

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 ІІІ-12

Стефанович Євгеній

**Тема роботи:** Файлова система в C++. Робота з бінарними файлами та текстовими файлами, маніпуляції символами й рядковими змінними, як типу `std::string`, так і `char*`. Ознайомлення з можливостями стандартної бібліотеки C++ для роботи з файлами та створенням власних бібліотек для розширення функціональності.

**Мета роботи:** Опанувати практичні навички роботи з файлами в мові C++: створення, зчитування та запис даних у бінарні й текстові файли. Засвоїти принципи роботи з рядковими змінними різних типів (`std::string` і `char*`), вивчити використання стандартних методів та функцій для маніпуляцій з ними. Дослідити основи створення та застосування власних бібліотек для зручності повторного використання коду й розширення можливостей стандартної бібліотеки C++.

**Джерела:**

CS50 course

University lectures

Google + chatGPT: string functions and memory allocation

**Виконання роботи:**

**Lab# programming: VNS Lab 6**

**Time expected: 30 min**

**Time spent: 30 min**

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.

## 16. Визначити які слова зустрічаються в рядку по одному разі.

```
C: > Users > Eugene > Desktop > epic_5 > vns_lab_6_task_eugenie_stefanovich.cpp > main()
1  #include <iostream>
2  #include <vector>
3  #include <cstring>
4  #include <unordered_map>
5
6  using namespace std;
7
8  int main() {
9      cout << "Введіть строку: ";
10     char base[256];
11     gets(base);
12
13
14     vector<string> slova;
15     vector<string> slova2;
16
17
18     char* word = strtok(base, " ");
19     while (word != nullptr) {
20         slova.push_back(string(word));
21         word = strtok(nullptr, " ");
22     }
23
24     unordered_map<string, int> sort;
25
26
27     for (const auto& word : slova) {
28         sort[word]++;
29     }
30
31
32     cout << "Слова, що зустрічаються лише один раз:" << endl;
33     for (auto final : sort) {
34         if (final.second == 1) {
35             cout << final.first << endl;
36         }
37     }
38     return 0;
39 }
40
```

```
hello
PS C:\Users\Eugene\Desktop\epic_5\output> cd 'c:\Users\Eugene\Desktop\epic_5\output'
PS C:\Users\Eugene\Desktop\epic_5\output> & .\vns_lab_6_task1.exe
Введіть строку: Hello i Vova Vova
Слова, що зустрічаються лише один раз:
i
Hello
PS C:\Users\Eugene\Desktop\epic_5\output> █
```

## Lab# programming: VNS Lab 8

**Time expected: 1 h**

**Time spent: 1.5 h – 2 h**

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.

16. Структура "Власник автомобіля":

- прізвище, ім'я, по батькові;
- номер автомобіля;
- телефон;
- номер техпаспорта.

Знищити елемент із заданим номером, додати 2 елементи перед елементом із заданим прізвищем.

```

> Users > Eugene > Desktop > epic_5 > G+ vns_lab_8_task_eugenie_stefanovich.cpp > printFile(const char *)
1  #include <iostream>
2  #include <cstring>
3  #include <cstdio>
4
5  using namespace std;
6
7  // Определяем структуру "Власник автомобіля"
8  struct CarOwner {
9      char priz[50]; // Прізвище
10     char name[50]; // Ім'я
11     char batok[50]; // По батькові
12     char carN[15]; // Номер автомобіля
13     char phone[15]; // Телефон
14     char techP[15];
15 };
16
17 // Функції - Створити
18 void createFile(const char* filename) {
19     FILE* f = fopen(filename, "wb");
20     if (f == nullptr) {
21         cerr << "Ошибка открытия файла для записи." << endl;
22         exit(0);
23     }
24
25     CarOwner owners[] = {
26         {"Грушевський", "Михайло", "Сергійович", "A1", "1", "TP1"},
27         {"Zuckerberg", "Mark", "Elliot", "B1", "2", "TP2"},
28         {"Стефанович", "Євгеній", "Володимирович", "B2", "0938256177", "TP3"}
29     };
30
31     for (const auto& owner : owners) {
32         fwrite(&owner, sizeof(CarOwner), 1, f);
33     }
34
35     fclose(f);
36     cout << "Файл успешно создан с начальными данными." << endl;
37 }
38
39 // Зчитати
40 void printFile(const char* filename) {
41     FILE* f = fopen(filename, "rb");
42     if (f == nullptr) {
43         cerr << "Ошибка открытия файла для чтения." << endl;
44         exit(0);
45     }
46
47     CarOwner owner;
48     cout << "Содержимое файла:" << endl;
49     cout << "-----" << endl;
50     while (fread(&owner, sizeof(CarOwner), 1, f) == 1) {
51         cout << "Прізвище: " << owner.priz << ", Ім'я: " << owner.name
52             << ", По батькові: " << owner.batok << endl;
53         cout << "Номер авто: " << owner.carN << ", Телефон: " << owner.phone
54             << ", Номер техпаспорта: " << owner.techP << endl;
55         cout << "-----" << endl;
56     }
57     fclose(f);
58 }
59
60 //видалити

```

```

> Users > Eugene > Desktop > epic_5 > vns_lab_8_task_eugenie_stefanovich.cpp > printFile(const char *)
39 void printFile(const char* filename) {
57     fclose(f);
58 }
59
60 //видалити
61 void deleteByCarNumber(const char* filename, const char* carNumber) {
62     FILE* f = fopen(filename, "rb");
63     FILE* bezN = fopen("bezN.dat", "wb");
64     if (f == nullptr || bezN == nullptr) {
65         cerr << "Ошибка открытия файла для удаления." << endl;
66         exit(0);
67     }
68
69     CarOwner owner;
70     bool found = false;
71
72     while (fread(&owner, sizeof(CarOwner), 1, f) == 1) {
73         if (strcmp(owner.carN, carNumber) != 0) {
74             fwrite(&owner, sizeof(CarOwner), 1, bezN);
75         } else {
76             found = true;
77         }
78     }
79
80     fclose(f);
81     fclose(bezN);
82
83     remove(filename);
84     rename("bezN.dat", filename);
85
86     if (found) {
87         cout << "Владелец с номером авто " << carNumber << " успешно удалён." << endl;
88     } else {
89         cout << "Владелец с номером авто " << carNumber << " не найден." << endl;
90     }
91 }
92
93 //додати
94 void addBeforeSurname(const char* filename, const CarOwner& owner1, const CarOwner& owner2,
95     FILE* f = fopen(filename, "rb");
96     FILE* dot = fopen("dot.dat", "wb");
97     if (f == nullptr || dot == nullptr) {
98         cerr << "Ошибка открытия файла для добавления." << endl;
99         exit(0);
100     }
101
102     CarOwner owner;
103     bool dodav = false;
104
105     while (fread(&owner, sizeof(CarOwner), 1, f) == 1) {
106         if (strcmp(owner.batok, surname) == 0 && !dodav) {
107             fwrite(&owner1, sizeof(CarOwner), 1, dot);
108             fwrite(&owner2, sizeof(CarOwner), 1, dot);
109             dodav = true;
110         }
111         fwrite(&owner, sizeof(CarOwner), 1, dot);
112     }
113
114     fclose(f);
115     fclose(dot);

```

```

C:\> Users > Eugene > Desktop > epic_5 > vns_lab_8_task_eugenie_stefanovich.cpp > printFile(const char *)
94 void addBeforeSurname(const char* filename, const CarOwner& owner1, const CarOwner& owner2, const char* surname) {
97     if (f == nullptr || dot == nullptr) {
99         exit(0);
100     }
101
102     CarOwner owner;
103     bool dodav = false;
104
105     while (fread(&owner, sizeof(CarOwner), 1, f) == 1) {
106         if (strcmp(owner.batok, surname) == 0 && !dodav) {
107             fwrite(&owner1, sizeof(CarOwner), 1, dot);
108             fwrite(&owner2, sizeof(CarOwner), 1, dot);
109             dodav = true;
110         }
111         fwrite(&owner, sizeof(CarOwner), 1, dot);
112     }
113
114     fclose(f);
115     fclose(dot);
116
117     remove(filename);
118     rename("dot.dat", filename);
119
120     if (dodav) {
121         cout << "Два владельца добавлены перед владельцем с призвищем " << surname << "." << endl;
122     } else {
123         cout << "Владелец с призвищем " << surname << " не найден. Новые владельцы добавлены в конец." << endl;
124     }
125 }
126
127 int main() {
128     const char* filename = "carowners.dat";
129
130     // Создаём файл и добавляем начальные данные
131     createFile(filename);
132
133     // Печать содержимого файла
134     printFile(filename);
135
136     // Удаляем владельца по номеру автомобиля
137     deleteByCarNumber(filename, "B2");
138     printFile(filename);
139
140     // Добавляем двух владельцев перед владельцем с заданным призвищем
141     CarOwner owner1 = {"Байден", "Джозеф", "Робинетт", "AB1", "123", "TP12"};
142     CarOwner owner2 = {"Иванов", "Иван", "Иванович", "AB2", "456", "TP21"};
143     addBeforeSurname(filename, owner1, owner2, "Elliot");
144     printFile(filename);
145
146     return 0;
147 }
148
149
150

```

## **Lab# programming: VNS Lab 9**

**Time expected: 1.5 h**

**Time spent: 2.5 h**

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

Виконати завдання.

16.

- 1) Скопіювати з файлу F1 у файл F2 всі рядки, що починаються на букву «А» і закінчуються на букву «З», розташовані між рядками з номерами N1 й N2.
- 2) Визначити кількість слів у першому рядку файлу F2.



```

C:\Users\Eugene\Desktop> cd /d epic_5 & g++ vns_lab_9_task_eugenie_stefanovich.cpp & main()
1  #include <iostream>
2  #include <vector>
3  #include <string>
4  #include <set>
5  #include <algorithm>
6  #include <fstream>
7  #include <sstream>
8  using namespace std;
9
10 int main() {
11     string F1 = "F1.txt";
12     string F2 = "F2.txt";
13
14
15     ofstream fileF1(F1);
16     fileF1 << "AB CD F3\n";
17     fileF1 << "123\n";
18     fileF1 << "A1 B2 C3\n";
19     fileF1 << "Hello\n";
20     fileF1 << "American \n";
21     fileF1 << "Ananas\n";
22     fileF1 << "pig 3\n";
23     fileF1 << "Albatros 3\n";
24     fileF1 << "Megamo3g\n";
25     fileF1 << "Diamant\n";
26     fileF1.close();
27
28
29     ifstream rF1(F1);
30     if (!rF1.is_open()){
31         cerr << "Не удалось открыть файл F1!" << endl;
32         exit(0);
33     }
34
35     int N1, N2;
36     cout << "Введіть номери рядків N1 та N2 (від 1 до 10): ";
37     cin >> N1 >> N2;
38
39
40
41
42     ofstream fileF2(F2);
43     if (!fileF2.is_open()) {
44         cerr << "Не вдалося відкрити файл F2" << endl;
45         exit(0);
46     }
47
48
49     string line;
50     int lineNumber = 0;
51     while (getline(rF1, line)) {
52         lineNumber++;
53     }

```

```

1 #include <iostream>
2 #include <string>
3
4 using namespace std;
5
6 void createAndWriteToFile(const char* filename);
7 void printFile(const char* filename);
8 bool hasDuplicateWords(const char* line);
9 void createNewFileWithNoDuplicates(const char* fromFileName, const char* toFileName);
10 int countVowelsInFirstLine(const char* filename);
11
12 int main() {
13     const char *filename1 = "F1.txt";
14     const char *filename2 = "F2.txt";
15     createAndWriteToFile(filename1);
16     printFile(filename1);
17     createNewFileWithNoDuplicates(filename1, filename2);
18     printFile(filename2);
19     cout << endl << "Number of vowels in the first line of F2: " << countVowelsInFirstLine(filename2) << endl;
20 }
21
22 void createAndWriteToFile(const char* filename) {
23     FILE *file = fopen(filename, "w");
24     if (file == NULL) {
25         cerr << "Не удалось создать файл для записи" << endl;
26         exit(1);
27     }
28
29     const char* lines[] = {
30         "line 1\n",
31         "line 2: line\n",
32         "line 3\n",
33         "line 4: line line\n",
34         "line 5\n",
35         "line 6\n",
36         "line 7: line\n",
37         "line 8\n",
38         "line 9: line line\n",
39         "line 10\n"
40     };
41
42     for (int i = 0; i < 10; i++) {
43         fputs(lines[i], file);
44     }
45
46     cout << "Завел новое название у файла " << filename << endl;
47     fclose(file);
48 }
49
50 void printFile(const char* filename) {
51     FILE *f = fopen(filename, "r");
52     if (f == NULL) {
53         cerr << "Не удалось прочитать файл для чтения" << endl;
54         exit(2);
55     }
56
57     char line[256];
58
59     cout << "Вывел файл " << filename << "\n";
60     cout << "-----\n";
61
62     while (fgets(line, sizeof(line), f)) {
63         cout << line;
64     }
65
66     fclose(f);
67 }
68
69 bool hasDuplicateWords(const char* line) {
70     char tapline[256];
71     strcpy(tapline, line, sizeof(tapline) - 1);
72     tapline[sizeof(tapline) - 1] = '\0';
73
74     const char* words[50];
75     int wordCount = 0;
76
77     char* word = strtok(tapline, " \n");
78     while (word != nullptr) {
79         for (int i = 0; i < wordCount; ++i) {
80             if (strcmp(words[i], word) == 0) {
81                 return true;
82             }
83         }
84
85         words[wordCount++] = word;
86         word = strtok(nullptr, " \n");
87     }
88     return false;
89 }
90
91 void createNewFileWithNoDuplicates(const char* fromFileName, const char* toFileName) {
92     FILE *from = fopen(fromFileName, "r");
93     FILE *to = fopen(toFileName, "w");
94     if (from == NULL || to == NULL) {
95         cerr << "Не удалось прочитать файл для чтения" << endl;
96         exit(3);
97     }
98
99     char line[256];
100
101     while (fgets(line, sizeof(line), from)) {
102         if (!hasDuplicateWords(line)) {
103             fputs(line, to);
104         }
105     }
106
107     fclose(from);
108     fclose(to);
109
110     cout << "Lines without repeated words copied successfully." << endl;
111 }
112
113 int countVowelsInFirstLine(const char* filename) {
114     FILE* f = fopen(filename, "r");
115
116     if (f == NULL) {
117         cerr << "Не удалось прочитать файл для чтения" << endl;
118         exit(4);
119     }
120
121     char line[256];
122     if (fgets(line, sizeof(line), f) == NULL) {
123         fclose(f);
124         return 0; // No lines in file
125     }
126
127     int counter = 0;
128
129     for (int i = 0; line[i] != '\0'; i++) {
130         char ch = tolower(line[i]);
131         if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
132             counter++;
133         }
134     }
135
136     return counter;
137 }
138
139 }

```

```

48
49     string line;
50     int lineNumber = 0;
51     while (getline(rF1, line)) {
52         lineNumber++;
53
54
55         if (lineNumber < N1 || lineNumber > N2) continue;
56
57
58         if (!line.empty() && line[0] == 'A' && line.back() == '3') {
59             fileF2 << line << endl;
60             cout << line << endl;
61         }
62     }
63     rF1.close();
64     fileF2.close();
65
66
67     ifstream rF2(F2);
68     if (!rF2.is_open()) {
69         cerr << "Не вдалося відкрити файл F2 " << endl;
70         exit(0);
71     }
72
73
74     if (getline(rF2, line)) {
75         int slovstrok = 0;
76         stringstream slov(line);
77         string word;
78         while (slov >> word) {
79             slovstrok++;
80         }
81
82         cout << "Кількість слів у першому рядку файлу F2: " << slovstrok << endl;
83     } else {
84         cout << "Файл F2 пуст." << endl;
85     }
86
87     rF2.close();
88     return 0;
89
90

```

```

PS C:\Users\Eugene\Desktop\epic_5\output> cd 'c:\Users\Eugene\Desktop\epic_5\output'
PS C:\Users\Eugene\Desktop\epic_5\output> & .\vns_lab_9_task_eugenie_stefanovich.exe'
Введіть номери рядків N1 та N2 (від 1 до 10): 1 5
AB CD F3
A1 B2 C3
Кількість слів у першому рядку файлу F2: 3
PS C:\Users\Eugene\Desktop\epic_5\output> █

```

# Lab# programming: Algotester Lab 4

## 4.3

### Lab 4v3

*Limits: 2 sec., 256 MiB*

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 видалити усі дублікати з масиву.

Виведіть результуючий масив.

### Input

У першому рядку  $N$  - кількість чисел.

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

### Output

У першому рядку  $M$  - кількість чисел у масиву

У другому рядку  $M$  посортованих за умовою чисел.

### Constraints

$$1 \leq N \leq 10^3$$

$$0 \leq a_i \leq 10^3$$

**Time expected: 30 min**

**Time spent: 20 min**

C:\Users\Eugene\Desktop> epic\_5 > C++ algotester\_lab\_4\_task\_2\_eugenie\_stefanovich.cpp > main()

```
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4
5  using namespace std;
6
7  int main() {
8      int N, wid;
9      vector<int> base0, base1, base2;
10     cin >> N;
11
12     for (int i = 0; i < N; i++) {
13         cin >> wid;
14         if (wid % 3 == 0)
15             base0.push_back(wid);
16         else if (wid % 3 == 1 || wid % 3 == -1)
17             base1.push_back(wid);
18         else if (wid % 3 == 2 || wid % 3 == -2)
19             base2.push_back(wid);
20     }
21
22     sort(base0.begin(), base0.end());
23     sort(base1.begin(), base1.end(), greater<int>());
24     sort(base2.begin(), base2.end());
25
26
27     base0.erase(unique(base0.begin(), base0.end()), base0.end());
28
29     base1.erase(unique(base1.begin(), base1.end()), base1.end());
30
31     base2.erase(unique(base2.begin(), base2.end()), base2.end());
32
33     cout << base0.size() + base1.size() + base2.size() << endl;
34
35     for (int num : base0) {
36         cout << num << " ";
37     }
38     for (int num : base1) {
39         cout << num << " ";
40     }
41     for (int num : base2) {
42         cout << num << " ";
43     }
44
45     return 0;
46 }
```

## 4.3

### Notes

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.

---

**Time expected: 45 min**

**Time spent: 1 h**

C:\> Users > Eugene > Desktop > epic\_5 > G+ algotester\_lab\_4\_task\_1\_eugenie\_stefanovich.cpp > main()

```
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4
5  using namespace std;
6
7  int main() {
8      int N, wid;
9      vector<int> base0;
10     vector<int> base1;
11     vector<int> base2;
12     cin >> N;
13
14     for(int i = 0; i < N; i++){
15         cin >> wid;
16         if(wid%3 == 0)
17             base0.push_back(wid);
18         else if(wid%3 == 1 || wid%3 == -1)
19             base1.push_back(wid);
20         else if(wid%3 == 2 || wid%3 == -2)
21             base2.push_back(wid);
22     }
23
24     sort(base0.begin(), base0.end());
25     sort(base1.begin(), base1.end());
26     reverse(base1.begin(), base1.end());
27     sort(base2.begin(), base2.end());
28
29     if (base0.size() > 1) {
30         for (int i = 0; i < base0.size() - 1; ) {
31             if (base0[i] == base0[i + 1]) {
32                 int k = 1;
33                 while (i + k < base0.size() && base0[i] == base0[i + k]) {
34                     k++;
35                 }
36                 base0.erase(base0.begin() + i + 1, base0.begin() + i + k);
37             } else {
38                 i++;
39             }
40         }
41     }
42
43     if (base1.size() > 1) {
44         for (int i = 0; i < base1.size() - 1; ) {
45             if (base1[i] == base1[i + 1]) {
46                 int k = 1;
47                 while (i + k < base1.size() && base1[i] == base1[i + k]) {
48                     k++;
49                 }
50                 base1.erase(base1.begin() + i + 1, base1.begin() + i + k);
51             } else {
52                 i++;
53             }
54         }
55     }
56
57     if (base2.size() > 1) {
58         for (int i = 0; i < base2.size() - 1; ) {
59             if (base2[i] == base2[i + 1]) {
60                 int k = 1;
61                 while (i + k < base2.size() && base2[i] == base2[i + k]) {
62                     k++;
63                 }
64                 base2.erase(base2.begin() + i + 1, base2.begin() + i + k);
65             } else {
66                 i++;
67             }
68         }
69     }
70 }
```

C: > Users > Eugene > Desktop > epic\_5 > G+ algotester\_lab\_4\_task\_1\_eugenie\_stefanovich.cpp > main()

```
7  int main() {
43      if (base1.size() > 1) {
44          for (int i = 0; i < base1.size() - 1; ) {
45              if (base1[i] == base1[i + 1]) {
47                  while (i + k < base1.size() && base1[i] == base1[i + k]) {
48                      k++;
49                  }
50                  base1.erase(base1.begin() + i + 1, base1.begin() + i + k);
51              } else {
52                  i++;
53              }
54          }
55      }
56
57      if (base2.size() > 1) {
58          for (int i = 0; i < base2.size() - 1; ) {
59              if (base2[i] == base2[i + 1]) {
60                  int k = 1;
61                  while (i + k < base2.size() && base2[i] == base2[i + k]) {
62                      k++;
63                  }
64                  base2.erase(base2.begin() + i + 1, base2.begin() + i + k);
65              } else {
66                  i++;
67              }
68          }
69      }
70
71      cout << base0.size() + base1.size() + base2.size() << endl;
72
73      for (int num : base0) {
74          cout << num << " ";
75      }
76
77      for (int num : base1) {
78          cout << num << " ";
79      }
80
81      for (int num : base2) {
82          cout << num << " ";
83      }
84
85
86      return 0;
87  }
```



# Lab# programming: Algotester Lab 6

## Lab 6v3

Limits: 1 sec., 256 MiB

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.

### Input

У першому рядку ціле число  $N$  - розмір поля для гри

У  $N$  наступних рядках стрічка  $row_i$  яка складається з  $N$  цифер -  $i$ -й рядок.

Ціле число  $Q$  - кількість запитань

У наступних  $Q$  рядках 2 цілих числа  $x_j, y_j$  - координати клітинок  $j$ -го запитання

**Time expected: 4h**

**Time spent: 4h**

C: > Users > Eugene > Desktop > epic\_5 > G+ algotester\_lab\_6\_task\_eugenie\_stefanovich.cpp > main()

```
1  #include <iostream>
2  #include <vector>
3  #include <string>
4  #include <set>
5  #include <algorithm>
6  using namespace std;
7
8
9  int main(){
10     int N , Q , x , y , k;
11     string row;
12     vector <int> base;
13     vector <int> wid;
14     vector <int> vert;
15     vector <int> gor;
16     vector <int> final;
17     set<int> razom;
18     cin >> N;
19
20     for(int i = N; i > 0; i--){
21         wid.push_back(i);
22     }
23     for(int i = 0; i < N; i++){
24         cin >> row;
25         for (char c : row) {
26             base.push_back(c - 48);
27         }
28     }
29
30     cout << endl;
31
32     cin >> Q;
33
34     for(int i = 0; i < Q; i++){
35         vert.clear();
36         gor.clear();
37         final.clear();
38         razom.clear();
39         cin >> x;
40         cin >> y;
41         if ( base[(((x - 1) * N) + y )-1] != 0){
42             cout << 1 << endl << base[(((x - 1) * N) + y - 1 ] << endl;
43             continue;
44         }
45
46         else if ( base[(((x - 1) * N) + y)-1 ] == 0){
47             k = y;
48             for(int l = 1; k < N ; l++ , k++){
49                 gor.push_back(base[(((x - 1) * N) + y + l)-1]);
50             }
51             k = y;
52             for(int l = 1; k > 1 ; l++ , k--){
53                 gor.push_back(base[(((x - 1) * N) + y - l - 1]);
54             }
55
56             k = x;
57
58             for(int l = 1; k < N ; l++ , k++){
59                 vert.push_back(base[(((x - 1 + 1) * N) + y )-1]);
60             }
```

C: > Users > Eugene > Desktop > epic\_5 > C algotester\_lab\_6\_task\_eugenie\_stefanovich.cpp > main()

```
9   int main(){
34   for(int i = 0; i < Q; i++){
41       if ( base[(((x - 1) * N) + y )-1] != 0){
45
46           else if ( base[(((x - 1) * N) + y)-1] == 0){
47               k = y;
48               for(int l = 1; k < N ; l++ , k++){
49                   gor.push_back(base[(((x - 1) * N) + y + 1)-1]);
50               }
51               k = y;
52               for(int l = 1; k > 1 ; l++ , k--){
53                   gor.push_back(base[(((x - 1) * N) + y - 1 - 1]);
54               }
55
56               k = x;
57
58               for(int l = 1; k < N ; l++ , k++){
59                   vert.push_back(base[(((x - 1 + 1) * N) + y )-1]);
60               }
61
62               k = x;
63
64               for(int l = 1; k > 1 ; l++ , k--){
65                   vert.push_back(base[(((x - 1 - 1) * N) + y - 1]);
66               }
67
68
69
70               for (int num : gor) {
71                   razom.insert(num);
72               }
73               for (int num : vert) {
74                   razom.insert(num);
75               }
76
77               for (int num : wid) {
78                   if (razom.find(num) == razom.end()) {
79                       final.push_back(num);
80                   }
81               }
82
83
84               if(final.size() == 0){
85                   cout << 0 << endl;
86                   continue;
87               }
88
89               cout << final.size() << endl;
90
91               sort(final.begin(), final.end());
92
93               for (int num : final)
94                   cout << num << " ";
95               cout << endl;
96
97           }
98
99   }
100
101   return 0;
```

**Practice# programming: Class Practice Task**

**Реалізувати функцію створення файлу 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 файлу.


```
enum FileOpResult { Success, Failure, ... };  
FileOpResult copy_file(char *file_from, char *file_to);
```

*Умови задачі:*

- копіювати вміст файлу 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 файлу.

**Time expected: 1 h**

**Time spent: 40 min**

: > Users > Eugene > Desktop > epic\_5 >  practice\_work\_team\_tasks\_eugenie\_stefanovich.cpp > C

```
1  #include <iostream>
2  #include <cstring>
3
4  using namespace std;
5
6  int main() {
7      const char* filename1 = "file1.txt";
8      const char* filename2 = "file2.txt";
9      char content[256];
10
11
12      cout << "Введіть вміст для запису у файл: ";
13      cin.getline(content, sizeof(content));
14
15
16
17      FILE* f1 = fopen(filename1, "w");
18
19      if (f1 == nullptr) {
20          cout << "Failure. File could not be created." << endl;
21          exit(0);
22      }
23
24      int len = strlen(content);
25      fwrite(content, sizeof(char), len, f1);
26      if (0 == len) {
27          fclose(f1);
28          cout << "Failure. File could not be created." << endl;
29          exit(0);
30      }
31
32      if (fclose(f1) != 0) {
33          cout << "Failure. File could not be created." << endl;
34          exit(0);
35      }
36
37      cout << "Success. File created." << endl;
```

```
PS C:\Users\Eugene\Desktop\epic_5\output> &
Введіть вміст для запису у файл: AFDKS
Success. File created.
Success. File is copied.
PS C:\Users\Eugene\Desktop\epic_5\output> █
```

## Practice# programming: Self Practice Task

### Рецепт

Обмеження: 2 сек., 256 МБ

Зенік хоче здивувати Марічку та спекти для неї торт. Але повар з нього поганий, тож він вирішив пошукати рецепт в інтернеті та знайшов такий дивний рецепт.

Усього в Зеніка є  $n$  інгредієнтів. Спочатку  $i$ -го інгредієнта є  $a_i$  грамів. Далі в рецепті написано, що кожної хвилини Зенік повинен взяти два інгредієнти, змішати їх та залишити рівно половину суміші, іншу половину викинути. І так поки на столі не залишиться лише один інгредієнт. Це й буде заготовка для торта.

Зенік також хоче спекти якнайбільший торт, щоб сильніше здивувати Марічку.

Допоможіть Зеніку та визначте, яку максимальну вагу може мати результуючий інгредієнт.

### Вхідні дані

У першому рядку задано ціле число  $n$  — кількість інгредієнтів.

У другому рядку  $n$  цілих чисел  $a_i$  — маса  $i$ -го інгредієнта в грамах.

### Вихідні дані

Виведіть єдине число — максимальну вагу заготовки. Відповідь вважатиметься правильною, якщо її абсолютна чи відносна похибка не буде більшою ніж  $10^{-7}$ .

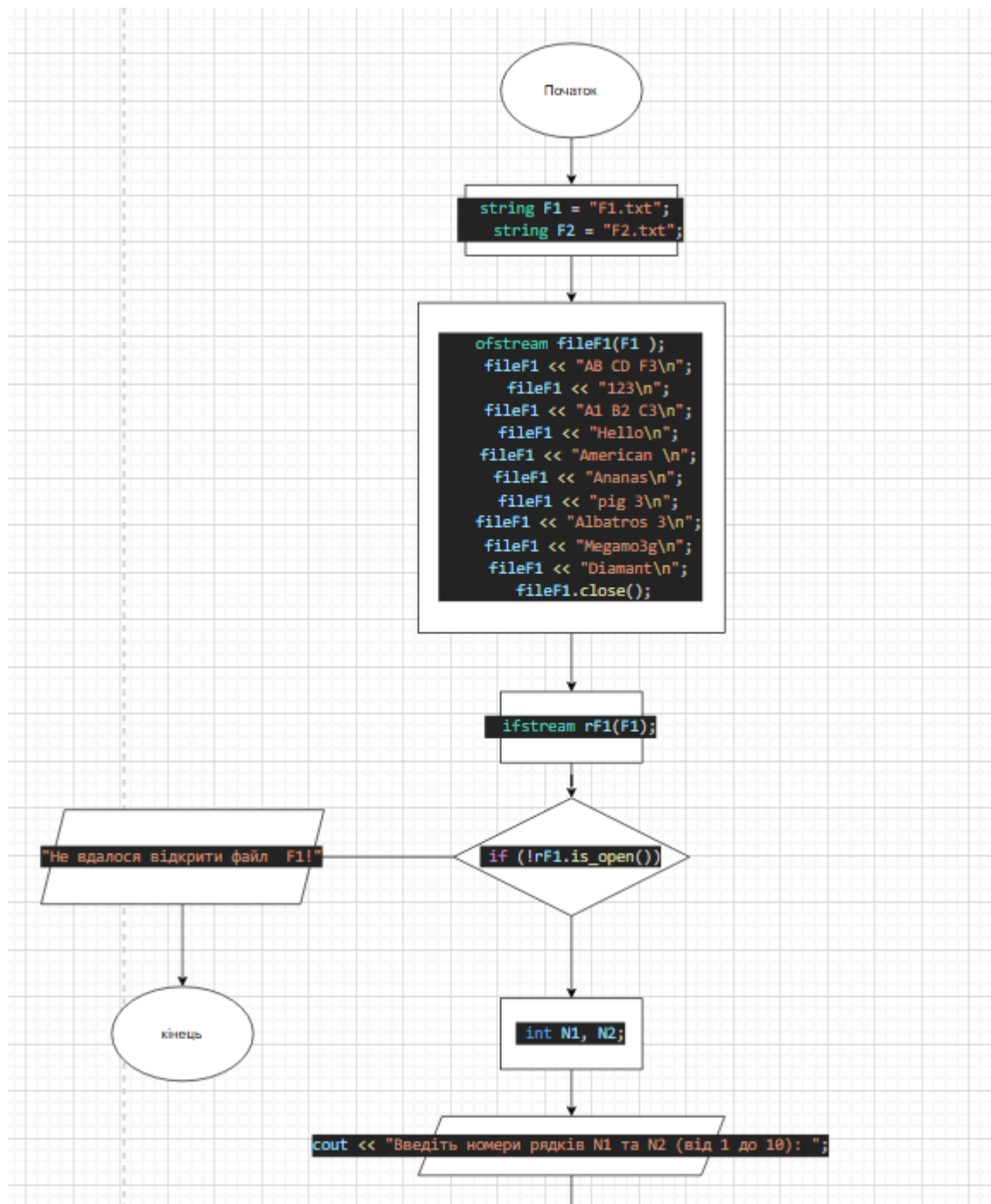
**Time expected: 30 min**

**Time spent: 30 min**

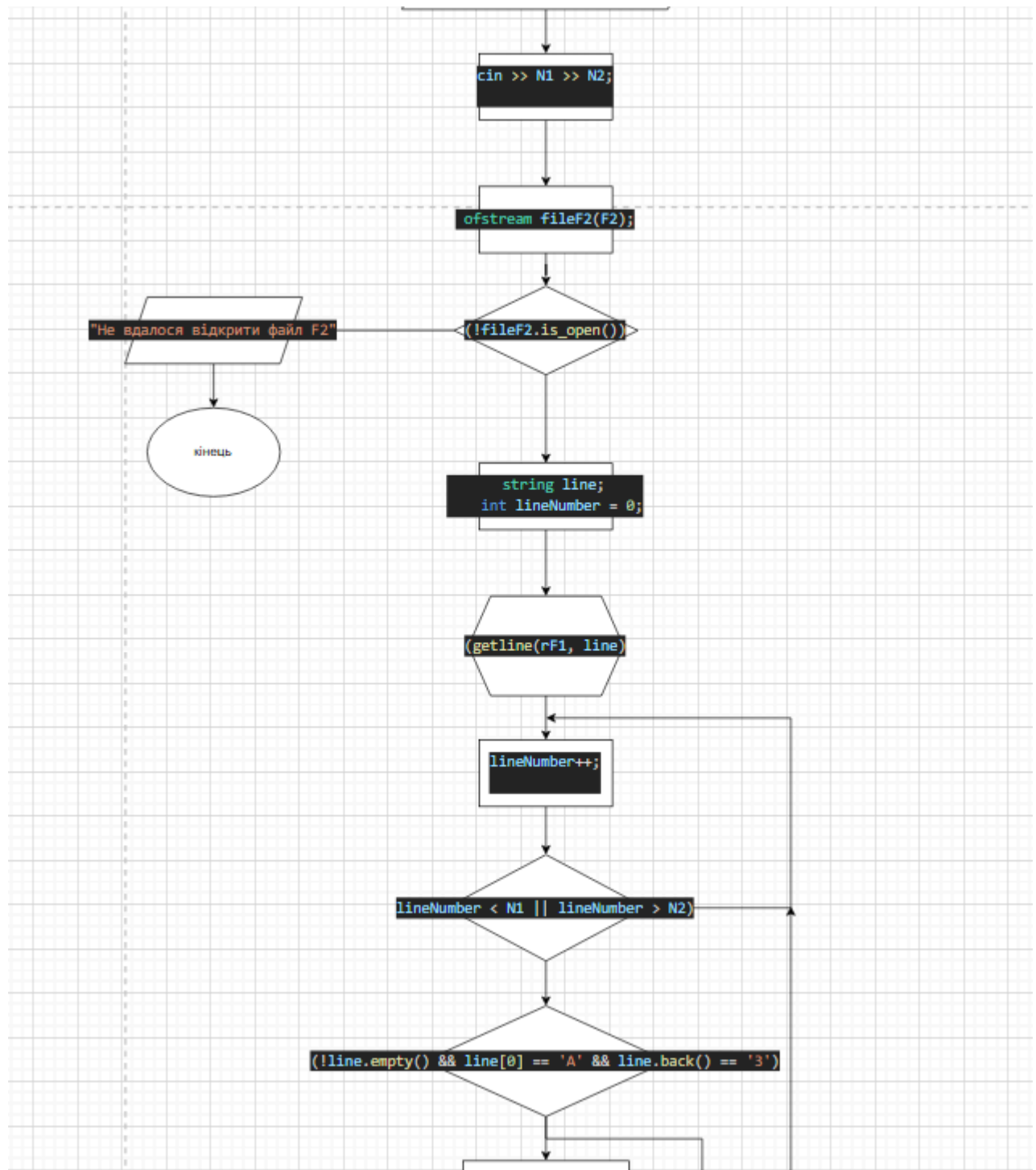
```
C: > Users > Eugene > Desktop > epic_5 > G+ practice_work_self_algotester_tasks_eugenie_stefanovich.cpp > main()
1  ∨ #include <iostream>
2  #include <vector>
3  #include <queue>
4  #include <iomanip>
5
6  using namespace std;
7
8  ∨ int main() {
9      int N;
10     double b, k;
11
12     priority_queue<double, vector<double>, greater<double>> pq;
13
14     cin >> N;
15     ∨ for (int i = 0; i < N; i++) {
16         cin >> b;
17         pq.push(b);
18     }
19
20     ∨ while (pq.size() > 1) {
21
22         double first = pq.top(); pq.pop();
23         double second = pq.top(); pq.pop();
24
25
26         k = (first + second) / 2.0;
27         pq.push(k);
28     }
29
30
31     cout << fixed << pq.top() << endl;
32     return 0;
33 }
34
35
```

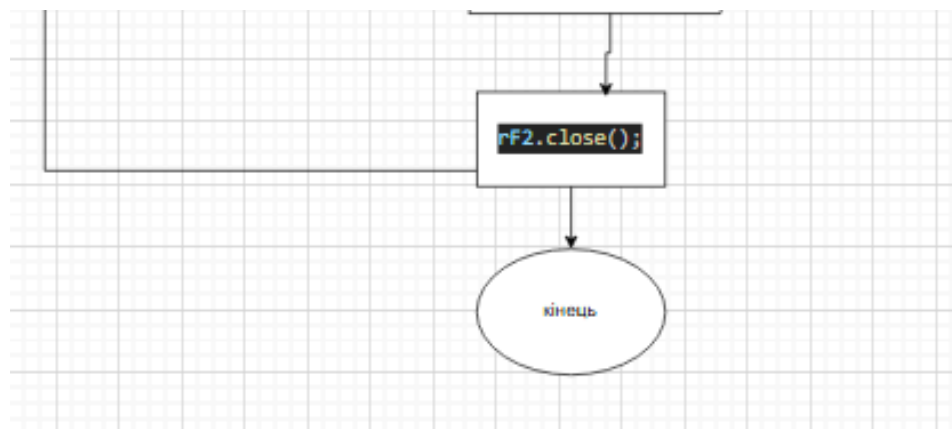
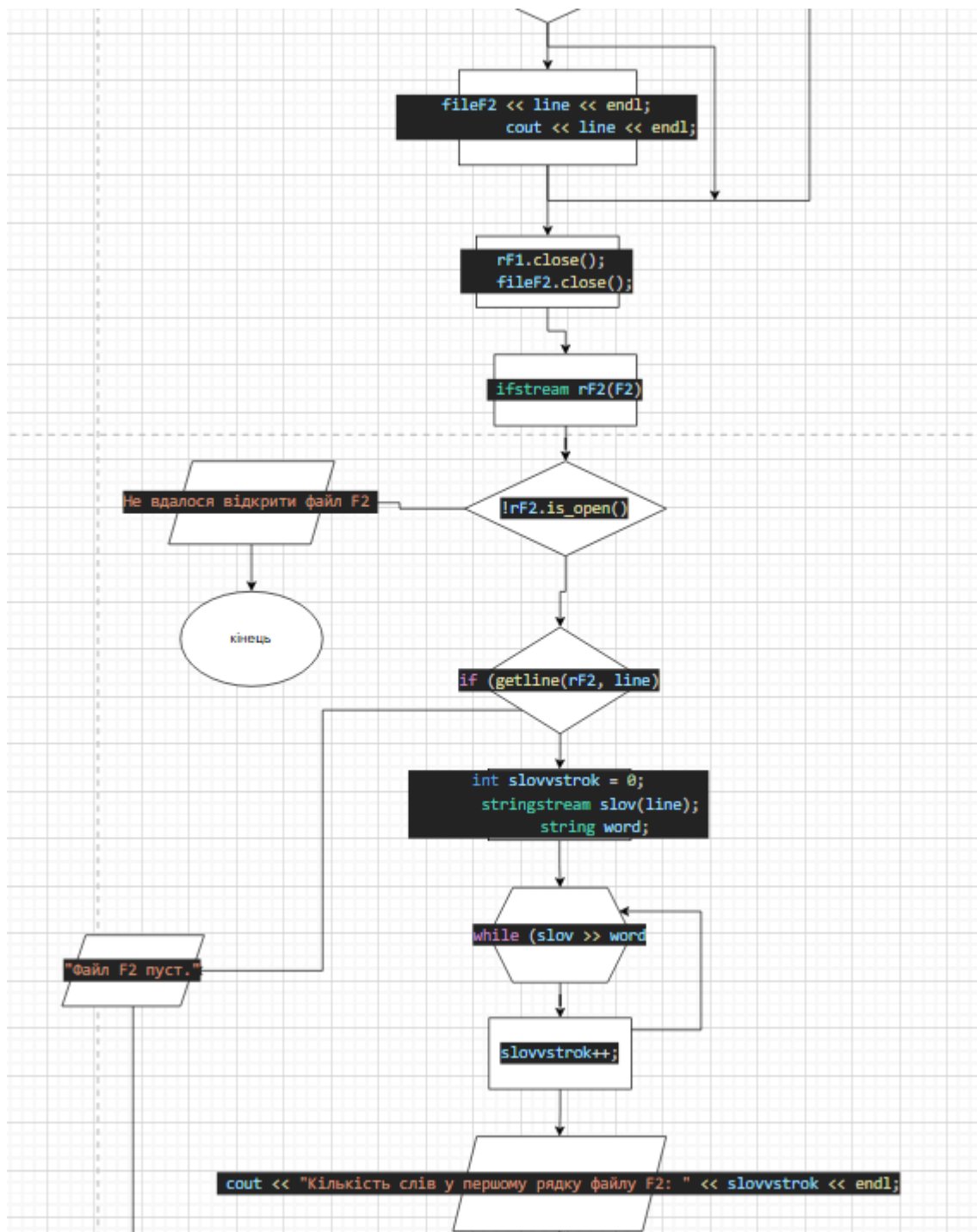


## схема к vns lab 9









Висновок: У ході роботи було вивчено основи роботи з файловою системою в C++: опрацьовано принципи обробки текстових і бінарних файлів, включаючи процеси запису, зчитування й редагування даних. Завдяки використанню різних типів рядкових змінних (`std::string` та `char*`) вдалося ознайомитися з різними підходами до зберігання й обробки текстових даних. Використання стандартної бібліотеки значно спростило роботу з файлами, дозволяючи зосередитися на вирішенні основних завдань.