

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-12
Гаврих Юрій Дмитрович

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:

Навчитись працювати з файловою системою в C++ , рядками типу `std::string` та `char`. Використання стандартної бібліотеки та створення власних бібліотек.

Теоретичні відомості:

[Базовий файловий ввід і вивід](#)

[Символьний тип даних char](#)

Виконання роботи

Task 3 - Lab# programming: VNS Lab 6

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.

15. Визначити яке слово зустрічається в рядку найчастіше.

Час виконання ~ 1,5 год.

Розв'язок:

```

1  #include <bits/stdc++.h>
2  #include<sstream>
3
4  using namespace std;
5
6  //Визначити яке слово зустрічається в рядку найчастіше.
7
8  int main() {
9      char s[256];
10     cout << "Введіть рядок : ";
11     gets(s);
12
13     // Видаляємо крапку в кінці, якщо вона є
14     int len = strlen(s);
15     if (len > 0 && s[len - 1] == '.') {
16         s[len - 1] = '\0';
17     }
18
19
20     stringstream ss(s);
21     string word;
22     map<string, int> word_count;
23
24     while (ss >> word) {
25         word_count[word]++;
26     }
27
28     vector<string> most_frequent_words;
29     int max_count = 0;
30
31     for (auto current: word_count)
32         max_count = max(max_count, current.second);
33
34     for (auto current: word_count) {
35         if (current.second == max_count) {
36             most_frequent_words.push_back(current.first);
37         }
38     }
39
40     if (!most_frequent_words.empty()) {
41         cout << "Найчастіше(ші) слово(а): { ";
42         for (auto most_frequent_word: most_frequent_words) cout << "\"" << most_frequent_word << "\" ";
43         cout << "} кількість повторів: " << max_count << endl;
44     } else {
45         cout << "Рядок порожній або не містить слів." << endl;
46     }
47
48     return 0;
49 }

```

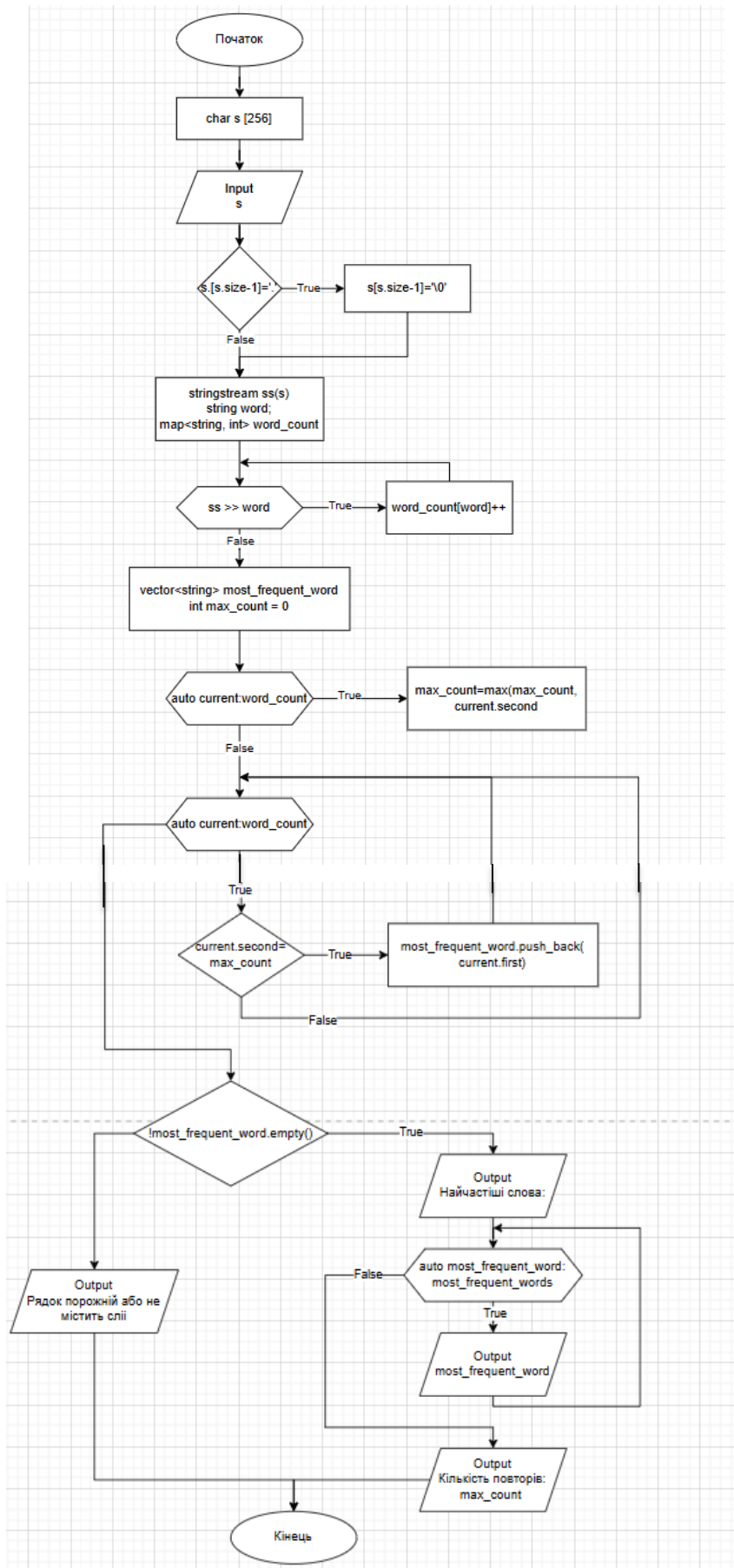
Результат виконання:

```

Введіть рядок : hi i am good hi am
Найчастіше(ші) слово(а): { "ам" "hi" } кількість повторів: 2

```

Блок-схема:



Task 4 - Lab# programming: VNS Lab 8

Час виконання ~ 2 год.

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.

Структура "Автомобіль":

- марка;
- рік випуску;
- ціна;
- кольори.

Знищити всі елементи, у яких рік випуску менше заданого, додати елемент на початок файлу.

Розв'язок:

```

1 #include <bits/stdc++.h>
2
3 using namespace std;
4
5 struct Car {
6     char brand[50];
7     int year;
8     double price;
9     char color[30];
10 };
11
12 const char *filename = "cars.bin";
13
14 void createFile() {
15     FILE *file = fopen(filename, "wb");
16     if (!file) {
17         cerr << "Помилка відкриття файну для запису." << endl;
18         return;
19     }
20
21     int n;
22     cout << "Введіть кількість автомобілів: ";
23     cin >> n;
24
25     for (int i = 0; i < n; ++i) {
26         Car car;
27         cout << "Марка: ";
28         cin.ignore();
29         cin.getline(car.brand, 50);
30         cout << "Пік випуску: ";
31         cin >> car.year;
32         cout << "Ціна: ";
33         cin >> car.price;
34         cout << "Колір: ";
35         cin.ignore();
36         cin.getline(car.color, 30);
37
38         fwrite(&car, sizeof(Car), 1, file);
39     }
40
41     fclose(file);
42 }
43
44 void printFile() {
45     FILE *file = fopen(filename, "rb");
46     if (!file) {
47         cerr << "Помилка відкриття файну для читання." << endl;
48         return;
49     }
50
51     Car car;
52     bool isEmpty = true;
53
54     while (fread(&car, sizeof(Car), 1, file)) {
55         isEmpty = false;
56         cout << "Марка: " << car.brand
57             << ", Пік випуску: " << car.year
58             << ", Ціна: " << car.price
59             << ", Колір: " << car.color << endl;
60     }
61
62     if (isEmpty) {
63         cout << "Будь попомнір." << endl;
64     }
65
66     fclose(file);
67 }
68
69 void deleteOldCars(int minYear) {
70     FILE *file = fopen(filename, "rb");
71     if (!file) {
72         cerr << "Помилка відкриття файну для читання." << endl;
73         return;
74     }
75
76     vector<Car> cars;
77     Car car;
78
79     while (fread(&car, sizeof(Car), 1, file)) {
80         if (car.year >= minYear) {
81             cars.push_back(car);
82         }
83     }
84
85     fclose(file);
86
87     file = fopen(filename, "wb");
88     if (!file) {
89         cerr << "Помилка відкриття файну для запису." << endl;
90         return;
91     }
92
93     for (auto car: cars) {
94         fwrite(&car, sizeof(Car), 1, file);
95     }
96
97     fclose(file);
98 }
99
100 void addCarToStart() {
101     Car newCar;
102     cout << "Додавання нового автомобіля:" << endl;
103     cout << "Марка: ";
104     cin.ignore();
105     cin.getline(newCar.brand, 50);
106     cout << "Пік випуску: ";
107     cin >> newCar.year;
108     cout << "Ціна: ";
109     cin >> newCar.price;
110     cout << "Колір: ";
111     cin.ignore();
112     cin.getline(newCar.color, 30);
113
114     FILE *file = fopen(filename, "rb");
115     if (!file) {
116         cerr << "Помилка відкриття файну для читання." << endl;
117         return;
118     }
119
120     vector<Car> cars;
121     Car car;
122
123     while (fread(&car, sizeof(Car), 1, file)) {
124         cars.push_back(car);
125     }
126
127     fclose(file);
128
129     file = fopen(filename, "wb");
130     if (!file) {
131         cerr << "Помилка відкриття файну для запису." << endl;
132         return;
133     }
134
135     fwrite(&newCar, sizeof(Car), 1, file);
136
137     for (auto car: cars) {
138         fwrite(&car, sizeof(Car), 1, file);
139     }
140
141     fclose(file);
142 }
143
144
145 int main() {
146     createFile();
147
148     cout << endl << "Вісст файну після створення:" << endl;
149     printFile();
150
151     int minYear;
152     cout << endl << "Мінімальний пік випуску для збереження: ";
153     cin >> minYear;
154     deleteOldCars(minYear);
155
156     cout << endl << "Вісст файну після видалення старих автомобілів:" << endl;
157     printFile();
158
159     addCarToStart();
160
161     cout << endl << "Вісст файну після додавання нового автомобіля:" << endl;
162     printFile();
163
164     return 0;
165 }

```

Результат виконання:

```
Введіть кількість автомобілів: 3
Марка: AUDI
Рік випуску: 2010
Ціна: 12000
Колір: green
Марка: Dodge
Рік випуску: 2018
Ціна: 30000
Колір: red
Марка: Toyota
Рік випуску: 2015
Ціна: 20000
Колір: blue

Вміст файлу після створення:
Марка: AUDI, Рік випуску: 2010, Ціна: 12000, Колір: green
Марка: Dodge, Рік випуску: 2018, Ціна: 30000, Колір: red
Марка: Toyota, Рік випуску: 2015, Ціна: 20000, Колір: blue

Мінімальний рік випуску для збереження: 2012

Вміст файлу після видалення старих автомобілів:
Марка: Dodge, Рік випуску: 2018, Ціна: 30000, Колір: red
Марка: Toyota, Рік випуску: 2015, Ціна: 20000, Колір: blue
Додавання нового автомобіля:
Марка: Honda
Рік випуску: 2012
Ціна: 13000
Колір: black

Вміст файлу після додавання нового автомобіля:
Марка: Honda, Рік випуску: 2012, Ціна: 13000, Колір: black
Марка: Dodge, Рік випуску: 2018, Ціна: 30000, Колір: red
Марка: Toyota, Рік випуску: 2015, Ціна: 20000, Колір: blue
```

Task 5 - Lab# programming: VNS Lab 9

Час виконання ~ 3 год.

Розв'язок:

```

1  #include <bits/stdc++.h>
2  #include <fstream>
3
4  using namespace std;
5
6  void createFileF1(const string &filename) {
7      ofstream file(filename);
8      if (!file) {
9          cerr << "Помилка відкриття файлу F1 для запису." << endl;
10         return;
11     }
12
13     vector<string> lines = {
14         "Акація цвіте",
15         "Вітер з моря",
16         "Дерева у тіні",
17         "Аромат весни",
18         "Квітка розквітла",
19         "Сонце гріє землю",
20         "Зелена трава",
21         "Хвилі на морі",
22         "Пісок гарячий",
23         "Магнолія розквітла"
24     };
25
26     for (auto line : lines) {
27         file << line << endl;
28     }
29
30     file.close();
31     cout << "Файл F1 створено." << endl;
32 }
33
34 void copyLinesToF2(const string &fileF1, const string &fileF2, int N1, int N2) {
35     ifstream file1(fileF1);
36     ofstream file2(fileF2);
37
38     if (!file1) {
39         cerr << "Помилка відкриття файлу F1 для читання." << endl;
40         return;
41     }
42     if (!file2) {
43         cerr << "Помилка відкриття файлу F2 для запису." << endl;
44         return;
45     }
46
47     string line;
48     int lineNumber = 0;
49
50     while (getline(file1, line)) {
51         lineNumber++;
52         if (lineNumber >= N1 && lineNumber <= N2 && line.back() == 'A') {
53             file2 << line << endl;
54         }
55     }
56
57     file1.close();
58     file2.close();
59     cout << "Рядки скопійовано у файл F2." << endl;
60 }
61
62 int findLineWithMaxA(const string &fileF2) {
63     ifstream file(fileF2);
64
65     if (!file) {
66         cerr << "Помилка відкриття файлу F2 для читання." << endl;
67         return -1;
68     }
69
70     string line;
71     int lineNumber = 0, maxLineNumber = -1, maxACount = 0;
72
73     while (getline(file, line)) {
74         lineNumber++;
75         int aCount = 0;
76         for (char ch : line) {
77             if (ch == 'A'){
78                 aCount++;
79             }
80         }
81         if (aCount > maxACount) {
82             maxACount = aCount;
83             maxLineNumber = lineNumber;
84         }
85     }
86
87     file.close();
88     return maxLineNumber;
89 }
90
91 int main() {
92     const string fileF1 = "F1.txt";
93     const string fileF2 = "F2.txt";
94
95     createFileF1(fileF1);
96
97     int N1, N2;
98     cout << "Введіть номер рядка N1, N2: ";
99     cin >> N1>>N2;
100
101     copyLinesToF2(fileF1, fileF2, N1, N2);
102
103
104     int maxLine = findLineWithMaxA(fileF2);
105     if (maxLine != -1) {
106         cout << "Номер рядка з найбільшою кількістю 'A' у файлі F2: " << maxLine << endl;
107     } else {
108         cout << "Файл F2 порожній або не знайдено рядків." << endl;
109     }
110
111     return 0;
112 }

```


Результат виконання:

```
Файл F1 створено.  
Введіть номер рядка N1, N2: 2 8  
Рядки скопійовано у файл F2.  
Номер рядка з найбільшою кількістю 'A' у файлі F2: 2
```

Task 6 - Lab# programming: Algotester [Lab 4.2](#)

Час виконання ~ 1 год.

Розв'язок №1:

```
1  #include<bits/stdc++.h>  
2  //4.2  
3  using namespace std;  
4  
5  int main(){  
6      int n,k;  
7      cin>>n>>k;  
8      vector<int>a(n);  
9      for(int i=0;i<n;i++){  
10         cin>>a[i];  
11     }  
12     sort(a.begin(), a.end());  
13     auto it= unique(a.begin(), a.end());  
14     a.erase(it, a.end());  
15  
16     n=a.size();  
17     k%=n;  
18     cout<<n<<endl;  
19     rotate(a.begin(),a.begin()+k, a.end());  
20     for(auto i:a){  
21         cout<<i<<" ";  
22     }  
23  
24 }
```

Розв'язок №2:

```

1  #include<bits/stdc++.h>
2  //4.2
3  using namespace std;
4
5  int main(){
6      int n,k;
7      cin>>n>>k;
8      vector<int>a(n),b,c;
9      for(int i=0;i<n;i++){
10         cin>>a[i];
11     }
12     sort(a.begin(), a.end());
13     b.push_back(a[0]);
14     for(int i=1;i<n;i++){
15         if(a[i]!=a[i-1]) b.push_back(a[i]);
16     }
17     n=b.size();
18     for(int i=0;i<n;i++){
19         c.push_back(b[ (i+k)%n ]);
20     }
21     cout<<n<<endl;
22     for(auto i:c)
23         cout<<i<<" ";
24 }

```

Результат виконання:

```

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

```

Algotester [Lab 4.3](#)

Час виконання ~ 2 год.

Розв'язок №1:

```

1  #include<bits/stdc++.h>
2  //4.3
3  using namespace std;
4
5  int main(){
6      int n;
7      cin>>n;
8      vector<int>a(n);
9      for(int i=0;i<n;i++){
10         cin>>a[i];
11     }
12     sort(a.begin(),a.end());
13     auto it= unique(a.begin(),a.end());
14     a.erase(it,a.end());
15
16     vector<int> mod0, mod1, mod2;
17     for (int x : a) {
18         if (x % 3 == 0) mod0.push_back(x);
19         else if (x % 3 == 1) mod1.push_back(x);
20         else mod2.push_back(x);
21     }
22
23     sort(mod0.begin(), mod0.end());
24     sort(mod1.rbegin(), mod1.rend()); //по спаданню
25     sort(mod2.begin(), mod2.end());
26
27     a.clear();
28     a.insert(a.end(), mod0.begin(), mod0.end());
29     a.insert(a.end(), mod1.begin(), mod1.end());
30     a.insert(a.end(), mod2.begin(), mod2.end());
31
32     cout<<a.size()<<endl;
33     for(auto i:a) cout<<i<<" ";
34 }

```

Розв'язок №2

```

1  #include <iostream>
2  #include <vector>
3  using namespace std;
4
5  void SplitByModul(const vector<int>& arr, vector<int>& mod0, vector<int>& mod1, vector<int>& mod2) {
6      for (int num : arr) {
7          if (num % 3 == 0) mod0.push_back(num);
8          else if (num % 3 == 1) mod1.push_back(num);
9          else mod2.push_back(num);
10     }
11 }
12
13 void quickSort(vector<int>& arr, int left, int right, bool ascending) {
14     if (left >= right) return;
15
16     int mid = arr[(left + right) / 2];
17     int i = left, j = right;
18
19     while (i <= j) {
20         while (ascending ? arr[i] < mid : arr[i] > mid) i++;
21         while (ascending ? arr[j] > mid : arr[j] < mid) j--;
22
23         if (i <= j) {
24             swap(arr[i], arr[j]);
25             i++;
26             j--;
27         }
28     }
29
30     if (left < j) quickSort(arr, left, j, ascending);
31     if (i < right) quickSort(arr, i, right, ascending);
32 }
33
34 vector<int> removeDuplicates(const vector<int>& arr) {
35     vector<int> result;
36     for (int num : arr) {
37         if (result.empty() || result.back() != num) {
38             result.push_back(num);
39         }
40     }
41     return result;
42 }
43
44 int main() {
45     int N;
46     cin >> N;
47     vector<int> arr(N);
48
49     for (int i = 0; i < N; i++) {
50         cin >> arr[i];
51     }
52     vector<int> mod0, mod1, mod2;
53     SplitByModul(arr, mod0, mod1, mod2);
54
55     quickSort(mod0, 0, mod0.size() - 1, true); // за зростанням
56     quickSort(mod1, 0, mod1.size() - 1, false); // за спаданням
57     quickSort(mod2, 0, mod2.size() - 1, true); // за зростанням
58
59
60     vector<int> result;
61     result.insert(result.end(), mod0.begin(), mod0.end());
62     result.insert(result.end(), mod1.begin(), mod1.end());
63     result.insert(result.end(), mod2.begin(), mod2.end());
64     result = removeDuplicates(result);
65
66     cout << result.size() << endl;
67     for (int num : result) {
68         cout << num << " ";
69     }
70     cout << endl;
71 }
72

```

Результат виконання:

```
10
1 33 4 8 6 5 2 7 5 0
9
0 6 33 7 4 1 2 5 8
```

Task 7 - Lab# programming: Algotester [Lab 6.2](#)

Час виконання ~ 2 год.

Розв'язок

```

1 #include<bits/stdc++.h>
2
3 using namespace std;
4 vector<string> board(8);
5
6 bool could_king(int x, int y) {
7     if (((0 <= x) && (x <= 7) && (0 <= y) && (y <= 7)) && (board[x][y] == 'K'))
8         return true;
9     else
10         return false;
11 }
12
13 bool king(int x, int y) {
14     vector<pair<int, int>> offset = {{1, 0},
15                                     {1, 1},
16                                     {1, -1},
17                                     {-1, 0},
18                                     {-1, 1},
19                                     {-1, -1},
20                                     {0, 1},
21                                     {0, -1}};
22     bool flag = false;
23     for (int i = 0; i < 8; i++)
24         if (could_king(x + offset[i].first, y + offset[i].second))
25             flag = true;
26     return flag;
27 }
28
29 bool tyra(int x, int y) {
30     bool flag = false;
31     for (int i = 0; i < 7; i++)
32         if ((board[x][i] == 'R') || (board[i][y] == 'R'))
33             flag = true;
34     return flag;
35 }
36
37 bool could_slon(int x, int y) {
38     if (((0 <= x) && (x <= 7) && (0 <= y) && (y <= 7)) && (board[x][y] == 'B'))
39         return true;
40     else
41         return false;
42 }
43
44 bool slon(int x, int y) {
45     bool flag = false;
46     vector<pair<int, int>> offset = {{1, 1},
47                                     {1, -1},
48                                     {-1, 1},
49                                     {-1, -1}};
50     for (int i = -7; i < 8; i++) {
51         for (int j = 0; j < 4; j++) {
52             if (could_slon(x + i * offset[j].first, y + i * offset[j].second))
53                 flag = true;
54         }
55     }
56     return flag;
57 }
58
59 bool could_horse(int x, int y) {
60     if (((0 <= x) && (x <= 7) && (0 <= y) && (y <= 7)) && (board[x][y] == 'N'))
61         return true;
62     else
63         return false;
64 }
65
66 bool horse(int x, int y) {
67     bool flag = false;
68     vector<pair<int, int>> offset = {{2, 1},
69                                     {2, -1},
70                                     {-2, 1},
71                                     {-2, -1},
72                                     {1, 2},
73                                     {1, -2},
74                                     {-1, 2},
75                                     {-1, -2}};
76     for (int i = 0; i < 8; i++) {
77         if (could_horse(x + offset[i].first, y + offset[i].second))
78             flag = true;
79     }
80     return (flag);
81 }
82
83
84 bool pishak(int x, int y) {
85     if (((0 <= x - 1) && (0 <= y - 1) && (board[x - 1][y - 1] == 'P')) ||
86         ((0 <= x - 1) && (y + 1 <= 7) && (board[x - 1][y + 1] == 'P')))
87         return true;
88     else
89         return false;
90 }
91
92 string find_result(int x, int y) {
93     string result = "";
94     if (slon(x, y)) result += "B";
95     if (king(x, y)) result += "K";
96     if (horse(x, y)) result += "N";
97     if (pishak(x, y)) result += "P";
98     if (slon(x, y) || tyra(x, y)) result += "Q";
99     if (tyra(x, y)) result += "R";
100     return result;
101 }
102
103 int main() {
104     for (int i = 0; i < 8; i++) {
105         cin >> board[i];
106     }
107     cin;
108     int q, x, y;
109     cin >> q;
110     for (int i = 0; i < q; i++) {
111         cin >> x >> y;
112         if (board[x-1][y-1] != '0')
113             cout<<board[x-1][y-1]<<endl;
114         string rez=find_result(x-1,y-1);
115         if(rez=="")
116             cout<<"0"<<endl;
117         else
118             cout<<rez<<endl;
119     }
120 }

```

Результат виконання:

```
00000000
0R000000
00N00000
0000P000
00000000
00000000
K0Q00000
0000000R
```

```
7
8 1
K
1 2
KQR
5 4
KP
5 1
O
6 2
KQR
8 4
O
6 7
O
```

Task 8 - Practice# programming: Class Practice Task

Час виконання~ 2 год.

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

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) Реалізувати функцію створення файла і запису в нього даних:

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.

Розв'язок:

```
1  #include<bits/stdc++.h>
2  #include<fstream>
3
4  using namespace std;
5
6  void write_to_file(char *file_name, char *content) {
7      ofstream file(file_name);
8      if (!file) {
9          cout << "Failure" << endl;
10         return;
11     }
12     file << content << endl;
13     file.close();
14     cout << "Success" << endl;
15 }
16
17
18 void copy_file(char *file_from_copy, char *file_to_insert) {
19     ofstream file_to(file_to_insert);
20     ifstream file_from(file_from_copy);
21     if ((!file_to) || (!file_from)) {
22         cout << "Failure" << endl;
23         return;
24     }
25     bool is_empty_file_from = true;
26
27     string line;
28     while (getline(file_from, line)) {
29         file_to << line << endl;
30
31         is_empty_file_from = false;
32     }
33     if (is_empty_file_from)
34         cout << "Failure file_from is empty" << endl;
35     else {
36         cout << "Success" << endl;
37     }
38 }
39
40 }
41
42 int main() {
43     char content[50];
44     char file_name[20];
45     cout << "Filename: ";
46     cin.getline(file_name, sizeof(file_name));
47     cout << "Content: ";
48     cin.getline(content, sizeof(content));
49
50     write_to_file(file_name, content);
51
52
53     char filename_from[20], filename_to[20];
54     cout << "Filename from which copy: ";
55     cin.getline(filename_from, sizeof(filename_from));
56     cout << "Filename in which insert: ";
57     cin.getline(filename_to, sizeof(filename_to));
58     copy_file(filename_from, filename_to);
59 }
```


Task 9 - Practice# programming: Self Practice Task

Час виконання 1,5 год

Algotester 2133 [Катеринка](#)

В цій задачі потрібно знайти мінімальну відстань між **q** парами вершин у зваженому графі, так щоб шлях проходив через задану третю вершину.

Розв'язок:

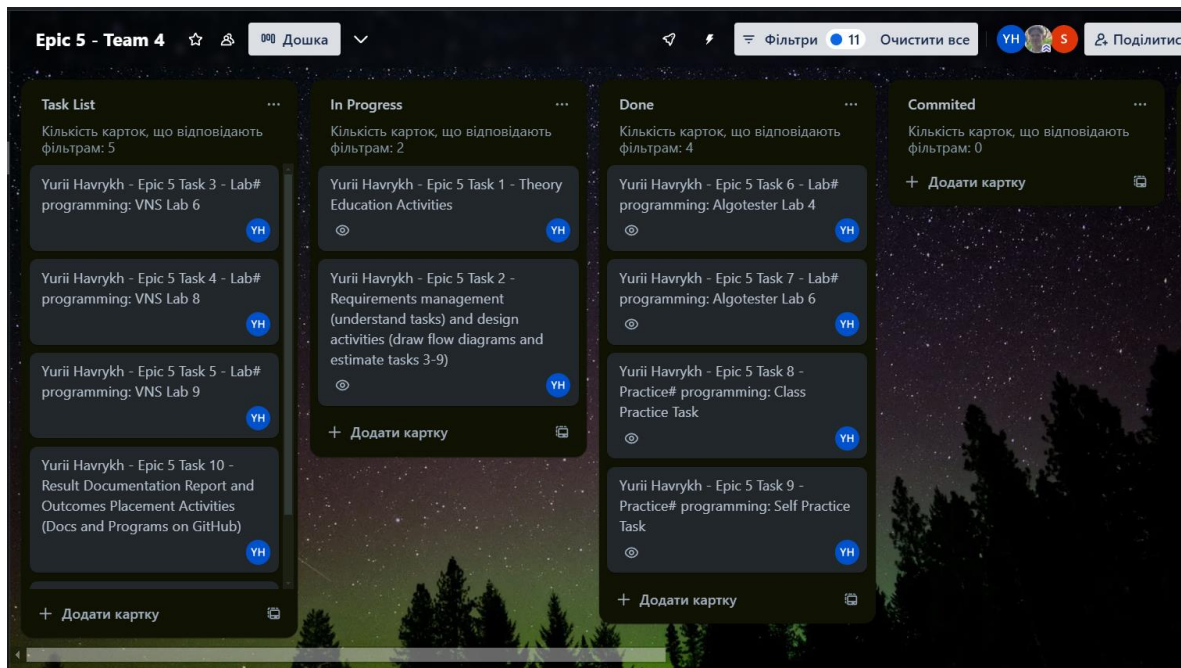
```
1  #include<bits/stdc++.h>
2
3  using namespace std;
4  const int inf = 2e8;
5  //2133 Катеринка
6  void dijkstra(set<pair<int, int>> &que, const vector<vector<pair<int, int>>> &graph, vector<int> &dist) {
7      while (!que.empty()) {
8
9          int cur = que.begin()->second;
10         que.erase(que.begin());
11         for (auto curent: graph[cur]) {
12             if (dist[curent.first] > dist[cur] + curent.second) {
13                 que.erase({dist[curent.first], curent.first});
14                 dist[curent.first] = dist[cur] + curent.second;
15                 que.insert({dist[curent.first], curent.first});
16             }
17         }
18     }
19 }
20
21 int main() {
22     int n, m, q, kat, from, to, d;
23     cin >> n >> m >> q >> kat;
24     vector<int> dist(n, inf);
25     vector<vector<pair<int, int>>> graph(n);
26     set<pair<int, int>> que;
27     vector<pair<int, int>> find(q);
28     for (int i = 0; i < m; i++) {
29         cin >> from >> to >> d;
30         graph[from - 1].push_back({to - 1, d});
31         graph[to - 1].push_back({from - 1, d});
32     }
33     for (int i = 0; i < q; i++) {
34         cin >> find[i].first >> find[i].second;
35     }
36     kat--;
37     que.insert({0, kat});
38     dist[kat] = 0;
39     dijkstra(que, graph, dist);
40     for (int i = 0; i < q; i++) {
41         cout << dist[find[i].first - 1] + dist[find[i].second - 1] << endl;
42     }
43 }
```

Результат виконання:

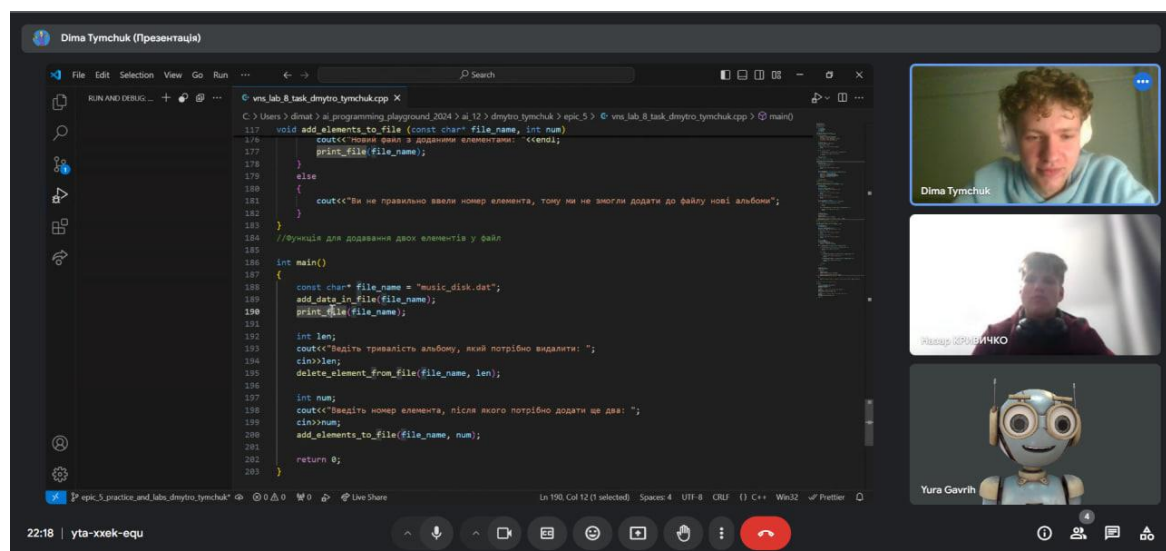
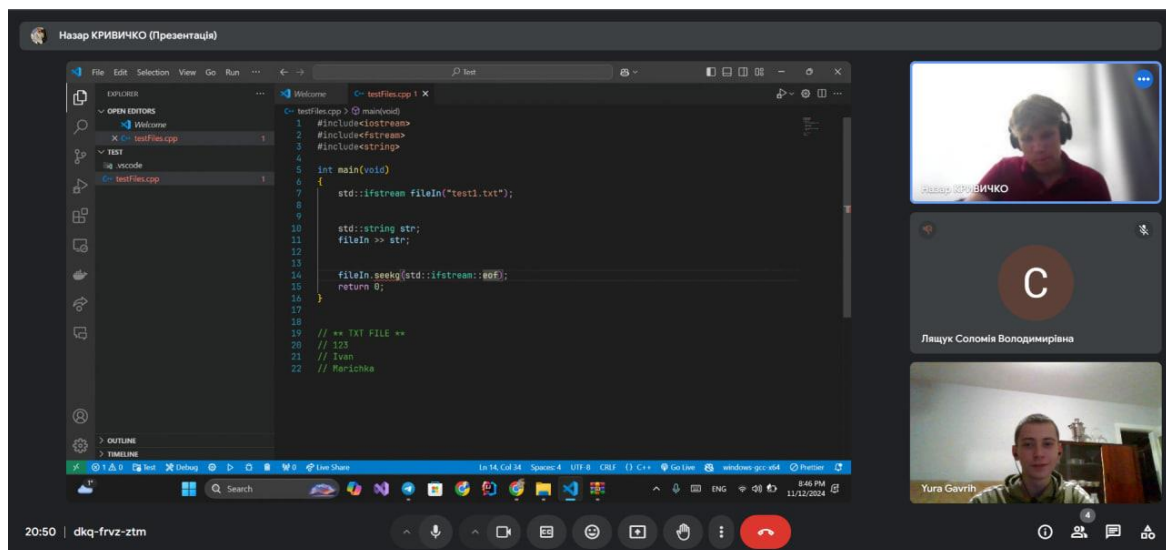
```
4 5 3 1
1 2 1
2 3 3
3 1 1
3 4 4
1 4 10
4 3
2 3
1 4
6
2
5
```

Task 10 - Result Documentation Report and Outcomes Placement Activities

Час виконання ~ 1,5 год.



Team meetings



Pull Request

Висновок:

В результаті виконання цієї роботи я навчився працювати з рядками типу `char` та `string`, закріпив свої знання роботи з файлами на `C` та `C++`.