

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(ка) групи ШІ-13

Яцишин Роман Олегович

Тема :

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

Мета :

Навчитись працювати з файлами, вносити у нього зміни. Опрацювати деталі роботи з файлами, набір команд у бібліотеці. Навчитись створювати власні бібліотеки та доцільно їх використовувати.

Теоретичні відомості:

1. Вступ до Роботи з Файлами:
 - Основні операції з файлами: відкриття, читання, запис, закриття
 - Робота з файловими дескрипторами
2. Символи і Рядкові Змінні:
 - Робота з `char` та `string`: основні операції і методи
3. Текстові Файли:
 - Особливості читання та запису текстових файлів
 - Обробка рядків з файлу: `getline`, `ignore`, `peek`
 - Форматування тексту при записі: `setw`, `setfill`, `setprecision`
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`
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

Виконання роботи:

1. Опрацювання завдання та вимог до програм та середовища:

Програмний код №1

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.

Програмний код №2

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.

Програмний код №3

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.

Програмний код №4

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.

Програмний код №4

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

Програмний код №5

У Клінта в черговий раз виключилось світло і йому немає чим зайнятися. Так як навіть це не заставить його подивитися збережені відео про програмування на ютубі - він вирішив придумати свою гру на основі sudoku.

Гра виглядає так:

Є поле розміром $N \times N$, в якому частина клітинок заповнена цифрами, а частина клітинок пусті (позначаються нулем). Також у нього є Q пар координат X та Y .

Завданням гри є написати до кожної координати скільки чисел туди можна вписати (якщо вона пуста) і які це числа (обов'язково в посортовані по зростанню!). В клітинку можна вписати лише ті числа, які не зустрічаються в рядку та стовбці, які перетинаються у цій клітинці.

Під час гри поле не міняється!

Також необов'язково, щоб це було валідне sudoku! Якщо є клітинка, в яку не можна вписати ніяку цифру - виведіть 0.

Також допускаються рядки та стовпці, в яких цифра записана кілька разів.

Програмний код №6

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.

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.

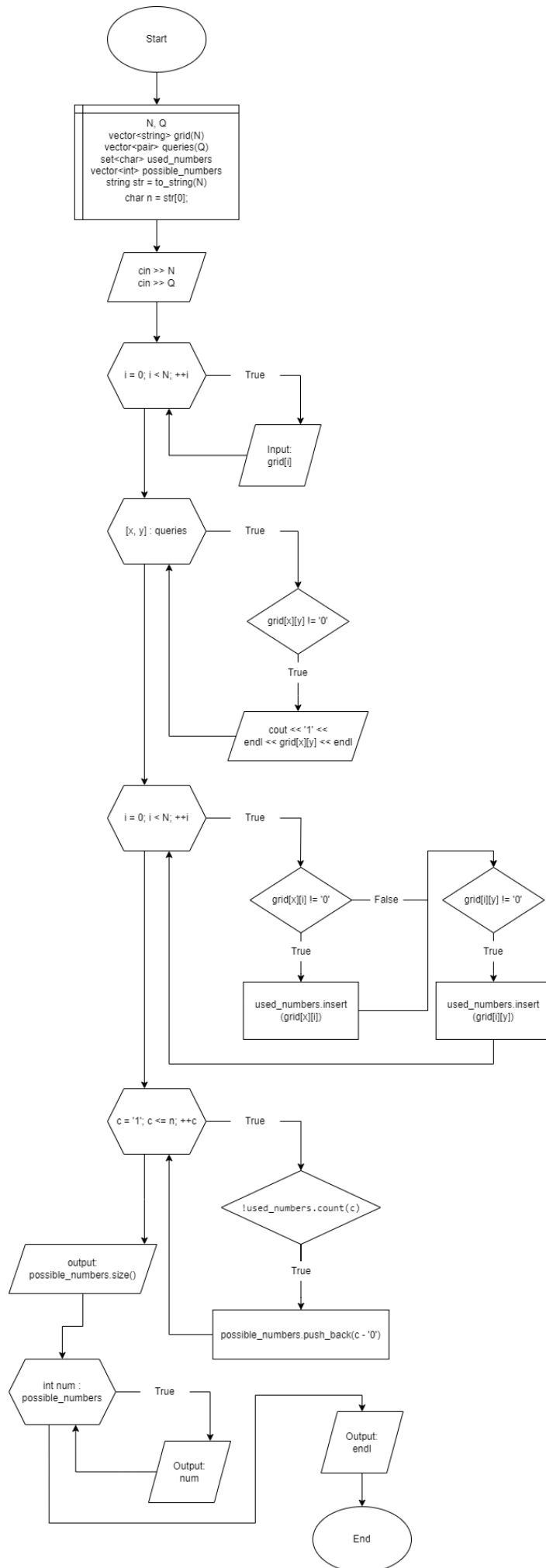
Програмний код №7

Після завершення основного туру олімпіади з програмування Зеник отримав невеличкий клаптик паперу, на якому було надруковане число x — кількість балів, що набрав Зеник. Зауважте, що згідно з кращими традиціями олімпіади з програмування, кількість балів Зеника не може бути нульовою чи від'ємною.

Помітивши не дуже щасливе обличчя Зеника, Марічка нагадала йому про щасливі цифри. Як ви вже напевно знаєте, щасливими вважають цифри 4 та 7. Марічка запевнила Зеника, що найкращим є не найбільший результат, а той, десятковий запис якого містить найбільше щасливих цифр.

Вам необхідно допомогти юному учаснику олімпіади з програмування та поррахувати кількість щасливих цифр у його результаті.

2. Дизайн:



3. Код програми:

1) Algotester Lab 4 Variant 2

```
1  <#include <iostream>
2  <#include <sstream>
3  <#include <algorithm>
4  <#include <vector>
5
6  using namespace std;
7
8  <int partition(vector<int> &vec, int low, int high) {
9
10     int pivot = vec[high];
11     int i = (low - 1);
12
13     for (int j = low; j <= high - 1; j++) {
14         if (vec[j] <= pivot) {
15             i++;
16             int temp = vec[i];
17             vec[i] = vec[j];
18             vec[j] = temp;
19         }
20     }
21     int temp2 = vec[i + 1];
22     vec[i + 1] = vec[high];
23     vec[high] = temp2;
24     return (i + 1);
25 }
26
27 <void quickSort(vector<int> &vec, int low, int high) {
28
29     if (low < high) {
30         int pi = partition(vec, low, high);
31         quickSort(vec, low, pi - 1);
32         quickSort(vec, pi + 1, high);
33     }
34 }
35
36 <void removeDuplicates(std::vector<int>& vec) {
37     int size = vec.size();
38 }
```

```

39     for (int i = 0; i < size; i++) {
40         for (int j = i + 1; j < size; j++) {
41             if (vec[i] == vec[j]) {
42                 for (int k = j; k < size - 1; k++) {
43                     vec[k] = vec[k + 1];
44                 }
45                 size--;
46                 j--;
47             }
48         }
49     }
50
51     vec.resize(size);
52 }
53
54
55 int main(){
56     int element = 0;
57     int N = 0;
58     int K = 0;
59     cin >> N >> K;
60
61     vector<int> arr;
62
63     for (int c = 0; c < N; c++) {
64         cin >> element;
65         arr.push_back(element);
66     }
67
68     removeDuplicates(arr);
69     quickSort(arr, 0, arr.size() - 1);
70
71     vector<int> temp(arr.begin(), arr.begin() + K);
72
73
74
75
76
77     for (int i = 0; i < K; i++) {
78         arr[arr.size() - K + i] = temp[i];
79     }
80
81     int M = arr.size();
82     cout << M << endl;
83     for (int num : arr) {
84         cout << num << " ";
85     }
86
87     return 0;
88 }

```

2) Self Practice task


```
1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5
6  int main() {
7
8      string x;
9      cin >> x;
10
11     int count = 0;
12
13     for (char digit : x) {
14         if (digit == '4' || digit == '7') {
15             count++;
16         }
17     }
18
19     cout << count << endl;
20
21     return 0;
22 }
```

3) Class practice task

```

1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  #include <cstring>
5
6  using namespace std;
7
8  enum FileOpResult {Success, Failure};
9  FileOpResult write_to_file(char *name, char *content){
10     ofstream outFile(name, ios::app);
11     if (outFile.is_open()) {
12         outFile << content;
13         outFile.close();
14     } else {
15         return Failure;
16     }
17
18     outFile.close();
19
20     return Success;
21 }
22
23 FileOpResult copy_file(char *file_from, char *file_to){
24     ifstream inFile(file_from, ios::app);
25     ofstream outFile(file_to, ios::app);
26
27     if (inFile.is_open())
28     {
29         if (outFile.is_open())
30         {
31             string content;
32             while (getline(inFile, content)) {
33                 outFile << content << endl;
34             }
35         } else {
36             cout << "file_to.txt is not found!";
37             return Failure;
38         }
39     }
40 }

```

```

39     } else {
40         cout << "file_from.txt is not found!";
41         return Failure;
42     }
43
44     inFile.close();
45     outFile.close();
46
47     return Success;
48 }
49
50 int main(){
51     string file_from = "file_from.txt";
52     string file_to = "file_to.txt";
53     string content;
54     getline(cin, content);
55
56     char* file_from_char = new char[file_from.length() + 1];
57     strcpy(file_from_char, file_from.c_str());
58
59     char* file_to_char = new char[file_to.length() + 1];
60     strcpy(file_to_char, file_to.c_str());
61
62     char* content_char = new char[content.length() + 1];
63     strcpy(content_char, content.c_str());
64
65
66     write_to_file(file_from_char, content_char);
67     copy_file(file_from_char, file_to_char);
68
69     delete[] file_from_char;
70     delete[] file_to_char;
71     delete[] content_char;
72 }
73

```

4) VNS Lab 6

```

1 // Надрукувати всі слова, які співпадають з її першим словом.
2
3 #include <iostream>
4 #include <sstream>
5 #include <string>
6 #include <vector>
7
8 using namespace std;
9
10 int main() {
11     string input;
12     getline(cin, input);
13     char delimiter = ' ';
14     istringstream iss(input);
15     string word;
16     vector<string> words;
17     vector<int> idxs;
18
19     while (getline(iss, word, delimiter)) {
20         words.push_back(word);
21     }
22
23     int idx = 0;
24     for (string w_now : words)
25     {
26         if ((w_now == words[0]) && (idx != 0))
27         {
28             cout << w_now << endl;
29         }
30         ++idx;
31     }
32
33     return 0;
34 }

```

5) VNS Lab 8

```

1  #include <iostream>
2  #include <fstream>
3  #include <iomanip>
4  #include <string>
5  #include <vector>
6
7  using namespace std;
8
9  struct Human {
10     string firstname;
11     string lastname;
12     string middlename;
13     string address;
14     string phonenumber;
15     string age;
16 };
17
18 void writeRecord(ofstream &outFile, const Human &person) {
19     size_t length;
20
21     // Write `firstname`
22     length = person.firstname.size();
23     outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
24     outFile.write(person.firstname.c_str(), length);
25
26     // Write `lastname`
27     length = person.lastname.size();
28     outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
29     outFile.write(person.lastname.c_str(), length);
30
31     // Write `middlename`
32     length = person.middlename.size();
33     outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
34     outFile.write(person.middlename.c_str(), length);
35
36     // Write `address`
37     length = person.address.size();
38     outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
39     outFile.write(person.address.c_str(), length);
40
41     // Write `phonenumber`
42     length = person.phonenumber.size();
43     outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
44     outFile.write(person.phonenumber.c_str(), length);
45
46     // Write `age`
47     length = person.age.size();
48     outFile.write(reinterpret_cast<const char*>(&length), sizeof(length));
49     outFile.write(person.age.c_str(), length);
50 }
51

```

```
52 void readRecord(istream &inFile, Human &person) {
53     size_t length;
54
55     // Read `firstname`
56     inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
57     person.firstname.resize(length);
58     inFile.read(&person.firstname[0], length);
59
60     // Read `lastname`
61     inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
62     person.lastname.resize(length);
63     inFile.read(&person.lastname[0], length);
64
65     // Read `middlename`
66     inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
67     person.middlename.resize(length);
68     inFile.read(&person.middlename[0], length);
69
70     // Read `address`
71     inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
72     person.address.resize(length);
73     inFile.read(&person.address[0], length);
74
75     // Read `phonenumber`
76     inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
77     person.phonenumber.resize(length);
78     inFile.read(&person.phonenumber[0], length);
79
80     // Read `age`
81     inFile.read(reinterpret_cast<char*>(&length), sizeof(length));
82     person.age.resize(length);
83     inFile.read(&person.age[0], length);
84 }
85
```

```

86 void displayRecord(const Human &person) {
87     cout << "-----" << endl;
88     cout << "First Name : " << person.firstname << endl;
89     cout << "Last Name : " << person.lastname << endl;
90     cout << "Middle Name: " << person.middlename << endl;
91     cout << "Address : " << person.address << endl;
92     cout << "Phone No. : " << person.phonenumber << endl;
93     cout << "Age : " << person.age << endl;
94     cout << "-----" << endl;
95 }
96
97
98 int main(){
99     int idx;
100     cout << "index of record, after which to put new record: ";
101     cin >> idx;
102
103     vector<Human> records;
104     Human person;
105     // Human people[3] = {
106     //     {"Roman", "Yatsyshyn", "Olehowych", "city Lviv", "+38666", "18"},
107     //     {"Vitsliy", "Portnikov", "Batkowych", "city Kyiv", "+38666069", },
108     //     {"Orest", "Melnyk", "Mykhaylowych", "city NONE", " ", "60"}
109     // };
110
111     Human newRecord = {"Olersandr", "Matrunych", "", "village Domazhyr", "+38096969696", "17"};
112
113     // ofstream outFile("humans.bin", ios::binary);
114     // if (!outFile) {
115     //     cerr << "Failed to open file for writing." << endl;
116     //     return 1;
117     // }
118
119     // // Write each record to the file
120     // for (const Human &person : people) {
121     //     writeRecord(outFile, person);
122     // }
123

```

```

124     // outFile.close();
125     // cout << "Records written successfully to 'humans.dat'." << endl;
126
127     ifstream file("humans.bin", ios::binary);
128
129     if (file){
130         while (file.peek() != EOF) { // Check for end of file
131             readRecord(file, person);
132             displayRecord(person);
133             records.push_back(person);
134         }
135     }
136     else {
137         cerr << "Failed to open file for writing." << endl;
138     }
139
140     file.close();
141
142     records.insert(records.begin() + idx + 1, newRecord);
143
144     ofstream tempFile("temp.bin", ios::binary);
145     if (!tempFile) {
146         cerr << "Failed to open temporary file for writing." << endl;
147         return 0;
148     }
149
150     for (const Human &record : records) {
151         if (record.age.empty() != true) {
152             writeRecord(tempFile, record);
153         }
154     }
155     tempFile.close();
156
157     string filename = "humans.bin";
158     string tempfilename = "temp.bin";
159
160     if (remove(filename.c_str()) != 0) {
161         cerr << "Error deleting original file." << endl;
162         return 0;
163     }
164     if (rename(tempfilename.c_str(), filename.c_str()) != 0) {
165         cerr << "Error renaming temporary file." << endl;
166         return 0;
167     }
168
169     cout << "Record added successfully." << endl;
170
171     return 0;
172 }
173

```

6) VNS Lab 9


```

1 // 1) Скопіювати з файлу F1 у файл F2 рядки, починаючи з 4.
2 // 2) Підрахувати кількість символів в останньому слові F2.
3
4 #include <iostream>
5 #include <fstream>
6 #include <iomanip>
7 #include <string>
8 #include <vector>
9
10 using namespace std;
11
12 int main(){
13     string filename_f1 = "F1.txt";
14     string filename_f2 = "F2.txt";
15     string arr[13] = {
16         "- .... ", "- - - - . . . . - - - -", "- . - . - - - -", "- - - -",
17         "- . . - - -", "- - . . . . - - -", "- . - - - . . . . -",
18         "- . . . .", "- - - -", "- . . . . - - - . . . .", "- - . . . . - - -",
19         "- . - - - - . . -", "- . . - - -"
20     };
21     size_t nmbr_of_symbols = 0;
22     int idx_last = 10;
23     int idx_target = 4;
24     int idx_now = 0;
25     string line;
26     ifstream file1(filename_f1, ios::app);
27     ifstream file2(filename_f2, ios::app);
28     // ofstream file1(filename_f1, ios::app);
29     // ofstream file2(filename_f2, ios::app);
30

```

```

30
31     if ((file1.is_open()) && (file2.is_open()))
32     {
33         // for (int i = 0; i < 13; i++)
34         // {
35         //     file1 << arr[i] << endl;
36         // }
37         // while (getline(file1, line))
38         // {
39         //     ++idx_now;
40         //     if (idx_now >= idx_target)
41         //     {
42         //         file2 << line << endl;
43         //     }
44         // }
45         while (getline(file2, line))
46         {
47             ++idx_now;
48             if (idx_now == idx_last)
49             {
50                 // cout << line << endl;
51                 nmbr_of_symbols = line.size();
52                 break;
53             }
54         }
55         cout << nmbr_of_symbols << endl;
56
57     } else {
58         cerr << "Error opening file for writing!\n";
59     }
60     file1.close();
61     file2.close();
62 }
63

```

7) Algotester Lab 4

```

1  ✓ #include <iostream>
2  #include <sstream>
3  #include <algorithm>
4  #include <vector>
5
6  using namespace std;
7
8  ✓ int main(){
9      int element = 0;
10     int n = 0;
11     cin >> n;
12
13     vector<int> arr;
14     vector<int> mod0, mod1, mod2;
15
16     ✓ for (int c = 0; c < n; c++) {
17         cin >> element;
18         arr.push_back(element);
19     }
20
21     ✓ for (int num : arr) {
22         if (num % 3 == 0) mod0.push_back(num);
23         else if (num % 3 == 1) mod1.push_back(num);
24         else mod2.push_back(num);
25     }
26
27     // Сортування
28     sort(mod0.begin(), mod0.end());
29     sort(mod1.begin(), mod1.end(), greater<int>());
30     sort(mod2.begin(), mod2.end());
31
32     // Об'єднання частин
33     vector<int> result;
34     result.insert(result.end(), mod0.begin(), mod0.end());
35     result.insert(result.end(), mod1.begin(), mod1.end());
36     result.insert(result.end(), mod2.begin(), mod2.end());
37
38     vector<int>::iterator it;
39     it = unique(result.begin(), result.end());
40
41     int M = distance(result.begin(), it);
42     result.resize(distance(result.begin(), it));
43
44     cout << M << endl;
45     ✓ for (int num : result) {
46         cout << num << " ";
47     }
48
49     return 0;
50 }

```

8) Algotester Lab 6

```

1  #include <iostream>
2  #include <vector>
3  #include <set>
4  #include <string>
5  #include <algorithm>
6
7  using namespace std;
8
9  int main(){
10     int N = 0;
11     cin >> N;
12
13     vector<string> grid(N);
14     for (int i = 0; i < N; ++i) {
15         cin >> grid[i];
16     }
17
18     int Q;
19     cin >> Q;
20
21     vector<pair<int, int>> queries(Q);
22     for (int i = 0; i < Q; ++i) {
23         cin >> queries[i].first >> queries[i].second;
24         --queries[i].first;
25         --queries[i].second;
26     }
27
28     for (const auto& [x, y] : queries) {
29         // cout << grid[x][y] << endl;
30         if (grid[x][y] != '0') {
31             cout << '1' << endl << grid[x][y] << endl;
32             continue;

```

```

33     }
34
35     set<char> used_numbers;
36     for (int i = 0; i < N; ++i) {
37         if (grid[x][i] != '0') {
38             used_numbers.insert(grid[x][i]);
39         }
40         if (grid[i][y] != '0') {
41             used_numbers.insert(grid[i][y]);
42         }
43     }
44
45     vector<int> possible_numbers;
46
47     string str = to_string(N);
48     char n = str[0];
49     for (char c = '1'; c <= n; ++c) {
50         if (!used_numbers.count(c)) {
51             possible_numbers.push_back(c - '0');
52         }
53     }
54
55     cout << possible_numbers.size() << endl;
56     for (int num : possible_numbers) {
57         cout << num << " ";
58     }
59     cout << endl;
60 }
61
62 return 0;
63 }

```

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

1) Algotester Lab 4

```

PS D:\Epic 5> & 'c:\Users\
soft-MIEngine-In-0yr1zfal.t
-MIEngine-Pid-o2ruu5i3.vc5'
10
1 33 4 8 6 5 2 7 5 0
9
0 6 33 7 4 1 2 5 8
PS D:\Epic 5>

```

Фактично затрачений час: 20 хв

2) Algotester Lab 4 variant 2

```
PS D:\Epic 5> & 'c:\Users\1\\.vs
soft-MIEngine-In-odrzz0a3.1n1' '-MIEngine-Pid-pyquztlk.lnp' '--d
10 11
5 6 2 3 1 2 3 3 4 7
7
2 3 4 5 6 7 1
PS D:\Epic 5> █
```

Фактично затрачений час: 20 хв

3) Algotester Lab 6

```
PS D:\Epic 5> & 'c:\Users\1\.v
soft-MIEngine-In-nj1dfiqf.t4l'
-MIEngine-Pid-0m2kd1m.hhf' '--
3
000
100
003
3
1 1
2 3
2 1
2
2 3
1
2
1
1
PS D:\Epic 5> █
```

Фактично затрачений час: 40 хв

4) VNS Lab 6

```
PS D:\Epic 5> & 'c:\Users\1\.vscode\extensions\ms-vscode-vscode-MIEngine-In-jvwvsiai.jby' '--stdo
-MIEngine-Pid-1lpjnqqw.mdt' '--dbgExe=
abc def abc abc ddd fff xyz abc
abc
abc
abc
PS D:\Epic 5> █
```

Фактично затрачений час: 30 хв

5) VNS Lab 8

```

Decoded Text
. . . . . R o m a n . . .
. . . . . Y a t s y s h y n .
. . . . . O l e h o w y c h .
. . . . . c i t y L v i v
. . . . . + 3 8 6 6 6 .
. . . . . 1 8 .
O l e r s a n d r .
. M a t r u n y c h .
. . . . . v i l l a g
e D o m a z h y r .
. . + 3 8 0 9 6 9 6 9 6 9 6 .
. . . . . 1 7 .
O r e s t . . . . . M e l
n y k . . . . . M y k h a
y l o w y c h . . . . . c
i t y N O N e . . . . .
. . . . . 6 0 +

```

Фактично затрачений час: 2.5 год.

6) VNS Lab 9

≡ F1.txt	≡ F2.txt
1 -	1 - - - -
2 - - - - - - -	2 - . . - - -
3 . - - . - - - -	3 - - - -
4 - - - -	4 . - - - - . - . - -
5 - . . - - -	5
6 - - - -	6 - - - -
7 . - - - - . - . - -	7 - - - . . - .
8	8 . - - - -
9 - - - -	9 - . - - - - . - -
10 . - . . - - - . . - .	10 - . . - - -
11 . - - - -	11
12 - . - - - - . -	
13 - . . - - -	
14	

Фактично затрачений час: 1.2 год.

7) Class practice task

```
PS D:\Epic 5> & 'c:\Users\soft-MIEngine-In-buekyadg.z-MIEngine-Pid-4tvd1fzb.e3d'
Hello, World!
PS D:\Epic 5>
```

```
file_from.txt
1 Hello, World!
```

```
file_to.txt
1 Hello, World!
2
```

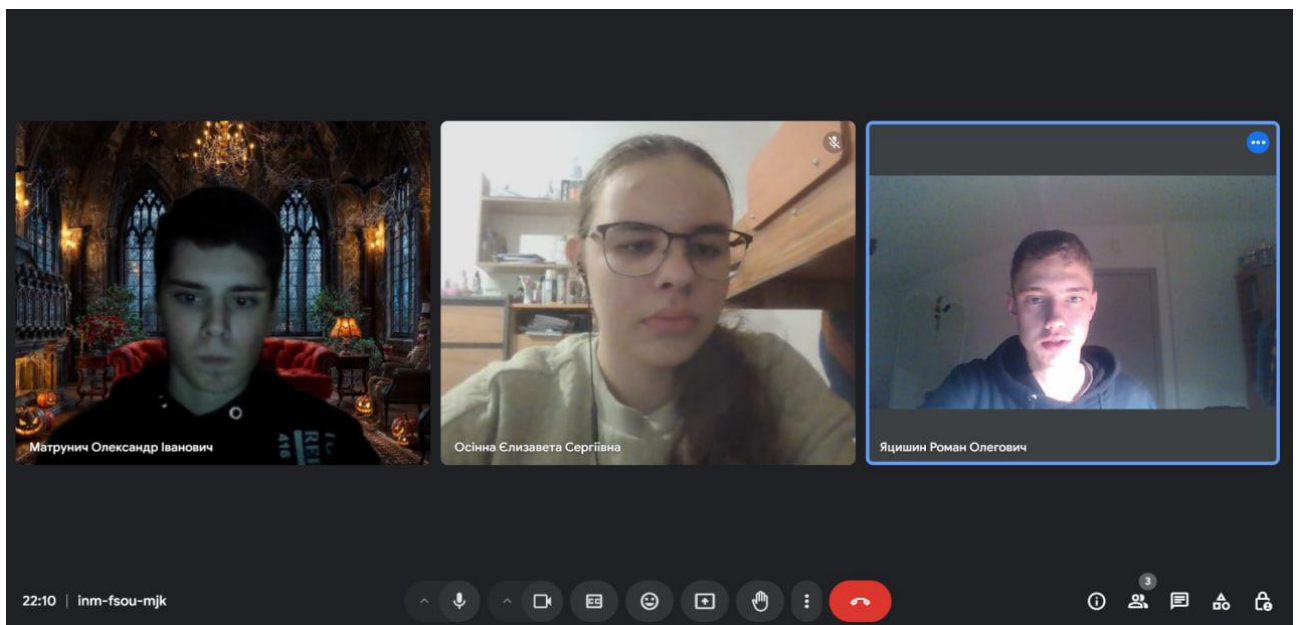
Фактично затрачений час: 90 хв.

8) Self practice task

```
PS D:\Epic 5> & 'c:\Users\soft-MIEngine-In-clnvprms.j-MIEngine-Pid-zf5mvwj0.20g'
54656416321351231
2
PS D:\Epic 5>
```

Фактично затрачений час: 10 хв.

6. Кооперація з командою:



Висновок:

У ході лабораторної роботи було здійснено роботу з текстовими та бінарними файлами, опрацьовано символи та рядкові змінні, засвоєно методи читання, запису та модифікації файлів за допомогою стандартної бібліотеки C++ та здійснено виконання завдання з використанням бібліотек та з власною реалізацією.

