[i] != NULL)

{

k++;

}

if (input[i] == ' ' || input[i] == '.' || input[i] == NULL)

{

if (k < minlength && k != 0)

{

minlength = k;

}

k = 0;

}

}

}

int maxlength = 0;

int j = 0;

for (int i = 0; i < 256; i++)

{

if (input[i] != NULL) {

if (input[i] != ' ' && input[i] != '.' && input[i] != NULL)

{

j++;

}

if (input[i] == ' ' || input[i] == '.' || input[i] == NULL)

{

if (j > maxlength)

{

maxlength = j;

}

j = 0;

}

}

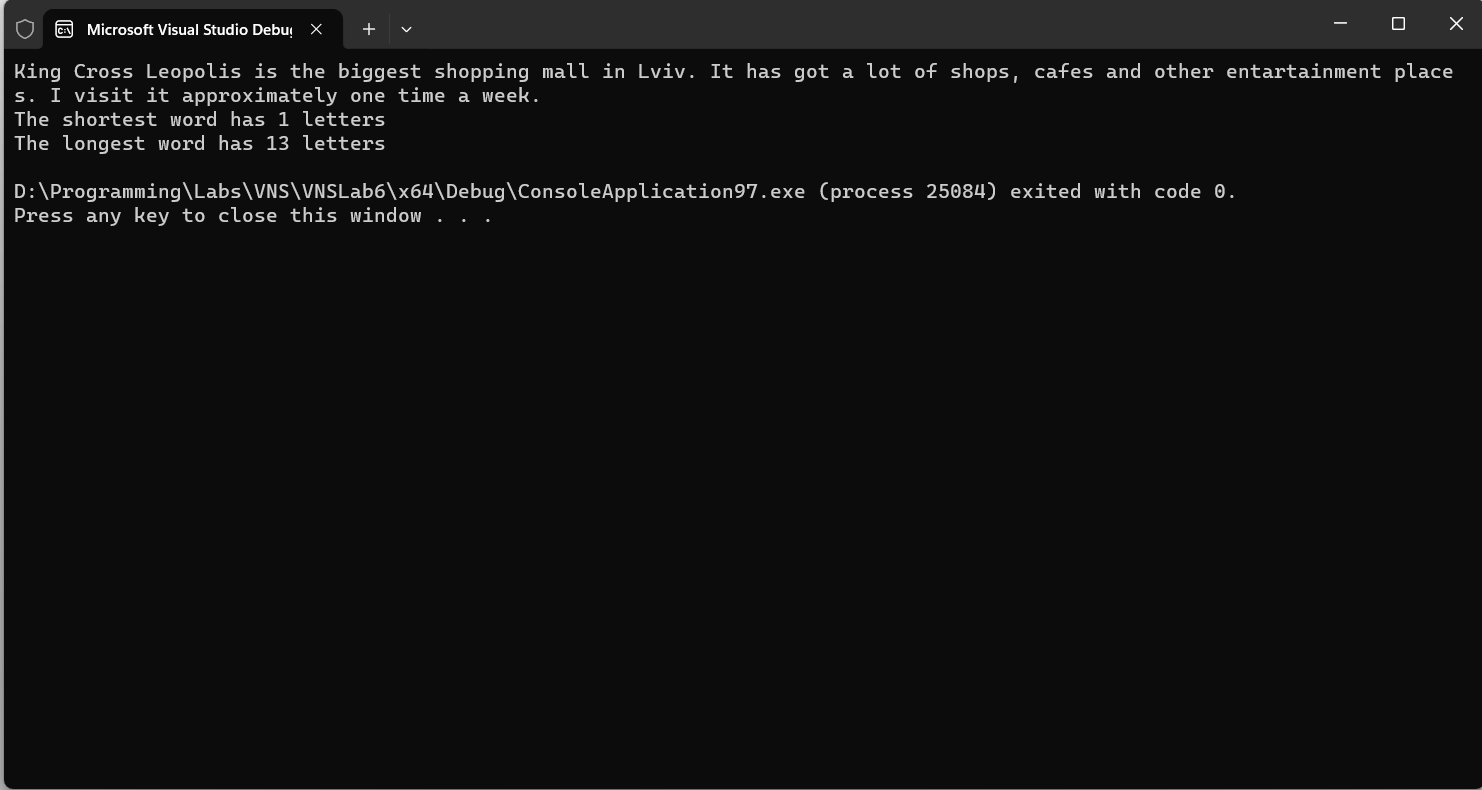
}

cout << "The shortest word has " << minlength << " letters" << endl;

cout << "The longest word has " << maxlength << " letters" << endl;

}

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



Час затрачений на виконання завдання – 45 хв

**VNS LAB 8**

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

Завдання №2 VNS Lab8

Варіант 23

Сформувати файл із елементів, заданої у варіанті структури,

роздрукувати його вмістиме, виконати знищення й додавання елементів у

відповідності зі своїм варіантом.

## 23. Структура "Стадіон":

## - назва;

## - рік будівлі;

## - кількість площадок;

## - види спорту.

## Знищити всі елементи, у яких рік будівлі меншее заданого, додати 2 елементи

## перед елементом із зазначеним номером.

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

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

using namespace std;

struct Stadium

{

string name;

int year;

int playgrounds;

string sports;

};

ofstream newfile1("TextLab1.txt");

ofstream newfile2("TextLab2.txt");

void getstruct(Stadium array[], int number)

{

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

{

cout << "Name = ";

getline(cin, array[i].name);

cout << "Building year = ";

cin >> array[i].year;

cout << "The number of playgrounds = ";

cin >> array[i].playgrounds;

cin.ignore(30000, '\n');

cout << "Available sports are ";

getline(cin, array[i].sports);

}

}

void printstruct(Stadium array[], int number)

{

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

{

newfile1 << "Stadium # " << i + 1 << ": " << endl;

newfile1 << "Name: " << array[i].name << endl;

newfile1 << "Building year: " << array[i].year << endl;

newfile1 << "Number of playgrounds: " << array[i].playgrounds << endl;

newfile1 << "Available sports: " << array[i].sports << endl;

newfile1 << endl;

}

}

void printstruct2(vector<Stadium> array, int number)

{

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

{

newfile2 << "Stadium # " << i + 1 << ": " << endl;

newfile2 << "Name: " << array[i].name << endl;

newfile2 << "Building year: " << array[i].year << endl;

newfile2 << "Number of playgrounds: " << array[i].playgrounds << endl;

newfile2 << "Available sports: " << array[i].sports << endl;

newfile2 << endl;

}

}

int main()

{

if (!newfile1 )

{

cerr << "Can't open file for writing in";

exit(1);

}

if (!newfile2)

{

cerr << "Can't open file for writing in";

exit(1);

}

int N;

cout << "Enter the number of stadiums: ";

cin >> N;

cin.ignore(30000, '\n');

Stadium\* array = new Stadium[N];

getstruct(array, N);

printstruct(array, N);

int year;

cout << "Enter the year: ";

cin >> year;

cin.ignore(30000, '\n');

vector<Stadium> newarray;

newarray.resize(N);

int j = 0;

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

{

if (array[i].year >= year)

{

newarray[j] = array[i];

j++;

}

}

newarray.resize(j);

Stadium additional1;

Stadium additional2;

cout << "Enter the additional stadiums: " << endl;

cout << "Name = ";

getline(cin, additional1.name);

cout << "Building year = ";

cin >> additional1.year;

cout << "The number of playgrounds = ";

cin >> additional1.playgrounds;

cin.ignore(30000, '\n');

cout << "Available sports are ";

getline(cin, additional1.sports);

cout << "Name = ";

getline(cin, additional2.name);

cout << "Building year = ";

cin >> additional2.year;

cout << "The number of playgrounds = ";

cin >> additional2.playgrounds;

cin.ignore(30000, '\n');

cout << "Available sports are ";

getline(cin, additional2.sports);

int number;

do

{

cout << "Enter the number";

cin >> number;

}

while (number > j || number <= 0);

cin.ignore(30000, '\n');

vector<Stadium> newarray2;

newarray2.resize(j + 2);

for (int i = 0; i < number - 1; i++)

{

newarray2[i].name = newarray[i].name;

newarray2[i].year = newarray[i].year;

newarray2[i].playgrounds = newarray[i].playgrounds;

newarray2[i].sports = newarray[i].sports;

}

newarray2[number - 1].name = additional1.name;

newarray2[number - 1].year = additional1.year;

newarray2[number - 1].playgrounds = additional1.playgrounds;

newarray2[number - 1].sports = additional1.sports;

newarray2[number].name = additional2.name;

newarray2[number].year = additional2.year;

newarray2[number].playgrounds = additional2.playgrounds;

newarray2[number].sports = additional2.sports;

for (int i = number + 1; i < j + 2; i++)

{

newarray2[i].name = newarray[i - 2].name;

newarray2[i].year = newarray[i - 2].year;

newarray2[i].playgrounds = newarray[i - 2].playgrounds;

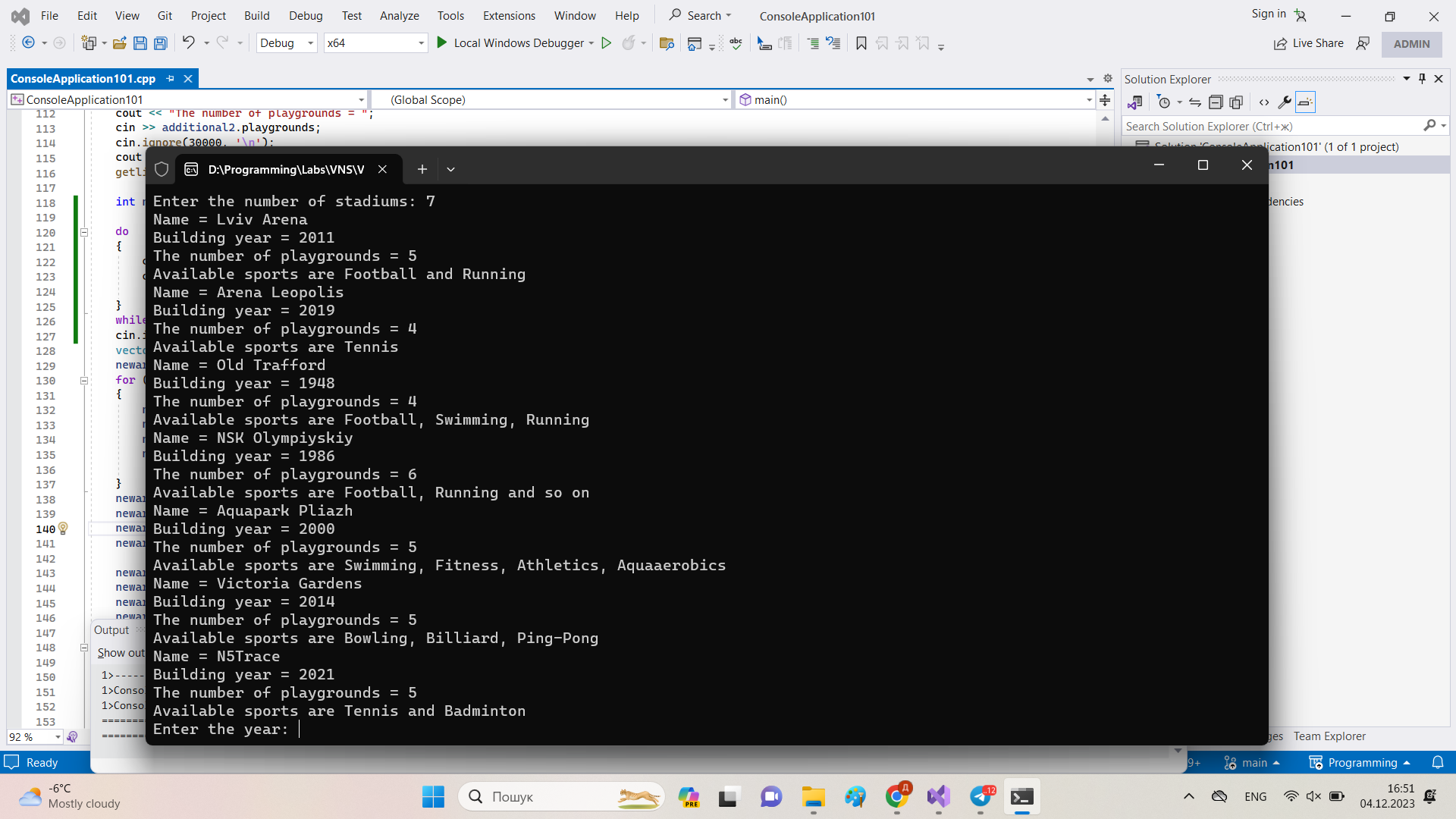
newarray2[i].sports = newarray[i - 2].sports;

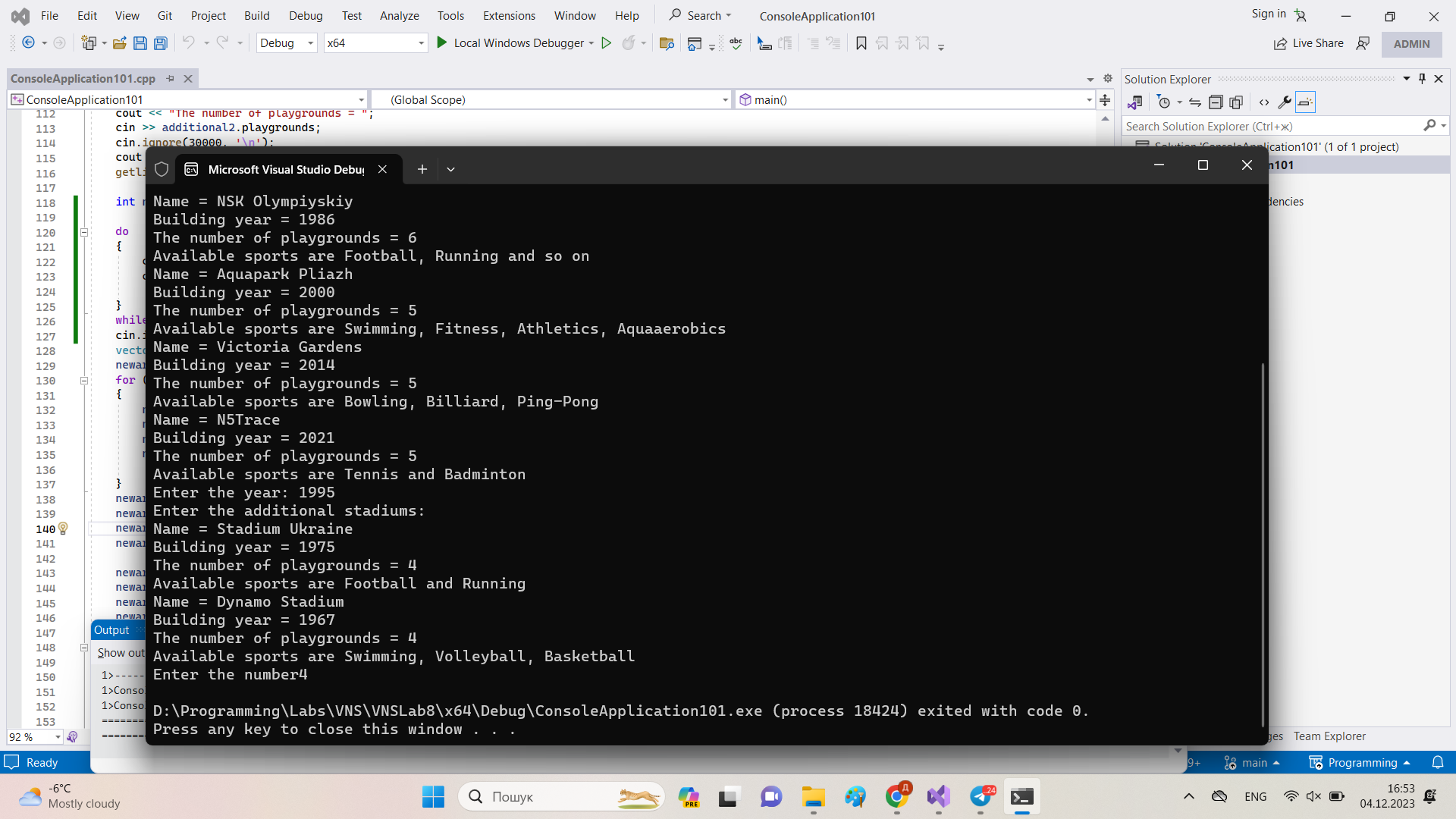
}

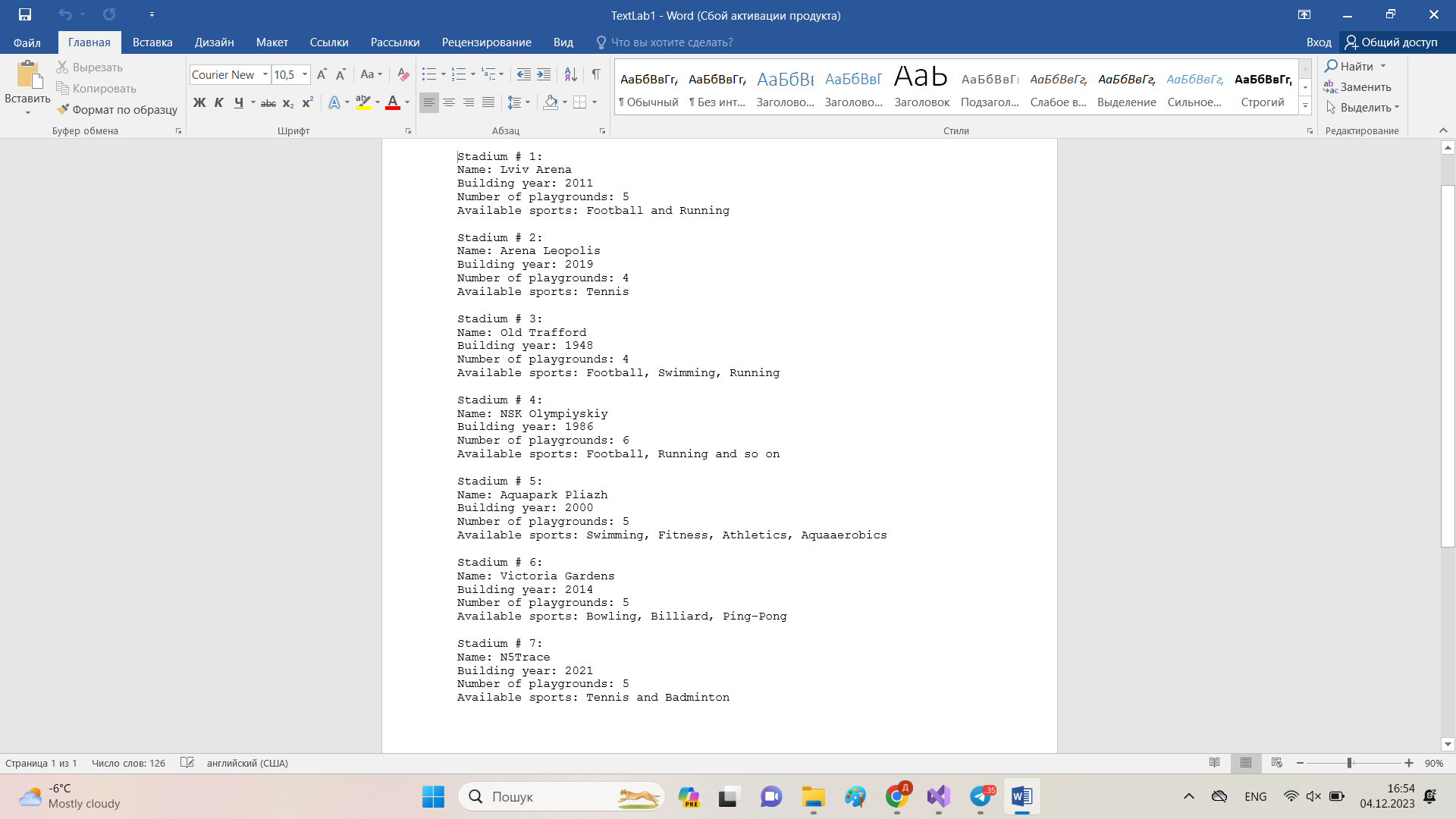
printstruct2(newarray2, j+2);

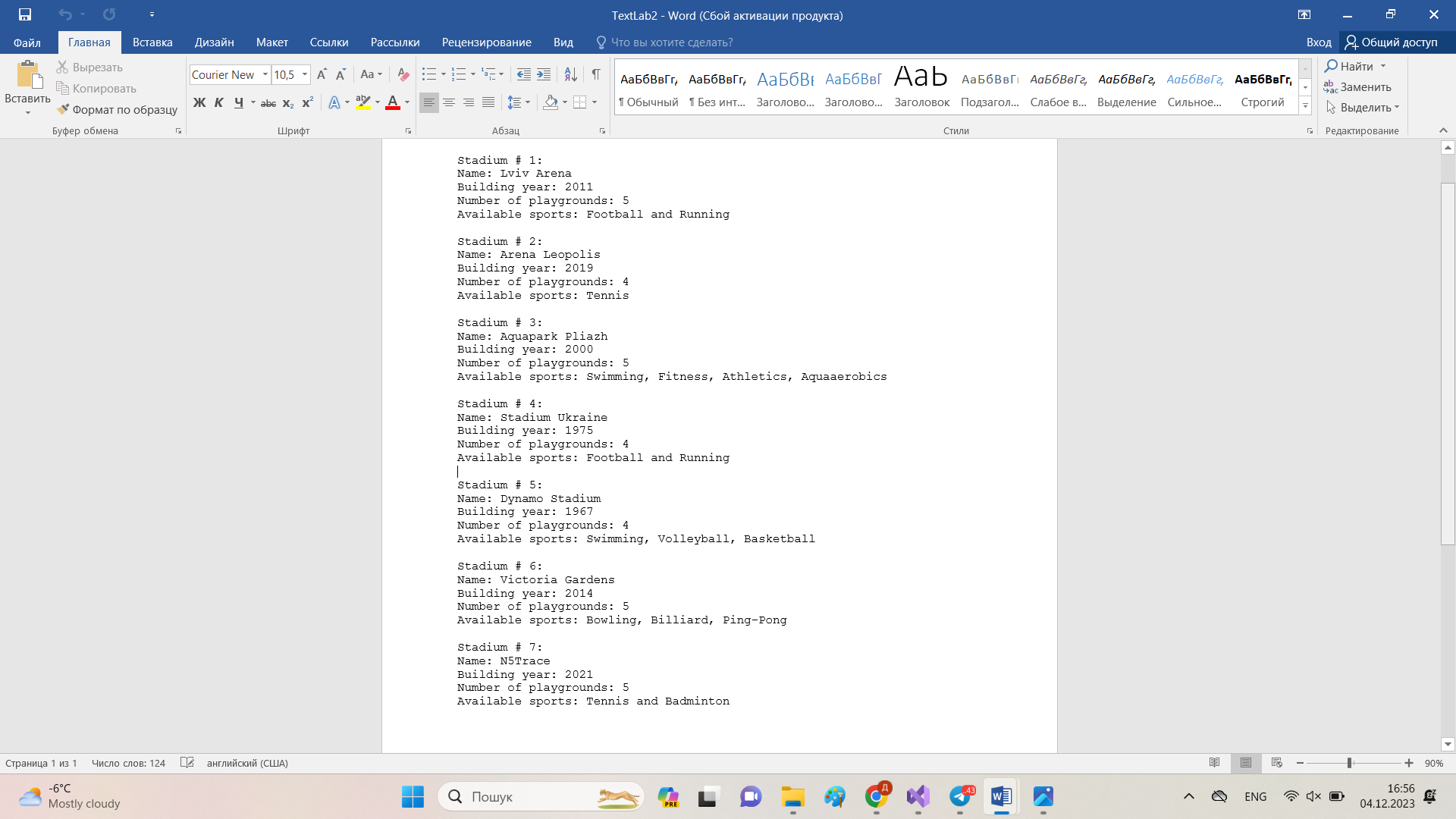
}

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









Час затрачений на виконання завдання – 120 хвилин

**VNS Lab 9**

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

Завдання 3 VNS Lab 9

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

інформацію

Виконати завдання.

23.

1) Скопіювати з файлу F1 у файл F2 всі рядки, у яких міститься два однакових

слова.

2) Визначити номер слова, у якому найбільше букв «a».

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

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

using namespace std;

int arraymaximum(vector<int> array, int size, int& n)

{

int max = array[0];

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

{

if (array[i] > max)

{

max = array[i];

n = i;

}

}

return max;

}

int arraymaximum(int array[], int size, int& n)

{

int max = array[0];

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

{

if (array[i] > max)

{

max = array[i];

n = i;

}

}

return max;

}

bool doubleword(string str)

{

bool dublicate = false;

vector<string> words;

words.resize(100);

int size = str.size();

int j = 0;

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

{

if (str[i] != ' ' && str[i] != NULL && str[i] != '.' && str[i] != ',')

{

words[j].push\_back(str[i]);

}

if (str[i] == ' ')

{

j++;

}

}

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

{

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

{

if (i != m && words[i] == words[m])

{

dublicate = true;

}

}

}

return dublicate;

}

int aCounter(string str, int& identificator)

{

vector<string> words;

words.resize(100);

int j = 0;

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

{

if (str[i] != ' ' && str[i] != NULL && str[i] != '.' && str[i] != ',')

{

words[j].push\_back(str[i]);

}

if (str[i] == ' ')

{

j++;

}

}

vector<int> amount;

amount.resize(j);

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

{

int count = 0;

for (int m = 0; m < words[i].size(); m++)

{

if (words[i][m] == 'a')

{

count++;

}

}

amount[i] = count;

}

int index = 0;

int x = arraymaximum(amount, j, index);

identificator = index;

return x;

}

int main()

{

ifstream input("file1.txt");

if (!input)

{

cerr << "Can't read the file";

}

else

{

vector<string> text;

text.resize(100);

string str;

cout << "The text in the file is:" << endl;

int j = 0;

while (getline(input, str))

{

cout << str << endl;

text[j] = str;

j++;

}

int rows = j;

ofstream output("file2.txt");

if (!output)

{

cerr << "Can't open the file for writing";

}

else

{

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

{

if (doubleword(text[i]))

{

output << text[i] << endl;

}

}

cout << "\033[1;32m" << "The file was written successfully" << "\033[0m" << endl;

}

int\* maxletters = new int[rows];

int\* counters = new int[rows] {0};

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

{

maxletters[i] = aCounter(text[i], counters[i]);

cout << "In line #" << i+1 << " the word #" << counters[i] + 1 << " has the biggest amount of letter a" << endl;

}

int identificator = 0;

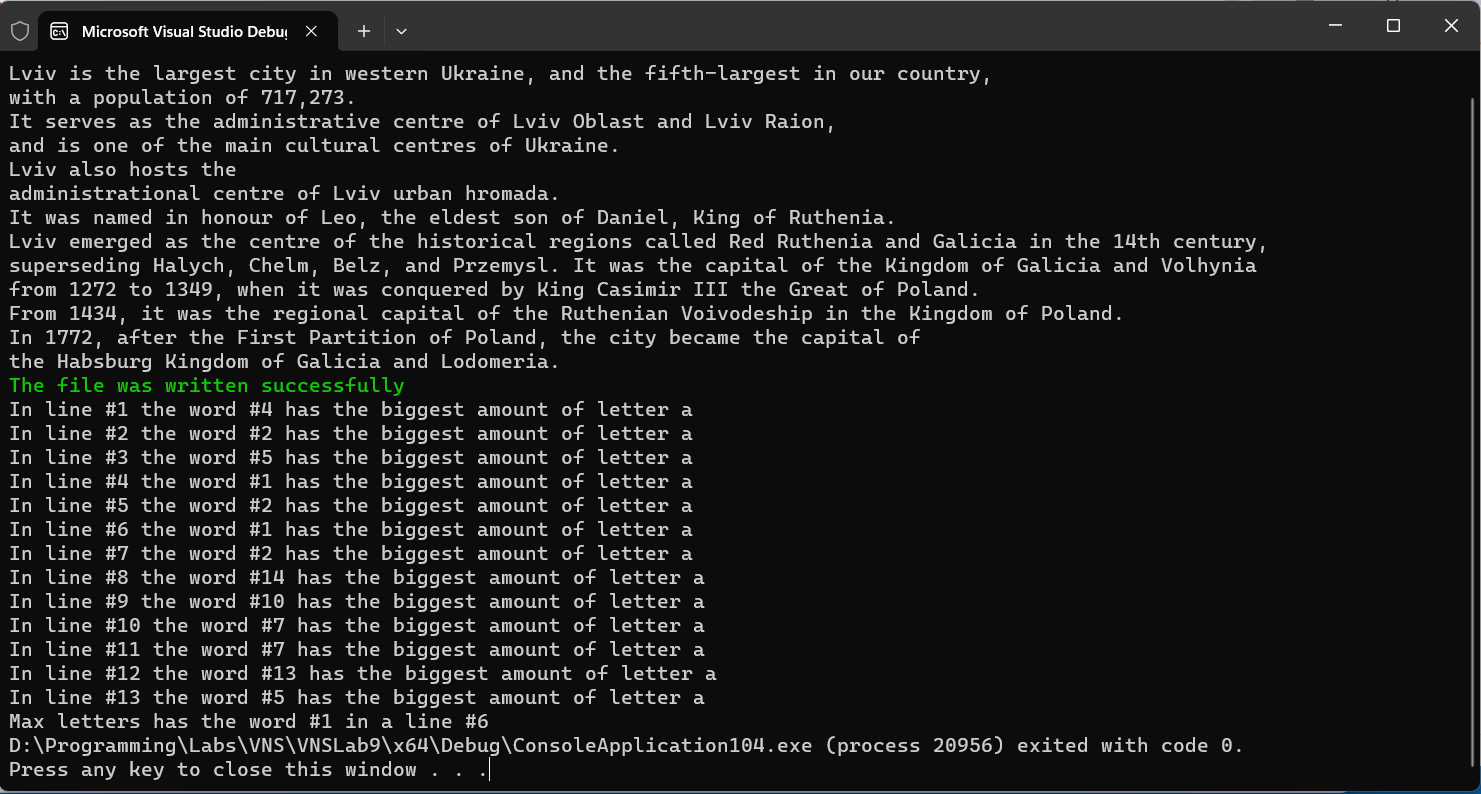
int max = arraymaximum(maxletters, rows, identificator);

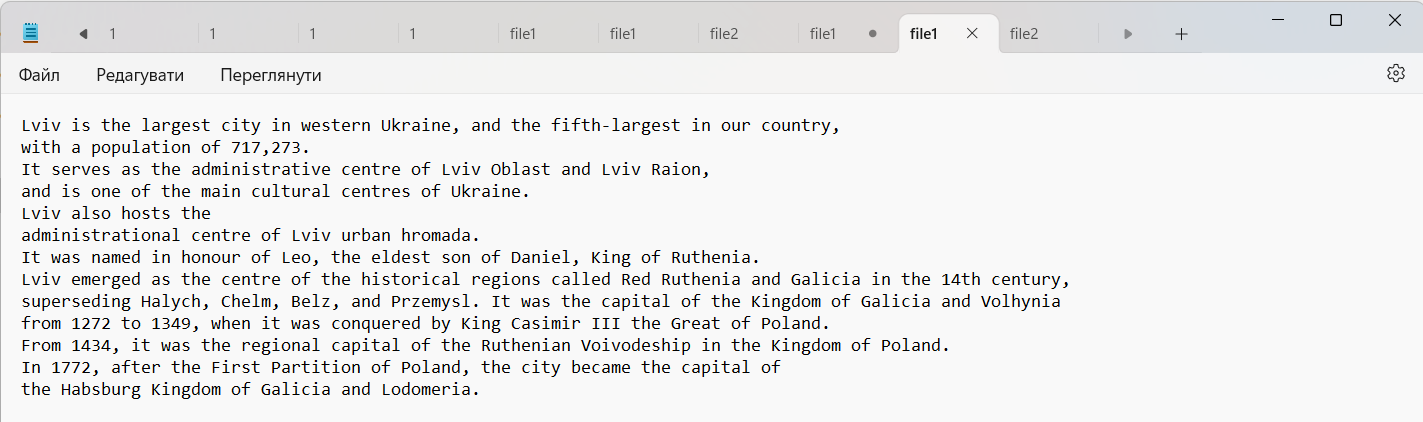
cout << "Max letters has the word #" << counters[identificator] + 1 << " in a line #" << identificator+1;

}

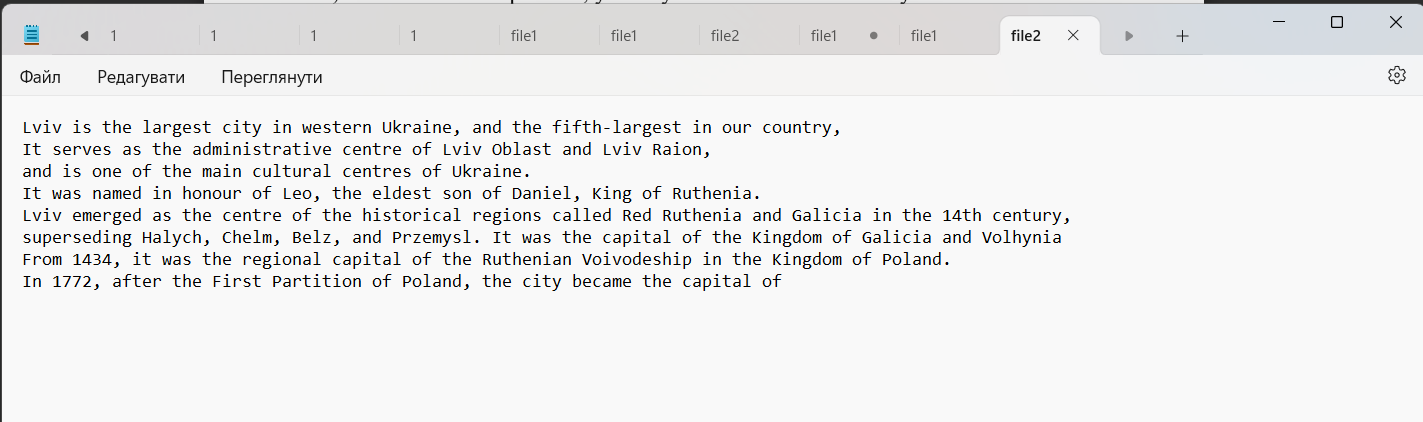
## }

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





Файл 1



Файл 2

Час затрачений на виконання завдання – 2 години

**Algotester Lab 4V2**

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

Завдання №4 Algotester Lab4v2

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

#include <iostream>

using namespace std;

void swap(int& a, int& b)

{

int x = a;

a = b;

b = x;

}

void upgradesort(int array[], int length)

{

for (int startindex = 0; startindex < length; startindex++)

{

int minindex = startindex;

for (int currentindex = startindex; currentindex < length; currentindex++)

{

if (array[currentindex] < array[minindex])

{

minindex = currentindex;

}

}

swap(array[startindex], array[minindex]);

}

}

int main()

{

int N, K;

cin >> N >> K;

int\* array1 = new int[N];

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

{

cin >> array1[i];

}

int\* array2 = new int[N];

int newsize = 0;

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

{

bool presence = false;

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

{

if (array2[j] == array1[i])

{

presence = true;

}

}

if (!presence)

{

array2[newsize] = array1[i];

newsize++;

}

}

upgradesort(array2, newsize);

int m = K % newsize;

int\* array3 = new int[newsize];

for (int t = m; t < newsize; t++)

{

array3[t - m] = array2[t];

}

for (int i = newsize - m, f = 0; f < m; i++, f++)

{

array3[i] = array2[f];

}

cout << newsize << endl;

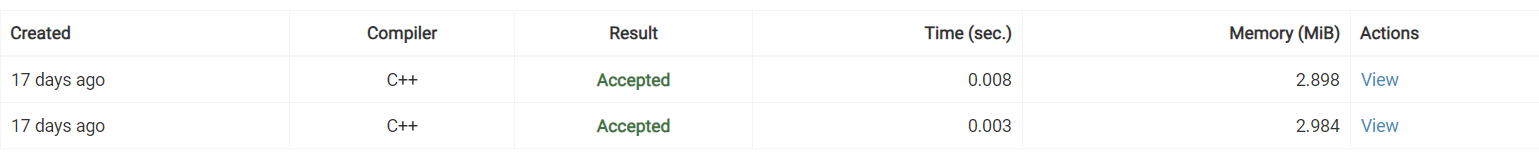
for (int i = 0; i < newsize; i++)

{

cout << array3[i] << " ";

}

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

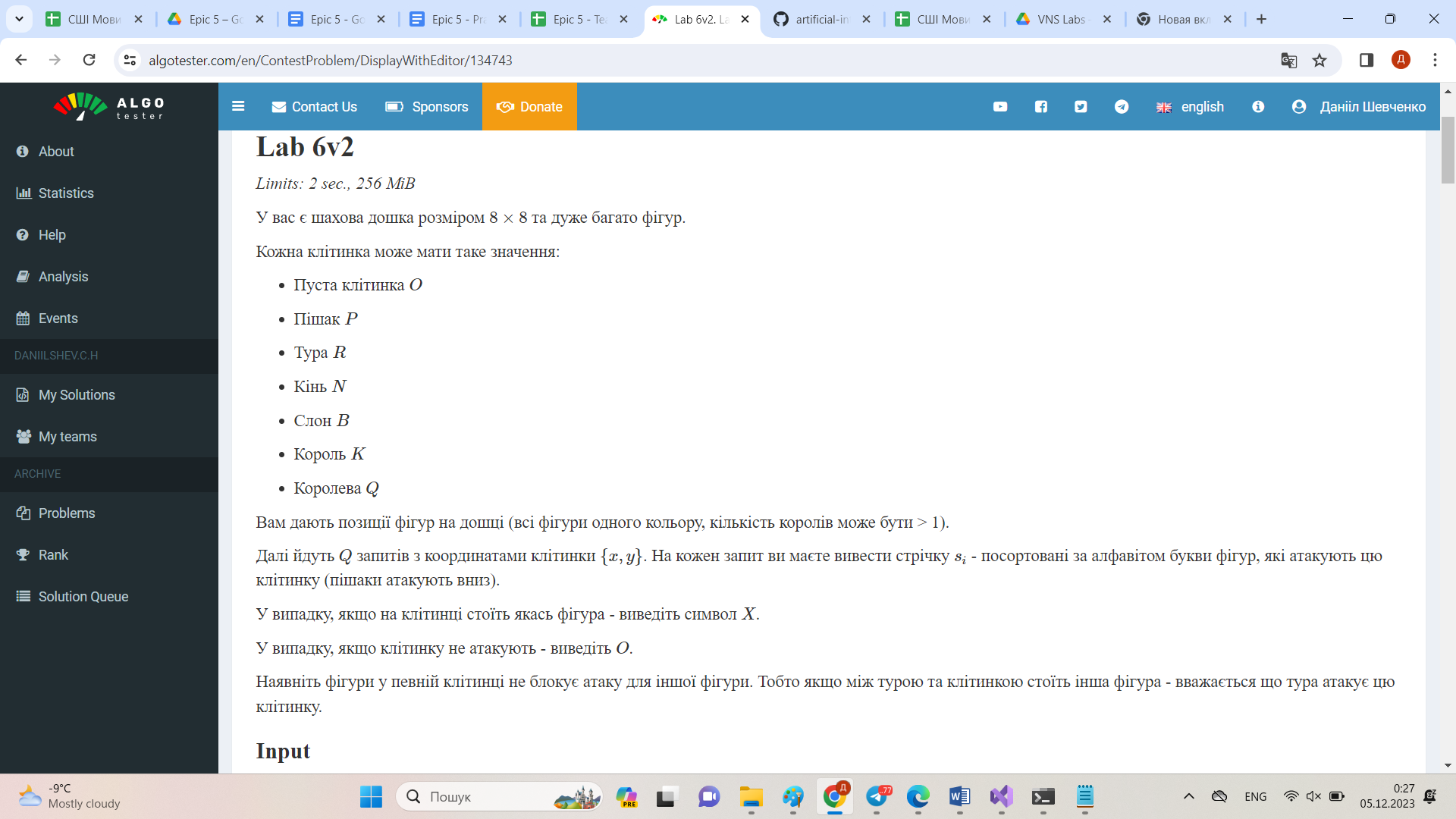


Час затрачений на виконання завдання – 60 хвилин

**Algotester Lab6v2**

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

Завдання №5 AlgotesterLab6v2



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

#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

int main()

{

string board[20];

for (int i = 0; i < 8; i++)

{

getline(cin, board[i]);

}

for (int i = 0; i < 20; i++)

{

board[i].resize(20);

}

int Q;

cin >> Q;

for (int m = 0; m < Q; m++)

{

int a;

int b;

cin >> a >> b;

int x = a - 1;

int y = b - 1;

string answer;

answer.resize(15);

int k = 0;

if (board[x][y] != 'O')

{

cout << 'X';

}

else

{

//пішка(P)

bool pishka = false;

if (((y>0 && x>0) && (board[x - 1][y - 1] == 'P')) || (x > 0 && (board[x - 1][y + 1] == 'P')))

{

pishka = true;

}

if (pishka)

{

answer[k] = 'P';

k++;

}

//тура(R)

bool tura = false;

for (int f = 0; f < 8; f++)

{

if (board[x][f] == 'R' || board[f][y] == 'R')

{

tura = true;

}

}

if (tura)

{

answer[k] = 'R';

k++;

}

//кінь(N)

bool horse = false;

if (y > 0 && (board[x + 2][y - 1] == 'N' ) || (board[x + 2][y + 1] == 'N')

|| ((x > 1 && y > 0) && (board[x - 2][y - 1] == 'N'))

|| (x > 1 && board[x - 2][y + 1] == 'N') || (y > 1 && board[x + 1][y - 2] == 'N')

|| (board[x + 1][y + 2] == 'N') || ((x > 0 && y > 1) && board[x - 1][y - 2] == 'N')

|| (x > 0 && board[x - 1][y + 2] == 'N'))

{

horse = true;

}

if (horse)

{

answer[k] = 'N';

k++;

}

//офіцер(B)

bool oficer = false;

for (int f = 0; f < 8; f++)

{

if (board[x + f][y + f] == 'B' || (y >= f && board[x + f][y - f] == 'B') || (x >= f && board[x - f][y + f] == 'B') || ((x >= f && y >= f) && board[x - f][y - f] == 'B' ))

{

oficer = true;

}

}

if (oficer)

{

answer[k] = 'B';

k++;

}

//король(K)

bool king = false;

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

(y > 0 && board[x + 1][y - 1] == 'K') || (board[x][y + 1] == 'K') ||

(y>0 && board[x][y - 1] == 'K') || (x>0 && board[x - 1][y + 1] == 'K') ||

(x>0 && board[x - 1][y] == 'K') || ((x > 0 && y > 0) && board[x - 1][y - 1] == 'K'))

{

king = true;

}

if (king)

{

answer[k] = 'K';

k++;

}

//королева(Q)

bool queen = false;

for(int u = 0; u < 8; u++)

if ((board[x + u][y + u] == 'Q') || (y >= u && board[x + u][y - u] == 'Q') || (x >= u && board[x - u][y + u] == 'Q') || ((x >= u && y >= u) && board[x - u][y - u] == 'Q') || (board[x][u] == 'Q') || (board[u][y] == 'Q'))

{

queen = true;

}

if (queen)

{

answer[k] = 'Q';

k++;

}

}

if (k == 0 && board[x][y] == 'O')

{

cout << 'O';

}

else

{

answer.resize(k);

sort(answer.begin(), answer.end());

cout << answer;

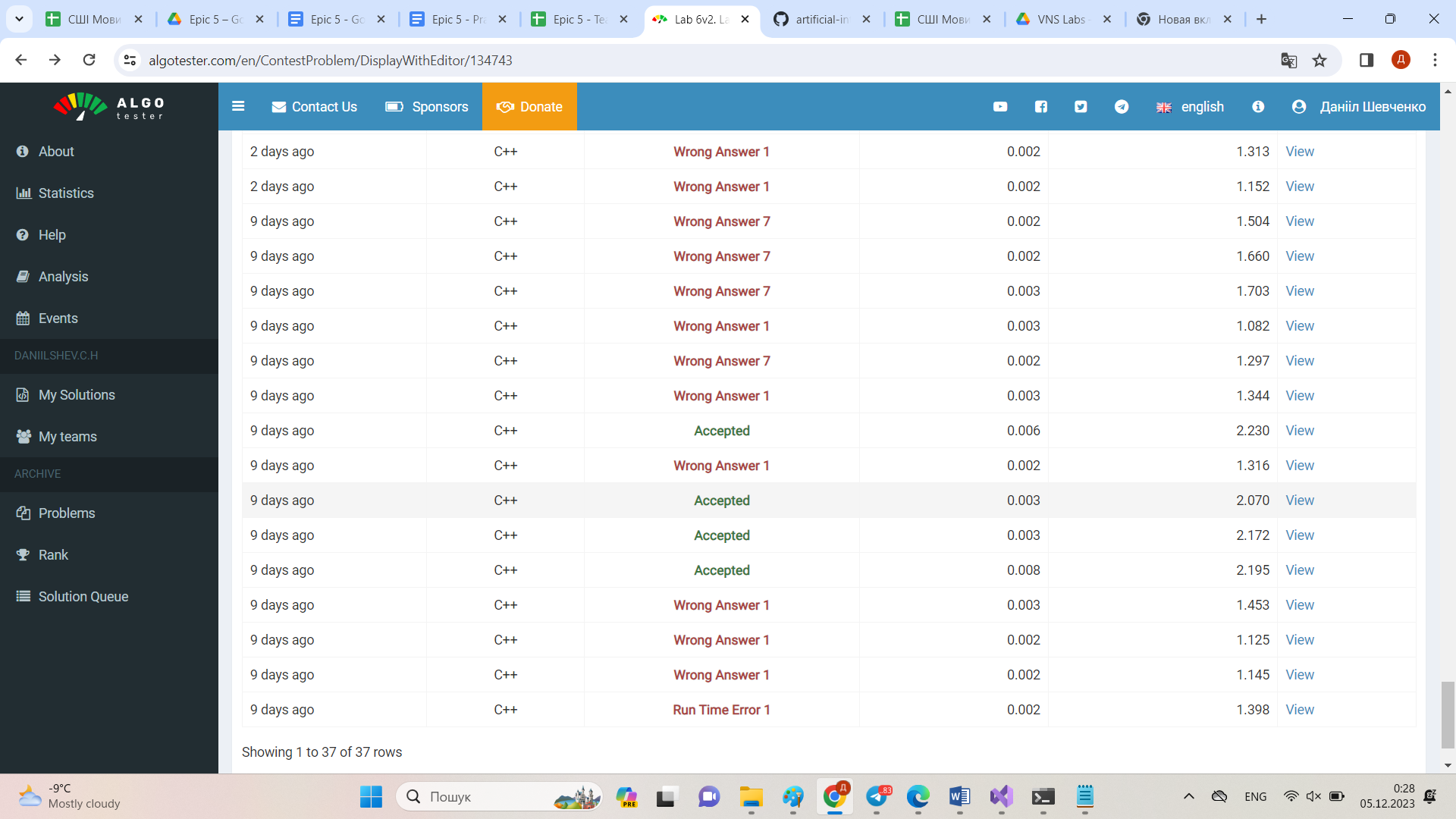
}

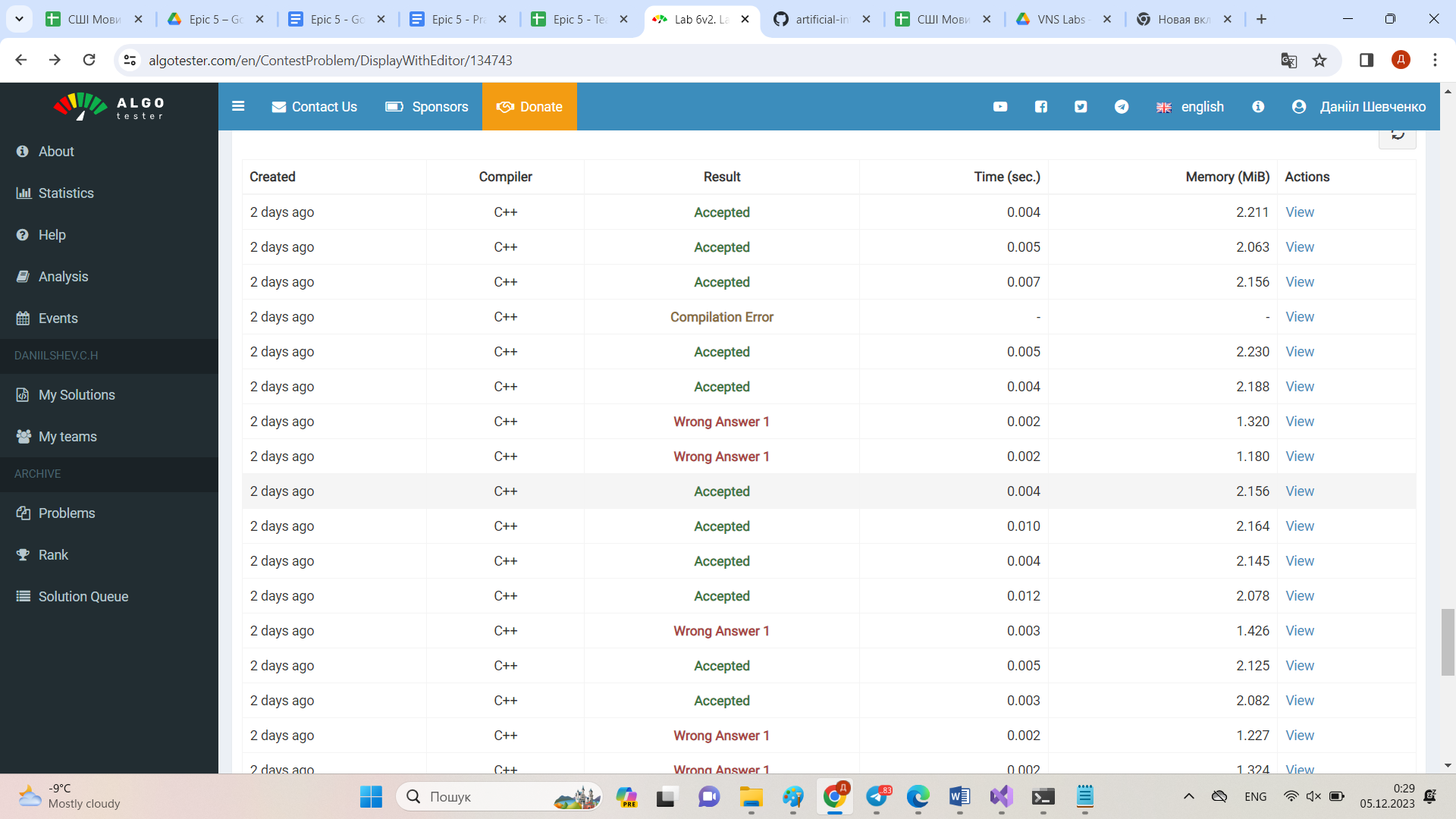
cout << endl;

}

}

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





Час затрачений на виконання завдання – 7 днів(проблема з перевіркою)  
**Practice Task # 5**

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

Завдання №6 Class-Practice Task # 6

## Задача №1 – Запис текстової стрічки у файл із заданим ім’ям

***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.

## Задача №2 – Копіювання вмісту файла у інший файл

***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.

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

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

using namespace std;

enum Status

{

Success,

Failure

};

Status filewriting(string name, string input)

{

ofstream out(name);

if (!out)

{

cerr << "Can't open file for writing" << endl;

return Failure;

}

else

{

out << input << endl;

cout << "Operation was completed successffully!" << endl;

return Success;

}

}

Status filecopying(string file\_from, string file\_to)

{

ifstream outof(file\_from);

ofstream into(file\_to);

if (!outof)

{

cerr << "Can't open file for reading" << endl;

return Failure;

}

else if (!into)

{

cerr << "Can't open file for writing" << endl;

return Failure;

}

else

{

vector<string> text;

text.resize(100);

string str;

cout << "The text in the file is:" << endl;

int j = 0;

while (getline(outof, str))

{

cout << str << endl;

text[j] = str;

j++;

}

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

{

into << text[i] << endl;

}

cout << "Operation was completed successfully";

return Success;

}

}

int main()

{

cout << "Part #1" << endl;

string a;

getline(cin, a);

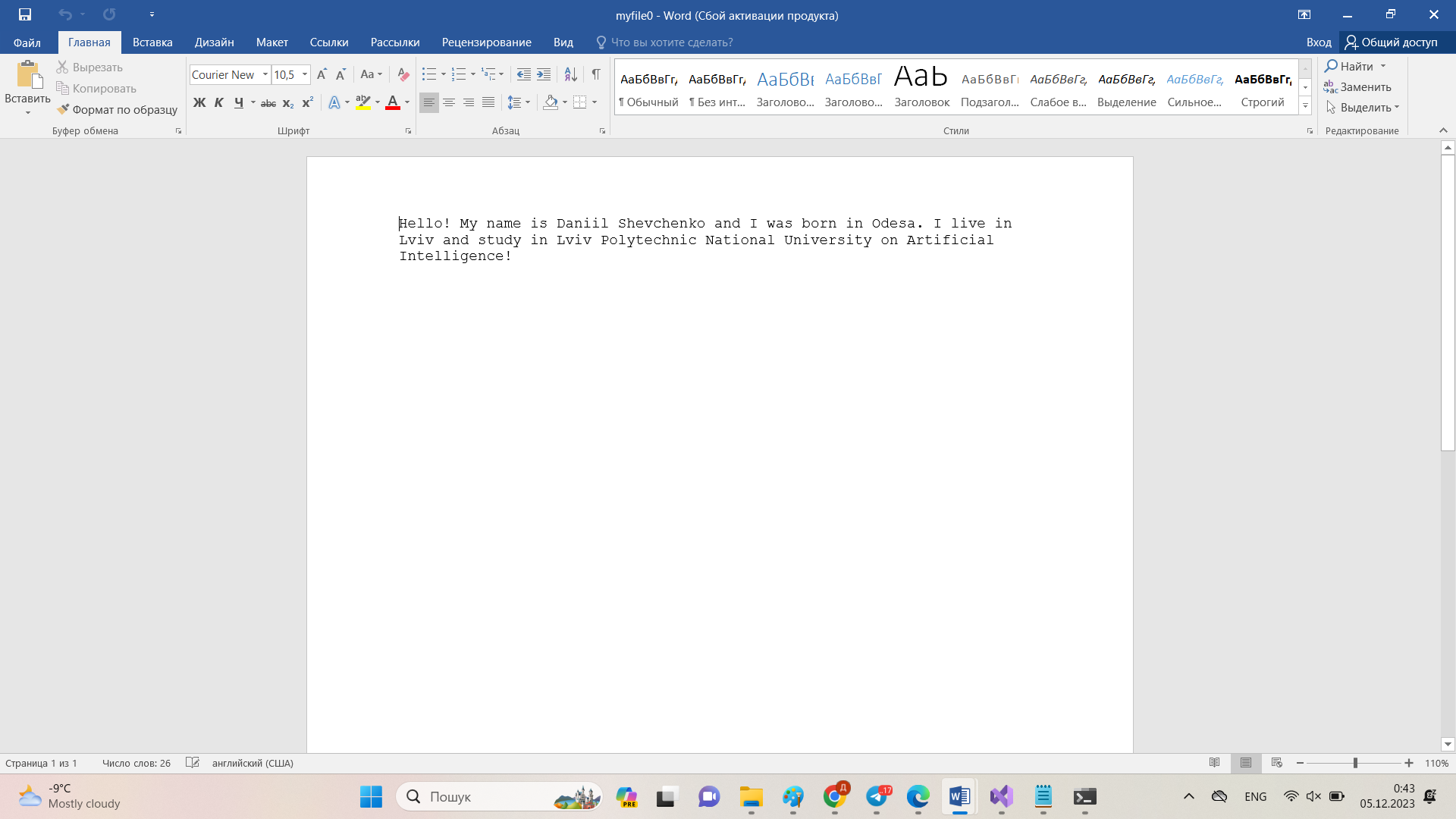
filewriting("myfile0.txt", a);

cout << "Part #2" << endl;

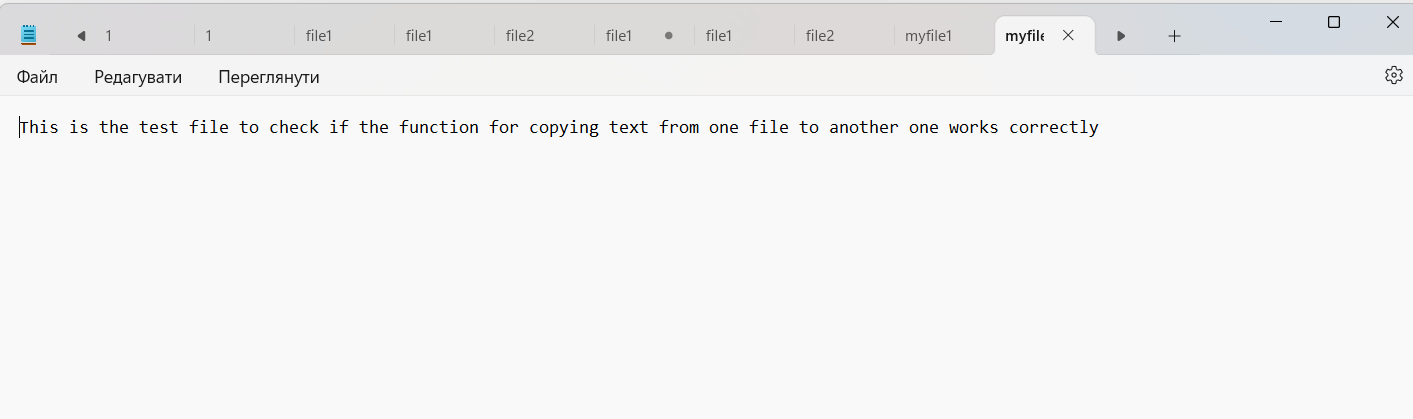
filecopying("myfile1.txt", "myfile2.txt");

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

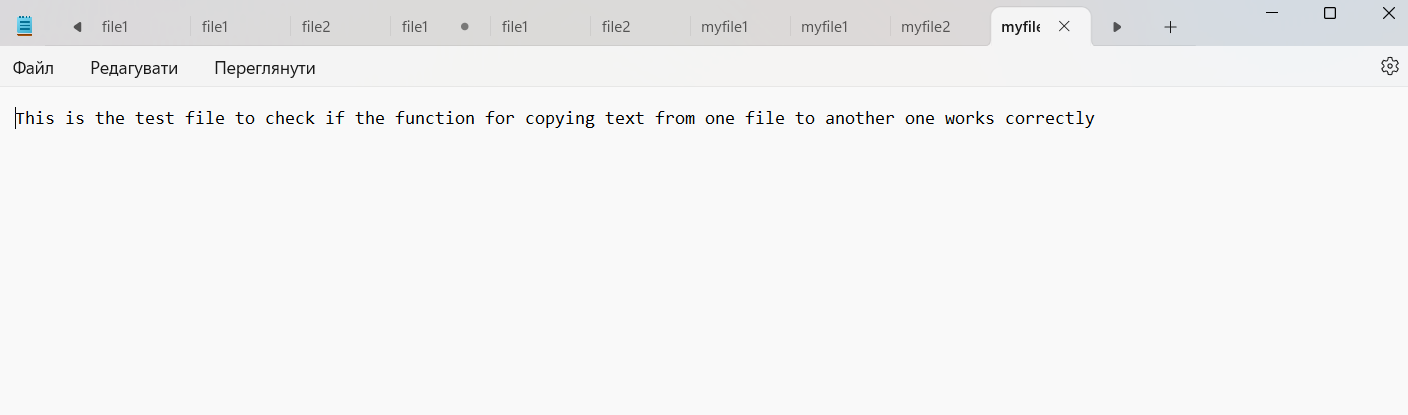




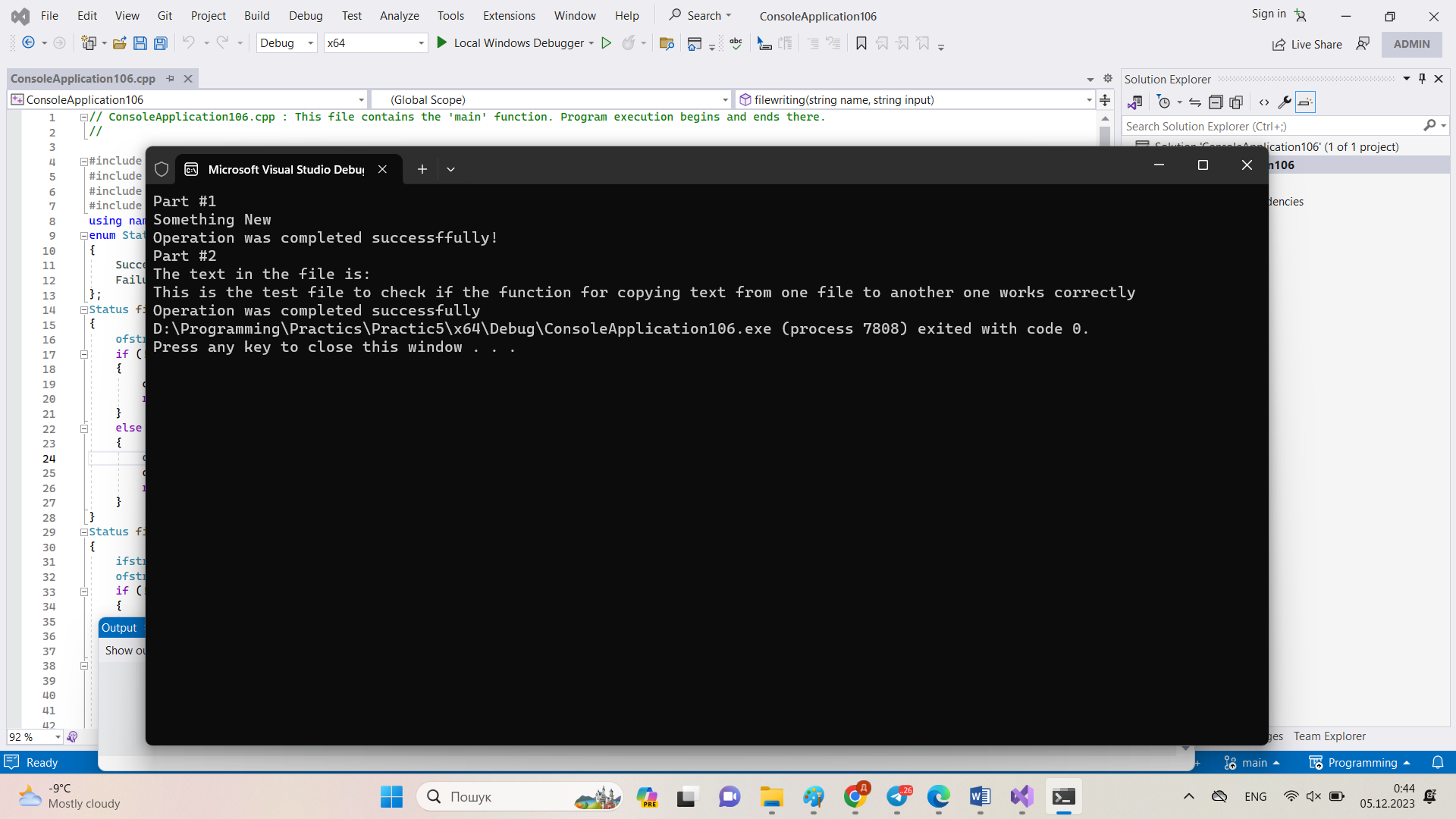
File 0



File 1



File 2



Час затрачений на виконання завдання – 30 хвилин

**Self-Practice(Algotester Lab4v1)**

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

Завдання №7 AlgotesterLab4v1



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

#include <iostream>

#include <vector>

using namespace std;

void upgradesort(vector<int>& array, int length)

{

for (int startindex = 0; startindex < length; startindex++)

{

int minindex = startindex;

for (int currentindex = startindex; currentindex < length; currentindex++)

{

if (array[currentindex] < array[minindex])

{

minindex = currentindex;

}

}

swap(array[startindex], array[minindex]);

}

}

int presentcounter(int element, vector<int> array, int size)

{

int count = 0;

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

{

if (array[i] == element)

{

count++;

}

}

return count;

}

int max(int x, int y)

{

if (x >= y)

{

return x;

}

else

{

return y;

}

}

int min(int x, int y)

{

if (x <= y)

{

return x;

}

else

{

return y;

}

}

int main()

{

int N;

cin >> N;

vector<int> array1;

array1.resize(N);

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

{

cin >> array1[i];

}

int M;

cin >> M;

vector<int> array2;

array2.resize(M);

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

{

cin >> array2[i];

}

// difference N-M

vector<int> difference12;

difference12.resize(N);

int indexdif12 = 0;

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

{

if (presentcounter(array1[i], difference12, indexdif12) == 0)

{

int x = presentcounter(array1[i], array1, N);

int y = presentcounter(array1[i], array2, M);

int k = x - y;

if (x > y)

{

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

{

difference12[indexdif12] = array1[i];

indexdif12++;

}

}

}

}

cout << indexdif12 << endl;

upgradesort(difference12, indexdif12);

for (int i = 0; i < indexdif12; i++)

{

cout << difference12[i] << " ";

}

cout << endl;

// difference M-N

vector<int> difference21;

difference21.resize(M);

int indexdif21 = 0;

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

{

if (presentcounter(array2[i], difference21, indexdif21) == 0)

{

int x = presentcounter(array2[i], array2, M);

int y = presentcounter(array2[i], array1, N);

int k = x - y;

if (x > y)

{

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

{

difference21[indexdif21] = array2[i];

indexdif21++;

}

}

}

}

cout << indexdif21 << endl;

upgradesort(difference21, indexdif21);

for (int i = 0; i < indexdif21; i++)

{

cout << difference21[i] << " ";

}

cout << endl;

// intersection

vector<int> intersection;

intersection.resize(N);

int indexinter = 0;

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

{

if (presentcounter(array1[i], intersection, indexinter) == 0)

{

int x = presentcounter(array1[i], array1, N);

int y = presentcounter(array1[i], array2, M);

int k = min(x, y);

if (k > 0)

{

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

{

intersection[indexinter] = array1[i];

indexinter++;

}

}

}

}

cout << indexinter << endl;

upgradesort(intersection, indexinter);

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

{

cout << intersection[i] << " ";

}

cout << endl;

// addition

vector<int> consolidation;

consolidation.resize(N+M);

int indexcon = 0;

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

{

if (presentcounter(array1[i], consolidation, indexcon) == 0)

{

int x = presentcounter(array1[i], array1, N);

int y = presentcounter(array1[i], array2, M);

int k = max(x, y);

if (k > 0)

{

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

{

consolidation[indexcon] = array1[i];

indexcon++;

}

}

}

}

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

{

if (presentcounter(array2[i], consolidation, indexcon) == 0)

{

int x = presentcounter(array2[i], array2, M);

int y = presentcounter(array2[i], array1, N);

int k = max(x, y);

if (k > 0)

{

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

{

consolidation[indexcon] = array2[i];

indexcon++;

}

}

}

}

cout << indexcon << endl;

upgradesort(consolidation, indexcon);

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

{

cout << consolidation[i] << " ";

}

cout << endl;

//symmetric difference

vector<int> symdif;

symdif.resize(N+M);

int indexsym = 0;

for (int i = 0; i < indexdif12; i++)

{

symdif[i] = difference12[i];

}

for (int i = 0; i < indexdif21; i++)

{

symdif[indexdif12 + i] = difference21[i];

}

upgradesort(symdif, indexdif12 + indexdif21);

cout << indexdif12 + indexdif21 << endl;

for (int i = 0; i < indexdif12 + indexdif21; i++)

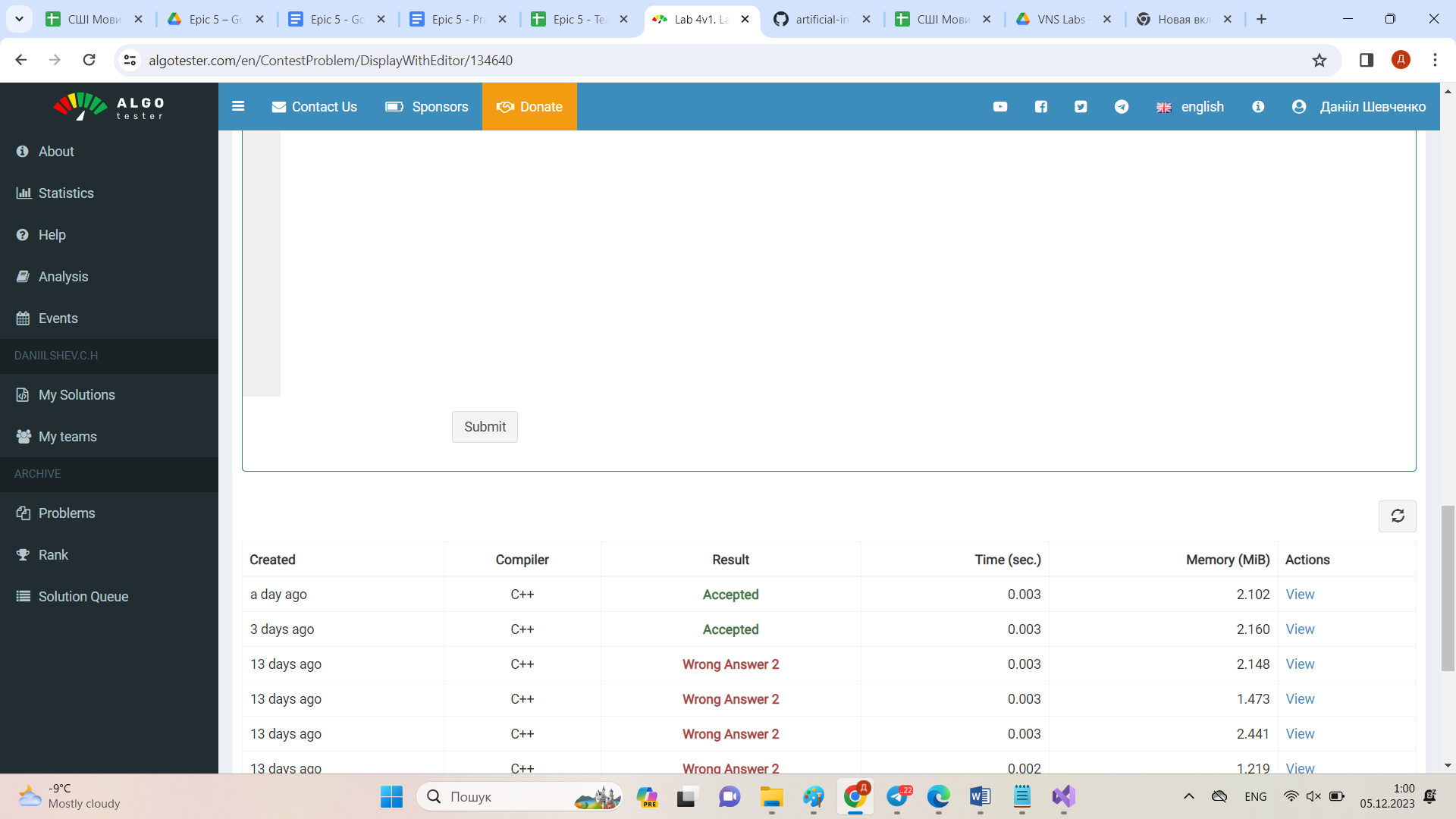
{

cout << symdif[i] << " ";

}

## }

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



**Self-Practice(Algotester Lab4v3)**

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

Завдання №8 AlgotesterLab4v3



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

## #include <iostream>

## using namespace std;

## void swap(int& a, int& b)

## {

## int x = a;

## a = b;

## b = x;

## }

## void upgradesort(int array[], int length)

## {

## for (int startindex = 0; startindex < length; startindex++)

## {

## int minindex = startindex;

## for (int currentindex = startindex; currentindex < length; currentindex++)

## {

## if (array[currentindex] < array[minindex])

## {

## minindex = currentindex;

## }

## }

## swap(array[startindex], array[minindex]);

## }

## }

## void downgradesort(int array[], int length)

## {

## for (int startindex = 0; startindex < length; startindex++)

## {

## int maxindex = startindex;

## for (int currentindex = startindex; currentindex < length; currentindex++)

## {

## if (array[currentindex] > array[maxindex])

## {

## maxindex = currentindex;

## }

## }

## swap(array[startindex], array[maxindex]);

## }

## }

## int main()

## {

## int N;

## cin >> N;

## int\* array = new int[N];

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

## {

## cin >> array[i];

## }

## int\* array0 = new int[N];

## int\* array1 = new int[N];

## int\* array2 = new int[N];

## int l0, l1, l2;

## l0 = 0;

## l1 = 0;

## l2 = 0;

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

## {

## if (array[i] % 3 == 0)

## {

## array0[l0] = array[i];

## l0++;

## }

## }

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

## {

## if (array[i] % 3 == 1)

## {

## array1[l1] = array[i];

## l1++;

## }

## }

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

## {

## if (array[i] % 3 == 2)

## {

## array2[l2] = array[i];

## l2++;

## }

## }

## upgradesort(array0, l0);

## upgradesort(array2, l2);

## downgradesort(array1, l1);

## int\* newarray = new int[N];

## for (int i = 0, k = 0; k < l0; i++, k++)

## {

## newarray[i] = array0[k];

## }

## for (int i = l0, k = 0; k < l1; i++, k++)

## {

## newarray[i] = array1[k];

## }

## for (int i = l0+l1, k = 0; k < l2; i++, k++)

## {

## newarray[i] = array2[k];

## }

## 

## int\* finalarray = new int[N];

## int newsize = 0;

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

## {

## bool presence = false;

## for (int j = 0; j < newsize; j++)

## {

## if (finalarray[j] == newarray[i])

## {

## presence = true;

## }

## }

## if (!presence)

## {

## finalarray[newsize] = newarray[i];

## newsize++;

## }

## }

## cout << newsize << endl;

## for (int i = 0; i < newsize; i++)

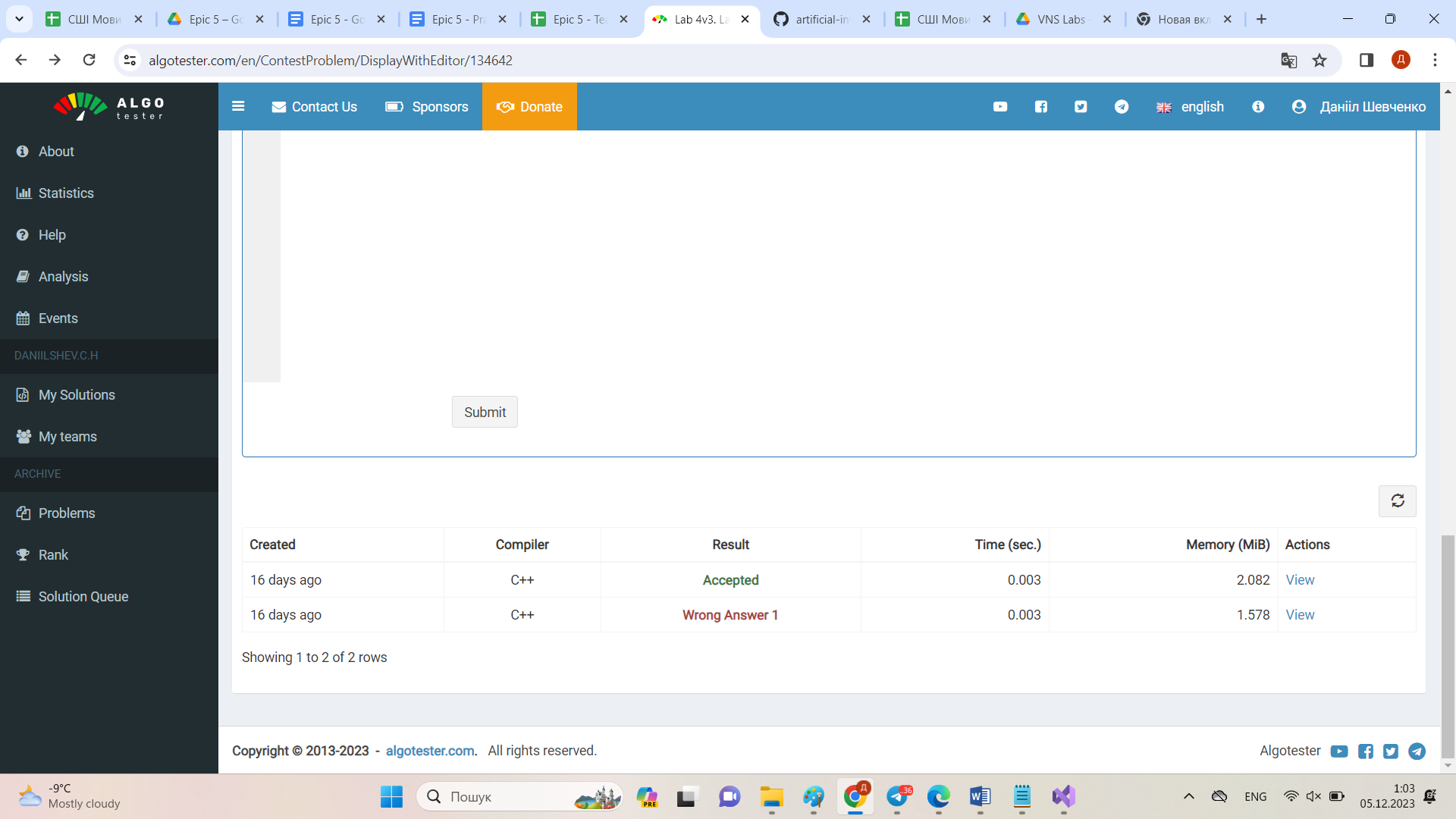
## {

## cout << finalarray[i] << " ";

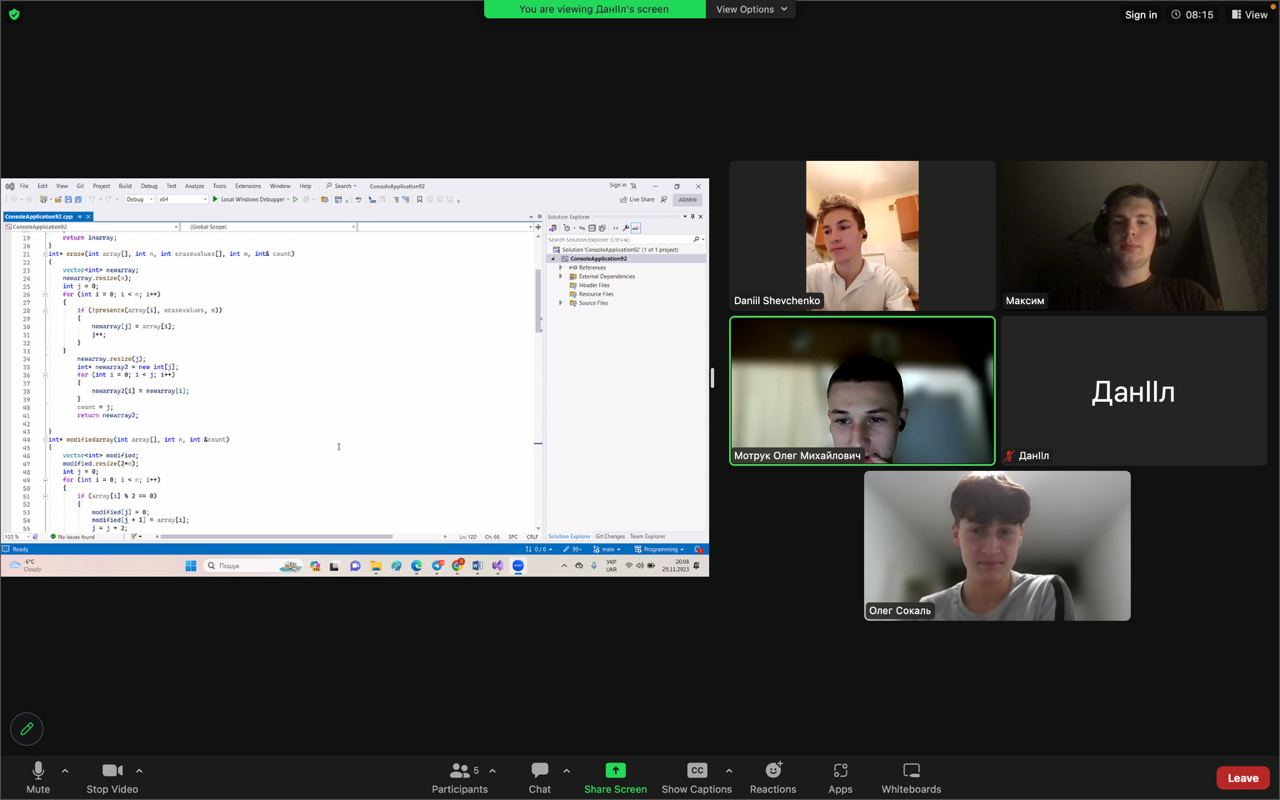
## }

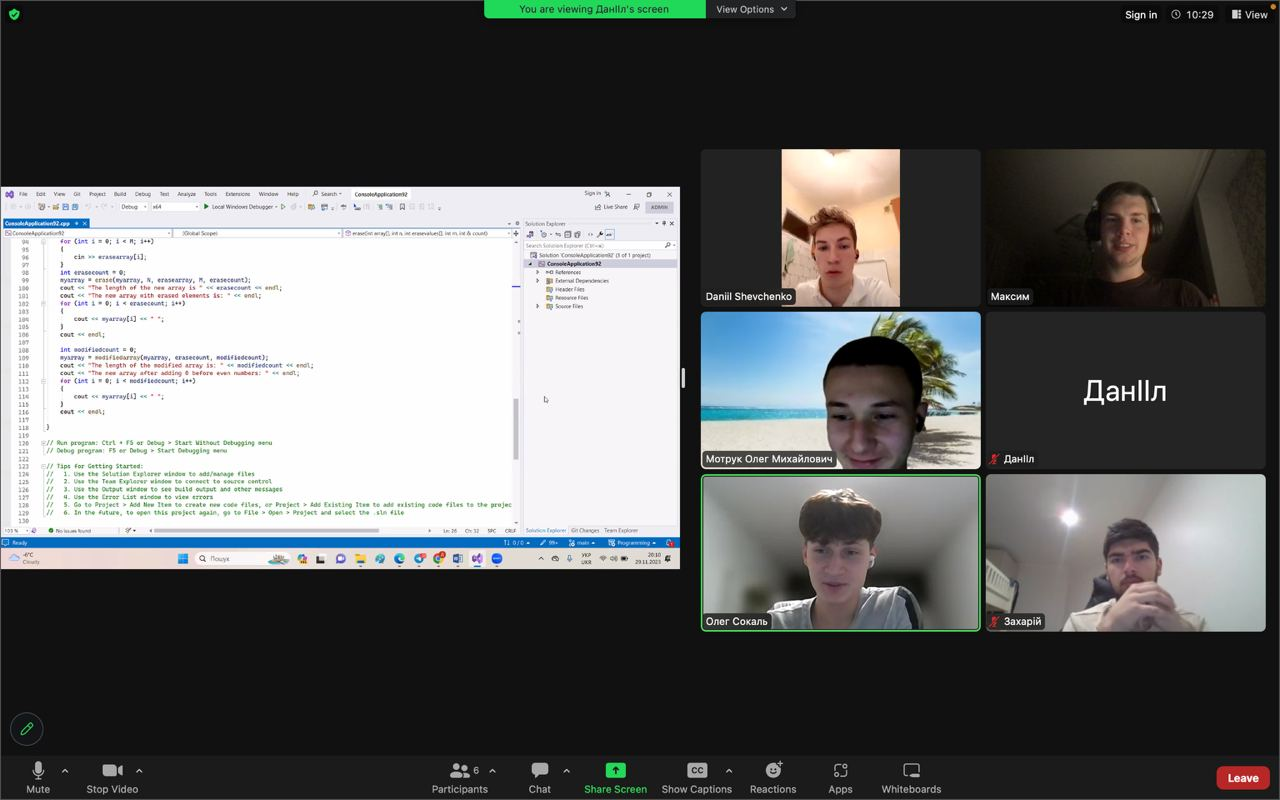
## }

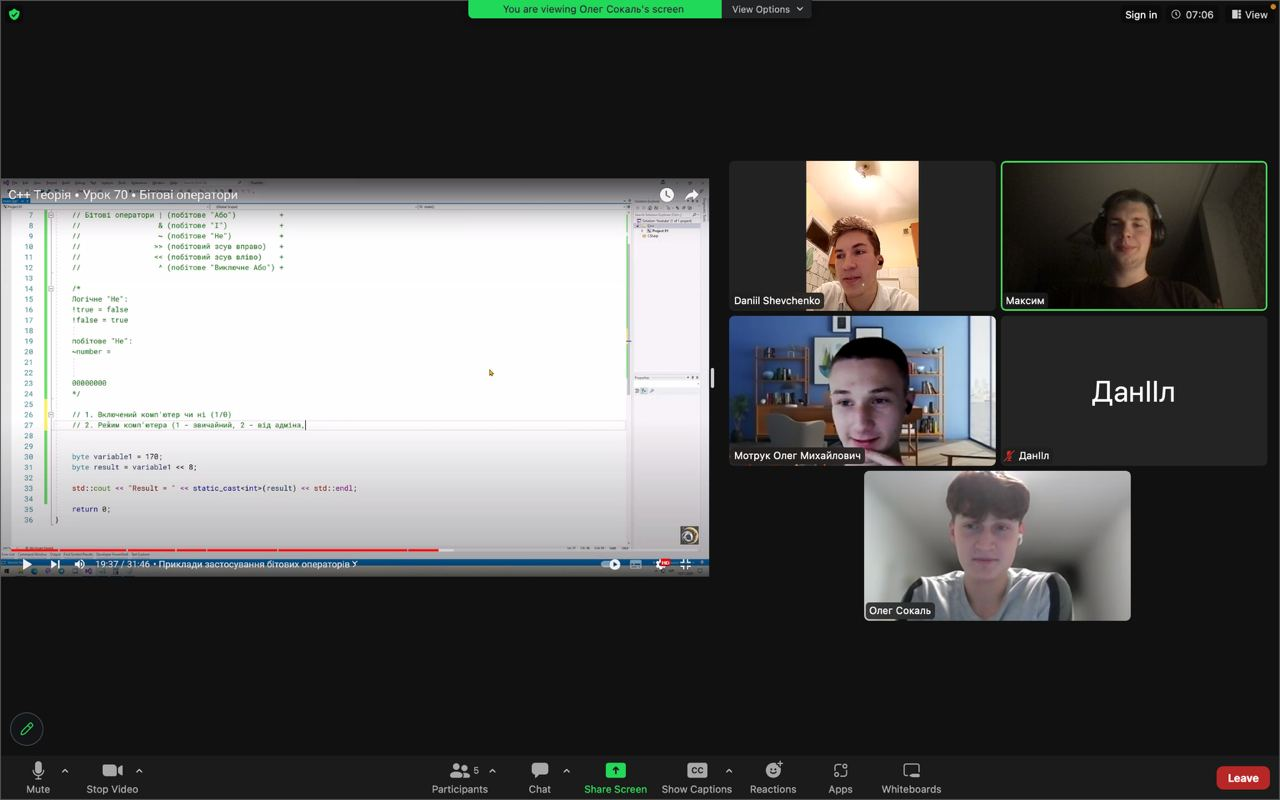
## **3. Результати виконання завдань, тестування та фактично затрачений час:**



**Докази командної роботи**







# **Висновки:**

Отже, в ході роботи над епіком були систематизовані основні знання про роботу зі стрічками, різні види стрічок, алгоритми їх введення та обробки. А окрему частину роботи над епіком було присвячено бібліотекам роботи з файлами та алгоритмам, що дозволяють створювати, редагувати та зчитувати файли

Pull-Request: