

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

Шийка Стефан

**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**

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.

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

```

G+ vns_lab_6_task_stefan_shyika.cpp > ...
1  #include <iostream>
2  #include <vector>
3  #include <cstring>
4  #include <algorithm>
5
6  using namespace std;
7
8  int main(){
9      puts("Введіть строку: ");
10     char s[256];
11     gets(s);
12
13     //remove the last "."
14     int len = strlen(s);
15     if(s[len-1] == '.') s[len-1] = '\0';
16
17     //add words strating with a number to a vector using tockens
18     vector<string> words;
19     char* token = strtok(s, " ");
20
21     while(token != nullptr){
22
23         string word(token);
24
25         if(isdigit(word[0])) words.push_back(word);
26
27         token = strtok(nullptr, " ");
28     }
29
30     //sort them and cout
31     sort(words.rbegin(), words.rend());
32
33     cout << "Слова, що починаються з цифри, у спадному порядку:" << endl;
34     for (const auto& word : words) {
35         cout << word << endl;
36     }
37 }

```

```

Введіть строку:
giwd 3fkelf 6ksdmf1ndng 9kfmglsfgnksfgnsgkfj slkf 2fk
Слова, що починаються з цифри, у спадному порядку:
9kfmglsfgnksfgnsgkfj
6ksdmf1ndng
3fkelf
2fk

```

**Lab# programming: VNS Lab 8**

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.

#### 18. Структура "Книга":

- назва;
- автор;
- рік видання;
- кількість сторінок.

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

```

1 #include <iostream>
2 #include <string>
3
4 using namespace std;
5
6 typedef struct{
7     char title[50];
8     char author[50];
9     int year;
10    int numberOfPages;
11 }Book;
12
13 void addInitialData(const char* filename);
14 void deleteThreeFirstBooks(const char* filename);
15 void addBeforeTheBook(const char* filename, const Book& newBook, const char* title);
16 void printFile(const char* filename);
17
18 int main(){
19     const char* filename = "books.dat";
20     addInitialData(filename);
21     deleteThreeFirstBooks(filename);
22
23     Book newBook = {"newBook", "Stefan", 2024, 999};
24
25     addBeforeTheBook(filename, newBook, "Books");
26
27 }
28
29 void addInitialData(const char* filename) {
30     FILE* f = fopen(filename, "wb");
31     if (f == NULL){
32         cerr << "Can't open the file." << endl;
33         exit(1);
34     }
35
36     Book initialBooks[] = {
37         {"Book1", "Author1", 1925, 218},
38         {"Book2", "Author2", 1968, 281},
39         {"Book3", "Author3", 1949, 328},
40         {"Book4", "Author4", 1813, 278},
41         {"Book5", "Author5", 1951, 277},
42         {"Book6", "Author6", 1851, 635}
43     };
44
45     for (int i = 0; i < 6; i++) {
46         if (fwrite(&initialBooks[i], sizeof(Book), 1, f) != 1) {
47             cerr << "Error writing to file." << endl;
48             exit(2);
49         }
50     }
51     fclose(f);
52     cout << "Initial books added successfully." << endl;
53     printFile(filename);
54 }
55
56 void deleteThreeFirstBooks(const char* filename){
57     FILE* f = fopen(filename, "rb");
58     FILE* tmp = fopen("tmp.dat", "wb");
59
60     if (f == NULL || tmp == NULL){
61         cerr << "Can't open the file." << endl;
62         exit(3);
63     }
64
65     Book book;
66     int counter = 0;
67
68     while(fread(&book, sizeof(Book), 1, f) == 1){
69         if(counter >= 3){
70             fwrite(&book, sizeof(Book), 1, tmp);
71         }
72         counter++;
73     }
74
75     fclose(tmp);
76     fclose(f);
77
78     remove(filename);
79     rename("tmp.dat", filename);
80     cout << "Deleted the first 3 books from " << filename << "." << endl;
81     printFile(filename);
82 }
83
84 void addBeforeTheBook(const char* filename, const Book& newBook, const char* title){
85     FILE* f = fopen(filename, "rb");
86     FILE* tmp = fopen("tmp.dat", "wb");
87
88     if (f == NULL || tmp == NULL) {
89         cerr << "Error opening file." << endl;
90         exit(4);
91     }
92
93     bool inserted = false;
94     Book book;
95
96     while(fread(&book, sizeof(Book), 1, f)){
97         if(strcmp(book.title, title) == 0 && !inserted){
98             fwrite(&newBook, sizeof(Book), 1, tmp);
99             inserted = true;
100         }
101         fwrite(&book, sizeof(Book), 1, tmp);
102     }
103
104     if (!inserted) {
105         cout << "No such title in the file :( Book inserted in the end." << endl;
106         fwrite(&newBook, sizeof(Book), 1, tmp);
107     }
108
109     fclose(f);
110     fclose(tmp);
111
112     remove(filename);
113     rename("tmp.dat", filename);
114     if (inserted){
115         cout << "Book added successfully before title: " << title << endl;
116     }
117     printFile(filename);
118 }
119
120 void printFile(const char* filename){
121     FILE* f = fopen(filename, "rb");
122     if (f == NULL) {
123         cerr << "Error opening file." << endl;
124         exit(5);
125     }
126
127     Book book;
128
129     cout << "Contents of " << filename << ":" << endl;
130     cout << "-----" << endl;
131
132     while (fread(&book, sizeof(Book), 1, f) == 1) {
133         cout << "Title: " << book.title << endl;
134         cout << "Author: " << book.author << endl;
135         cout << "Year: " << book.year << endl;
136         cout << "Number of Pages: " << book.numberOfPages << endl;
137         cout << "-----" << endl;
138     }
139
140     fclose(f);
141 }
142

```

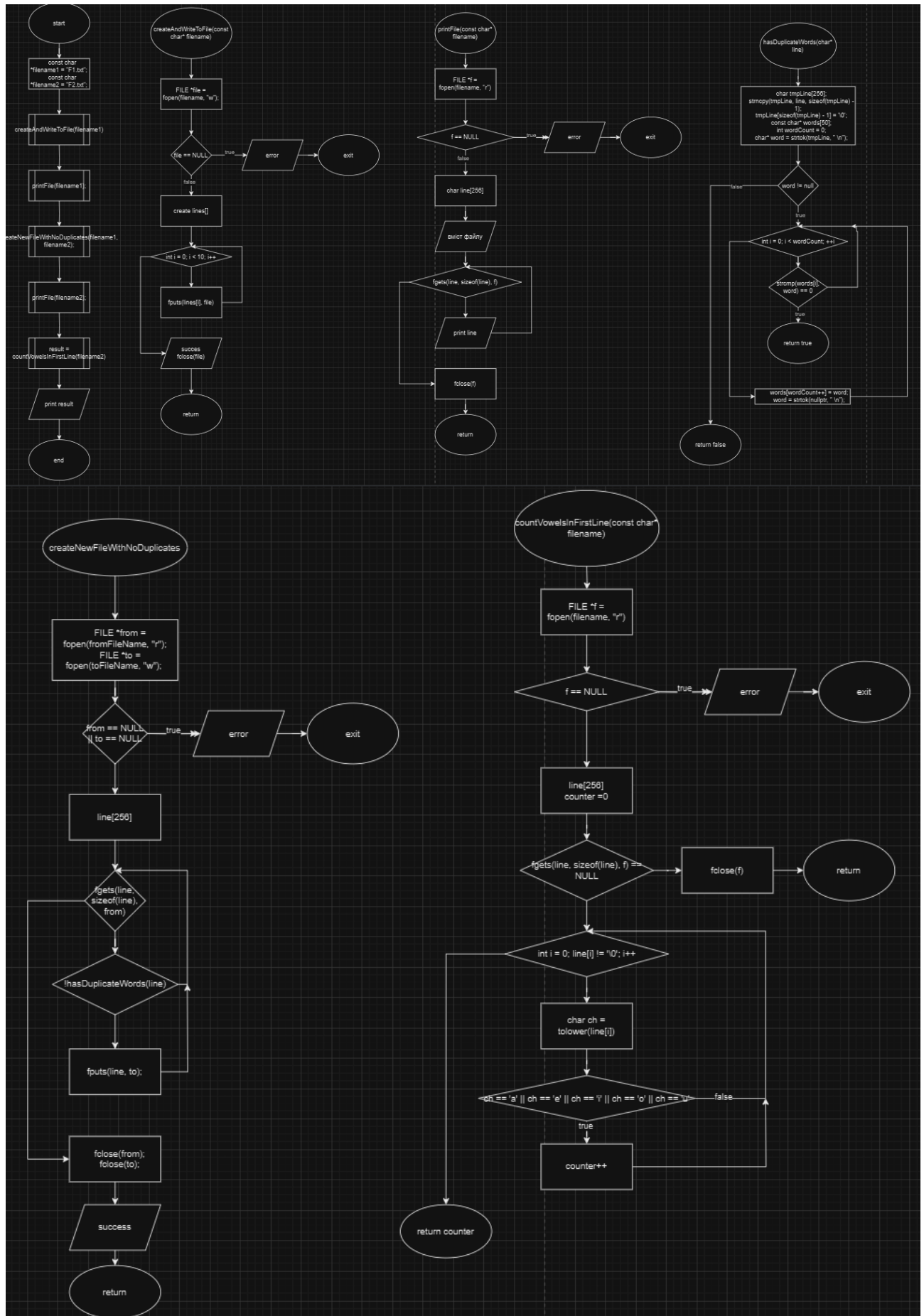
## **Lab# programming: VNS Lab 9**

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

Виконати завдання.

18.

- 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.



```

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 вдалося відкрити файл для запису" << 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 << "Дані успішно записані у файл " << 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 вдалося відкрити файл для читання" << endl;
54         exit(2);
55     }
56
57     char line[256];
58
59     cout << "Вміст файлу " << 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(char* line) {
70     char tmpLine[256];
71     strcpy(tmpLine, line, sizeof(tmpLine) - 1);
72     tmpLine[sizeof(tmpLine) - 1] = '\0';
73
74     const char* words[50];
75     int wordCount = 0;
76
77     char* word = strtok(tmpLine, " \n");
78     while (word != nullptr) {
79
80         for (int i = 0; i < wordCount; ++i) {
81             if (strcmp(words[i], word) == 0) {
82                 return true;
83             }
84         }
85
86         words[wordCount++] = word;
87         word = strtok(nullptr, " \n");
88     }
89     return false;
90 }
91
92 void createNewFileWithNoDuplicates(const char* fromFileName, const char* toFileName) {
93     FILE *from = fopen(fromFileName, "r");
94     FILE *to = fopen(toFileName, "w");
95     if (from == NULL || to == NULL) {
96         cerr << "Не вдалося відкрити файл для читання" << endl;
97         exit(3);
98     }
99
100     char line[256];
101
102     while (fgets(line, sizeof(line), from)) {
103         if (!hasDuplicateWords(line)) {
104             fputs(line, to);
105         }
106     }
107
108     fclose(from);
109     fclose(to);
110
111     cout << "Lines without repeated words copied successfully." << endl;
112 }
113
114 int countVowelsInFirstLine(const char* filename) {
115     FILE* f = fopen(filename, "r");
116
117     if (f == NULL) {
118         cerr << "Не вдалося відкрити файл для читання" << endl;
119         exit(4);
120     }
121
122     char line[256];
123     if (fgets(line, sizeof(line), f) == NULL) {
124         fclose(f);
125         return 0; // No lines in #file
126     }
127
128     int counter = 0;
129
130     for (int i = 0; line[i] != '\0'; i++) {
131         char ch = tolower(line[i]);
132         if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
133             counter++;
134         }
135     }
136
137     return counter;
138 }
139

```



```

Дані успішно записані у файл F1.txt
Вміст файлу F1.txt:
-----
line 1
line 2: line
line 3
line 4: line line
line 5
line 6
line 7: line
line 8
line 9: line line
line 10
Lines without repeated words copied successfully.
Вміст файлу F2.txt:
-----
line 1
line 3
line 5
line 6
line 8
line 10

Number of vowels in the first line of F2: 2
PS C:\Users\user\Desktop\c++\epic5>

```

## Lab# programming: Algotester Lab 4

### 4.2

#### Lab 4v2

*Limits: 1 sec., 256 MiB*

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.

#### Input

У першому рядку цілі числа  $N$  та  $K$

У другому рядку  $N$  цілих чисел - елементи масиву  $a$

#### Output

У першому рядку ціле число  $N$  - розмір множини  $a$

У наступному рядку  $N$  цілих чисел - множина  $a$

```
algotester_lab_4_task_stefan_shyika_v1.cpp > main()
1  #include <iostream>
2  #include <set>
3
4  using namespace std;
5
6  int main(){
7      int N, K;
8      set<int> numSet;
9      cin >> N >> K;
10     int tmp;
11
12     for(int i = 0; i < N; i++){
13         cin >> tmp;
14         numSet.insert(tmp);
15     }
16
17     int len = numSet.size();
18     cout << len << endl;
19
20     for(int i = K; i < len + K; i++){
21         int index = i % len;
22         auto it = numSet.begin();
23         advance(it, index);
24
25         if(i != len + K - 1){
26             cout << *it << " ";
27         }else{
28             cout << *it;
29         }
30     }
31 }
```

```

algotester_lab_4_task_stefan_shyika_v2.cpp > main()
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4
5  using namespace std;
6
7  int main() {
8      int N, K;
9      cin >> N >> K;
10
11     vector<int> a(N);
12     for (int i = 0; i < N; i++) {
13         cin >> a[i];
14     }
15
16     sort(a.begin(), a.end());
17
18     auto last = unique(a.begin(), a.end());
19     a.erase(last, a.end());
20
21     int len = a.size();
22     K = K % len;
23     rotate(a.begin(), a.begin() + K, a.end());
24
25     cout << len << endl;
26     for(int i = 0; i < len; i++){
27         if(i != len-1){
28             cout << a[i] << " ";
29         }else{
30             cout << a[i];
31         }
32     }
33 }

```

2 days ago

Lab 4v2 - Lab 4v2

C++ 23

Accepted

0.003

1.219

1858136

## 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 видаліть 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.

algotester\_lab\_4\_variant\_2\_stefan\_shyika\_v1.cpp > main()

```
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  #include <set>
5
6  using namespace std;
7
8  int main(){
9      int N;
10     cin >> N;
11
12     vector<int> vec(N);
13
14     for(int i = 0; i < N; i++){
15         cin >> vec[i];
16     }
17
18     sort(vec.begin(), vec.end());
19     auto last = unique(vec.begin(), vec.end());
20     vec.erase(last, vec.end());
21
22     vector<int> mod0, mod1, mod2;
23     for (int num : vec) {
24         if (num % 3 == 0) {
25             mod0.push_back(num);
26         } else if (num % 3 == 1) {
27             mod1.push_back(num);
28         } else {
29             mod2.push_back(num);
30         }
31     }
32
33     sort(mod0.begin(), mod0.end());
34     sort(mod2.begin(), mod2.end());
35     sort(mod1.begin(), mod1.end(), greater<int>());
36
37     vector<int> result;
38     result.insert(result.end(), mod0.begin(), mod0.end());
39     result.insert(result.end(), mod1.begin(), mod1.end());
40     result.insert(result.end(), mod2.begin(), mod2.end());
41
42     int len = result.size();
43     cout << len << endl;
44     for (int i = 0; i < len; i++) {
45         if(i != len -1){
46             cout << result[i] << " ";
47         }else{
48             cout << result[i];
49         }
50     }
51 }
```

```

1  #include <iostream>
2  #include <vector>
3
4  using namespace std;
5
6  int partition(vector<int>& arr, int start, int end) {
7      int pivot = arr[end];
8      int i = start - 1;
9      for (int j = start; j < end; j++) {
10         if (arr[j] < pivot) {
11             i++;
12             swap(arr[i], arr[j]);
13         }
14     }
15     swap(arr[i + 1], arr[end]);
16     return i + 1;
17 }
18
19 void quickSort(vector<int>& arr, int start, int end) {
20     if (start < end) {
21         int pi = partition(arr, start, end);
22         quickSort(arr, start, pi - 1);
23         quickSort(arr, pi + 1, end);
24     }
25 }
26
27 vector<int> removeDuplicatesAfterSorting(const vector<int>& arr) {
28     vector<int> uniqueArr;
29     for (size_t i = 0; i < arr.size(); i++) {
30         if (i == 0 || arr[i] != arr[i - 1]) {
31             uniqueArr.push_back(arr[i]);
32         }
33     }
34     return uniqueArr;
35 }
36
37
38 int main() {
39     int N;
40     cin >> N;
41
42     vector<int> vec(N);
43     for (int i = 0; i < N; i++) {
44         cin >> vec[i];
45     }
46
47     quickSort(vec, 0, N - 1);
48
49     vec = removeDuplicatesAfterSorting(vec);
50
51
52     vector<int> mod0, mod1, mod2;
53     for (int num : vec) {
54         if (num % 3 == 0) {
55             mod0.push_back(num);
56         } else if (num % 3 == 1) {
57             mod1.push_back(num);
58         } else {
59             mod2.push_back(num);
60         }
61     }
62
63     //mod0 and mod1 are already sorted
64     for (size_t i = 0; i < mod1.size() / 2; i++) {
65         swap(mod1[i], mod1[mod1.size() - 1 - i]);
66     }
67
68
69     vector<int> result;
70     result.insert(result.end(), mod0.begin(), mod0.end());
71     result.insert(result.end(), mod1.begin(), mod1.end());
72     result.insert(result.end(), mod2.begin(), mod2.end());
73
74     int len = result.size();
75     cout << len << endl;
76     for (int i = 0; i < len; i++) {
77         if (i != len - 1) {
78             cout << result[i] << " ";
79         } else {
80             cout << result[i];
81         }
82     }
83 }

```

19 hours ago	Lab 4v3 - Lab 4v3	C++ 23	Accepted	0.003	1.242	1858386
--------------	-------------------	--------	----------	-------	-------	---------

# Lab# programming: Algotester Lab 6

## Lab 6v2

Limits: 2 sec., 256 MiB

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

Кожна клітинка може мати таке значення:

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

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

Далі йдуть  $Q$  запитів з координатами клітинки  $\{x, y\}$ . На кожен запит ви маєте вивести стрічку  $s_i$  - посортовані за алфавітом букви фігур, які атакують цю клітинку (пішаки атакують вниз).

У випадку, якщо на клітинці стоїть якась фігура - виведіть символ  $X$ .

У випадку, якщо клітинку не атакують - виведіть  $O$ .

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

## Input

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

У наступному рядку ціле число  $Q$  - кількість записів

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

## Output

$Q$  разів відповідь у наступному форматі:

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

17 hours ago	Lab 6v2 - Lab 6v2	C++ 23	Accepted	0.003	1.332	1858492
--------------	-------------------	--------	----------	-------	-------	---------

```

1  #include <iostream>
2  #include <vector>
3  #include <set>
4  #include <algorithm>
5
6  using namespace std;
7
8  bool validCoordinates(int x, int y) {
9      return x >= 0 && x < 8 && y >= 0 && y < 8;
10 }
11
12 void addPawnAttacks(vector<set<char>>& attacks, int x, int y) {
13     if (validCoordinates(x + 1, y - 1)) attacks[(x + 1) * 8 + (y - 1)].insert('P');
14     if (validCoordinates(x + 1, y + 1)) attacks[(x + 1) * 8 + (y + 1)].insert('P');
15 }
16
17 void addRookAttacks(vector<set<char>>& attacks, int x, int y, char piece) {
18     for (int i = 0; i < 8; i++) {
19         if (i != y) attacks[x * 8 + i].insert(piece);
20         if (i != x) attacks[i * 8 + y].insert(piece);
21     }
22 }
23
24 void addBishopAttacks(vector<set<char>>& attacks, int x, int y, char piece) {
25     for (int i = -7; i <= 7; i++) {
26         if (i != 0) {
27             if (validCoordinates(x + i, y + i)) attacks[(x + i) * 8 + (y + i)].insert(piece);
28             if (validCoordinates(x + i, y - i)) attacks[(x + i) * 8 + (y - i)].insert(piece);
29         }
30     }
31 }
32
33 void addKnightAttacks(vector<set<char>>& attacks, int x, int y) {
34     int knightMoves[8][2] = {{-2, -1}, {-2, 1}, {2, -1}, {2, 1}, {-1, -2}, {1, -2}, {-1, 2}, {1, 2}};
35     for (auto& move : knightMoves) {
36         int x1 = x + move[0], y1 = y + move[1];
37         if (validCoordinates(x1, y1)) attacks[x1 * 8 + y1].insert('N');
38     }
39 }
40
41 void addKingAttacks(vector<set<char>>& attacks, int x, int y) {
42     for (int dx = -1; dx <= 1; dx++) {
43         for (int dy = -1; dy <= 1; dy++) {
44             if (dx != 0 || dy != 0) {
45                 int x1 = x + dx, y1 = y + dy;
46                 if (validCoordinates(x1, y1)) attacks[x1 * 8 + y1].insert('K');
47             }
48         }
49     }
50 }
51
52 int main() {
53     vector<string> board(8);
54     for (int i = 0; i < 8; i++) {
55         cin >> board[i];
56     }
57
58     vector<set<char>> attacks(64);
59
60     for (int i = 0; i < 8; i++) {
61         for (int j = 0; j < 8; j++) {
62             char piece = board[i][j];
63             switch (piece) {
64                 case 'P':
65                     addPawnAttacks(attacks, i, j);
66                     break;
67                 case 'R':
68                     addRookAttacks(attacks, i, j, 'R');
69                     break;
70                 case 'N':
71                     addKnightAttacks(attacks, i, j);
72                     break;
73                 case 'B':
74                     addBishopAttacks(attacks, i, j, 'B');
75                     break;
76                 case 'K':
77                     addKingAttacks(attacks, i, j);
78                     break;
79                 case 'Q': //queen = rook + bishop
80                     addRookAttacks(attacks, i, j, 'Q');
81                     addBishopAttacks(attacks, i, j, 'Q');
82                     break;
83                 default:
84                     break;
85             }
86         }
87     }
88
89     int Q;
90     cin >> Q;
91     for (int i = 0; Q - i > 0; i++) {
92         int x, y;
93         cin >> x >> y;
94         x--; y--;
95
96         char cell = board[x][y];
97         if (cell != 'O') {
98             cout << "X\n";
99         } else {
100             if (attacks[x * 8 + y].empty()) {
101                 cout << "O\n";
102             } else {
103                 vector<char> attackers(attacks[x * 8 + y].begin(), attacks[x * 8 + y].end());
104                 sort(attackers.begin(), attackers.end());
105                 for (char attacker : attackers) {
106                     cout << attacker;
107                 }
108                 cout << "\n";
109             }
110         }
111     }
112 }

```

## 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.



```

1  #include <iostream>
2  #include <cstring>
3
4  using namespace std;
5
6  enum FileOpResult { Success, Failure };
7
8  FileOpResult write_to_file(const char *name, const char *content);
9  FileOpResult copy_file(const char *file_from, const char *file_to);
10
11 int main() {
12     const char* filename1 = "file1.txt";
13     const char* filename2 = "file2.txt";
14     char content[256];
15
16     cout << "Введіть вміст для запису у файл: ";
17     cin.getline(content, sizeof(content));
18
19     FileOpResult result1 = write_to_file(filename1, content);
20     if (result1 == Success) {
21         cout << "Success. File created." << endl;
22     } else {
23         cout << "Failure. File could not be created." << endl;
24     }
25
26     FileOpResult result2 = copy_file(filename1, filename2);
27     if (result2 == Success) {
28         cout << "Success. File is copied." << endl;
29     } else {
30         cout << "Failure coping file." << endl;
31     }
32 }
33
34 FileOpResult write_to_file(const char *name, const char *content) {
35     if (name == nullptr || strlen(name) == 0) {
36         return FileOpResult::Failure;
37     }
38
39     FILE* f = fopen(name, "w");
40     if (f == NULL) {
41         return FileOpResult::Failure;
42     }
43
44     int len = strlen(content);
45     size_t written = fwrite(content, sizeof(char), len, f);
46
47     if (written != len) {
48         fclose(f);
49         return FileOpResult::Failure;
50     }
51
52     if (fclose(f) != 0) {
53         return FileOpResult::Failure;
54     }
55
56     return FileOpResult::Success;
57 }
58
59 FileOpResult copy_file(const char *file_from, const char *file_to){
60     if (file_from == nullptr || strlen(file_from) == 0 || file_to == nullptr || strlen(file_to) == 0) {
61         return FileOpResult::Failure;
62     }
63
64     FILE* f1 = fopen(file_from, "r");
65     FILE* f2 = fopen(file_to, "w");
66
67     if (f1 == NULL || f2 == NULL) {
68         return FileOpResult::Failure;
69     }
70
71     char buffer[512];
72     while(fgets(buffer, sizeof(buffer), f1) != NULL){
73         if(fputs(buffer, f2) == EOF){
74             fclose(f1);
75             fclose(f2);
76             return FileOpResult::Failure;
77         }
78     }
79
80     if (fclose(f1) != 0 || fclose(f2) != 0) {
81         return FileOpResult::Failure;
82     }
83
84     return FileOpResult::Success;
85 }
86

```

```
Введіть вміст для запису у файл: roses are red, violets are blue
Success. File created.
Success. File is copied.
```

## Practice# programming: Self Practice Task

### Числа для малят

Limits: 2 sec., 256 MB

Малата, напишіть на клаптику паперу ціле додатне число  $n$ . Тепер уявіть собі, що Ви можете переставляти цифри у його десятковому записі як завгодно. Єдина умова — не повинно бути нулів на початку запису числа. Вам потрібно визначити мінімальне та максимальне числа, які можна отримати таким способом.

Бажаю успіху! І не забувайте, що ви все знаєте, просто можливо щось забули.

#### Input

У єдиному рядку задано одне ціле число  $n$ .

#### Output

У єдиному рядку виведіть два цілих числа — мінімальне та максимальне числа, які можна отримати перестановкою цифр числа  $n$ .

14 hours ago	0889 - Числа для малят	C++ 23	Accepted	0.003	1.215	1858604
--------------	------------------------	--------	----------	-------	-------	---------

```
practice_work_self_algotester_tasks_stefan_shyika.cpp > ...
1  #include <iostream>
2  #include <string>
3  #include <algorithm>
4
5  using namespace std;
6
7  int main() {
8      string n;
9      cin >> n;
10
11     string minNumber = n;
12     sort(minNumber.begin(), minNumber.end());
13
14     // Перемішуємо першу ненульову цифру на початок, якщо є нуль на початку
15     if (minNumber[0] == '0') {
16         for (int i = 1; i < minNumber.size(); i++) {
17             if (minNumber[i] != '0') {
18                 swap(minNumber[0], minNumber[i]);
19                 break;
20             }
21         }
22     }
23
24     string maxNumber = n;
25     sort(maxNumber.begin(), maxNumber.end(), greater<char>());
26
27     cout << minNumber << " " << maxNumber;
28
29 }
```

## Pull

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.