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

Кисіль Сергій Дмитрович

# **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:

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

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

<https://acode.com.ua/urok-211-dostup-do-symvoliv-std-string-konvertatsiya-std-string-v-ryadky-c-style/>

Team meats

Одногрупники та однопотоківці

* + Що опрацьовано:

Ініціалізація строк типу C-Style

Усвідомлення структуру строк C-Style у вигляді масиву елементів типу char

* + Статус: Ознайомлений/ Ознайомлений частково / Не ознайомлений
  + Початок опрацювання теми: 01.11.2023
  + Завершення опрацювання теми: 29.11.2023

Тема №2: Строки std::string

* + Джерела Інформації:

Team meats

<https://acode.com.ua/urok-208-ryadkovi-klasy-std-string-i-std-wstring/>

<https://acode.com.ua/urok-209-stvorennya-znyshhennya-i-konvertatsiya-std-string/>

<https://acode.com.ua/urok-210-dovzhyna-i-yemnist-std-string/>

<https://acode.com.ua/urok-212-prysvoyuvannya-i-perestanovka-znachen-z-std-string/>

<https://acode.com.ua/urok-213-dodavannya-do-std-string/>

<https://acode.com.ua/urok-214-vstavka-symvoliv-i-ryadkiv-v-std-string/>

(також кожну тему опрацьовував на <https://vns.lpnu.ua/pluginfile.php?file=%2F1116827%2Fmod_resource%2Fcontent%2F1%2Fkonspekt_lekcyi_ukr_1.pdf> )

* + Що опрацьовано:

Бібліотека <string> та її застосування

Різниця між строками C-Style та std::string

Методи обробки std::string

* + Статус: Ознайомлений/ Ознайомлений частково / Не ознайомлений
  + Початок опрацювання теми: 01.11.2023
  + Завершення опрацювання теми: 29.11.2023

Тема №3: Обробка вводу стрічок, функція getline

* + Джерела Інформації:

<https://acode.com.ua/urok-215-potoky-vvodu-i-vyvodu/>

<https://acode.com.ua/urok-216-funktsional-klasu-istream/>

<https://acode.com.ua/urok-217-funktsional-klasiv-ostream-i-ios-formatuvannya-vyvodu/>

Team meats

Одногрупники

* + Що опрацьовано:

Застосування функції getline

Відмінності getline від cin

Використання функції cin.ignore() в парі з cin та getline

Статус: Ознайомлений/ Ознайомлений частково / Не ознайомлений

* + Початок опрацювання теми: 01.11.2023
  + Завершення опрацювання теми: 29.11.2023

Тема №4: Бібліотека fstream та використання її для роботи з файлами

* + Джерела Інформації:

<https://acode.com.ua/urok-220-bazovyj-fajlovyj-vvid-i-vyvid/>

Одногрупники

* + 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/ Ознайомлений частково / Не ознайомлений

* + Початок опрацювання теми: 01.11.2023
  + Звершення опрацювання теми: 29.11.2023

Тема №5: Інші бібліотеки файлової обробки

* + Джерела Інформації:

<https://acode.com.ua/urok-221-randomnyj-fajlovyj-vvid-i-vyvid/>

* + Що опрацьовано:

Алгоритми обробки файлів мови С, що були актуальні в колишньому

* + Статус: Ознайомлений/ Ознайомлений частково / Не ознайомлений
  + Початок опрацювання теми: 01.11.2023
  + Звершення опрацювання теми: 29.11.2023

# **Виконання роботи:**

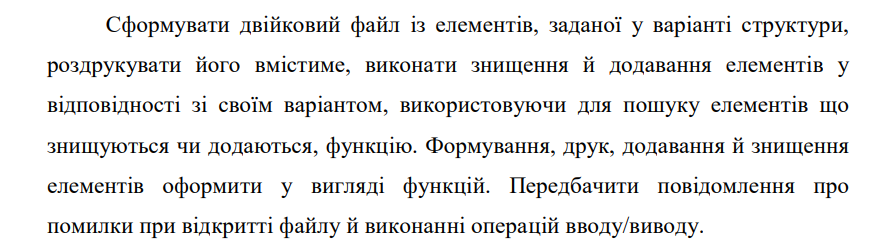
## **1. Опрацювання завдання та вимог до програм та середовища:**

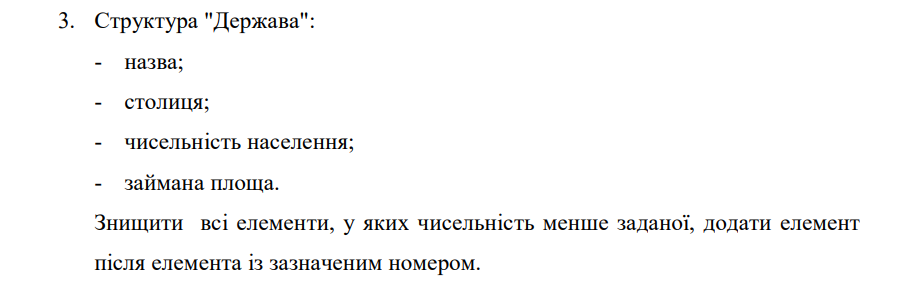
* Завдання №1 Лаб# Програмування: VNS Lab6 (Завдання 1)
* Варіант завдання: 3
* Деталі завдання:

Малюнок(1) - умова завдання 1.

Завдання №2 Лаб# Програмування: VNS Lab8 (Завдання 1)

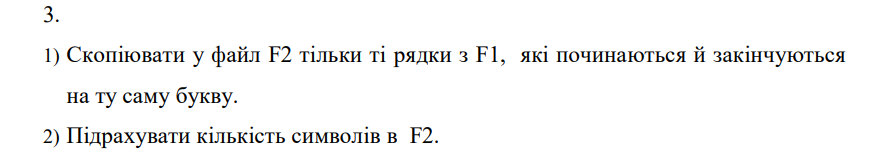
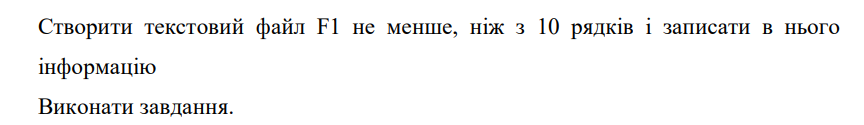
Варіант завдання: 3

* Деталі завдання:
* 

 Малюнок(2) - умова завдання 2.

Завдання №3 Лаб# Програмування: VNS Lab9 (Завдання 1)

Варіант завдання: 3

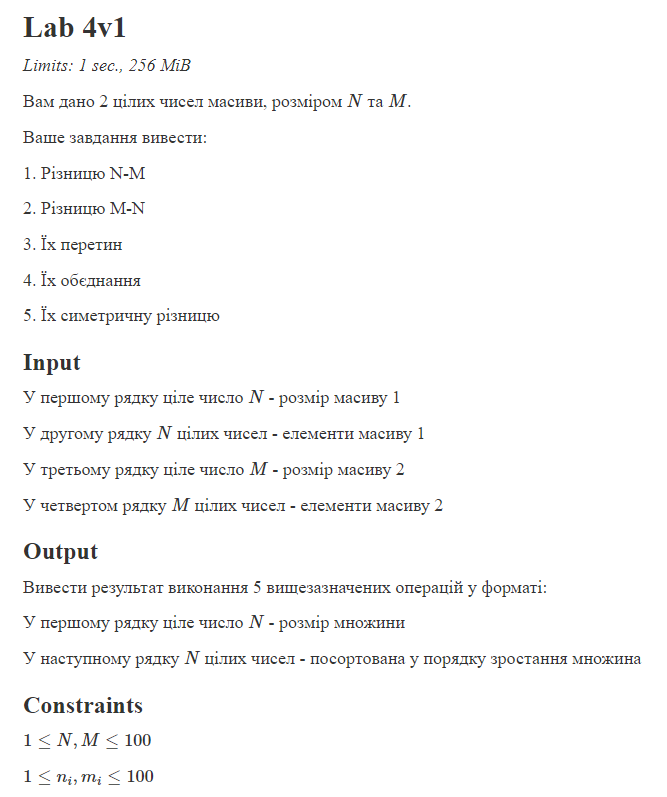
* Деталі завдання:

Малюнок(3) - умова завдання 3.

Завдання №4 Лаб# Програмування: Algotester Lab 4

Варіант завдання 1

* Деталі завдання:

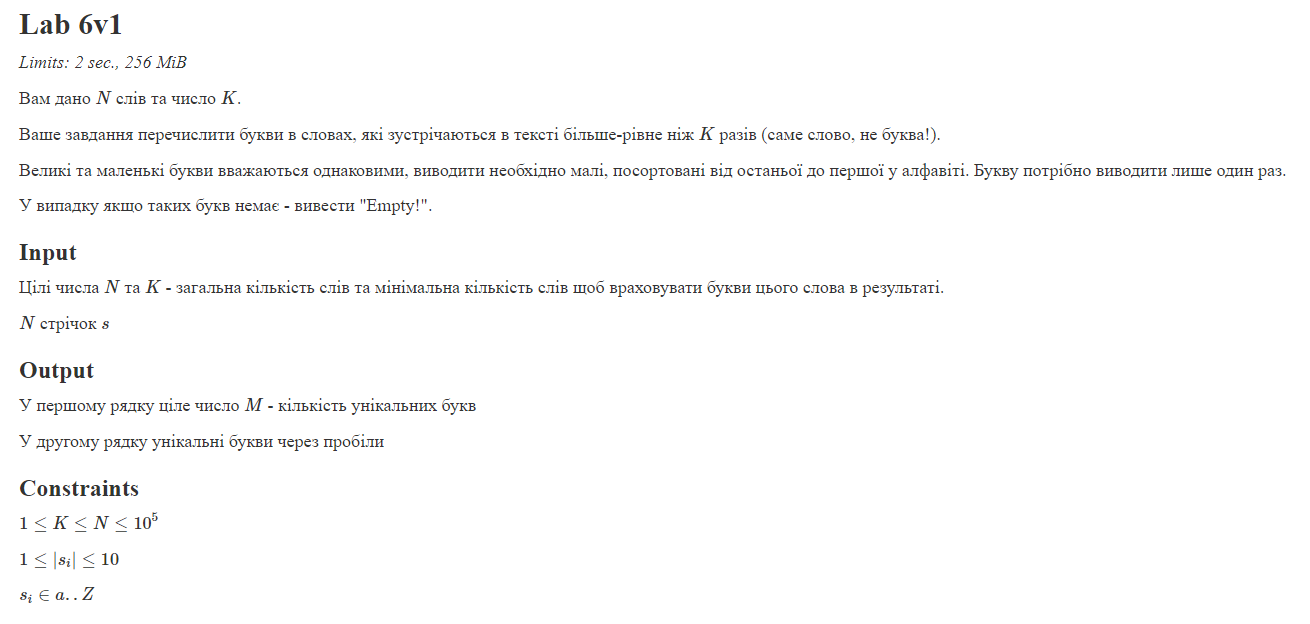


Малюнок(4) - умова завдання 4.

Завдання №5 Лаб# Програмування: Algotester Lab 6

Варіант завдання 1

* Деталі завдання:



Малюнок(5) - умова завдання 5.

Завдання №6 Практична# Програмування: Class Practice Task

Варіант завдання (відсутній)

* Деталі завдання:

## **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.

### Мета задачі

***Розуміння методів роботи з файлами:*** Робота з файлами є одним з базових навиків програмування. Реалізація функції створення та запису в файл допоможе освоїти практичні навики роботи з файлами з використанням стандартної бібліотеки С++. Для виконання завдання студент має навчитись використовувати методи відкриття файла, запису масиву даних у файл, закриття файла та обробки помилок чи станів операції на кожному з етапів.

***Розвиток алгоритмічне мислення:*** Запис у файл включає набір операції, які якнайкраще вкладаються в концепцію алгоритма, як списка детальних кроків. Імплементація цієї функції наочно демонструє створення алгоритмів у програмуванні.

***Освоїти навики роботи з текстовими стрічками:*** завдання допоможе освоїти роботу з С стрічка, які є масивами з нульовим символом в кінці. Типові концепції при роботі з С стрічками це арифметика вказівників, ітерація по стрічці, копіювання частини стрічки, розбиття на токени по заданому символу.

***Розвинути навички розв’язувати задачі:*** Запис у файл може супроводжуватись набором станів (немає доступу на створення, недостатньо місця, ін.), які необхідно передбачити у алгоритмі. Аналіз цих станів дозволяє розвинути навик розв’язання інженерних задач у програмуванні.

## **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.

### Мета задачі

***Розуміння методів роботи з файлами:*** Робота з файлами є одним з базових навиків програмування. Реалізація функції копіювання вмісту файла допоможе освоїти практичні навики роботи з файлами з використанням стандартної бібліотеки С++. Для виконання завдання студент має навчитись використовувати методи відкриття файла, читання вмісту файла, запису масиву даних у файл, закриття файла та обробки помилок чи станів операції на кожному з етапів.

***Розвиток алгоритмічне мислення:*** Читання та запис у файл включає набір операцій, які якнайкраще вкладаються в концепцію алгоритма, як списка детальних кроків. Імплементація цієї функції наочно демонструє створення алгоритмів у програмуванні.

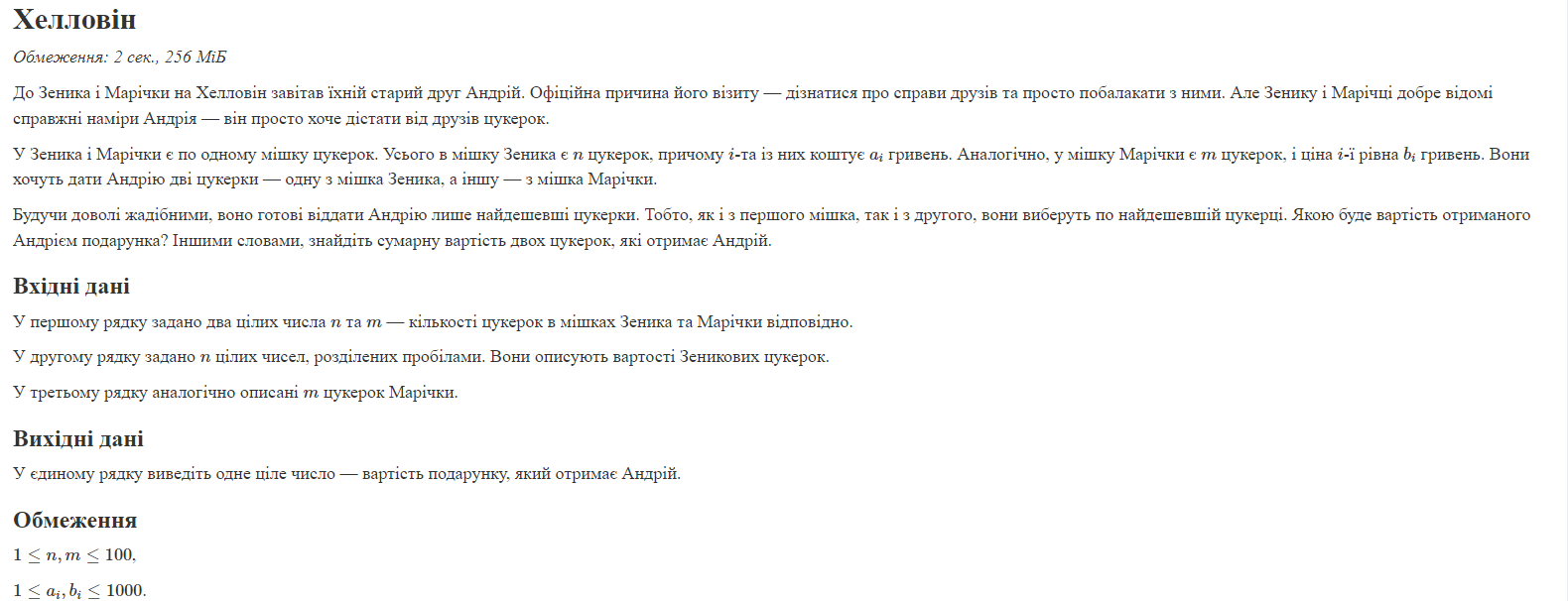
***Освоїти навики роботи з потоком даних:*** завдання допоможе освоїти роботу з потоками даних (концепція реалізована в STL як набір класів \*stream\* - fstream, stringstream, streambuf та ін.). Концепція потоку даних дозволяє абстрагувати роботу з джерелами та приймачами даних та писати з її допомогою високорівневий код.

***Розвинути навички розв’язувати задачі:*** Операції читання з файла та запис у файл можуть супроводжуватись набором різних станів (немає доступу на читання чи створення, недостатньо місця, ін.), які необхідно передбачити у алгоритмі. Аналіз цих станів дозволяє розвинути навик розв’язання інженерних задач у програмуванні.

Завдання №7 Практична# Програмування: Self Practice Task

Варіант завдання (відсутній)

* Деталі завдання:



Малюнок(6) - умова завдання 7.

Завдання №8 Лаб# Програмування: Algotester Lab 4(2)

Варіант завдання 1

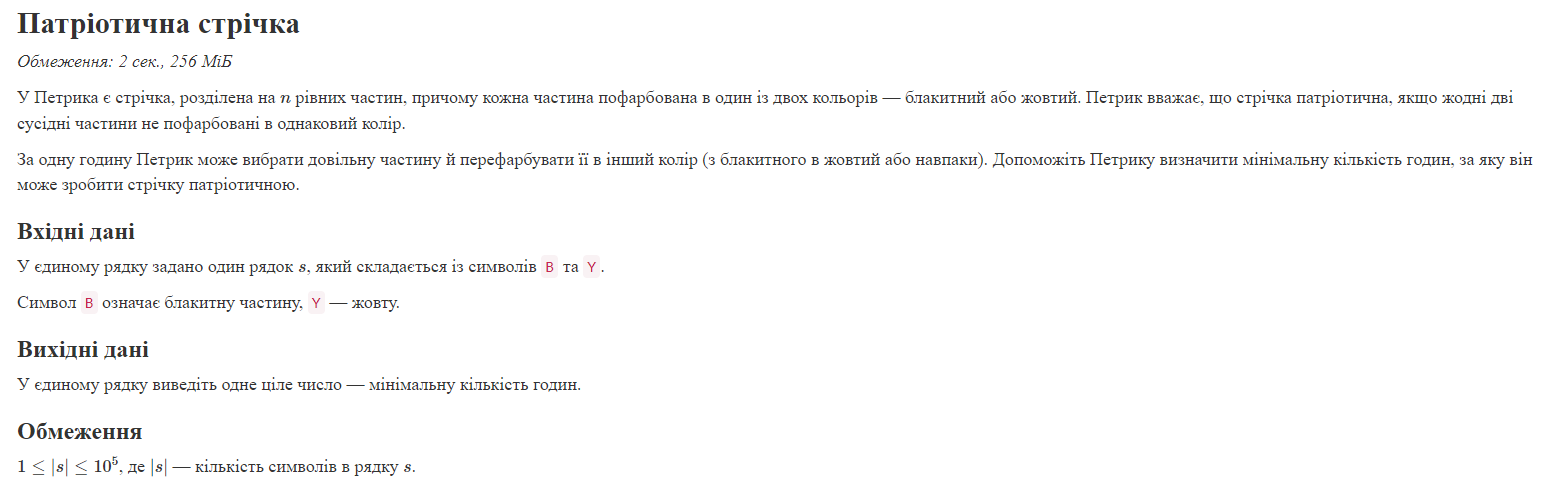
* Деталі завдання:

аналогічна задача до завдання №4 тільки без використання засобів STL (std::set\_intersection, std::set\_symmetric\_difference, std::set\_difference, std::set\_union)

Завдання №9 Практична# Програмування: Self Practice Task

Варіант завдання (відсутній)

* Деталі завдання:



Малюнок(7) - умова завдання 9.

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

Програма №1 VNS Lab 6 (Завдання 1)

* Планований час на реалізацію: 70 хвилин.

Програма №2 Лаб# Програмування: VNS Lab 8 (Завдання 1)

* Планований час на реалізацію: 400 хвилин.

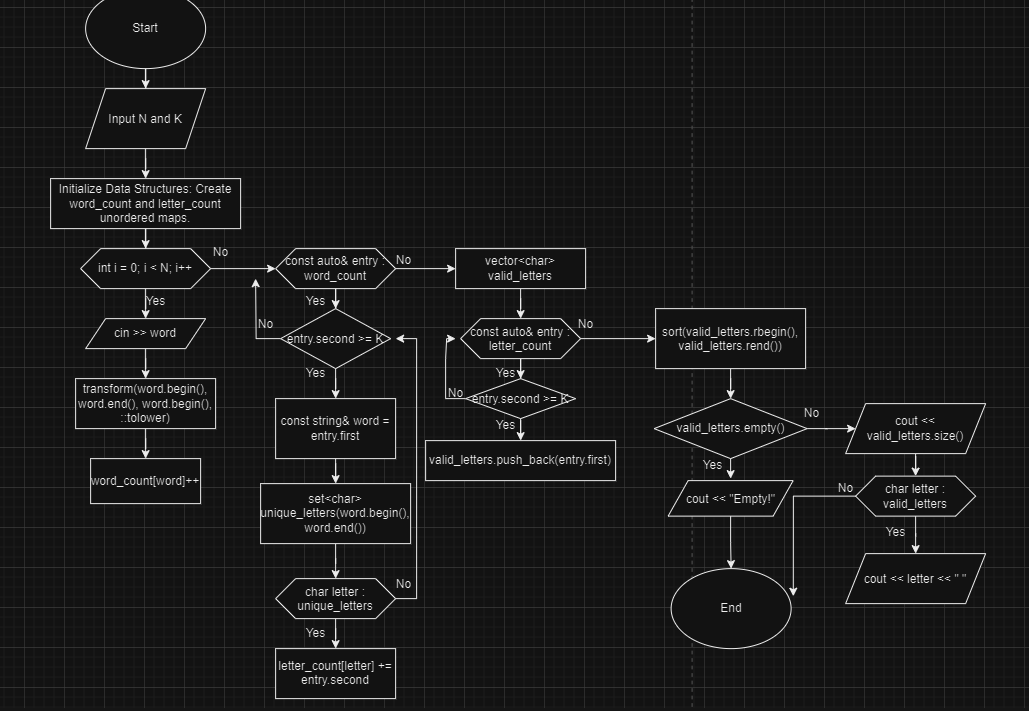
Програма №3: Лаб# Програмування: VNS Lab 9 (Завдання 1)

* Планований час на реалізацію 200 хвилин.

Програма №4: Лаб# Програмування: Algotester Lab 4

* Планований час на реалізацію 1.5 години.

Програма №5: Лаб# Програмування: Algotester Lab 6



Малюнок(8) - блок-схема до Програми № 6.

* Планований час на реалізацію 140 хвилин.

Програма №6: Практична# Програмування: Class Practice Task

* Планований час на реалізацію 80 хвилин.

Програма №7: Практична# Програмування: Self Practice Task

* Планований час на реалізацію 20 хвилин.

Програма №8: Лаб# Програмування: Algotester Lab 4(2)

* Планований час на реалізацію 100 хвилин.

Програма №9: Практична# Програмування: Self Practice Task

* Планований час на реалізацію 30 хвилин.

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

Завдання №1: Лаб# Програмування: VNS Lab 6 (Завдання 1)

#include <iostream>

#include <cstring>

#include <cctype>

using namespace std;

int main(){

char s[256];

printf("Введіть рядок: ");

gets(s);

const char \*delimiters = " .";

char \*word = strtok(s, delimiters);

while (word != NULL){

int digit\_count = 0;

for (int i = 0; i < strlen(word); i++){

if (isdigit(word[i])){

digit\_count++;

}

}

if (digit\_count == 1){

cout << word << endl;

}

word = strtok(NULL, delimiters);

}

return 0;

}

Завдання №2: Лаб# Програмування: VNS Lab 8 (Завдання 1)

#include <iostream>

#include <vector>

#include <fstream>

#include <cstring>

using namespace std;

struct State {

char name[30];

char capital[30];

int population;

double area;

};

void FillFile(const vector<State>& states, const char\* FileName) {

ofstream fout(FileName, ios::binary);

if (!fout) {

cout << "Error opening/creating the file" << endl;

return;

}

for (const State& s : states) {

fout.write(reinterpret\_cast<const char\*>(&s), sizeof(State));

}

fout.close();

}

void DeleteStatesWithLowPopulation(vector<State>& states, int populationThreshold, const char\* FileName) {

vector<State> updatedStates;

for (const State& s : states) {

if (s.population >= populationThreshold) {

updatedStates.push\_back(s);

}

}

states = updatedStates;

ofstream fout(FileName, ios::binary);

if (!fout) {

cout << "Error opening the file for writing" << endl;

return;

}

for (const State& s : updatedStates) {

fout.write(reinterpret\_cast<const char\*>(&s), sizeof(State));

}

fout.close();

}

bool AddState(vector<State>& states, const char\* FileName) {

State newState;

cout << "Enter data for the state you want to add:" << endl;

cout << "Name: ";

cin.ignore();

cin.getline(newState.name, sizeof(newState.name));

cout << "Capital: ";

cin.getline(newState.capital, sizeof(newState.capital));

cout << "Population: ";

cin >> newState.population;

cout << "Area: ";

cin >> newState.area;

string beforeStateName;

cout << "After which state would you like to insert the new one? Enter the name: ";

cin.ignore();

getline(cin, beforeStateName);

ifstream fin(FileName, ios::binary);

if (!fin) {

cout << "Error reading data from the file" << endl;

return false;

}

ofstream fout("temp.bin", ios::binary);

if (!fout) {

cout << "Error writing data to the file" << endl;

return false;

}

bool found = false;

State state;

while (fin.read(reinterpret\_cast<char\*>(&state), sizeof(State))) {

if (strcmp(state.name, beforeStateName.c\_str()) == 0) {

fout.write(reinterpret\_cast<const char\*>(&newState), sizeof(State));

found = true;

}

fout.write(reinterpret\_cast<const char\*>(&state), sizeof(State));

}

if (!found) {

cout << "State not found" << endl;

return false;

}

fin.close();

fout.close();

remove(FileName);

rename("temp.bin", FileName);

return true;

}

void PrintFile(const char\* FileName) {

ifstream fin(FileName, ios::binary);

if (!fin) {

cout << "Error reading the file" << endl;

return;

}

State state;

while (fin.read(reinterpret\_cast<char\*>(&state), sizeof(State))) {

cout << "Name: " << state.name << endl;

cout << "Capital: " << state.capital << endl;

cout << "Population: " << state.population << endl;

cout << "Area: " << state.area << endl << endl;

}

fin.close();

}

int main() {

vector<State> states = {

{"Ukraine\t", "Kyiv\t", 44000000, 603500.0},

{"Germany", "Berlin", 83000000, 357022.0},

{"France\t", "Paris\t", 67000000, 551695.0},

{"Italy\t", "Rome\t", 60300000, 301340.0},

{"Spain\t", "Madrid", 47000000, 505992.0}

};

FillFile(states, "states.bin");

PrintFile("states.bin");

DeleteStatesWithLowPopulation(states, 500000, "states.bin");

if (!AddState(states, "states.bin")) {

return 1;

}

PrintFile("states.bin");

return 0;

}

Завдання №3: Лаб# Програмування: VNS Lab 9 (Завдання 1)

#include <iostream>

#include <fstream>

#include <string>

int main() {

std::ifstream inputFile("F1.txt");

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

std::cerr << "Fail F1.txt" << std::endl;

return 1;

}

std::ofstream outputFile("F2.txt");

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

std::cerr << "Fail F2.txt" << std::endl;

inputFile.close();

return 1;

}

std::string line;

int totalCharactersInF2 = 0;

while (std::getline(inputFile, line)) {

if (!line.empty() && line.front() == line.back()) {

outputFile << line << std::endl;

totalCharactersInF2 += line.length();

}

}

inputFile.close();

outputFile.close();

std::cout << "Amount F2: " << totalCharactersInF2 << std::endl;

return 0;

}

Завдання №4: Лаб# Програмування: Algotester Lab 4

#include <iostream>

#include <vector>

#include <algorithm>

#include <set>

using namespace std;

void print\_vector(const vector<int>& v){

for (int i : v) {

cout << i << ' ';

}

cout << '\n';

}

int main(){

int N, M;

cin >> N;

if (N < 1 || N > 100){

return 1;

}

vector<int> array1(N);

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

cin >> array1[i];

}

cin >> M;

if (M < 1 || M > 100){

return 1;

}

vector<int> array2(M);

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

cin >> array2[i];

}

cout << endl;

sort(array1.begin(), array1.end());

sort(array2.begin(), array2.end());

vector<int> diff\_N\_M;

set\_difference(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(diff\_N\_M));

vector<int> diff\_M\_N;

set\_difference(array2.begin(), array2.end(), array1.begin(), array1.end(), back\_inserter(diff\_M\_N));

vector<int> intersection;

set\_intersection(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(intersection));

vector<int> union\_set;

set\_union(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(union\_set));

vector<int> symm\_diff;

set\_symmetric\_difference(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(symm\_diff));

cout << diff\_N\_M.size() << endl;

print\_vector(diff\_N\_M);

cout << diff\_M\_N.size() << endl;

print\_vector(diff\_M\_N);

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

print\_vector(intersection);

cout << union\_set.size() << endl;

print\_vector(union\_set);

cout << symm\_diff.size() << endl;

print\_vector(symm\_diff);

return 0;

}

Завдання №5: Лаб# Програмування: Algotester Lab 6

#include <iostream>

#include <string>

#include <unordered\_map>

#include <set>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

int N, K;

cin >> N >> K;

unordered\_map<string, int> word\_count;

unordered\_map<char, int> letter\_count;

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

string word;

cin >> word;

transform(word.begin(), word.end(), word.begin(), ::tolower);

word\_count[word]++;

}

for (const auto& entry : word\_count) {

if (entry.second >= K) {

const string& word = entry.first;

set<char> unique\_letters(word.begin(), word.end());

for (char letter : unique\_letters) {

letter\_count[letter] += entry.second;

}

}

}

vector<char> valid\_letters;

for (const auto& entry : letter\_count) {

if (entry.second >= K) {

valid\_letters.push\_back(entry.first);

}

}

sort(valid\_letters.rbegin(), valid\_letters.rend());

if (valid\_letters.empty()) {

cout << "Empty!" << endl;

} else {

cout << valid\_letters.size() << endl;

for (char letter : valid\_letters) {

cout << letter << " ";

}

cout << endl;

}

return 0;

}

Завдання №6: Практична# Програмування: Class Practice Task

#include <iostream>

#include <fstream>

#include <string>

enum FileOpResult { Success, Failure };

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

std::ofstream file(name, std::ios::out);

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

return Failure;

}

file << content;

file.close();

return Success;

}

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

std::ifstream source(file\_from, std::ios::binary);

std::ofstream destination(file\_to, std::ios::binary);

if (!source.is\_open() || !destination.is\_open()) {

return Failure;

}

destination << source.rdbuf();

if (!destination) {

return Failure;

}

return Success;

}

int main() {

const char \*file\_name = "file\_from.txt";

const char \*file\_content = "Hello, and wellcome to the LosPolos Germanos family, my name is Gustavo but you can call me Gus!";

const char \*file\_copy\_from = "file\_from.txt";

const char \*file\_copy\_to = "file\_to.txt";

// Завдання №1 - Запис у файл

FileOpResult write\_result = write\_to\_file(file\_name, file\_content);

if (write\_result == Success) {

std::cout << "Write to file: Success" << std::endl;

} else {

std::cout << "Write to file: Failure" << std::endl;

}

// Завдання №2 - Копіювання файлу

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

if (copy\_result == Success) {

std::cout << "Copy file: Success" << std::endl;

} else {

std::cout << "Copy file: Failure" << std::endl;

}

return 0;

}

Завдання №7: Практична# Програмування: Self Practice Task

#include <iostream>

using namespace std;

int main()

{

int N;

int M;

cin >> N >> M;

int arr1[N];

int arr2[M];

int min = 1000;

int res;

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

{

cin >> arr1[i];

if(arr1[i]<min)

min = arr1[i];

}

res = min;

min = 1000;

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

{

cin >> arr2[i];

if(arr2[i]<min)

min = arr2[i];

}

res+=min;

cout << res;

return 0;

}

Завдання №8: Лаб# Програмування: Algotester Lab 4(2)

#include <iostream>

#include <vector>

#include <algorithm>

#include <set>

using namespace std;

int main() {

int N, M;

cin >> N;

if (N < 1 || N > 100){

return 1;

}

vector<int> array1(N);

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

cin >> array1[i];

}

cin >> M;

if (M < 1 || M > 100){

return 1;

}

vector<int> array2(M);

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

cin >> array2[i];

}

cout << endl;

vector<int> differenceNM;

set\_difference(array1.begin(), array1.end(), array2.begin(), array2.end(), back\_inserter(differenceNM));

vector<int> differenceMN;

set\_difference(array2.begin(), array2.end(), array1.begin(), array1.end(), back\_inserter(differenceMN));

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

for (int num : differenceNM){

cout << num << " ";

}

cout << endl;

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

for (int num : differenceMN){

cout << num << " ";

}

cout << endl;

set<int> intersectionSet;

for (int num : array1){

if (find(array2.begin(), array2.end(), num) != array2.end()){

intersectionSet.insert(num);

}

}

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

for (int num : intersectionSet){

cout << num << " ";

}

cout << endl;

set<int> unionSet(array1.begin(), array1.end());

unionSet.insert(array2.begin(), array2.end());

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

for (int num : unionSet){

cout << num << " ";

}

cout << endl;

vector<int> symmetricDifference;

for (int num : array1){

if (find(array2.begin(), array2.end(), num) == array2.end()){

symmetricDifference.push\_back(num);

}

}

for (int num : array2) {

if (find(array1.begin(), array1.end(), num) == array1.end()){

symmetricDifference.push\_back(num);

}

}

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

for (int num : symmetricDifference){

cout << num << " ";

}

cout << endl;

return 0;

}

Завдання №9: Практична# Програмування: Self Practice Task

#include <iostream>

using namespace std;

int main()

{

string s;

cin >> s;

int length = s.length();

char masB[length];

char masY[length];

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

if(i%2==0){

masB[i]='B';

masY[i]='Y';

}

else{

masB[i]='Y';

masY[i]='B';

}

}

int counterForB=0;

int counterForY=0;

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

if(masB[i]!=s.at(i))

counterForB++;

if(masY[i]!=s.at(i))

counterForY++;

}

if(counterForB<counterForY)

cout << counterForB;

else

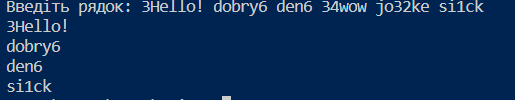
cout << counterForY;

return 0;

}

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

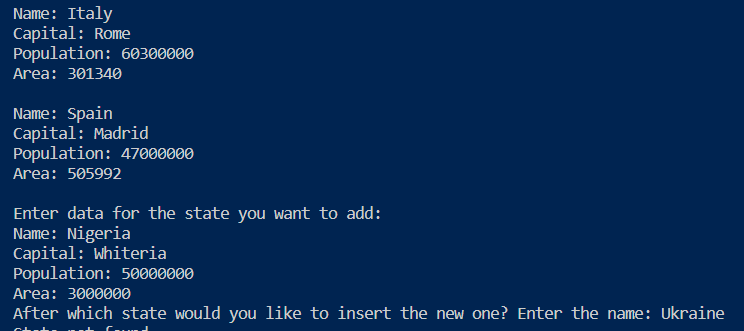
Завдання №1: Лаб# Програмування: VNS Lab 6 (Завдання 1)



Малюнок(9) - результат виконання завдання 1.

Затрачений час: 200 хв.

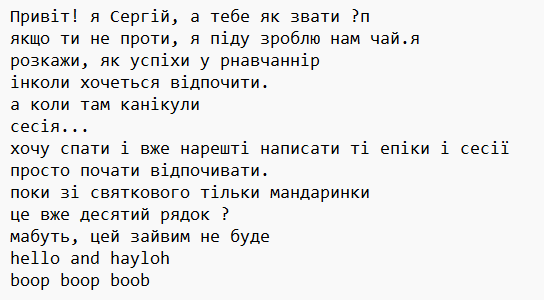
Завдання №2: Лаб# Програмування: VNS Lab 8 (Завдання 1)



Малюнок(10) - результат виконання завдання 2.

Затрачений час: 5 годин.

Завдання №3: Лаб# Програмування: VNS Lab 9 (Завдання 1)



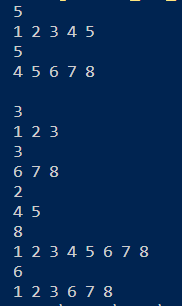




Малюнок(11) - результат виконання завдання 3.

Затрачений час: 3 години.

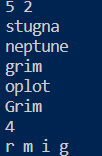
Завдання №4: Лаб# Програмування: Algotester Lab 4



Малюнок(12) - результат виконання завдання 4.

Затрачений час: 2 години.

Завдання №5: Лаб# Програмування: Algotester Lab 6



Малюнок(13) - результат виконання завдання 5.

Затрачений час: 360 хвилин.

Завдання №6: Практична# Програмування: Class Practice Task



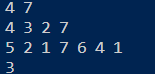


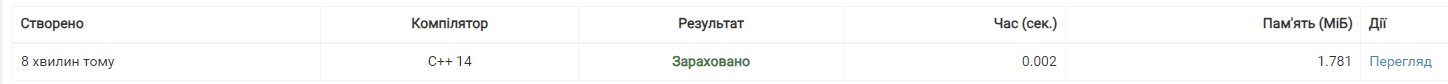


Малюнок(14) - результат виконання завдання 6.

Затрачений час: 300 хвилин.

Завдання №7: Практична# Програмування: Self Practice Task

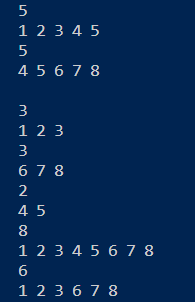




Малюнок(15) - умова завдання 7.

Затрачений час: 30 хвилин.

Завдання №8: Лаб# Програмування: Algotester Lab 4(2)

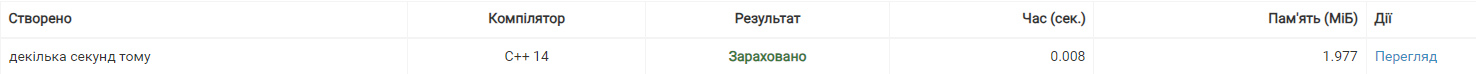


Малюнок(16) - результат виконання завдання 8.

Затрачений час: 600 хвилин.

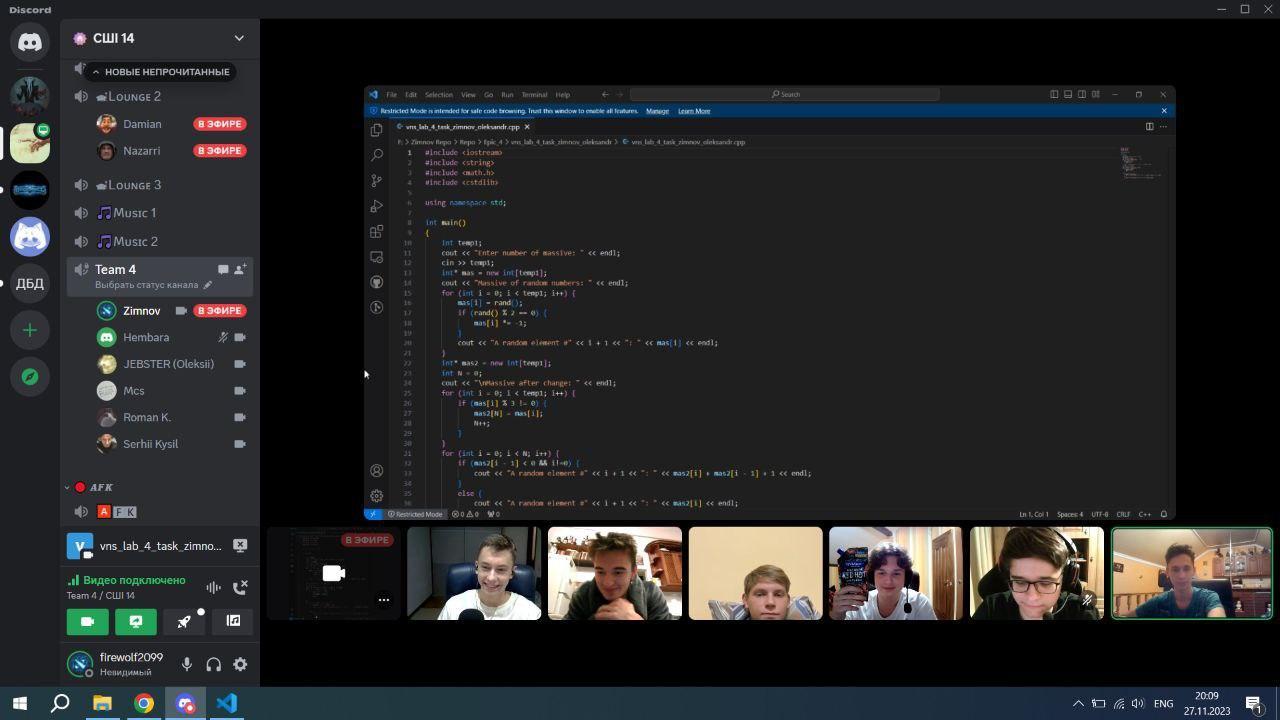
Завдання №9: Практична# Програмування: Self Practice Task

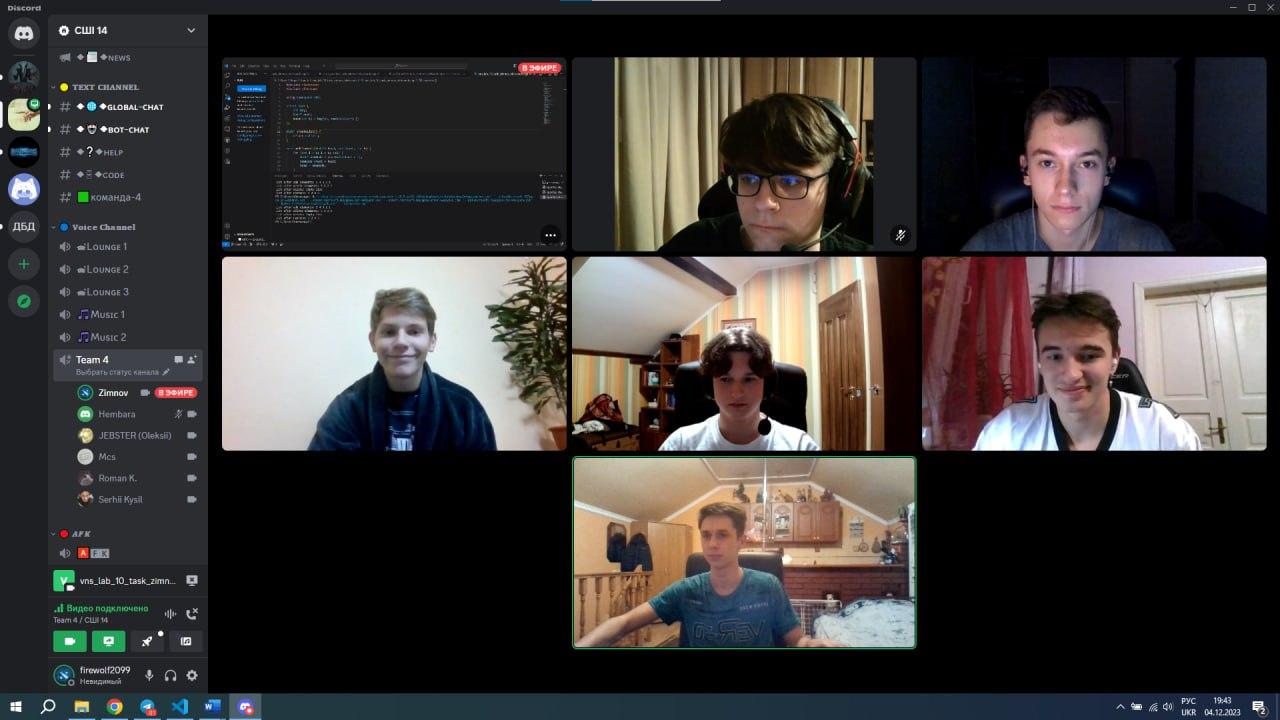


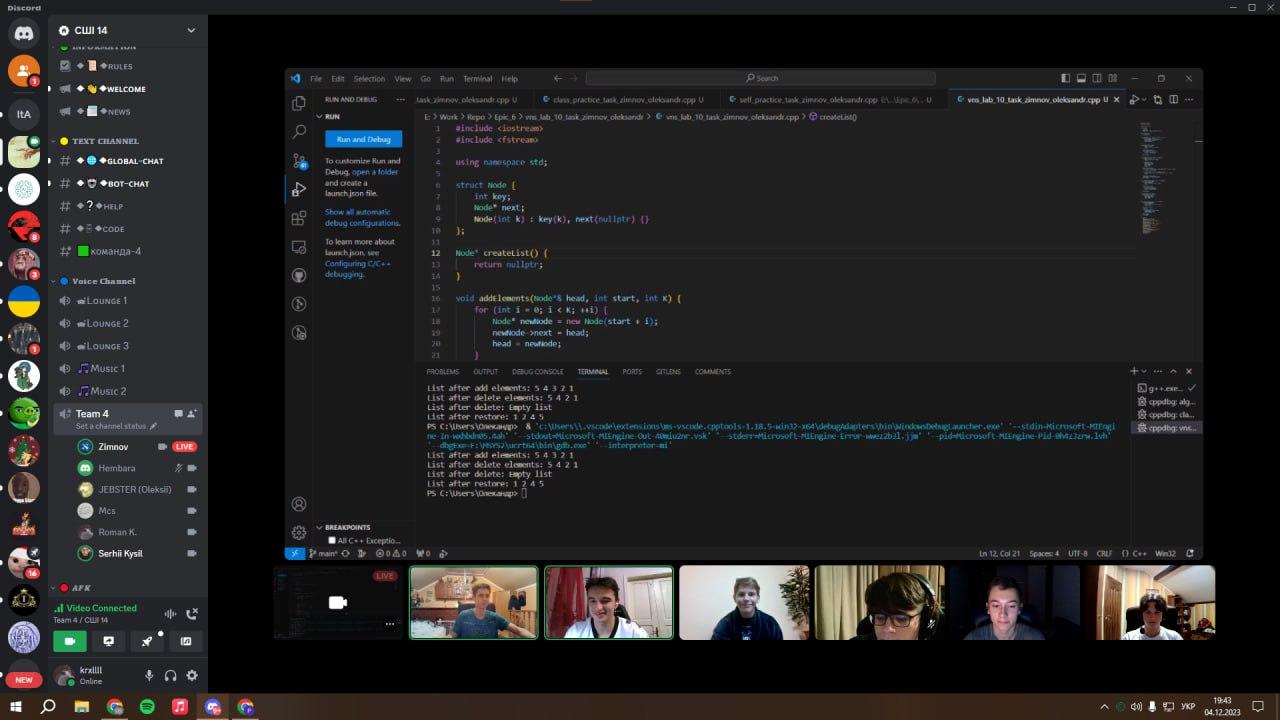


Малюнок(17) - умова завдання 9.

Затрачений час: 30 хвилин.







Малюнки(18-20) - скріни-підтвердження активності команди.

# **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.

PR: <https://github.com/artificial-intelligence-department/ai_programming_playground/pull/815>