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:
Студент групи ШІ-13
Кобзар Артем Сергійович

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

Мета: ознайомитися з операціями з файлами, бібліотеками, їх створенням та використанням.

Теоретичні відомості:

- лекції, практичні
- вказівки до лабораторних робіт ВНС
- https://www.programiz.com/cpp-programming
- geeksforgeeks.org
- w3schools.com/cpp

Виконання роботи

Завдання №1.1

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.

Завдання №1.2

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.

Завдання №2 VNS Lab#6(Варіант 18)

Всі слова рядка, які починаються із цифри відсортувати за спаданням. 18. Всі слова рядка, які починаються із цифри відсортувати за спаданням.

Завдання №3 VNS Lab#8(Варіант 18)

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.
Структура "Книга":
- назва;
- автор;
- рік видання;
- кількість сторінок.

Знищити 3 елементи з початку файлу, додати елемент перед елементом із
зазначеною назвою.

Завдання №4 VNS Lab#9(Варіант 18)

Створити текстовий файл F1 не менше, ніж з 10 рядків і записати в нього інформацію
Виконати завдання:
1) Скопіювати з файлу F1 у файл F2 всі рядки, у яких немає однакових слів.
2) Визначити кількість голосних букв у першому рядку файлу F2.

Завдання №5 Algotester Lab 4v1

Вам дано 2 цілих чисел масиви, розміром N та M.
Ваше завдання вивести:
1. Різницю N-M
2. Різницю M-N
3. Їх перетин
4. Їх обєднання
5. Їх симетричну різницю

Вхідні дані

У першому рядку ціле число N - розмір масиву 1
У другому рядку N цілих чисел - елементи масиву 1
У третьому рядку ціле число M - розмір масиву 2
У четвертом рядку M цілих чисел - елементи масиву 2

Вихідні дані

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

Завдання №6 Algotester Lab 4v2

Вам дано масив a з N цілих чисел.
Спочатку видаліть масиву a усі елементи що повторюються, наприклад масив [1, 3, 3, 4] має перетворитися у [1, 3, 4].
Після цього оберніть посортовану версію масиву a на K, тобто при K = 3 масив [1, 2, 3, 4, 5, 6, 7] перетвориться на [4, 5, 6, 7, 1, 2, 3].
Виведіть результат.

Вхідні дані

У першому рядку цілі числа N та K
У другому рядку N цілих чисел - елементи масиву a

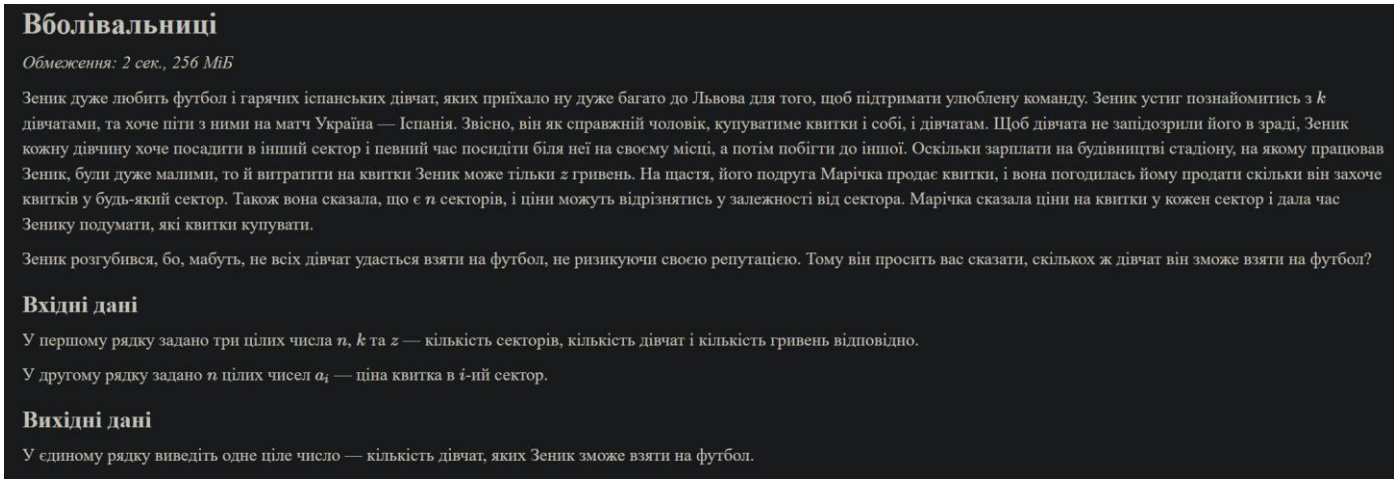
Вихідні дані

У першому рядку ціле число N - розмір множини a
У наступному рядку N цілих чисел - множина a

Завдання №7 Algotester Lab 6v2

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.

Завдання №8 Algotester self-practice task



Вболівальниці

Обмеження: 2 сек., 256 МіБ

Зеник дуже любить футбол і гарячих іспанських дівчат, яких приїхало ну дуже багато до Львова для того, щоб підтримати улюблену команду. Зеник устиг познайомитись з k дівчатами, та хоче піти з ними на матч Україна — Іспанія. Звісно, він як справжній чоловік, купуватиме квитки і собі, і дівчатам. Щоб дівчата не запідозрили його в зраді, Зеник кожну дівчину хоче посадити в інший сектор і певний час посидіти біля неї на своєму місці, а потім побігти до іншої. Оскільки зарплати на будівництві стадіону, на якому працював Зеник, були дуже малими, то й витратити на квитки Зеник може тільки z гривень. На щастя, його подруга Марічка продає квитки, і вона погодилась йому продати скільки він захоче квитків у будь-який сектор. Також вона сказала, що є n секторів, і ціни можуть відрізнятись у залежності від сектора. Марічка сказала ціни на квитки у кожен сектор і дала час Зенику подумати, які квитки купувати.

Зеник розгубився, бо, мабуть, не всіх дівчат удасться взяти на футбол, не ризикуючи своєю репутацією. Тому він просить вас сказати, скількох ж дівчат він зможе взяти на футбол?

Вхідні дані

У першому рядку задано три цілих числа n, k та z — кількість секторів, кількість дівчат і кількість гривень відповідно.

У другому рядку задано n цілих чисел a_i — ціна квитка в i-ий сектор.

Вихідні дані

У єдиному рядку виведіть одне ціле число — кількість дівчат, яких Зеник зможе взяти на футбол.

Код, дизайн та оцінка часу

Завдання №1.1

Запис текстової стрічки у файл із заданим ім’ям;

Завдання №1.2

Копіювання вмісту файла у інший файл

```cpp
#include <iostream>
#include <fstream>
#include <cstring>

using namespace std;

enum FileOpResult { Failure, Success };

FileOpResult write_to_file(char* name, char* content) {
    if (name == nullptr || strlen(name) == 0) {
        cout << "Cannot create file without a name" << endl;
        return Failure;
    }

    ofstream myFile(name);
    if (!myFile.is_open()) {
        cout << "Can't open file: " << name << endl;
        return Failure;
    }

    myFile << content << endl;
    if (myFile.fail()) {
        cout << "Error while writing data to file: " << name << endl;
        return Failure;
    }

    myFile.close();
    if (myFile.fail()) {
        cout << "Error closing file: " << name << endl;
        return Failure;
    }

    return Success;
}

FileOpResult copy_file(char* file_from, char* file_to) {
    if (file_from == nullptr || file_to == nullptr || strlen(file_from) == 0 || strlen(file_to) == 0) {
        cout << "File names cannot be empty" << endl;
        return Failure;
    }
```

```
41
42      ifstream inputFile(file_from);
43      if (!inputFile.is_open()) {
44          cout << "No such file: " << file_from << endl;
45          return Failure;
46      }
47
48      ofstream outputFile(file_to);
49      if (!outputFile.is_open()) {
50          cout << "Can't open destination file: " << file_to << endl;
51          inputFile.close();
52          return Failure;
53      }
54
55      char buffer[256];
56      while (inputFile.getline(buffer, sizeof(buffer))) {
57          outputFile << buffer << endl;
58          if (outputFile.fail()) {
59              cout << "Error while writing to file: " << file_to << endl;
60              inputFile.close();
61              outputFile.close();
62              return Failure;
63          }
64      }
65
66      inputFile.close();
67      outputFile.close();
68
69      return Success;
70  }
71
72  int main() {
73      char filename1[256], content[256];
74      cout << "Enter the name of the file: ";
75      cin.getline(filename1, sizeof(filename1));
76
77      cout << "Enter content for the file: ";
78      cin.getline(content, sizeof(content));
79
```

```
79
80      if (write_to_file(filename1, content) == Success) {
81          cout << "File written successfully!" << endl;
82      } else {
83          cout << "Failed to write to file." << endl;
84          return 1;
85      }
86
87      char filename2[256];
88      cout << "Enter the name of the file to copy to: ";
89      cin.getline(filename2, sizeof(filename2));
90
91      if (copy_file(filename1, filename2) == Success) {
92          cout << "File copied successfully!" << endl;
93      } else {
94          cout << "Failed to copy the file." << endl;
95          return 1;
96      }
97
98      return 0;
99  }
```

```
Enter the name of the file: photos
Enter content for the file: jpg
File written successfully!
Enter the name of the file to copy to: more photos
File copied successfully!
```

Витрачено приблизно 3.5 години

Завдання №2 VNS Lab#6(Варіант 18)

```cpp
#include <iostream>
#include <vector>
#include <cstring>
#include <algorithm>
#include <cctype>

using namespace std;

int main() {
    cout << "Enter string: ";
    char s[256];
    cin.getline(s, 256);

    int len = strlen(s);
    if (len > 0 && s[len - 1] == '.') {
        s[len - 1] = '\0';
    }

    vector<string> words;
    char* token = strtok(s, " ");

    while (token != nullptr) {
        string word(token);

        if (!word.empty() && isdigit(word[0])) {
            words.push_back(word);
        }

        token = strtok(nullptr, " ");
    }

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

    cout << "Words that started with nums:" << endl;
    for (const auto& word : words) {
        cout << word << endl;
    }

    return 0;
}
```

```
Enter string: 123bro SHADOW777 789ENEMY !love
Words that started with nums:
789ENEMY
123bro
```

Витрачено приблизно 1.5 години

Завдання №3 VNS Lab#8(Варіант 18)

```cpp
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
using namespace std;

// Structure "Book"
struct Book {
    string title;
    string author;
    int year;
    int pages;
};

void writeToFile(const string& filename, const vector<Book>& books) {
    ofstream outFile(filename, ios::binary);
    if (!outFile) {
        cerr << "Error opening file for writing!" << endl;
        return;
    }
    for (const auto& book : books) {
        outFile.write(reinterpret_cast<const char*>(&book), sizeof(Book));
    }
    outFile.close();
}

void printFile(const string& filename) {
    ifstream inFile(filename, ios::binary);
    if (!inFile) {
        cerr << "Error opening file for reading!" << endl;
        return;
    }
    Book book;
    while (inFile.read(reinterpret_cast<char*>(&book), sizeof(Book))) {
        cout << "Title: " << book.title << ", Author: " << book.author
             << ", Year: " << book.year << ", Pages: " << book.pages << endl;
    }
    inFile.close();
}

void deleteFirstThree(const string& filename) {
    ifstream inFile(filename, ios::binary);
    if (!inFile) {
        cerr << "Error opening file for reading!" << endl;
        return;
    }
    vector<Book> books;
    Book book;
    int index = 0;
    while (inFile.read(reinterpret_cast<char*>(&book), sizeof(Book))) {
        if (index++ >= 3) {
            books.push_back(book);
        }
    }
    inFile.close();

    ofstream outFile(filename, ios::binary);
    if (!outFile) {
        cerr << "Error opening file for writing!" << endl;
        return;
    }
    for (const auto& b : books) {
        outFile.write(reinterpret_cast<const char*>(&b), sizeof(Book));
    }
    outFile.close();
}

void addBeforeTitle(const string& filename, const string& targetTitle, const Book& newBook) {
    ifstream inFile(filename, ios::binary);
    if (!inFile) {
        cerr << "Error opening file for reading!" << endl;
        return;
    }
    vector<Book> books;
    Book book;
    bool inserted = false;
    while (inFile.read(reinterpret_cast<char*>(&book), sizeof(Book))) {
        if (!inserted && book.title == targetTitle) {
            books.push_back(newBook);
            inserted = true;
```

```
    while (inFile.read(reinterpret_cast<char*>(&book), sizeof(Book))) {
        if (!inserted && book.title == targetTitle) {
            books.push_back(newBook);
            inserted = true;
        }
        books.push_back(book);
    }
    inFile.close();

    ofstream outFile(filename, ios::binary);
    if (!outFile) {
        cerr << "Error opening file for writing!" << endl;
        return;
    }
    for (const auto& b : books) {
        outFile.write(reinterpret_cast<const char*>(&b), sizeof(Book));
    }
    outFile.close();
}

int main() {
    string filename = "books.bin";
    vector<Book> books = {
        {"Book1", "Author1", 2001, 100},
        {"Book2", "Author2", 2002, 200},
        {"Book3", "Author3", 2003, 300},
        {"Book4", "Author4", 2004, 400}
    };

    writeToFile(filename, books);

    cout << "Initial file content:" << endl;
    printFile(filename);

    deleteFirstThree(filename);
    cout << "\nAfter deleting first 3 elements:" << endl;
    printFile(filename);

    Book newBook = {"NewBook", "NewAuthor", 2024, 150};
    addBeforeTitle(filename, "Book4", newBook);
    cout << "\nAfter adding a new element before 'Book4':" << endl;
    printFile(filename);

    return 0;
}
```

```
Initial file content:
Title: Book1, Author: Author1, Year: 2001, Pages: 100
Title: Book2, Author: Author2, Year: 2002, Pages: 200
Title: Book3, Author: Author3, Year: 2003, Pages: 300
Title: Book4, Author: Author4, Year: 2004, Pages: 400
```

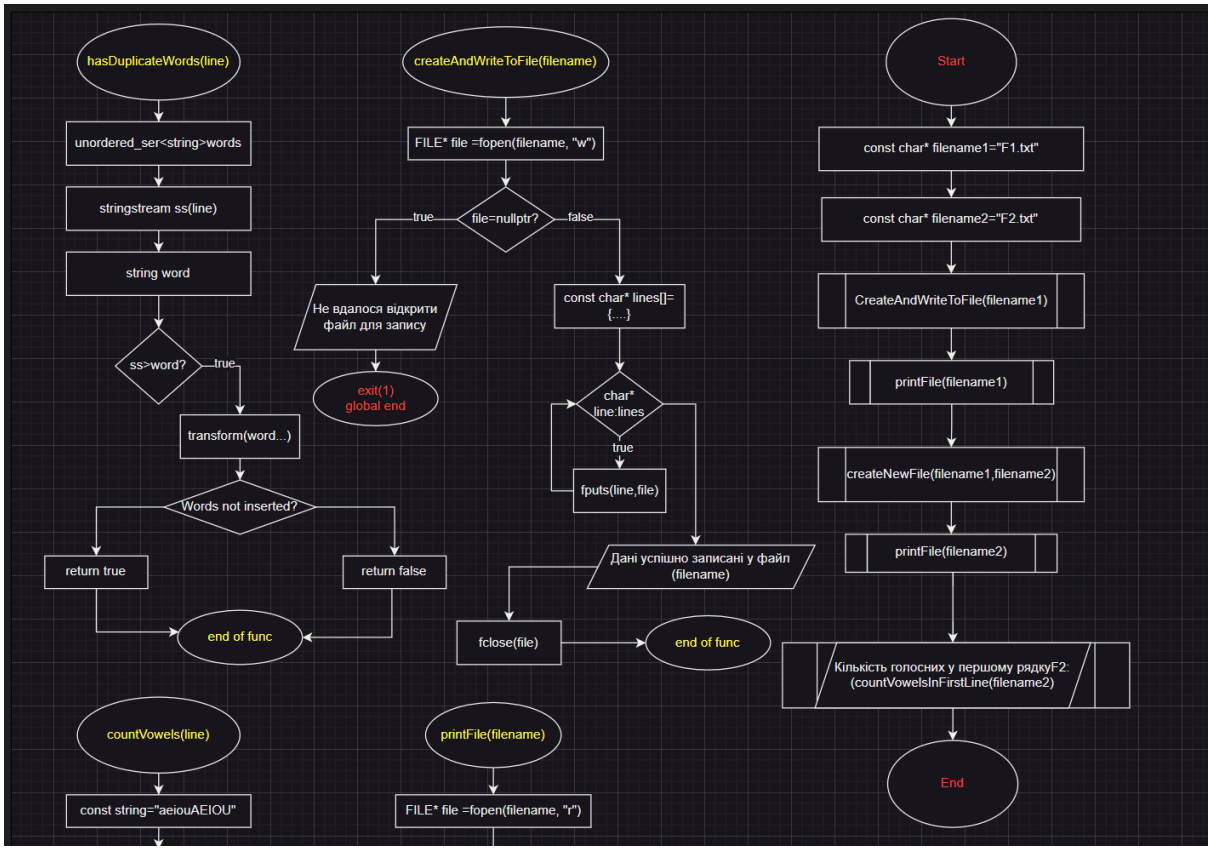
Витрачено приблизно 4 години

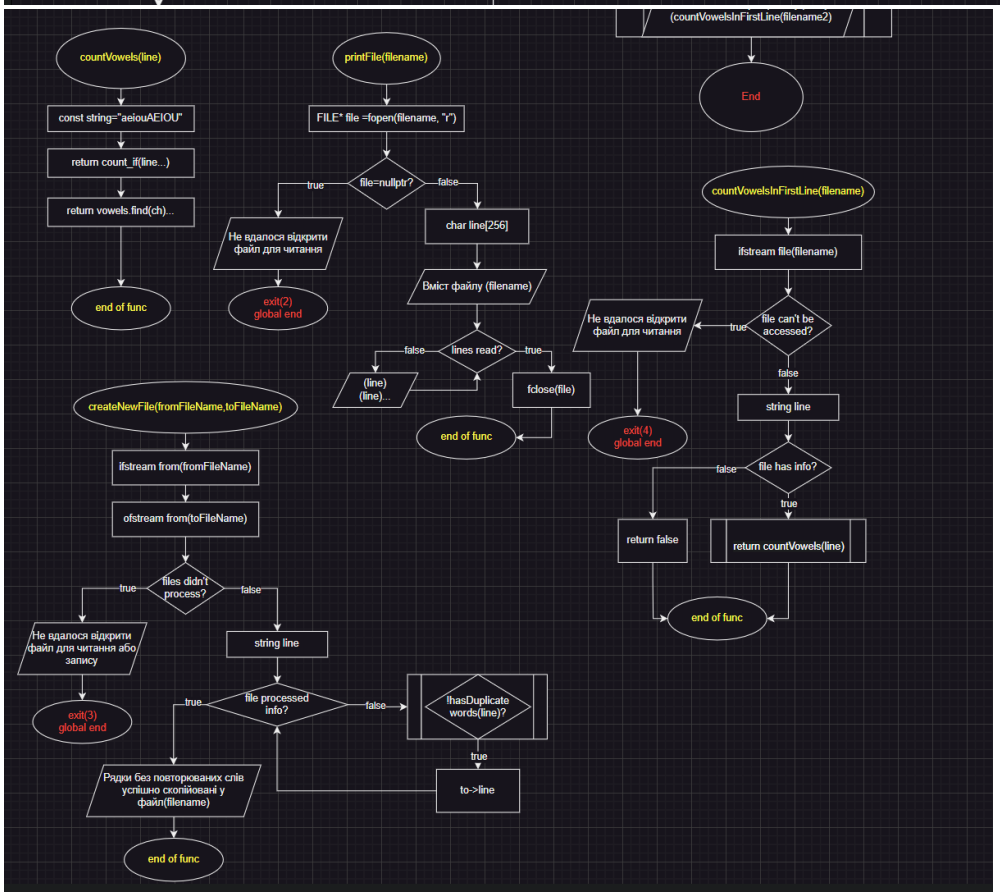
Завдання №4 VNS Lab#9(Варіант 18)

```cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <unordered_set>
#include <cstring>
#include <cctype>
#include <algorithm>

using namespace std;

bool hasDuplicateWords(const string& line) {
    unordered_set<string> words;
    stringstream ss(line);
    string word;
    while (ss >> word) {
        transform(word.begin(), word.end(), word.begin(), ::tolower);
        // Якщо вставлення повертає false, значить слово вже є в множині, і це дубль
        if (!words.insert(word).second) {
            return true;
        }
    }
    return false;
}

int countVowels(const string& line) {
    const string vowels = "aeiouAEIOU";
    return count_if(line.begin(), line.end(), [&vowels](char ch) { return vowels.find(ch) != string::npos; });
}

void createAndWriteToFile(const char* filename) {
    FILE* file = fopen(filename, "w");
    if (file == nullptr) {
        cerr << "Не вдалося відкрити файл для запису." << endl;
        exit(1);
    }

    const char* lines[] = {
        "This is a test line.\n",
        "This line contains duplicate duplicate words.\n",
        "Unique line is here.\n",
        "Another line line with duplicates.\n",
        "Programming is fun and rewarding.\n",
        "Learn to code and build something fun 'yep'.\n",
        "Learn to code and build something fun 'yep'.\n",
        "Building projects is fun fun fun.\n",
        "You must broaden your mindset in order to become powerful\n",
        "Growth is lifelong lifelong journey.\n",
        "Easter egg:(=+=PRESS F TO PAY RESPECT=+=).\n"
    };

    for (const char* line : lines) {
        fputs(line, file);
    }

    cout << "Дані успішно записані у файл " << filename << endl;
    fclose(file);
}

void printFile(const char* filename) {
    FILE* file = fopen(filename, "r");
    if (file == nullptr) {
        cerr << "Не вдалося відкрити файл для читання." << endl;
        exit(2);
    }

    char line[256];
    cout << "Вміст файлу " << filename << ":\n";
    cout << "--------------------------------\n";
    while (fgets(line, sizeof(line), file)) {
        cout << line;
    }
    fclose(file);
}

void createNewFile(const char* fromFileName, const char* toFileName) {
    ifstream from(fromFileName);
    ofstream to(toFileName);

    if (!from || !to) {
        cerr << "Не вдалося відкрити файл для читання або запису." << endl;
        exit(3);
    }

```

```
82
83      string line;
84      while (getline(from, line)) {
85          if (!hasDuplicateWords(line)) {
86              to << line << endl;
87          }
88      }
89
90      cout << "Рядки без повторюваних слів успішно скопійовані у файл " << toFileName << endl;
91  }
92
93  int countVowelsInFirstLine(const char* filename) {
94      ifstream file(filename);
95      if (!file) {
96          cerr << "Не вдалося відкрити файл для читання." << endl;
97          exit(4);
98      }
99
100     string line;
101     if (getline(file, line)) {
102         return countVowels(line);
103     }
104     return 0;
105 }
106
107 int main() {
108     const char* filename1 = "F1.txt";
109     const char* filename2 = "F2.txt";
110
111     createAndWriteToFile(filename1);
112
113     printFile(filename1);
114
115     createNewFile(filename1, filename2);
116
117     printFile(filename2);
118
119     cout << endl
120         << "Кількість голосних у першому рядку файлу F2: " << countVowelsInFirstLine(filename2) << endl;
121
122     return 0;
123 }
```

```
Дані успішно записані у файл F1.txt
Вміст файлу F1.txt:
---------------------------------
This is a test line.
This line contains duplicate duplicate words.
Unique line is here.
Another line line with duplicates.
Programming is fun and rewarding.
Learn to code and build something fun 'yep'.
Building projects is fun fun fun.
You must broaden your mindset in order to become powerful
Growth is lifelong lifelong journey.
Easter egg:(=+=PRESS F TO PAY RESPECT=+=).
Рядки без повторюваних слів успішно скопійовані у файл F2.txt
Вміст файлу F2.txt:
---------------------------------
This is a test line.
Unique line is here.
Programming is fun and rewarding.
Learn to code and build something fun 'yep'.
You must broaden your mindset in order to become powerful
Easter egg:(=+=PRESS F TO PAY RESPECT=+=).

Кількість голосних у першому рядку файлу F2: 6
```





Витрачено приблизно 4 години
На блок-схеми пішло 3 години

Завдання №5 Algotester Lab 4v1

```cpp
#include <iostream>
#include <vector>
#include <set>
#include <algorithm>
using namespace std;

void PrintResult(const vector<int>& result) {
    cout << result.size() << endl;
    for (int elm : result) {
        cout << elm << " ";
    }
    cout << endl << endl;
}

int main() {
    int sizeN, sizeM;
    cin >> sizeN;
    vector<int> N(sizeN);
    for (int i = 0; i < sizeN; i++) {
        cin >> N[i];
    }

    cin >> sizeM;
    vector<int> M(sizeM);
    for (int i = 0; i < sizeM; i++) {
        cin >> M[i];
    }

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

    vector<int> result;

    result.resize(N.size());
    auto it = set_difference(N.begin(), N.end(), M.begin(), M.end(), result.begin());
    result.resize(it - result.begin());
    PrintResult(result);

    result.resize(M.size());
    it = set_difference(M.begin(), M.end(), N.begin(), N.end(), result.begin());
    result.resize(it - result.begin());
    PrintResult(result);

    result.resize(min(N.size(), M.size()));
    it = set_intersection(N.begin(), N.end(), M.begin(), M.end(), result.begin());
    result.resize(it - result.begin());
    PrintResult(result);

    result.resize(N.size() + M.size());
    it = set_union(N.begin(), N.end(), M.begin(), M.end(), result.begin());
    result.resize(it - result.begin());
    PrintResult(result);

    result.resize(N.size() + M.size());
    it = set_symmetric_difference(N.begin(), N.end(), M.begin(), M.end(), result.begin());
    result.resize(it - result.begin());
    PrintResult(result);

    return 0;
}
```

```
5
1 2 3 4 5
5
4 5 6 7 8
3
1 2 3

3
6 7 8

2
4 5

8
1 2 3 4 5 6 7 8

6
1 2 3 6 7 8
```

7 годин тому	Lab 4v1 - Lab 4v1	C++ 23	Зараховано	0.003	1.422	1909234

Витрачено приблизно 1.5 години

Завдання №6 Algotester Lab 4v2

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    int N, K;
    cin >> N >> K;
    vector<int> arr(N);

    for (int i = 0; i < N; i++) {
        cin >> arr[i];
    }

    sort(arr.begin(), arr.end());
    auto last_unique = unique(arr.begin(), arr.end());
    arr.erase(last_unique, arr.end());

    int rotation_index = K % arr.size();
    rotate(arr.begin(), arr.begin() + rotation_index, arr.end());

    cout << arr.size() << "\n";
    for (int num : arr) {
        cout << num << " ";
    }
    cout << "\n";

    return 0;
}
```

```
10 3
1 2 2 3 3 3 4 5 6 7
7
4 5 6 7 1 2 3
```

2 дні тому	Lab 4v2 - Lab 4v2	C++ 23	Зараховано	0.003	1.281	1907901

Витрачено приблизно 45 хвилин

Завдання №7 Algotester Lab 6v2

```cpp
#include <iostream>
#include <vector>
#include <unordered_set>
#include <algorithm>

using namespace std;

bool isValidCell(int row, int col) {
    return row >= 0 && row < 8 && col >= 0 && col < 8;
}

void markPawnAttacks(vector<unordered_set<char>>& attackMap, int row, int col) {
    if (isValidCell(row + 1, col - 1)) attackMap[(row + 1) * 8 + (col - 1)].insert('P');
    if (isValidCell(row + 1, col + 1)) attackMap[(row + 1) * 8 + (col + 1)].insert('P');
}

void markRookAttacks(vector<unordered_set<char>>& attackMap, int row, int col, char rookType) {
    for (int i = 0; i < 8; ++i) {
        if (i != col) attackMap[row * 8 + i].insert(rookType);
        if (i != row) attackMap[i * 8 + col].insert(rookType);
    }
}

void markBishopAttacks(vector<unordered_set<char>>& attackMap, int row, int col, char bishopType) {
    for (int i = 1; i < 8; ++i) {
        if (isValidCell(row + i, col + i)) attackMap[(row + i) * 8 + (col + i)].insert(bishopType);
        if (isValidCell(row + i, col - i)) attackMap[(row + i) * 8 + (col - i)].insert(bishopType);
        if (isValidCell(row - i, col + i)) attackMap[(row - i) * 8 + (col + i)].insert(bishopType);
        if (isValidCell(row - i, col - i)) attackMap[(row - i) * 8 + (col - i)].insert(bishopType);
    }
}

void markKnightAttacks(vector<unordered_set<char>>& attackMap, int row, int col) {
    int knightMoves[8][2] = {{-2, -1}, {-2, 1}, {2, -1}, {2, 1}, {-1, -2}, {1, -2}, {-1, 2}, {1, 2}};
    for (auto& move : knightMoves) {
        int newRow = row + move[0], newCol = col + move[1];
        if (isValidCell(newRow, newCol)) attackMap[newRow * 8 + newCol].insert('N');
    }
}

```

```cpp
void markKingAttacks(vector<unordered_set<char>>& attackMap, int row, int col) {
    for (int dx = -1; dx <= 1; ++dx) {
        for (int dy = -1; dy <= 1; ++dy) {
            if (dx != 0 || dy != 0) {
                int newRow = row + dx, newCol = col + dy;
                if (isValidCell(newRow, newCol)) attackMap[newRow * 8 + newCol].insert('K');
            }
        }
    }
}

int main() {
    vector<string> board(8);
    for (int i = 0; i < 8; ++i) {
        cin >> board[i];
    }

    vector<unordered_set<char>> attackMap(64);

    for (int row = 0; row < 8; ++row) {
        for (int col = 0; col < 8; ++col) {
            char piece = board[row][col];
            switch (piece) {
                case 'P': markPawnAttacks(attackMap, row, col); break;
                case 'R': markRookAttacks(attackMap, row, col, 'R'); break;
                case 'B': markBishopAttacks(attackMap, row, col, 'B'); break;
                case 'N': markKnightAttacks(attackMap, row, col); break;
                case 'K': markKingAttacks(attackMap, row, col); break;
                case 'Q':
                    markRookAttacks(attackMap, row, col, 'Q');
                    markBishopAttacks(attackMap, row, col, 'Q');
                    break;
                default: break;
            }
        }
    }
    int Q;
    cin >> Q;
```

```cpp
    int Q;
    cin >> Q;
    while (Q--) {
        int x, y;
        cin >> x >> y;
        --x; --y;

        if (board[x][y] != 'O') {
            cout << "X\n";
        } else if (attackMap[x * 8 + y].empty()) {
            cout << "O\n";
        } else {
            vector<char> attackers(attackMap[x * 8 + y].begin(), attackMap[x * 8 + y].end());
            sort(attackers.begin(), attackers.end());
            for (char attacker : attackers) {
                cout << attacker;
            }
            cout << "\n";
        }
    }
    return 0;
}
```

```
KOOOOOOO
OOOOOOOO
OOOOOOOO
OOOOOOOO
OOOOOOOO
OOOOOOOO
OOOOOOOO
OOOOOOOO

5
1 1
X
1 2
K
2 1
K
2 2
K
3 1
O
```

6 годин тому	Lab 6v2 - Lab 6v2	C++ 23	Зараховано	0.003	1.301	1909279

Витрачено приблизно 4 години

Завдання №8 Algotester self-practice task

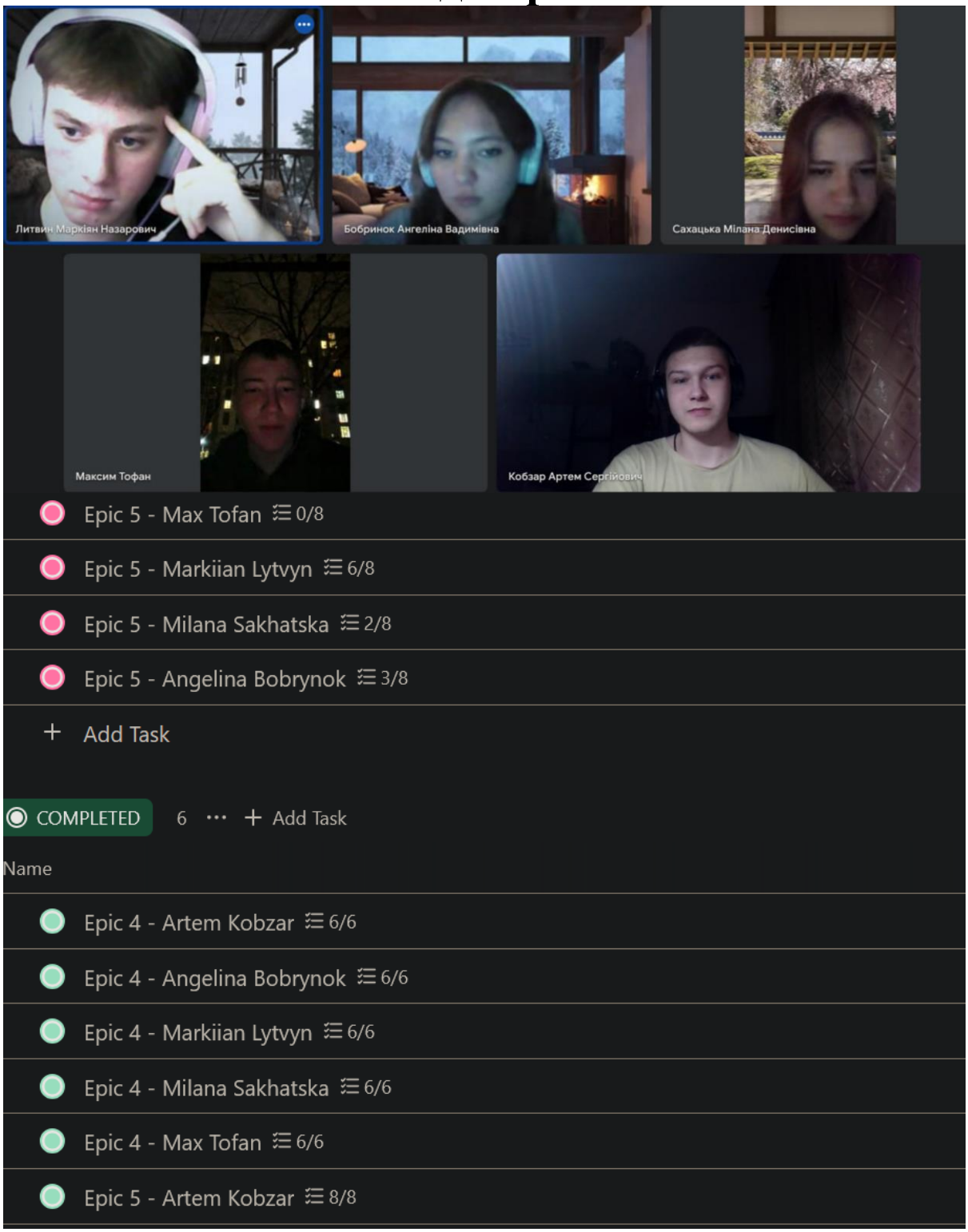
```cpp
#include <iostream>
#include <algorithm>

int main() {
    int n, k, z;
    std::cin >> n >> k >> z;
    int* prices = new int[n];

    for (int i = 0; i < n; ++i) {
        std::cin >> prices[i];
        prices[i] *= 2;
    }
    std::sort(prices, prices + n);
    int girls = 0;

    for (int i = 0; i < n; ++i) {
        if (z >= prices[i]) {
            z -= prices[i];
            ++girls;
        } else {
            break;
        }
    }
    std::cout << std::min(girls, k) << std::endl;
    delete[] prices;

    return 0;
}
```

```
4 7 44
4 47 7 74
2
```

8 годин тому	0522 - Вболівальниці	C++ 23	Зараховано	0.003	1.180	1909153

Витрачено приблизно 40 хвилин

Командна робота



Висновок: у цьому блоці я застосував операції над файлами у програмах, а також використав структури, вектори, рядки та функції.