

Міністерство освіти і науки України  
Національний університет «Львівська політехніка»  
Кафедра систем штучного інтелекту



## Звіт

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

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

**з дисципліни:** «Основи програмування»

до:

ВНС Лабораторної Роботи № 6  
ВНС Лабораторної Роботи № 8  
ВНС Лабораторної Роботи № 9  
Алготестер Лабораторної Роботи №4  
Алготестер Лабораторної Роботи №6  
Практичних Робіт до блоку №5

**Виконав(ла):**  
Студент групи ІІІ-11  
Зубрицький Арсеній Юрійович

Львів 2024

## **Тема роботи:**

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

## **Мета роботи:**

Ознайомитися з основними принципами та методами роботи з файлами в C++, включаючи текстові та бінарні файли. Навчитися використовувати стандартну бібліотеку для роботи з файлами, а також створювати власні бібліотеки для ефективного управління кодом у проектах.

## **Теоретичні відомості:**

Теоретичні відомості з переліком важливих тем:

1. Вступ до Роботи з Файлами:
  - Основні операції з файлами: відкриття, читання, запис, закриття
  - Робота з файловими дескрипторами
  - C-style читання з файлу та запис до файлу
  - Перевірка стану файлу: перевірка помилок, кінець файлу
  - Базові приклади читання та запису в файл
2. Символи і Рядкові Змінні:
  - Робота з char та string: основні операції і методи
  - Стрічкові літерали та екранування символів
  - Конкатенація, порівняння та пошук у рядках
3. Текстові Файли:
  - Особливості читання та запису текстових файлів
  - Обробка рядків з файлу: getline, ignore, peek
  - Форматування тексту при записі: setw, setfill, setprecision
  - Парсинг текстових файлів: розділення на слова, аналіз структури
  - Обробка помилок при роботі з файлами
4. Бінарні Файли:
  - Вступ до бінарних файлів: відмінності від текстових, приклади (великі дані, ігрові ресурси, зображення)
  - Читання та запис бінарних даних
  - Робота з позиціонуванням у файлі: seekg, seekp
  - Серіалізація об'єктів у бінарний формат
5. Стандартна бібліотека та робота з файлами:
  - Огляд стандартної бібліотеки для роботи з файлами
  - Потoki вводу/виводу: ifstream, ofstream, fstream

- Обробка помилок при роботі з файлами
- 6. Створення й використання бібліотек:
  - Вступ до створення власних бібліотек у C++
  - Правила розбиття коду на header-и(.h) та source(.cpp) файли
  - Статичні проти динамічних бібліотек: переваги та використання
  - Інтерфейси бібліотек: створення, документування, версіонування
  - Використання сторонніх бібліотек у проектах
- Індивідуальний план опрацювання теорії:
- **Тема №1:** Вступ до Роботи з Файлами:
  - Джерела Інформації
    - <https://www.programiz.com/c-programming/c-file-input-output>
  - Що опрацьовано:
    - Ознайомився з основними операціями над файлами
    - C-style читання та запис до файлу
    - Опрацював базові приклади застосування читання та запису в файлі
  - Статус: Ознайомлений
  - Початок опрацювання теми: 25.11.2024
  - Звершення опрацювання теми: 25.11.2024
  - Витрачено часу: 20 хв
- **Тема №2:** Символи і Рядкові Змінні:
  - Джерела Інформації:
    - <https://www.geeksforgeeks.org/cpp-string-functions/>
  - Що опрацьовано:
    - Ознайомився з роботою char та string операціями над ними
  - Статус: Ознайомлений
  - Початок опрацювання теми: 25.11.2024
  - Звершення опрацювання теми: 25.11.2024
  - Витрачено часу: 10 хв
- **Тема №3:** Текстові Файли:
  - Джерела Інформації:
    - <https://acode.com.ua/urok-220-bazovyj-fajlovyj-vvid-i-vyvid/>
  - Що опрацьовано:
    - Опрацював особливості читання та запису
    - Ознайомився з обробкою рядків з файлу: getline, ignore, peek
    - ознайомився з форматуванням тексту при записі: setw, setfill, setprecision. Парсингом текстових файлів: розділення на слова, аналіз структури. Обробку помилок при роботі з файлами
  - Статус: Ознайомлений
  - Початок опрацювання теми: 25.11.2024
  - Звершення опрацювання теми: 25.11.2024
  - Витрачено часу: 40 хв
- **Тема №4:** Бінарні Файли:

- Джерела Інформації:
  - <https://acode.com.ua/urok-221-randomnyj-fajlovyj-vvid-i-vyvid/>
  - <https://abitap.com/6-0-tekstovi-ta-binarni-fajly/>
  - <https://foxminded.ua/serializatsiia/>
- Що опрацьовано:
  - Опрацював відмінності бінарних від текстових, приклади (великі дані, ігрові ресурси, зображення)
  - Читання та запис бінарних даних
  - Роботу з позиціонуванням у файлі: seekg, seekp
  - Зрозумів що таке серіалізація об'єктів у бінарний формат
- Статус: Ознайомлений
- Початок опрацювання теми: 25.11.2024
- Звершення опрацювання теми: 25.11.2024
- Витрачено часу: 50 хв
- **Тема №5:** Стандартна бібліотека та робота з файлами:
  - Джерела Інформації:
    - Лекції та практичні
  - Що опрацьовано:
    - Опрацював потоки вводу/виводу: ifstream, ofstream, fstream
    - Обробка помилок при роботі з файлами
  - Статус: Ознайомлений
  - Початок опрацювання теми: 25.11.2024
  - Звершення опрацювання теми: 25.11.2024
  - Витрачено часу: 20 хв
- **Тема №6:** Створення й використання бібліотек:
  - Джерела Інформації:
    - Лекції та практичні
  - Що опрацьовано:
    - Опрацював створення власних бібліотек у C++
    - Правила розбиття коду на header-и(.h) та source(.cpp) файли
    - Статичні проти динамічних бібліотек: переваги та використання
    - Інтерфейси бібліотек: створення, документування, версіонування
    - Використання сторонніх бібліотек у проектах
  - Статус: Ознайомлений
  - Початок опрацювання теми: 25.11.2024
  - Звершення опрацювання теми: 25.11.2024
  - Витрачено часу: 40 хв

## **Виконання роботи:**

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

Завдання №1 Vns Lab\_6\_task\_2\_4

#### **2. Постановка 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.

4. Надрукувати всі слова, які співпадають з її першим словом

Завдання №2 VNS Lab\_8\_task\_1\_4

#### **2. Постановка завдання**

Сформувати двійковий файл із елементів, заданої у варіанті структури, роздрукувати його вміст, 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.

4. Структура "Людина":

- прізвище, ім'я, по батькові;
- домашня адреса;
- номер телефону;
- вік.

Знищити усі елементи із заданим віком, додати елемент після елемента із заданим номером.

Завдання №3 VNS Lab\_9\_task\_1\_4

#### **2. Постановка завдання**

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

Виконати завдання.

4.

1) Скопіювати з файлу F1 у файл F2 рядки, починаючи з 4.

2) Підрахувати кількість символів в останньому слові F2.

## Завдання №4, 5 Algotester Lab4v3

### Lab 4v3

Обмеження: 2 сек., 256 МБ

Вам дано масив, який складається з  $N$  додатніх цілих чисел.

Ваше завдання - розділити його на три частини, по остачі від ділення на 3, по зростанню остачі (тобто спочатку йдуть числа, у яких остача 0, далі числа з остачею 1 і тоді нарешті числа з остачею 2).

Далі необхідно ті елементи, остача від ділення на 3 яких парна посортувати по зростанню, а ті, у яких остача 1 - по спаданню.

Після цього видаліть усі дублікати з масиву.

Виведіть результуючий масив.

#### Вхідні дані

У першому рядку  $N$  - кількість чисел.

У другому рядку  $N$  чисел  $a_i$  - елементи масиву.

#### Вихідні дані

У першому рядку  $M$  - кількість чисел у масиву

У другому рядку  $M$  посортованих за умовою чисел.

#### Примітки

**Пам'ятайте, ви маєте написати 2 варіанти розв'язку, один з використанням засобів STL (власноруч написаний компаратор або `std::partition + std::sort + std::unique`), інший зі своєю реалізацією. Алгоритм сортування можна вибрати будь-який, окрім сортування бульбашкою і має працювати за  $N \cdot \log N$  часу.**

## Завдання №6 Algotester Lab6v2

### Lab 6v2

Обмеження: 2 сек., 256 МБ

У вас є шахова дошка розміром  $8 \times 8$  та дуже багато фігур.

Кожна клітинка може мати таке значення:

- Пуста клітинка  $O$
- Пішак  $P$
- Тура  $R$
- Кінь  $N$
- Слон  $B$
- Король  $K$
- Королева  $Q$

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

Далі йдуть  $Q$  запитів з координатами клітинки  $\{x, y\}$ . На кожен запит ви маєте вивести стрічку  $s_i$  - 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  $O$ .

Наявність фігури у певній клітинці не блокує атаку для іншої фігури. Тобто якщо між турою та клітинкою стоїть інша фігура - вважається що тура атакує цю клітинку.

#### Вхідні дані

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

У наступному рядку ціле число  $Q$  - кількість запитів

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

#### Вихідні дані

$Q$  разів відповідь у наступному форматі:

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

## Завдання №7 Робота з текстовими файлами

# Робота з текстовими файлами

## 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 навички роботи з файлами з використанням стандартної бібліотеки C++. 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 операцій, які якнайкраще вкладаються в концепцію алгоритма, як списка детальних кроків. Імплементация цієї функції наочно демонструє створення алгоритмів у програмуванні.

**Освоїти навички роботи з текстовими стрічками:** завдання допоможе освоїти роботу з C стрічка, які є масивами з нульовим символом в кінці. Типові концепції при роботі з C 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 навик роботи з файлами з використанням стандартної бібліотеки C++. 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 списку детальних кроків. Імплементация цієї функції наочно демонструє створення алгоритмів у програмуванні.

**Освоїти навик роботи з потоком даних:** завдання допоможе освоїти роботу з потоками даних (концепція реалізована в STL як набір класів *\*stream\** - *fstream*, *stringstream*, *stringstream* 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.

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

### Завдання №1

```
1  #include <iostream>
2  #include <cstring>
3  #include <cstdlib>
4
5  using namespace std;
6
7  // Функція для пошуку та друку слів, які співпадають з першим словом
8  void FindMatchingWords(char str[]) {
9      // Розділяємо рядок на слова, використовуючи пробіли
10     char* firstWord = strtok(str, " "); // Знаходимо перше слово
11     if (firstWord == nullptr) {
12         cout << "Рядок не містить слів!" << endl;
13         return;
14     }
15
16     cout << "Перше слово: " << firstWord << endl;
17     cout << "Слова, які співпадають з першим словом: " << endl;
18
19     // Перебираємо інші слова і порівнюємо їх з першим
20     char* currentWord = strtok(nullptr, " ");
21     while (currentWord != nullptr) {
22         // Порівнюємо поточне слово з першим
23         if (strcmp(currentWord, firstWord) == 0) {
24             cout << currentWord << endl; // Друкуємо, якщо співпадає
25         }
26         currentWord = strtok(nullptr, " ");
27     }
28 }
29
30 int main() {
31     char s[256]; // Масив для збереження введеного рядка
32
33     // Ввід рядка користувачем
34     cout << "Введіть рядок (до 255 символів, завершуйте крапкою):";
35     gets(s); // Зчитуємо рядок
36
37     // Видаляємо крапку з кінця рядка, якщо вона є
38     size_t len = strlen(s);
39     if (len > 0 && s[len - 1] == '.') {
40         s[len - 1] = '\0';
41     }
42
43     // Викликаємо функцію для пошуку та друку слів
44     FindMatchingWords(s);
45
46     return 0;
47 }
48
```



Посилання на pull-request:

[https://github.com/artificial-intelligence-department/ai\\_programming\\_playground\\_2024/pull/687/files#diff-c7e8d1e0c691b6a55a46c3297ac4b926003c0bd6b69bdcf624335a69e117522a](https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/687/files#diff-c7e8d1e0c691b6a55a46c3297ac4b926003c0bd6b69bdcf624335a69e117522a)

## Завдання №2

```
1  #include <iostream>
2  #include <fstream>
3  #include <cstring>
4
5  using namespace std;
6
7  // Структура "Людина"
8  struct Person {
9      char lastName[50];
10     char firstName[50];
11     char middleName[50];
12     char address[100];
13     char phoneNumber[20];
14     int age;
15 };
16
17 // Функція для заповнення однієї структури
18 void fillPerson(Person& p) {
19     cout << "Введіть прізвище: ";
20     cin >> p.lastName;
21     cout << "Введіть ім'я: ";
22     cin >> p.firstName;
23     cout << "Введіть по батькові: ";
24     cin >> p.middleName;
25     cout << "Введіть домашню адресу: ";
26     cin.ignore(); // Очищаємо буфер
27     cin.getline(p.address, 100);
28     cout << "Введіть номер телефону: ";
29     cin >> p.phoneNumber;
30     cout << "Введіть вік: ";
31     cin >> p.age;
32 }
33
34 // Функція для виведення даних про людину
35 void printPerson(const Person& p) {
36     cout << "Прізвище: " << p.lastName << "\nІм'я: " << p.firstName << "\nПо батькові: " << p.middleName
37         << "\nАдреса: " << p.address << "\nТелефон: " << p.phoneNumber << "\nВік: " << p.age << "\n-----" << endl;
38 }
39
```

```

40 // Функція для запису структури у файл
41 void writeToFile(const char* filename, const Person& p) {
42     ofstream outFile(filename, ios::binary | ios::app);
43     if (!outFile) {
44         cerr << "Помилка відкриття файлу для запису!" << endl;
45         return;
46     }
47     outFile.write(reinterpret_cast<const char*>(&p), sizeof(Person));
48     outFile.close();
49 }
50
51 // Функція для читання всіх елементів з файлу
52 void readFromFile(const char* filename) {
53     ifstream inFile(filename, ios::binary);
54     if (!inFile) {
55         cerr << "Помилка відкриття файлу для читання!" << endl;
56         return;
57     }
58     Person p;
59     while (inFile.read(reinterpret_cast<char*>(&p), sizeof(Person))) {
60         printPerson(p);
61     }
62     inFile.close();
63 }
64
65 // Функція для видалення людей із заданим віком
66 void removeByAge(const char* filename, int ageToRemove) {
67     ifstream inFile(filename, ios::binary);
68     if (!inFile) {
69         cerr << "Помилка відкриття файлу для читання!" << endl;
70         return;
71     }
72
73     ofstream tempFile("temp.dat", ios::binary);
74     if (!tempFile) {
75         cerr << "Помилка відкриття тимчасового файлу для запису!" << endl;
76         return;
77     }
78
79     Person p;
80     while (inFile.read(reinterpret_cast<char*>(&p), sizeof(Person))) {
81         if (p.age != ageToRemove) {
82             tempFile.write(reinterpret_cast<const char*>(&p), sizeof(Person));
83         }
84     }
85
86     inFile.close();
87     tempFile.close();
88
89     remove(filename);
90     rename("temp.dat", filename);
91     cout << "Елементи з віком " << ageToRemove << " успішно видалено." << endl;
92 }

```

```

124 // Функція для додавання нового елемента після заданого
125 void addAfterPerson(const char* filename, int personNumber, const Person& newPerson) {
126     ifstream inFile(filename, ios::binary);
127     if (!inFile) {
128         cerr << "Помилка відкриття файлу для читання!" << endl;
129         return;
130     }
131
132     ofstream tempFile("temp.dat", ios::binary);
133     if (!tempFile) {
134         cerr << "Помилка відкриття тимчасового файлу для запису!" << endl;
135         return;
136     }
137
138     Person p;
139     int currentIndex = 0;
140     while (inFile.read(reinterpret_cast<char*>(&p), sizeof(Person))) {
141         tempFile.write(reinterpret_cast<const char*>(&p), sizeof(Person));
142         currentIndex++;
143         if (currentIndex == personNumber) {
144             tempFile.write(reinterpret_cast<const char*>(&newPerson), sizeof(Person));
145         }
146     }
147
148     inFile.close();
149     tempFile.close();
150
151     remove(filename);
152     rename("temp.dat", filename);
153     cout << "Нова людина додана після " << personNumber << "-го елемента." << endl;
154 }
155
156 // Функція для очищення файлу
157 void clearFile(const char* filename) {
158     ofstream outFile(filename, ios::binary | ios::trunc);
159     if (!outFile) {
160         cerr << "Помилка відкриття файлу для очищення!" << endl;
161         return;
162     }
163     outFile.close();
164     cout << "Файл успішно очищений." << endl;
165 }
166
167 // Функція для видалення файлу
168 void deleteFile(const char* filename) {
169     if (remove(filename) != 0) {
170         cerr << "Помилка при видаленні файлу!" << endl;
171     } else {
172         cout << "Файл успішно видалено." << endl;
173     }
174 }
175

```

```

146 int main() {
147     const char* filename = "people.dat";
148
149     // Введення кількості людей, яких хочемо додати в файл
150     int count;
151     cout << "Введіть кількість людей: ";
152     cin >> count;
153
154     // Додаємо людей у файл
155     for (int i = 0; i < count; i++) {
156         Person p;
157         cout << "Введіть дані для " << i + 1 << "-ї людини:" << endl;
158         fillPerson(p);
159         writeToFile(filename, p);
160     }
161
162     // Читання всіх елементів з файлу
163     cout << "\nВміст файлу:\n";
164     readFromFile(filename);
165
166     // Видалення людей із заданим віком
167     int ageToRemove;
168     cout << "\nВведіть вік для видалення: ";
169     cin >> ageToRemove;
170     removeByAge(filename, ageToRemove);
171
172     // Читання всіх елементів після видалення
173     cout << "\nВміст файлу після видалення:\n";
174     readFromFile(filename);
175
176     // Додавання нового елемента
177     Person newPerson;
178     cout << "\nВведіть дані для нової людини, яку потрібно додати після певного елемента:" << endl;
179     fillPerson(newPerson);
180     int personNumber;
181     cout << "Введіть номер елемента після якого потрібно додати нового: ";
182     cin >> personNumber;
183     addAfterPerson(filename, personNumber, newPerson);
184
185     // Читання всіх елементів після додавання
186     cout << "\nВміст файлу після додавання:\n";
187     readFromFile(filename);
188
189     // Очищення файлу
190     char choice;
191     cout << "\nЧи хочете ви очистити файл? (y/n): ";
192     cin >> choice;
193     if (choice == 'y' || choice == 'Y') {
194         clearFile(filename);
195     }
196
197     // Видалення файлу
198     cout << "\nЧи хочете ви видалити файл? (y/n): ";
199     cin >> choice;
200     if (choice == 'y' || choice == 'Y') {
201         deleteFile(filename);
202     }
203
204     return 0;
205 }
206

```

Посилання на pull-request:

[https://github.com/artificial-intelligence-department/ai\\_programming\\_playground\\_2024/pull/687/files#diff-dbabd572f29ad3c0211cdc7c3eb0acb728aace4ccaa7aa08a55a7b99f77d12ff](https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/687/files#diff-dbabd572f29ad3c0211cdc7c3eb0acb728aace4ccaa7aa08a55a7b99f77d12ff)

### Завдання №3

```
1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  #include <sstream>
5
6  using namespace std;
7
8  // Функція для копіювання рядків з F1 у F2, починаючи з 4-го
9  void copyLinesFromFile(const string& inputFile, const string& outputFile) {
10     ifstream inFile(inputFile);
11     if (!inFile) {
12         cerr << "Помилка відкриття файлу " << inputFile << endl;
13         return;
14     }
15
16     ofstream outFile(outputFile);
17     if (!outFile) {
18         cerr << "Помилка відкриття файлу " << outputFile << endl;
19         return;
20     }
21
22     string line;
23     int lineCount = 0;
24
25     // Читаємо рядки з файлу і записуємо в F2, починаючи з 4-го
26     while (getline(inFile, line)) {
27         lineCount++;
28         if (lineCount >= 4) {
29             outFile << line << endl;
30         }
31     }
32
33     inFile.close();
34     outFile.close();
35 }
36
37 // Функція для підрахунку кількості символів в останньому слові файлу
38 int countLastWordLength(const string& fileName) {
39     ifstream inFile(fileName);
40     if (!inFile) {
41         cerr << "Помилка відкриття файлу " << fileName << endl;
42         return 0;
43     }
44
45     string line;
46     string lastWord;
47     while (getline(inFile, line)) {
48         // Розбиваємо рядок на слова
49         stringstream ss(line);
50         string word;
51         while (ss >> word) {
52             lastWord = word; // Оновлюємо останнє слово
53         }
54     }
55
56     inFile.close();
57
58     // Повертаємо довжину останнього слова
59     return lastWord.length();
60 }
```

```

62  int main() {
63      const string inputFile = "F1.txt";
64      const string outputFile = "F2.txt";
65
66      // Створюємо файл F1 і записуємо в нього інформацію
67      ofstream outFile(inputFile);
68      if (!outFile) {
69          cerr << "Помилка відкриття файлу для запису!" << endl;
70          return 1;
71      }
72
73      // Записуємо 10 рядків у файл F1
74      outFile << "Перше слово" << endl;
75      outFile << "Друге слово" << endl;
76      outFile << "Третє слово" << endl;
77      outFile << "Четверте слово" << endl;
78      outFile << "П'яте слово" << endl;
79      outFile << "Шосте слово" << endl;
80      outFile << "Сьоме слово" << endl;
81      outFile << "Восьме слово" << endl;
82      outFile << "Дев'яте слово" << endl;
83      outFile << "Десяте слово" << endl;
84
85      outFile.close();
86
87      // Копіюємо рядки з F1 у F2, починаючи з 4-го
88      copyLinesFromFile(inputFile, outputFile);
89      cout << "Рядки скопійовано з F1 в F2, починаючи з 4-го." << endl;
90
91      // Підраховуємо кількість символів в останньому слові файлу F2
92      int lastWordLength = countLastWordLength(outputFile);
93      cout << "Кількість символів в останньому слові файлу F2: " << lastWordLength << endl;
94
95      return 0;
96  }
97

```

Посилання на pull-request:

[https://github.com/artificial-intelligence-department/ai\\_programming\\_playground\\_2024/pull/687/files#diff-be7be864a6d871bcf3377b24bda94c955ff7136078075d91e2918bd1081d0a0f](https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/687/files#diff-be7be864a6d871bcf3377b24bda94c955ff7136078075d91e2918bd1081d0a0f)

Завдання №4

```

1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  #include <iterator>
5
6  using namespace std;
7
8  int main() {
9      int N;
10     cin >> N;
11
12     vector<int> arr(N);
13     for (int i = 0; i < N; ++i) {
14         cin >> arr[i];
15     }
16
17     // Розділяємо на три групи за остачею від ділення на 3
18     vector<int> group0, group1, group2;
19
20     for (int num : arr) {
21         if (num % 3 == 0) group0.push_back(num);
22         else if (num % 3 == 1) group1.push_back(num);
23         else group2.push_back(num);
24     }
25
26     // Сортуюмо за умовами: group0 та group2 по зростанню, group1 по спаданню
27     sort(group0.begin(), group0.end());
28     sort(group1.begin(), group1.end(), greater<int>());
29     sort(group2.begin(), group2.end());
30
31     // Об'єднуємо всі три групи
32     vector<int> result;
33     result.insert(result.end(), group0.begin(), group0.end());
34     result.insert(result.end(), group1.begin(), group1.end());
35     result.insert(result.end(), group2.begin(), group2.end());
36
37     // Видаляємо дублікати
38     result.erase(unique(result.begin(), result.end()), result.end());
39
40     // Виводимо результат
41     cout << result.size() << endl;
42     for (int num : result) {
43         cout << num << " ";
44     }
45     cout << endl;
46
47     return 0;
48 }
49

```

Посилання на pull-request:

[https://github.com/artificial-intelligence-department/ai\\_programming\\_playground\\_2024/pull/687/files#diff-8452669e4a613d21ad5949915cf51743bf899f244e8e88a41765183097d04ab3](https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/687/files#diff-8452669e4a613d21ad5949915cf51743bf899f244e8e88a41765183097d04ab3)

## Завдання №5

```
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4
5  using namespace std;
6
7  void sortGroup0(vector<int>& group) {
8      sort(group.begin(), group.end());
9  }
10
11 void sortGroup1(vector<int>& group) {
12     sort(group.begin(), group.end(), greater<int>());
13 }
14
15 void sortGroup2(vector<int>& group) {
16     sort(group.begin(), group.end());
17 }
18
19 bool contains(const vector<int>& result, int value) {
20     return find(result.begin(), result.end(), value) != result.end();
21 }
22
23 int main() {
24     int N;
25     cin >> N;
26
27     vector<int> arr(N);
28     for (int i = 0; i < N; ++i) {
29         cin >> arr[i];
30     }
31
32     // Розділяємо на три групи за остачею від ділення на 3
33     vector<int> group0, group1, group2;
34
35     for (int num : arr) {
36         if (num % 3 == 0) group0.push_back(num);
37         else if (num % 3 == 1) group1.push_back(num);
38         else group2.push_back(num);
39     }
40
41     // Сортуюмо групи
42     sortGroup0(group0);
43     sortGroup1(group1);
44     sortGroup2(group2);
45
46     // Об'єднуємо всі три групи
47     vector<int> result;
48     result.insert(result.end(), group0.begin(), group0.end());
49     result.insert(result.end(), group1.begin(), group1.end());
50     result.insert(result.end(), group2.begin(), group2.end());
51 }
```



```

52     // Видаляємо дублікати вручну
53     vector<int> finalResult;
54     for (int num : result) {
55         if (!contains(finalResult, num)) {
56             finalResult.push_back(num);
57         }
58     }
59
60     // Виводимо результат
61     cout << finalResult.size() << endl;
62     for (int num : finalResult) {
63         cout << num << " ";
64     }
65     cout << endl;
66
67     return 0;
68 }
69

```

Посилання на pull-request:

[https://github.com/artificial-intelligence-department/ai\\_programming\\_playground\\_2024/pull/687/files#diff-3f580ce006ae498decea0c018b28911951ead719febacd416f87de3c1b8ac36c](https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/687/files#diff-3f580ce006ae498decea0c018b28911951ead719febacd416f87de3c1b8ac36c)

## Завдання №6

```

1  #include <iostream>
2  #include <vector>
3  #include <string>
4  #include <set>
5  #include <algorithm>
6  #include <cmath>
7
8  using namespace std;
9
10 // Константа для розміру дошки
11 const int BOARD_SIZE = 8;
12
13 // Функція перевірки атаки фігури
14 bool isUnderAttack(char piece, int px, int py, int tx, int ty) {
15     if (piece == 'R') {
16         return px == tx || py == ty;
17     }
18     if (piece == 'B') {
19         return abs(px - tx) == abs(py - ty);
20     }
21     if (piece == 'N') {
22         return (abs(px - tx) == 2 && abs(py - ty) == 1) || (abs(px - tx) == 1 && abs(py - ty) == 2);
23     }
24     if (piece == 'P') {
25         return px + 1 == tx && abs(py - ty) == 1;
26     }
27     if (piece == 'K') {
28         return max(abs(px - tx), abs(py - ty)) == 1;
29     }
30     if (piece == 'Q') {
31         return px == tx || py == ty || abs(px - tx) == abs(py - ty);
32     }
33     return false;
34 }

```

```

37 int main() {
38     vector<string> chessboard(BOARD_SIZE);
39
40     // Зчитування стану шахової дошки
41     for (int i = 0; i < BOARD_SIZE; i++) {
42         cin >> chessboard[i];
43     }
44
45     int queries;
46     cin >> queries;
47
48     vector<string> outputs(queries);
49
50     // Обробка кожного запиту
51     for (int q = 0; q < queries; q++) {
52         int x, y;
53         cin >> x >> y;
54         x--, y--; // Перехід до 0-індексації
55
56         // Якщо клітинка зайнята фігурою
57         if (chessboard[x][y] != '0') {
58             outputs[q] = "X";
59             continue;
60         }
61
62         set<char> attackers;
63         // Перевіряємо, які фігури атакують цільову клітинку
64         for (int i = 0; i < BOARD_SIZE; i++) {
65             for (int j = 0; j < BOARD_SIZE; j++) {
66                 char piece = chessboard[i][j];
67                 if (piece != '0' && isUnderAttack(piece, i, j, x, y)) {
68                     attackers.insert(piece);
69                 }
70             }
71         }
72
73         // Формуємо результат
74         if (attackers.empty()) {
75             outputs[q] = "0";
76         } else {
77             string attackList(attackers.begin(), attackers.end());
78             outputs[q] = attackList;
79         }
80     }
81
82     // Вивід результатів
83     for (const string& output : outputs) {
84         cout << output << endl;
85     }
86
87     return 0;
88 }
89

```

Посилання на pull-request:

[https://github.com/artificial-intelligence-department/ai\\_programming\\_playground\\_2024/pull/687/files#diff-b3c7d052b4f0334dfdf313205b8287690eeecd831538cfcf5c5cc6381b2560a6](https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/687/files#diff-b3c7d052b4f0334dfdf313205b8287690eeecd831538cfcf5c5cc6381b2560a6)

## Завдання №7

```
1  #include <iostream>
2  #include <fstream>
3
4  enum FileOpResult { Success, Failure };
5
6  FileOpResult write_to_file(const char* name, const char* content) {
7      std::ofstream outFile(name, std::ios::trunc);
8      if (!outFile) return Failure; // Якщо файл не відкрився для запису
9      outFile << content;
10     if (outFile.fail()) return Failure; // Помилка при записі
11     outFile.close();
12     return outFile.is_open() ? Failure : Success; // Перевірка закриття файлу
13 }
14
15 FileOpResult copy_file(const char* file_from, const char* file_to) {
16     std::ifstream inFile(file_from, std::ios::in);
17     if (!inFile) return Failure; // Якщо файл не відкрився для читання
18
19     std::ofstream outFile(file_to, std::ios::trunc);
20     if (!outFile) return Failure; // Якщо файл не відкрився для запису
21
22     outFile << inFile.rdbuf();
23     if (inFile.fail() || outFile.fail()) return Failure; // Помилка при копіюванні
24     inFile.close();
25     outFile.close();
26     return Success;
27 }
28
29 int main() {
30     // Задача 1: Запис в файл
31     const char* filename = "output.txt";
32     const char* content = "Це текст, який потрібно записати у файл.";
33     if (write_to_file(filename, content) == Success)
34         std::cout << "Текст успішно записано в файл!" << std::endl;
35     else
36         std::cout << "Не вдалося записати текст у файл." << std::endl;
37
38     // Задача 2: Копіювання вмісту одного файлу в інший
39     const char* file_from = "input.txt";
40     const char* file_to = "copied_output.txt";
41     if (copy_file(file_from, file_to) == Success)
42         std::cout << "Вміст успішно скопійовано!" << std::endl;
43     else
44         std::cout << "Не вдалося скопіювати вміст файлів." << std::endl;
45
46     return 0;
47 }
48
```

Посилання на pull-request:

[https://github.com/artificial-intelligence-department/ai\\_programming\\_playground\\_2024/pull/687/files#diff-1a703342c73ef2ae2c17f6dde514e7366d2958bf2f20944f2e95d50a48f0585b](https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/687/files#diff-1a703342c73ef2ae2c17f6dde514e7366d2958bf2f20944f2e95d50a48f0585b)

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

##### Завдання №1

```
Введіть рядок (до 255 символів, завершуйте крапкою):dog cat fish bird dog cat.  
Перше слово: dog  
Слова, які співпадають з першим словом:  
dog
```

Час затрачений на виконання завдання: 1 год

##### Завдання №2

```
Введіть кількість людей: 2  
Введіть дані для 1-ї людини:  
Введіть прізвище: Black  
Введіть ім'я: John  
Введіть по батькові:  
Batkovych  
Введіть домашню адресу: shevchenka_12  
Введіть номер телефону: 098654726  
Введіть вік: 25  
Введіть дані для 2-ї людини:  
Введіть прізвище: Zubrytskyi  
Введіть ім'я: Arsenii  
Введіть по батькові: Yuriyovych  
Введіть домашню адресу: Levka-Lykyanenko  
Введіть номер телефону: 0985739883  
Введіть вік: 17  
  
Вміст файлу:  
Прізвище: Black  
Ім'я: John  
По батькові: Batkovych  
Адреса: shevchenka_12  
Телефон: 098654726  
Вік: 25  
-----  
Прізвище: Zubrytskyi  
Ім'я: Arsenii  
По батькові: Yuriyovych  
Адреса: Levka-Lykyanenko  
Телефон: 0985739883  
Вік: 17  
-----  
  
Введіть вік для видалення: 25  
Елементи з віком 25 успішно видалено.  
  
Вміст файлу після видалення:  
Прізвище: Zubrytskyi  
Ім'я: Arsenii  
По батькові: Yuriyovych  
Адреса: Levka-Lykyanenko  
Телефон: 0985739883  
Вік: 17  
-----  
  
Введіть дані для нової людини, яку потрібно додати після певного елемента:  
Введіть прізвище: dmytro  
Введіть ім'я: dzmil  
Введіть по батькові: batkovych  
Введіть домашню адресу: pokrovska_1  
Введіть номер телефону: 063888227  
Введіть вік: 23  
Введіть номер елемента після якого потрібно додати нового: 0  
Нова людина додана після 0-го елемента.
```

Час затрачений на виконання завдання: 4 год

### Завдання №3

```
Рядки скопійовано з F1 в F2, починаючи з 4-го.  
Кількість символів в останньому слові файлу F2: 10
```

Час затрачений на виконання завдання: 1 год

### Завдання №4

```
9  
4 5 10 9 8 7 6 3 11  
9  
3 6 9 10 7 4 5 8 11
```

Час затрачений на виконання завдання: 1 год

### Завдання №5

```
9  
2 3 4 6 7 23 33 8 9  
9  
3 6 9 33 7 4 2 8 23
```

Час затрачений на виконання завдання: 40 хв

### Завдання №6

```
R0000P00  
00000000  
00000000  
00000000  
00000000  
00000000  
00000000  
00000000  
00000000  
  
4  
1 1  
1 5  
1 6  
1 8  
X  
R  
X  
R
```

Час затрачений на виконання завдання: 3 год

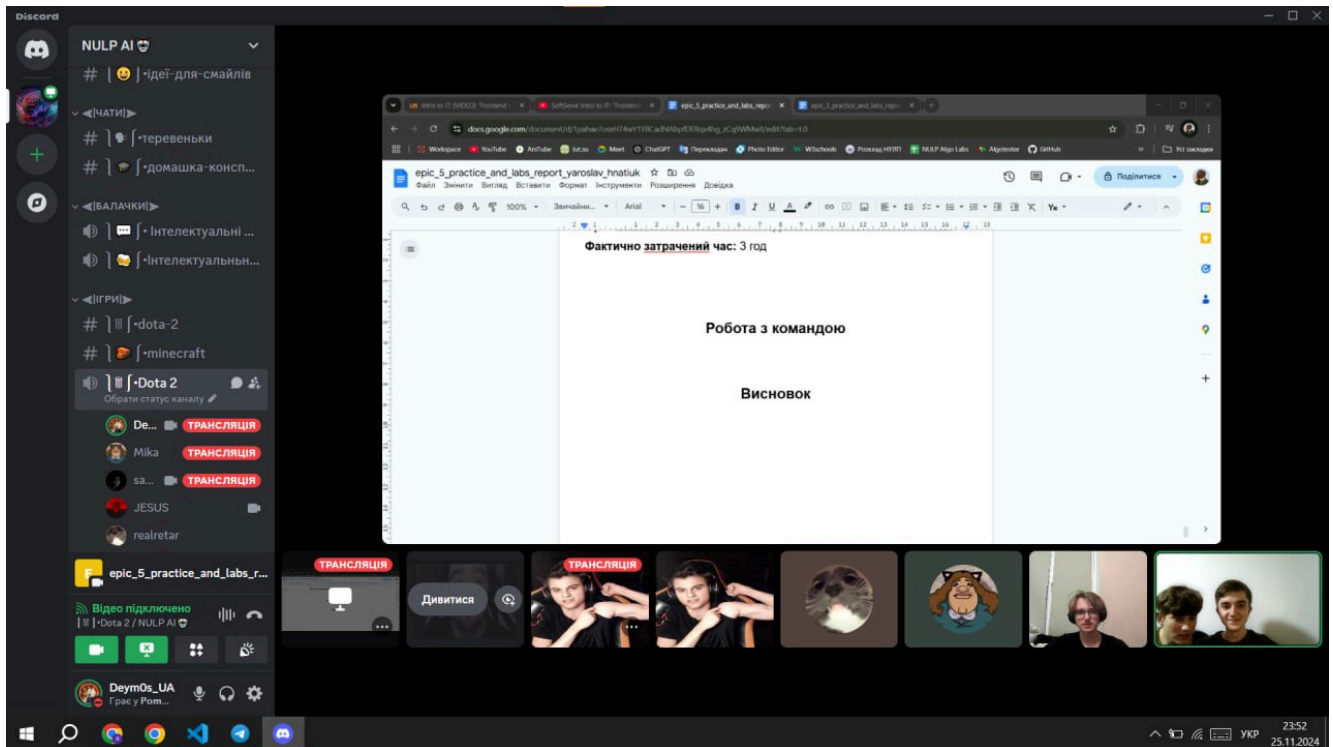
## Завдання №7

Текст успішно записано в файл!  
Вміст успішно скопійовано!

Час затрачений на виконання завдання: 1 год

### 6. Кооперація з командою:

- Скрін з зустрічі по обговоренню задач Епіку та Скрін прогресу по Трелло



**Висновки:** В ході виконання робіт з еріс\_5 я опрацював Файли. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами. На практиці застосував та закріпив вивчені знання.