Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

NO_CAPTION null

**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»

***Виконала:***

студентка групи ШІ-11

Труш Соломія Володимирівна

# **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:**

Засвоїти теоретичні знання по темі за виконанням практичних завдань у вигляді лабораторних.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: **Файли, системи числення.**
* Тема №2: **Символи і рядки**.
* Тема №3: **Стандартна бібліотека. Створення і використання бібліотек.**

1. Індивідуальний план опрацювання теорії:

* Тема №1: **Файли, системи числення.**
  + Джерела Інформації
    - Стаття. <https://www.udacity.com/blog/2023/04/c-streams.html#:~:text=In%20C%2B%2B%2C%20a%20stream,and%20other%20I%2FO%20devices>.
    - Лекційний матеріал на ВНС
  + Що опрацьовано:
    - Файли, робота з файлами, системи числення
  + Статус: Ознайомлений
  + Початок опрацювання теми: 10.12
  + Звершення опрацювання теми: 14.12
* Тема №2: **Символи і рядки**.
  + Джерела Інформації:
    - Відео.
    - Стаття 1. <https://acode.com.ua/urok-82-ryadky-c-style/>
    - Стаття 2. <https://www.digitalocean.com/community/tutorials/reverse-string-c-plus-plus>
    - Лекційний матеріал на ВНС
    - [chat.openai.com](https://chat.openai.com/)
  + Що опрацьовано:
    - Символи і рядки, робота з ними
  + Статус: Ознайомлений
  + Початок опрацювання теми: 10.12
  + Звершення опрацювання теми: 14.12
* Тема №3: **Стандартна бібліотека. Створення і використання бібліотек.**
  + Джерела Інформації:
    - Відео. <https://www.youtube.com/watch?v=oNnatJtPWjI&ab_channel=code_report>
    - Стаття 1. <https://www.programiz.com/cpp-programming/set>
    - Стаття 2. <https://www.programiz.com/cpp-programming/map>
    - Лекційний матеріал на ВНС
    - [chat.openai.com](https://chat.openai.com/)
  + Що опрацьовано:
    - Контейнери set, map, функції set\_difference, set\_intersection, set\_symmetric\_difference, set\_union
  + Статус: Ознайомлений
  + Початок опрацювання теми: 10.12
  + Звершення опрацювання теми: 14.12

# **Виконання роботи:**

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

Завдання **VNS Lab 6 - Task 1**

* Варіант 11
* Деталі завдання

11. Перетворити рядок таким чином, щоб всі слова в ньому були надруковані

навпаки.

Завдання **VNS Lab 8 - Task 1**

* Варіант 11
* Деталі завдання

Сформувати двійковий файл із елементів, заданої у варіанті структури,

роздрукувати його вмістиме, виконати знищення й додавання елементів у

відповідності зі своїм варіантом, використовуючи для пошуку елементів що

знищуються чи додаються, функцію. Формування, друк, додавання й знищення

елементів оформити у вигляді функцій. Передбачити повідомлення про

помилки при відкритті файлу й виконанні операцій вводу/виводу.

* Важливі деталі для врахування в імплементації програми

Структура "Відеокасета":

- назва фільму;

- режисер;

- тривалість;

- ціна.

Знищити всі елементи із ціною вищою заданої, додати 3 елементи в кінець

файлу.

Завдання **VNS Lab 9 - Task 1**

* Варіант 11
* Деталі завдання

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

інформацію.

1) Скопіювати з файлу F1 у файл F2 всі рядки, крім того рядка, що містить

найкоротше слово.

2) Надрукувати номер цього рядка.

* Важливі деталі для врахування в імплементації програми

Завдання **Algotester Lab 4**

* Варіант 1
* Деталі завдання

Вам дано 2 цілих чисел масиви, розміром N M

Ваше завдання вивести:

1. Різницю N-M

2. Різницю M-N

3. Їх перетин

4. Їх об'єднання

5. Їх симетричну різницю

Вивести результат виконання 5 вищезазначених операцій у форматі:

У першому рядку ціле число N - розмір множини. У наступному рядку N цілих чисел - посортована у порядку зростання множина.

* Важливі деталі для врахування в імплементації програми

1<=N,M<=100  
1<=n,m<=100

Завдання **Algotester Lab 6**

* Варіант 1
* Деталі завдання

Вам дано N слів та число K.

Ваше завдання перечислити букви в словах, які зустрічаються в тексті більше-рівне ніж K разів (саме слово, не буква!).

Великі та маленькі букви вважаються однаковими, виводити необхідно малі, посортовані від останньої до першої у алфавіті. Букву потрібно виводити лише один раз.

У випадку якщо таких букв немає - вивести "Empty!".

* Важливі деталі для врахування в імплементації програми

1<=K<=M<=10^5

1<=|s|<=10

Завдання **Class Practice Work**

* Деталі завдання

1. Реалізувати функцію створення файла і запису в нього даних:

enum FileOpResult { Success, Failure, … };

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

Умови задачі:

- створити файл із заданим ім’ям; якщо файл існує – перезаписати його вміст

- написати код стійкий до різних варіантів вхідних параметрів

- name – ім’я, може не включати шлях

- записати у файл вміст стрічки content, прочитати content із стандартного вводу

- повернути статус операції: Success – все пройшло успішно, Failure – файл не вдалося створити, або збій операції відкриття, запису даних, чи закриття файла.

1. 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.

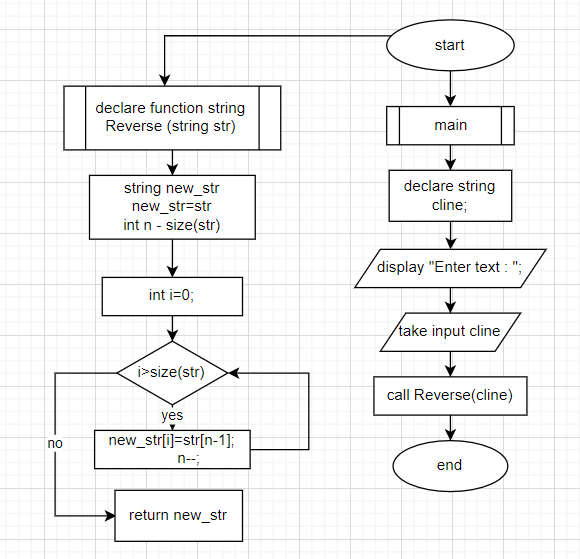
Завдання **Self Practice Work**

* Деталі завдання
* Важливі деталі для врахування в імплементації програми

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

Програма **VNS Lab 6 - Task 1**

* Блок-схема



*Figure 1. Flowchart for VNS Lab 6 - Task 1*

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

Середовище сконфігуровано під час виконання Epic 1.

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

Завдання **VNS Lab 6 - Task 1**

Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <string>

using namespace std;

string Reverse (string str) {

string new\_str;

new\_str=str;

int n = size(str);

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

new\_str[i]=str[n-1];

n--;

}

return new\_str;

}

int main () {

string cline;

cout << "Enter text : ";

getline(cin, cline);

cout << Reverse(cline);

return 0;

}

Завдання **VNS Lab 8 - Task 1**

Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <vector>

#include <fstream>

using namespace std;

struct Video\_Tape {

string title;

string director;

double duration;

double price;

};

void print\_file(const vector<Video\_Tape>& tapes) {

for (const auto& tape : tapes) {

cout << tape.title << " " << tape.director << " " << tape.duration << " " << tape.price << endl;

}

}

void write\_file(const vector<Video\_Tape>& tapes, const string& filename) {

ofstream file(filename, ios::binary | ios::out);

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

cerr << "Error opening file for writing" << endl;

return;

}

for (const auto& tape : tapes) {

file.write(reinterpret\_cast<const char\*>(&tape), sizeof(Video\_Tape));

}

file.close();

}

void Delete\_high\_price(vector<Video\_Tape>& tapes, double high\_price) {

vector<Video\_Tape> newTapes;

for (const auto& tape : tapes) {

if (!(tape.price > high\_price)) {

newTapes.push\_back(tape);

}

}

tapes = move(newTapes);

}

void AddTape (vector<Video\_Tape>& tapes) {

for (int i=0; i<3; i++) {

Video\_Tape newTape;

cout << endl;

cout << "Enter title: " << endl;

cin >> newTape.title;

cout << "Enter director: " << endl;

cin >> newTape.director;

cout << "Enter duration: " << endl;

cin >> newTape.duration;

cout << "Enter price: " << endl;

cin >> newTape.price;

tapes.push\_back(newTape);

}

}

int main() {

vector<Video\_Tape> tapes = {

{"Kill Bill", "Quentin Tarantino", 120, 9.5},

{"Openheimer", "Christopher Nolan", 150, 15.7},

{"Thor 4", "Taika Waititi", 150, 11.2},

};

cout << "Original file : " << endl;

print\_file(tapes);

double high\_price;

cout << endl;

cout << "Enter the maximum price : ";

cin >> high\_price;

Delete\_high\_price(tapes, high\_price);

cout << endl;

cout << "File after deletion : " << endl;

print\_file(tapes);

AddTape(tapes);

cout << endl;

cout << "File after adding new tapes : " << endl;

print\_file(tapes);

write\_file(tapes, "video\_tapes");

return 0;

}

Завдання **VNS Lab 9 - Task 1**

Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <fstream>

#include <sstream>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

ofstream fileF1("F1.txt");

if (!fileF1.is\_open()) {

cerr << "Unable to open " << endl;

return 1;

}

fileF1 << "This is a sample text" << endl;

fileF1 << "Hello, world!" << endl;

fileF1 << "Shortest word" << endl;

fileF1 << "Programming" << endl;

fileF1 << "This file contains some text" << endl;

fileF1 << "Clamor Spe" << endl;

fileF1 << "Quick brown fox" << endl;

fileF1 << "Slow old turtle" << endl;

fileF1 << "Hide and seek" << endl;

fileF1 << "End of the file" << endl;

fileF1.close();

ifstream fileF1Read("F1.txt");

if (!fileF1Read.is\_open()) {

cerr << "Unable to open " << endl;

return 1;

}

string line;

string shortestWord;

int shortestLine = 0;

int currentLine = 0;

while (getline(fileF1Read, line)) {

currentLine++;

istringstream iss(line);

vector<string> words;

string word;

while (iss >> word) {

words.push\_back(word);

}

for (const auto& word : words) {

if (shortestWord.empty() || word.length() < shortestWord.length()) {

shortestWord = word;

shortestLine = currentLine;

}

}

}

fileF1Read.close();

fileF1Read.open("F1.txt");

ofstream fileF2("F2.txt");

if (!fileF1Read.is\_open() || !fileF2.is\_open()) {

cerr << "Unable to open files for reading or writing." << endl;

return 1;

}

currentLine = 0;

while (getline(fileF1Read, line)) {

currentLine++;

if (currentLine != shortestLine) {

fileF2 << line << endl;

}

}

cout << "Line with the shortest word: " << shortestLine << endl;

fileF1Read.close();

fileF2.close();

return 0;

}

Завдання **Algotester Lab 4**

Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main () {

vector<int> U, I, D1, D2, DS;

int n, m;

cin >> n;

vector<int> N(n);

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

cin >> N[i];

}

cin >> m;

vector<int> M(m);

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

cin >> M[j];

}

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

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

set\_difference(N.begin(), N.end(), M.begin(), M.end(), back\_inserter(D1)); // різниця N-M

set\_difference(M.begin(), M.end(), N.begin(), N.end(), back\_inserter(D2)); // M-N

set\_intersection(N.begin(), N.end(), M.begin(), M.end(), back\_inserter(I)); // перетин

set\_union(N.begin(), N.end(), M.begin(), M.end(), back\_inserter(U)); // об'єднання

set\_symmetric\_difference(N.begin(), N.end(), M.begin(), M.end(), back\_inserter(DS)); // симетрична різниця

cout << "\n" << D1.size() << endl;

for (auto i = D1.begin(); i!=D1.end(); i++) {

cout << \*i << " " ;

}

cout << endl;

cout << "\n" << D2.size() << endl;

for (auto i = D2.begin(); i!=D2.end(); i++) {

cout << \*i << " " ;

}

cout << endl;

cout << "\n" << I.size() << endl;

for (auto i = I.begin(); i!=I.end(); i++) {

cout << \*i << " " ;

}

cout << endl;

cout << "\n" << U.size() << endl;

for (auto i = U.begin(); i!=U.end(); i++) {

cout << \*i << " " ;

}

cout << endl;

cout << "\n" << DS.size() << endl;

for (auto i = DS.begin(); i!=DS.end(); i++) {

cout << \*i << " " ;

}

cout << endl;

}

Завдання **Algotester Lab 6**

Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <vector>

#include <algorithm>

#include <map>

#include <set>

using namespace std;

int main() {

int n, k;

cin >> n >> k;

vector<string> words(n);

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

cin >> words[i];

transform(words[i].begin(), words[i].end(), words[i].begin(), ::tolower);

}

map<string, int> wordCount;

set<string> rep;

set<char> reeesult;

for (const string& word : words) {

wordCount[word]++;

if (wordCount[word] >= k) {

rep.insert(word);

for (char c : word) {

reeesult.insert(c);

}

}

}

if (rep.empty()) {

cout << "Empty!";

return 0;

}

vector<char> result(reeesult.begin(), reeesult.end());

sort(result.rbegin(), result.rend());

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

for (char a : result) {

cout << a << " ";

}

return 0;

}

Завдання **Class Practice Work**

Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

enum FileOpResult { Success, Failure };

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

if (!name || !content) {

cerr << "Invalid input: Name or content is null." << endl;

return Failure;

}

ofstream file(name);

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

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

return Failure;

}

file << content;

if (file.fail()) {

cerr << "Error writing to file." << endl;

file.close();

return Failure;

}

file.close();

return Success;

}

int main() {

const char\* filename = "source.txt";

string fileContent;

cout << "Enter content: ";

getline(cin, fileContent);

FileOpResult result = write\_to\_file(filename, fileContent.c\_str());

if (result == Success) {

cout << "File successfully created or updated." << endl;

} else {

cout << "Failed to create or update the file." << endl;

}

return 0;

}

#include <iostream>

#include <fstream>

using namespace std;

enum FileOpResult { Success, Failure };

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

if (!file\_from || !file\_to) {

cerr << "Invalid input: file\_from or file\_to is null." << endl;

return Failure;

}

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

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

cerr << "Error opening source file: " << file\_from << endl;

return Failure;

}

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

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

cerr << "Error opening destination file: " << file\_to << endl;

source.close();

return Failure;

}

destination << source.rdbuf();

if (destination.fail()) {

cerr << "Error copying file content." << endl;

source.close();

destination.close();

return Failure;

}

source.close();

destination.close();

return Success;

}

int main() {

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

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

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

if (result == Success) {

cout << "File successfully copied." << endl;

} else {

cout << "Failed to copy the file." << endl;

}

return 0;

}

Завдання **Self Practice Work**

Посилання на файл програми у пул-запиті GitHub

#include <iostream>

using namespace std;

int main () {

unsigned m, n, res;

cin >> m >> n;

res = m\*n;

if (res%2==0) {

cout << "Dragon";

} else {

cout << "Imp";

}

}

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

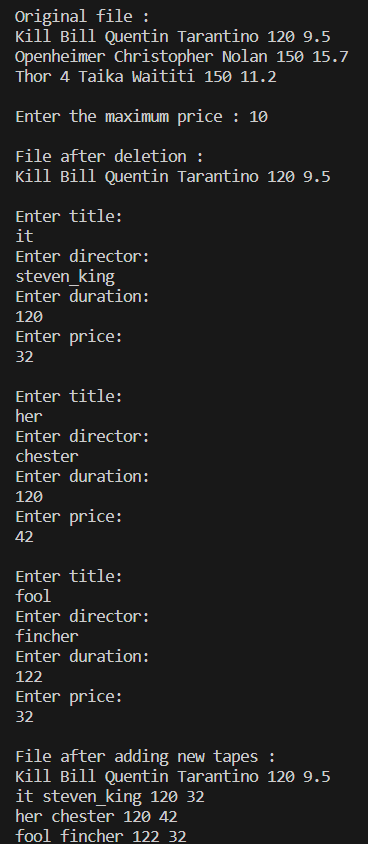
Завдання **VNS Lab 6 - Task 1** варіант 11



*Figure 2. VNS Lab 6 results*

Планований час виконання - 30хв , фактичний - 20хв

Завдання **VNS Lab 8 - Task 1** варіант 11



*Figure 3. VNS Lab 8 results*

Планований час виконання - 3 год , фактичний - 4 год

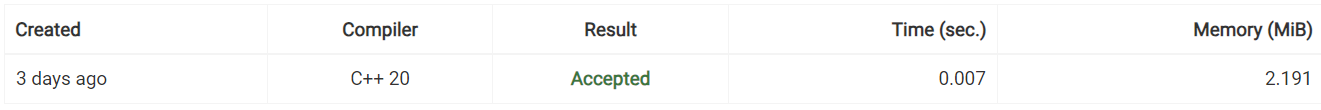
Завдання **VNS Lab 9 - Task 1** варіант 11



*Figure 4. VNS Lab 9 results*

Планований час виконання - 3 год , фактичний - 3 год

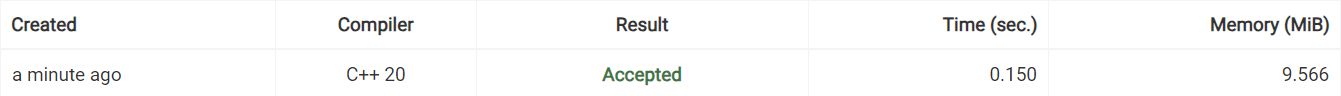
Завдання **Algotester Lab 4** варіант 1



*Figure 5. Algotester Lab 4 results*

Планований час виконання - 2 год , фактичний - 3 год

Завдання **Algotester Lab 6** варіант 1



*Figure 6. Algotester Lab 6 results*

Планований час виконання - 3 год , фактичний - 6 год

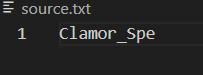
Завдання **Class Practice Work**

****

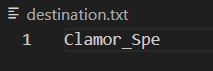
*Figure 7. Class Practice results*

****

*Figure 8. Class Practice results*

****

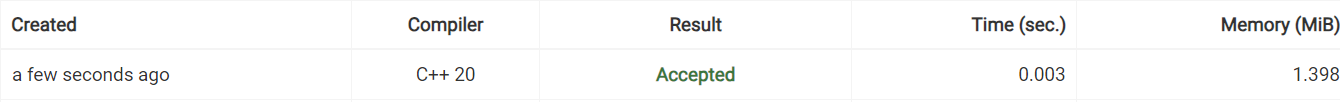
*Figure 9. Class Practice results*

****

*Figure 10. Class Practice results*

Планований час виконання - 3 год , фактичний - 4 год

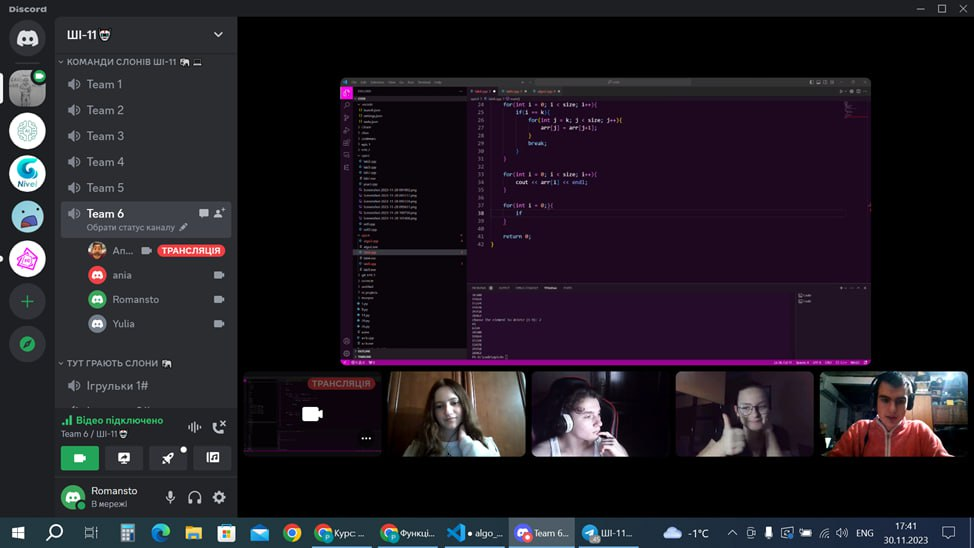
Завдання **Self Practice Work**

****

*Figure 11. Self Practice results*

Планований час виконання - 15 хв , фактичний - 15 хв

## **6. Кооперація з командою:**



*Figure 12. Team Cooperation*

# **Висновки:**

Ознайомилась з файлами і принципами роботи з ними, рядками і символами, стандартною бібліотекою і деякими її функціями.