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-13

Степанюк Артем Костянтинович

# **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:** Вивчити операції з файлами та системи числення, розглянути бінарні файли, символьні та рядкові змінні, а також текстові файли.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Системи числення.
* Тема №2: Бінарні файли.
* Тема №3: Символи і Рядкові Змінні та Текстові Файли
* Тема №4: Стандартна бібліотека та деталі/методи роботи з файлам.
* Тема №5: Створення й використання бібліотек.

1. Індивідуальний план опрацювання теорії:

* Тема №1: Системи числення
  + Джерела Інформації
    - [C++ Теорія ⦁ Урок 68 ⦁ Системи числення (youtube.com)](https://www.youtube.com/watch?v=c5FSYMpsLN0)
  + Що опрацьовано:
    - Вивчив системи числення по відео на ютубі
  + Статус: Ознайомлено
  + Початок опрацювання теми: 12.12.2023
  + Звершення опрацювання теми: 14.12.2023
* Тема №2: Бінарні файли
  + Джерела Інформації:
    - [C++ Tutorial for Beginners 29 - Binary Files in C++ | Structure In Binary File I/O - YouTube](https://www.youtube.com/watch?v=tyXq2d5g8-0)
  + Що опрацьовано:
    - Вивчив використання бінарних файлів по відео на ютубі
  + Статус: Ознайомлено
  + Початок опрацювання теми: 12.12.2023
  + Звершення опрацювання теми: 14.12.2023
* Тема №3: Символи і Рядкові Змінні та Текстові Файли
  + Джерела Інформації:
    - [Reading and Writing to Files (ifstream and ofstream) - C++ Tutorial 25 - YouTube](https://www.youtube.com/watch?v=Cz4fl-TUjVk)
  + Що опрацьовано:
    - Ознайомився з символами і Рядковіми Змінами та Текстовими Файлами по роліку на ютубі
  + Статус: Ознайомлено
  + Початок опрацювання теми: 12.12.2023
  + Звершення опрацювання теми: 14.12.2023
* Тема №4: Стандартна бібліотека та деталі/методи роботи з файлам
  + Джерела Інформації:
    - [Header Files - C++ Tutorial For Beginners #14 - YouTube](https://www.youtube.com/watch?v=0TCh01BBlrM)
  + Що опрацьовано:
    - Вивчив застосування стандартних бібліотек та деталей роботи з файлами по відеоролику на ютубі
  + Статус: Ознайомлено
  + Початок опрацювання теми: 12.12.2023

Звершення опрацювання теми: 14.12.2023

* Тема №5: Створення й використання бібліотек.
  + Джерела Інформації:
    - [Using Libraries in C++ (Static Linking) - YouTube](https://www.youtube.com/watch?v=or1dAmUO8k0)
  + Що опрацьовано:
    - Вивчив створення та використання бібліотек по відеоролику на ютубі
  + Статус: Ознайомлено
  + Початок опрацювання теми: 12.12.2023

Звершення опрацювання теми: 14.12.2023

# **Виконання роботи:**

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

Завдання №1 РЯДКИ

**Деталі завдання:**

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.

**Важливі деталі для врахування в імплементації програми:**

Визначити яке слово зустрічається в рядку найчастіше.

**Завдання №2** Блоковий ввід-вивід

**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 "Автомобіль":

- марка;

- рік випуску;

- ціна;

- кольори.

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

початок файлу.

**Завдання №3 Ввід-вивід рядків**

**Деталі завдання:** Створити текстовий файл F1 не менше, ніж з 10 рядків і записати в нього

інформацію

**Важливі деталі для врахування в імплементації програми:**

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

розташовані між рядками з номерами N1 й N2.

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

**Завдання №4 Algotester Lab 4v2**.

**Деталі завдання:** Вам дано масивaзNцiлих чисел.Спочатку видалiть масивуaусi елементи що повторюються, наприклад масив [1, 3, 3, 4] маєперетворитися у [1, 3, 4].Пiсля цього обернiть посортовану версiю масивуaнаK, тобто приK= 3масив [1, 2, 3, 4, 5, 6,7] перетвориться на [4, 5, 6, 7, 1, 2, 3].Виведiть результат.

**Важливі деталі для врахування в імплементації програми:**

Вхiднi данi:

У першому рядку цiлi числаNтаKУ другому рядкуNцiлих чисел - елементи масивуaВихiднi данiУ першому рядку цiле числоN- розмiр множиниaУ наступному рядку N

цiлих чисел - множина

Обмеження

1≤N, K≤10000≤ai≤100

**Завдання №5 Algotester 6v2**

**Деталі завдання:** У вас є шахова дошка розміром

8×8 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 si - 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.

**Важливі деталі для врахування в імплементації програми:**

**Вхiднi данi**

**У перших 8 рядках стрiчка rowi- станi-го рядка дошки.**

**У наступному рядку цiле число Q- кiлькiсть записiв**

**У наступних Q рядках 2 цiлих числаx та y- координати клiтинки**

**Вихiднi данi Q разiв вiдповiдь у наступному форматi:**

**Строка result- усi фiгури, якi атакують клiтинку з запиту.**

**Обмеження|rowi|=Nrowi∈{O, P, R, N, B, K, Q}1≤Q≤641≤x, y≤8**

**Завдання №6 Практична**

**Деталі завдання:** Реалізувати функцію створення файла і запису в нього даних:

enum FileOpResult { Success, Failure, … };

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

**Важливі деталі для врахування в імплементації програми:**

Умови задачі:

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

- написати код стійкий до різних варіантів вхідних параметрів

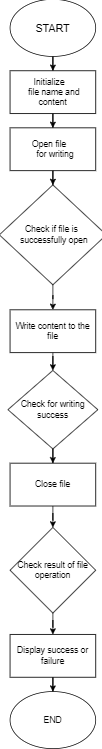
- name – ім’я, може не включати шлях

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

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

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

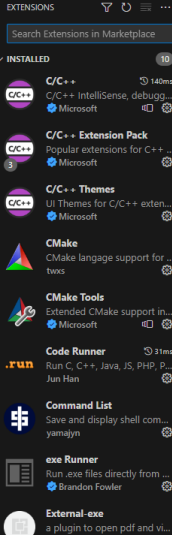
Завдання №6: Практична



*Блок-схема для практичної*

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

Завдання №1. Конфігурація середовища

. ****

*Конфігурація мого середовища*

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

**Завдання №1: РЯДКИ**

**#include <iostream>**

**#include <string>**

**#include <vector>**

**#include <map>**

**int main() {**

**std::string s;**

**std::cout << "Введіть строку: ";**

**std::getline(std::cin, s);**

**std::vector<std::string> words;**

**for (size\_t i = 0; i < s.length(); i++) {**

**if (s[i] != ' ') {**

**std::string word = "";**

**while (i < s.length() && s[i] != ' ') {**

**word += s[i];**

**i++;**

**}**

**words.push\_back(word);**

**}**

**}**

**std::map<std::string, int> word\_counts;**

**for (const std::string &word : words) {**

**word\_counts[word]++;**

**}**

**int max\_count = 0;**

**std::string most\_common\_word;**

**for (const auto &pair : word\_counts) {**

**if (pair.second > max\_count) {**

**max\_count = pair.second;**

**most\_common\_word = pair.first;**

**}**

**}**

**std::cout << "Найчастіше зустрічається слово: " << most\_common\_word << std::endl;**

**return 0;**

**}**

**Завдання №2 Блоковий ввід-вивід**

#include <iostream>

#include <fstream>

#include <cstring>

using namespace std;

struct Car {

char brand[50];

int year;

float price;

char color[20];

};

void printCar(const Car& car) {

cout << "Марка: " << car.brand << "\n";

cout << "Рік випуску: " << car.year << "\n";

cout << "Ціна: " << car.price << "\n";

cout << "Колір: " << car.color << "\n\n";

}

void createFile(const char\* filename, const Car\* cars, int size) {

ofstream file(filename, ios::binary);

if (!file) {

cerr << "Помилка при відкритті файлу для запису.\n";

exit(EXIT\_FAILURE);

}

file.write(reinterpret\_cast<const char\*>(cars), sizeof(Car) \* size);

if (!file) {

cerr << "Помилка при записі в файл.\n";

exit(EXIT\_FAILURE);

}

file.close();

}

void readFile(const char\* filename) {

ifstream file(filename, ios::binary);

if (!file) {

cerr << "Помилка при відкритті файлу для читання.\n";

exit(EXIT\_FAILURE);

}

Car car;

while (file.read(reinterpret\_cast<char\*>(&car), sizeof(Car))) {

printCar(car);

}

file.close();

}

void modifyFile(const char\* filename, int minYear, const Car& newCar) {

fstream file(filename, ios::in | ios::out | ios::binary);

if (!file) {

cerr << "Помилка при відкритті файлу для модифікації.\n";

exit(EXIT\_FAILURE);

}

Car car;

while (file.read(reinterpret\_cast<char\*>(&car), sizeof(Car))) {

if (car.year < minYear) {

cout << "Вилучено елемент:\n";

printCar(car);

} else {

file.seekp(-static\_cast<std::ios::off\_type>(sizeof(Car)), ios::cur);

file.write(reinterpret\_cast<const char\*>(&car), sizeof(Car));

}

}

file.seekp(0, ios::beg);

file.write(reinterpret\_cast<const char\*>(&newCar), sizeof(Car));

file.close();

}

int main() {

Car cars[] = {

{"Toyota", 2020, 25000.0, "Червоний"},

{"Ford", 2018, 20000.0, "Синій"},

{"PESPATRON", 2022, 30000.0, "Чорний"},

{"BMW", 2019, 40000.0, "Чорний"},

};

const int size = sizeof(cars) / sizeof(Car);

createFile("cars.dat", cars, size);

cout << "Вміст файлу до модифікації:\n";

readFile("cars.dat");

int minYear = 2020;

Car newCar = {"Nissan", 2021, 28000.0, "Зелений"};

modifyFile("cars.dat", minYear, newCar);

cout << "\nВміст файлу після модифікації:\n";

readFile("cars.dat");

return 0;

}

**Завдання №3: Ввід-вивід рядків**

#include <stdio.h>

#include <string.h>

#define MAXLINE 255

int main() {

FILE \*f1, \*f2;

char buf[MAXLINE];

int N1, N2, currentLine = 0;

int maxACount = 0, maxALine = 0;

f1 = fopen("F1.txt", "r");

if (f1 == NULL) {

printf("Файл F1.txt не існує. Створюю новий файл.\n");

f1 = fopen("F1.txt", "w");

fputs("Рядок 1\n", f1);

fputs("Рядок 2\n", f1);

fclose(f1);

f1 = fopen("F1.txt", "r");

}

f2 = fopen("F2.txt", "w");

if (f2 == NULL) {

perror("Помилка відкриття файлу F2");

fclose(f1);

return 1;

}

printf("Введіть N1 та N2: ");

scanf("%d %d", &N1, &N2);

while (fgets(buf, MAXLINE, f1) != NULL) {

currentLine++;

size\_t len = strlen(buf);

if (len > 0 && buf[len - 1] == 'A') {

if (currentLine >= N1 && currentLine <= N2) {

fputs(buf, f2);

int ACount = 0;

for (size\_t i = 0; i < len; i++) {

if (buf[i] == 'A') {

ACount++;

}

}

if (ACount > maxACount) {

maxACount = ACount;

maxALine = currentLine;

}

}

}

}

fclose(f1);

fclose(f2);

printf("Файл F2 був успішно створений та заповнений.\n");

printf("Номер рядка з найбільшою кількістю букв 'А' в F2: %d\n", maxALine);

return 0;

}

**Завдання №4 Algotester Lab 4.**

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

int N, K;

cin >> N >> K;

vector<int> a(N);

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

cin >> a[i];

}

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

a.erase(unique(a.begin(), a.end()), a.end());

K = K % a.size();

rotate(a.begin(), a.begin() + K, a.end());

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

for (const auto& element : a) {

cout << element << " ";

}

return 0;

}

**Завдання №5 Algotester lab 6.**

#include <iostream>

#include <vector>

#include <bitset>

#include <algorithm>

using namespace std;

const int boardSize = 8;

enum FigureType { Empty, Rook, Pawn, Knight, Bishop, King, Queen };

struct Figure {

char symbol;

FigureType type;

bool canMove(int x, int y, const vector<string>& board) const {

return true;

}

};

Figure getFigure(char symbol) {

switch (symbol) {

case 'R':

return {'R', Rook};

case 'P':

return {'P', Pawn};

case 'N':

return {'N', Knight};

case 'B':

return {'B', Bishop};

case 'K':

return {'K', King};

case 'Q':

return {'Q', Queen};

default:

return {'O', Empty};

}

}

bool canMoveInDirection(int x, int y, const vector<string>& board, int dx, int dy, FigureType figureType) {

int i = x + dx;

int j = y + dy;

while (i >= 0 && i < boardSize && j >= 0 && j < boardSize) {

if (getFigure(board[i][j]).type == figureType) {

return true;

}

i += dx;

j += dy;

}

return false;

}

bool canMoveInDirections(int x, int y, const vector<string>& board, int dx[], int dy[], int n, FigureType figureType) {

int i = 0;

while (i < n) {

int nx = x + dx[i];

int ny = y + dy[i];

if (nx >= 0 && nx < boardSize && ny >= 0 && ny < boardSize && getFigure(board[nx][ny]).type == figureType) {

return true;

}

i++;

}

return false;

}

bool canMovePawn(int x, int y, const vector<string>& board) {

return (x - 1 >= 0 && (y - 1 >= 0 && getFigure(board[x - 1][y - 1]).type == Pawn || y + 1 < boardSize && getFigure(board[x - 1][y + 1]).type == Pawn));

}

bool canMoveRook(int x, int y, const vector<string>& board) {

int dx[] = {-1, 1, 0, 0};

int dy[] = {0, 0, -1, 1};

for (int i = 0; i < 4; ++i) {

if (canMoveInDirection(x, y, board, dx[i], dy[i], Rook)) {

return true;

}

}

return false;

}

bool canMoveKnight(int x, int y, const vector<string>& board) {

int knight\_dx[] = {-2, -1, 1, 2, 2, 1, -1, -2};

int knight\_dy[] = {1, 2, 2, 1, -1, -2, -2, -1};

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

int nx = x + knight\_dx[i];

int ny = y + knight\_dy[i];

if (nx >= 0 && nx < boardSize && ny >= 0 && ny < boardSize && getFigure(board[nx][ny]).type == Knight) {

return true;

}

}

return false;

}

bool canMoveBishop(int x, int y, const vector<string>& board) {

int dx[] = {-1, -1, 1, 1};

int dy[] = {-1, 1, -1, 1};

for (int i = 0; i < 4; ++i) {

if (canMoveInDirection(x, y, board, dx[i], dy[i], Bishop)) {

return true;

}

}

return false;

}

bool canMoveKing(int x, int y, const vector<string>& board) {

int king\_dx[] = {-1, -1, -1, 0, 0, 1, 1, 1};

int king\_dy[] = {-1, 0, 1, -1, 1, -1, 0, 1};

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

int nx = x + king\_dx[i];

int ny = y + king\_dy[i];

if (nx >= 0 && nx < boardSize && ny >= 0 && ny < boardSize && getFigure(board[nx][ny]).type == King) {

return true;

}

}

return false;

}

bool canMoveQueen(int x, int y, const vector<string>& board) {

int dx[] = {-1, -1, -1, 0, 0, 1, 1, 1};

int dy[] = {-1, 0, 1, -1, 1, -1, 0, 1};

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

if (canMoveInDirection(x, y, board, dx[i], dy[i], Queen)) {

return true;

}

}

return false;

}

string processQuery(int x, int y, const vector<string>& board) {

Figure figure = getFigure(board[x][y]);

if (figure.type != Empty) {

return "X";

} else {

string answer;

const vector<Figure> figures = {getFigure('B'), getFigure('K'), getFigure('N'), getFigure('P'), getFigure('Q'), getFigure('R')};

for (const Figure& figure : figures) {

switch (figure.type) {

case Bishop:

if (canMoveBishop(x, y, board)) answer += figure.symbol;

break;

case King:

if (canMoveKing(x, y, board)) answer += figure.symbol;

break;

case Knight:

if (canMoveKnight(x, y, board)) answer += figure.symbol;

break;

case Pawn:

if (canMovePawn(x, y, board)) answer += figure.symbol;

break;

case Queen:

if (canMoveQueen(x, y, board)) answer += figure.symbol;

break;

case Rook:

if (canMoveRook(x, y, board)) answer += figure.symbol;

break;

default:

break;

}

}

return answer.empty() ? "O" : answer;

}

}

vector<string> processQueries(const vector<string>& board, const vector<pair<int, int>>& queries) {

vector<string> results;

for (const auto& query : queries) {

int x = query.first;

int y = query.second;

results.push\_back(processQuery(x, y, board));

}

return results;

}

int main() {

vector<string> board(boardSize);

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

cin >> board[i];

}

int Q;

cin >> Q;

vector<pair<int, int>> queries;

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

int a, b;

cin >> a >> b;

queries.push\_back({a - 1, b - 1});

}

vector<string> results = processQueries(board, queries);

for (const string& result : results) {

cout << result << endl;

}

return 0;

}

**Завдання №5 Algotester lab 6.**

**#include <iostream>**

**#include <fstream>**

**#include <string>**

**enum FileOpResult { Success, Failure };**

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

**if (!name || !content) {**

**return FileOpResult::Failure;**

**}**

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

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

**std::cerr << "Помилка відкривая файл: " << name << std::endl;**

**return FileOpResult::Failure;**

**}**

**file << content;**

**if (file.fail()) {**

**std::cerr << "Помилка запису у файл: " << name << std::endl;**

**file.close();**

**return FileOpResult::Failure;**

**}**

**file.close();**

**return FileOpResult::Success;**

**}**

**int main() {**

**const char fileName[] = "primertexta.txt";**

**const char content[] = "Hello, World!";**

**if (write\_to\_file(fileName, content) == FileOpResult::Success) {**

**std::cout << "Файл успішно записано.\n";**

**} else {**

**std::cout << "Не вдалося записати у файл..\n";**

**}**

**return 0;**

**}**

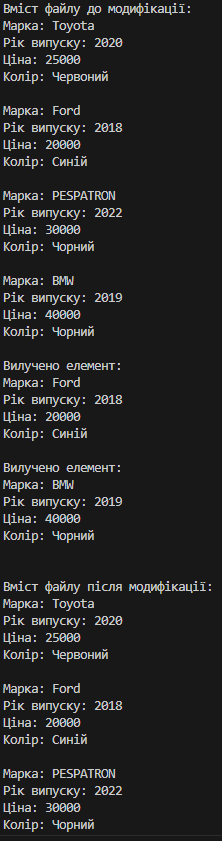
## **5. Результати виконання завдань, тестування та фактично затрачений час:**

Завдання №1 Робота з одновимірними масивами

**

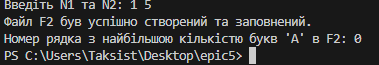
*Те, що мені видає програма 1*

Завдання №2 **Блоковий ввід-вивід**

**

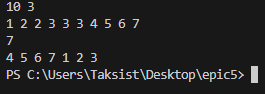
*Те, що мені видає програма 2*

Завдання №3: Ввід-вивід рядків



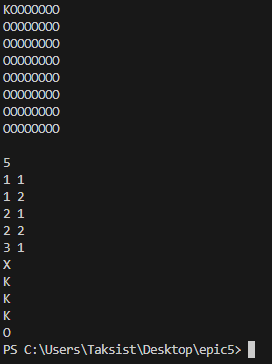
*Те, що мені видає програма 3*

Завдання №4 Algotester Lab 4.

**

*Те, що мені видає програма 4*

Завдання №5 Algotester Lab 6.



*Те, що мені видає програма 5*

Завдання №6 Class Practice



*Те, що мені видає програма 6*

**Висновки: В ході виконання лабораторної роботи з теми "Файли. Системи числення. Бінарні файли. Символи і Рядкові Змінні та Текстові Файли" було вивчено основні концепції роботи з файлами, системами числення та використання текстових та бінарних форматів файлів у програмуванні. Зокрема, були розглянуті поняття символів та рядкових змінних, а також вивчено роботу з бінарними файлами.**