

Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний  
інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 9 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

«Дослідження алгоритмів обходу масивів»

Варіант 25

Виконав ПІ-15, Плугатирьов Дмитро Валерійович

студент (шифр, прізвище, ім'я, по батькові)

Перевірив Вечерковська Анастасія Сергіївна

(прізвище, ім'я, по батькові)

Київ 2021

## Лабораторна робота 9

### Дослідження алгоритмів обходу масивів

**Мета** – дослідити алгоритми обходу масивів, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

### Завдання

Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (табл. 1).
2. Ініціювання змінної, що описана в п.1 даного завдання.
3. Обчислення змінної, що описана в п.1, згідно з варіантом (табл. 1).

25	Задано матрицю дійсних чисел $A[m,n]$ . У кожному рядку матриці знайти останній мінімальний елемент $X$ і його місцезнаходження. Обміняти знайдене значення $X$ з елементом середнього стовбця.
----	---

### 1. Постановка задачі

Створити матрицю дійсних чисел. В кожному рядку матриці знайти останній мінімальний елемент, його індекс та обміняти першозгаданого з іншим із середнього стовпця.

### 2. Побудова математичної моделі

Змінна	Тип	Ім'я	Призначення
Матриця	Дійсний	<code>matrix[row][col]</code>	Початкові дані, результат
Рядки матриці	Цілочисельний	<code>row</code>	Початкові дані

Стовпці матриці	Цілочисельний	col	Початкові дані
Мінімальний елемент	Дійсний	min	Проміжні дані
Середній елемент	Дійсний	x	Проміжні дані
Лічильник циклу	Цілочисельний	i	Проміжні дані
Лічильник вкладеного циклу	Цілочисельний	y	Проміжні дані
Параметр для визначення випадкового числа	Цілочисельний	temp	Проміжні дані
Індекс середнього елемента рядка	Цілочисельний	midI	Проміжні дані
Індекс найменшого елемента рядка	Цілочисельний	colI	Проміжні дані
Індекс рядка з найменшим елементом	Цілочисельний	rowI	Проміжні дані
Пошук елемента, його індексу та обмін із середнім	Підпрограма	find_elem_and_exchange	Проміжні дані
Обмін елементу з із середнім	Підпрограма	Exchange_min_mid	Проміжні дані

Створення матриці та її ініціювання	Підпрограма	create_and_init_matrix	Проміжні дані
---	-------------	------------------------	------------------

Розмірність матриці визначається користувачем. Прохід по рядкам матриці відбувається з використанням вкладених циклів з умовами на відбір елементу та індексу. Мінімальний елемент та його індекс заносяться у змінні-утримувачі. Виклик підпрограми обмінює середній елемент рядка місцями зі знайденим.

За парної кількості стовпців матриці середнім елементом вважається крайній справа.

Дія  $x := \text{rand}() \% \text{num1} - \text{num2}$  означає присвоєння згенерованого псевдовипадковим чином цілого числа до змінної  $x$  в межах від  $\text{num1} - \text{num2} - 1$  до  $-\text{num2}$ .

### **3. Розв'язання**

Програмні специфікації записати у псевдокоді та графічній формі у вигляді блок-схеми.

*Крок 1.* Визначити основні дії.

*Крок 2.* Визначення розмірності та ініціалізація матриці.

*Крок 3.* Знаходження мінімальних елементів, їх індексів та обмін між із середніми в рядках.

### **4. Псевдокод**

## **О с н о в н а   п р о г р а м а :**

### *Крок 1*

#### **початок**

визначення розмірності та ініціалізація матриці

знаходження мінімальних елементів, їх індексів та обмін між із середніми в рядках

#### **кінець**

### *Крок 2*

#### **початок**

**ввести row**

**ввести col**

**create\_and\_init\_matrix(matrix, row, col)**

знаходження мінімальних елементів, їх індексів та обмін між із середніми в рядках

#### **кінець**

### *Крок 3*

#### **початок**

**ввести row**

**ввести col**

**create\_and\_init\_matrix(matrix, row, col)**

**find\_elem\_and\_exchange(matrix, row, col, x)**

#### **кінець**

## **П і д п р о г р а м и:**

**create\_and\_init\_matrix(matrix, row, col)**

**повторити для i від 0 до row**

**повторити для y від 0 до col**

**temp := rand() % 3**

**якщо temp == 2**

**то**

**matrix[i][y] := rand() % 1000 \* 2 - 1000**

**все якщо**

**інакше якщо temp == 1**

**то**

**matrix[i][y] := (rand() % 1000 \* 2 - 1000) / 12.4**

**все якщо**

**інакше**

**matrix[i][y] := 0**

**все якщо**

**все повторити**

**все повторити**

**кінець**

**find\_elem\_and\_exchange(matrix, row, col, x)**

$\text{midI} := \text{row} / 2$

**повторити для i від 0 до row**

$\text{rowI} := i$

$x := \text{matrix}[i][0]$

**повторити для y від 0 до col**

**якщо**  $x \geq \text{matrix}[i][y]$

**то**

$x := \text{matrix}[i][y]$

$\text{colI} := y$

**все якщо**

**якщо**  $\text{colI} \neq \text{row} / 2$  **або**  $\text{colI} \neq \text{row} / 2 + 1$

**то**

**Exchange\_min\_mid(matrix, rowI, colI, x, midI)**

**все якщо**

**все повторити**

**все повторити**

**кінець**

**Exchange\_min\_mid(matrix, rowI, colI, x, midI)**

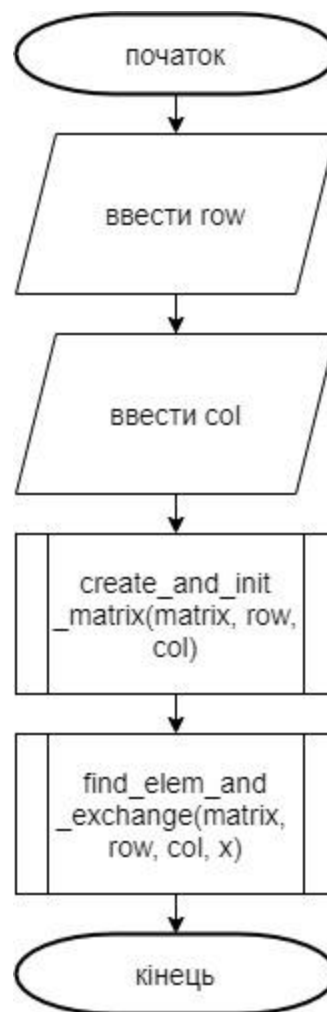
$\text{matrix}[\text{rowI}][\text{colI}] := \text{matrix}[\text{rowI}][\text{midI}]$

$\text{matrix}[\text{rowI}][\text{midI}] := x$

**кінець**

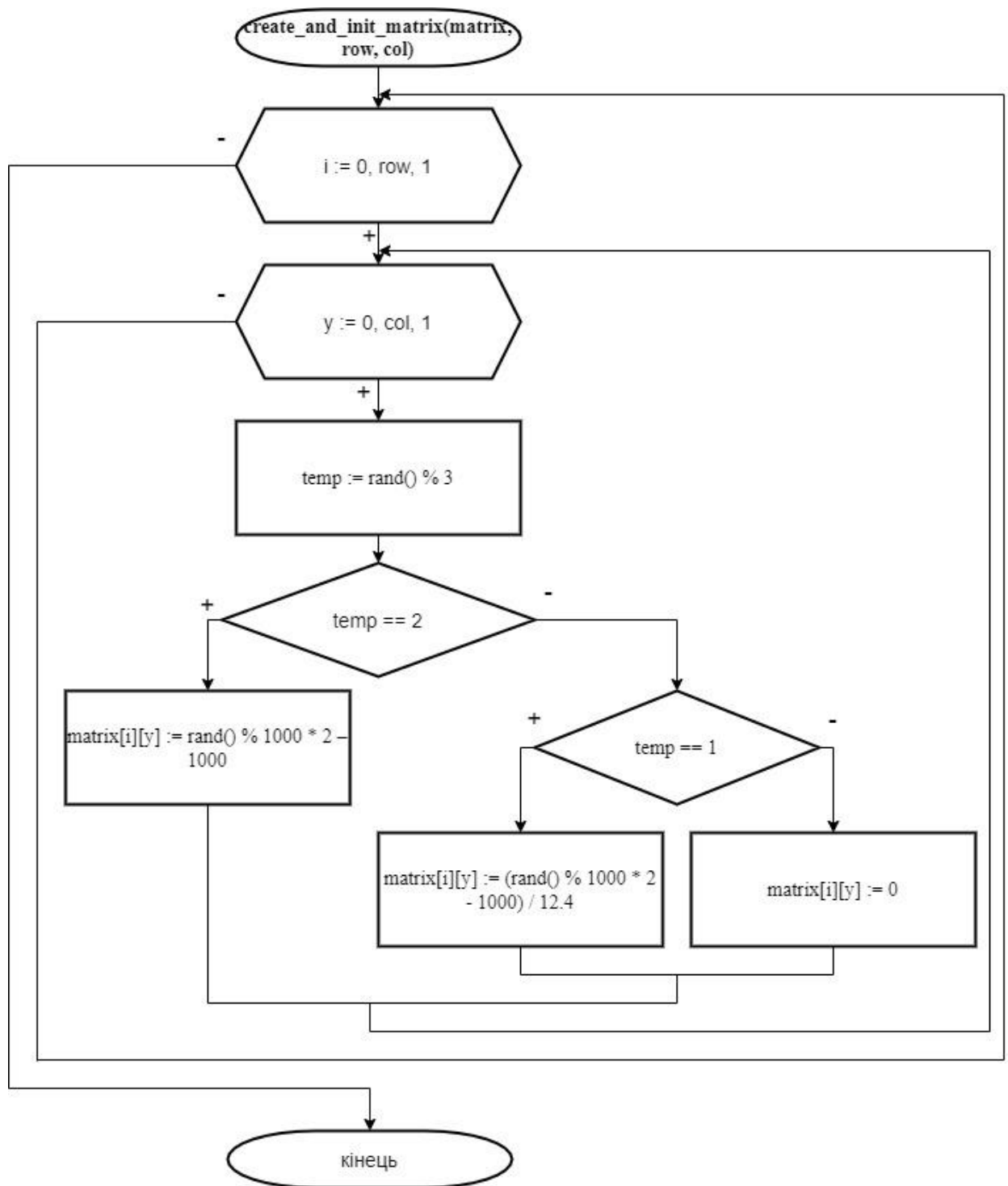
*Блок-схема*

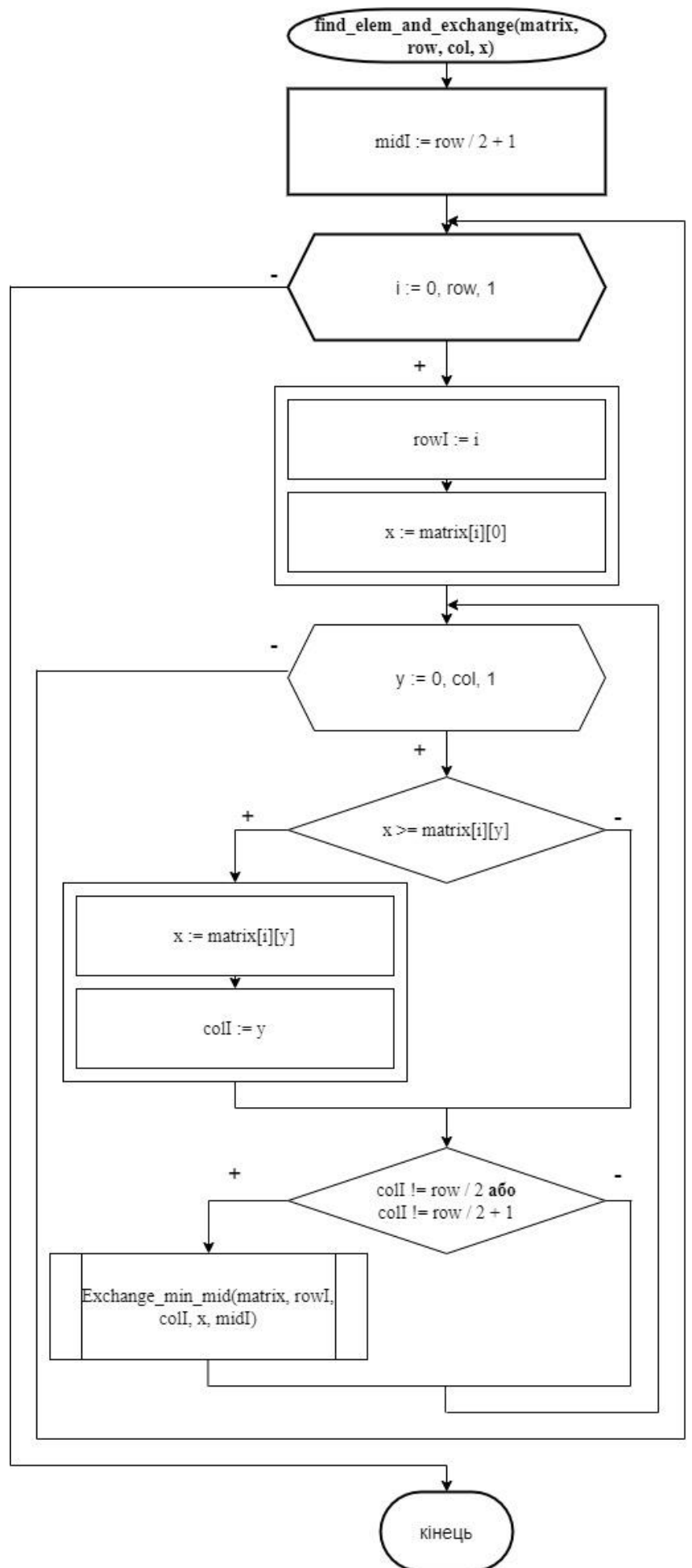
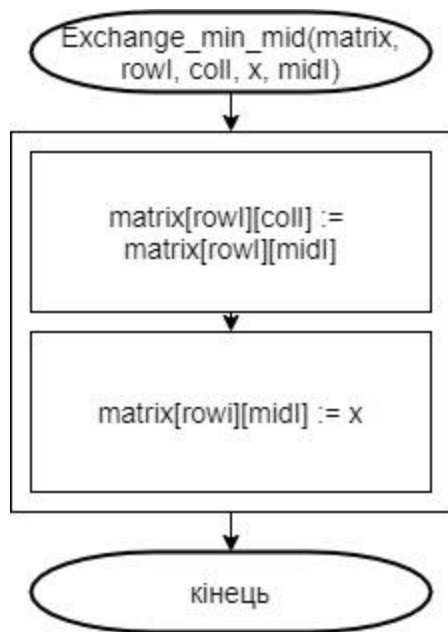
## Основна програма:



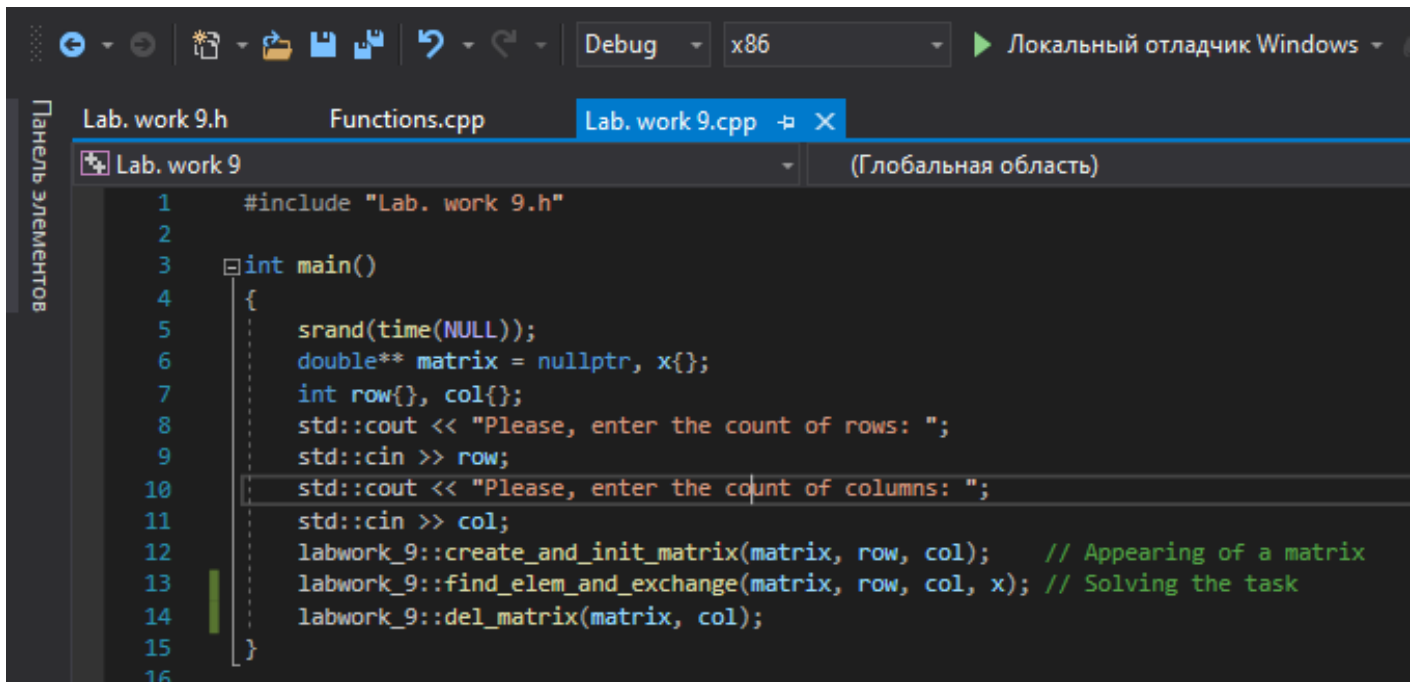
## Підпрограми:



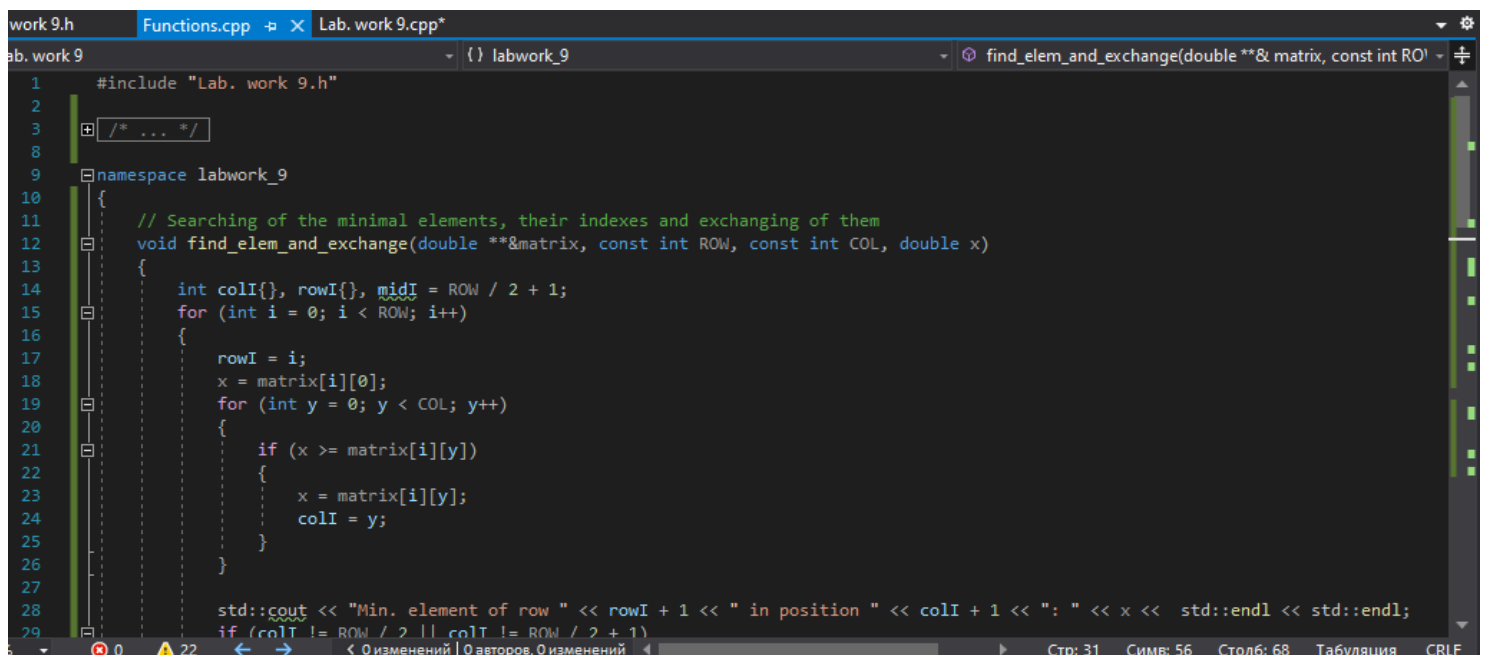




## 5. Код программы



```
1  #include "Lab. work 9.h"
2
3  int main()
4  {
5      srand(time(NULL));
6      double** matrix = nullptr, x{};
7      int row{}, col{};
8      std::cout << "Please, enter the count of rows: ";
9      std::cin >> row;
10     std::cout << "Please, enter the count of columns: ";
11     std::cin >> col;
12     labwork_9::create_and_init_matrix(matrix, row, col); // Appearing of a matrix
13     labwork_9::find_elem_and_exchange(matrix, row, col, x); // Solving the task
14     labwork_9::del_matrix(matrix, col);
15 }
16
```



```
1  #include "Lab. work 9.h"
2
3  /* ... */
4
5  namespace labwork_9
6  {
7      // Searching of the minimal elements, their indexes and exchanging of them
8      void find_elem_and_exchange(double **&matrix, const int ROW, const int COL, double x)
9      {
10         int colI{}, rowI{}, midI = ROW / 2 + 1;
11         for (int i = 0; i < ROW; i++)
12         {
13             rowI = i;
14             x = matrix[i][0];
15             for (int y = 0; y < COL; y++)
16             {
17                 if (x >= matrix[i][y])
18                 {
19                     x = matrix[i][y];
20                     colI = y;
21                 }
22             }
23         }
24
25         std::cout << "Min. element of row " << rowI + 1 << " in position " << colI + 1 << ": " << x << std::endl << std::endl;
26         if (colI != ROW / 2 || colI != ROW / 2 + 1)
27         {
28             // ...
29         }
30     }
31 }
```

```
69     template<typename T>
70     void print_arr(T**matrix, const int ROW, const int COL)
71     {
72         std::cout << "Here is a matrix:" << std::endl;
73         for (int i = 0; i < ROW; i++)
74         {
75             for (int y = 0; y < COL; y++)
76             {
77                 std::cout << std::setw(10) << matrix[i][y];
78             }
79             std::cout << std::endl;
80         }
81         std::cout << std::endl << std::endl;
82     }
83
84     template<typename T>
85     void print_arr(T* sequence, const int SIZE)
86     {
87         std::cout << "Here is a sequence:" << std::endl;
88         for (int i = 0; i < SIZE; i++)
89         {
90             std::cout << sequence[i] << " ";
91         }
92         std::cout << std::endl << std::endl;
93     }
94 }
```

```
work 9.h  Functions.cpp  Lab. work 9.cpp*
b. work 9  {} labwork_9  find_elem_and_exchange(double **& matrix, co
29  if (colI != ROW / 2 || colI != ROW / 2 + 1)
30  {
31      Exchange_min_mid(matrix, ROW, rowI, colI, x, midI);
32      print_arr(matrix[rowI], COL);
33  }
34  else
35      std::cout << "The element is already in the mid of the row" << std::endl << std::endl;
36  }
37  std::cout << "The final matrix is: " << std::endl;
38  print_arr(matrix, ROW, COL);
39  }
40
41  // Exchanging two elements of the row
42  void Exchange_min_mid(double **&matrix, const int ROW, const int ROWI, const int COLI, const double X, const int MID_I)
43  {
44      matrix[ROWI][COLI] = matrix[ROWI][MID_I];
45      matrix[ROWI][MID_I] = X;
46  }
47
48  // Creating of the matrix and its initialization
49  void create_and_init_matrix(double **&matrix, const int ROW, const int COL)
50  {
51      matrix = new double* [ROW];
52      for (int i = 0; i < ROW; i++)
53      {
```

```
Lab. work 9.h  Functions.cpp  Lab. work 9.cpp*
Lab. work 9  {} labwork_9  find_elem_and_exchange(double **& matrix, con
53  {
54      matrix[i] = new double[COL]{};
55  }
56
57  double temp{};
58  for (int i = 0; i < ROW; i++)
59  {
60      for (int y = 0; y < COL; y++)
61      {
62          temp = rand() % 3;
63          matrix[i][y] = (temp == 2) ? rand() % 1000 * 2 - 1000 : (temp == 1) ? (rand() % 1000 * 2 - 1000) / 12.4 : 0;
64      }
65      print_arr(matrix, ROW, COL);
66  }
67
68
69  template<typename T>
70  void print_arr(T** matrix, const int ROW, const int COL) { ... }
83
84  template<typename T>
85  void print_arr(T* sequence, const int SIZE) { ... }
94
```

```
Lab. work 9.h  Functions.cpp  Lab. work 9.cpp*
Lab. work 9  {} labwork_9  print_arr<T>(T* sequence, const int SIZE)
4  #include <ctime>
5  #include <cstdlib>
6  #include <iomanip>
7
8  namespace labwork_9
9  {
10     void find_elem_and_exchange(double**& matrix, const int ROW, const int COL, double x);
11     void Exchange_min_mid(double**& matrix, const int ROW, const int ROWI, const int COLI, const double X, const int MID_I);
12     void create_and_init_matrix(double**& matrix, const int ROW, const int COL);
13     template<typename T>
14     void print_arr(T** matrix, const int ROW, const int COL);
15     template<typename T>
16     void print_arr(T* sequence, const int SIZE);
17 }
```

```
Lab. work 9.h  Functions.cpp  Lab. work 9.cpp
Lab. work 9
88     for (int i = 0; i < SIZE; i++)
89     {
90         std::cout << sequence[i] << " ";
91     }
92     std::cout << std::endl << std::endl;
93 }
94
95 template<typename T>
96 void del_matrix(T** matrix, const int COL)
97 {
98     for (int i = 0; i < COL; i++)
99     {
100         delete[] matrix[i];
101     }
102     delete[] matrix;
103 }
104 }
```

## Тестування

```
Консоль отладки Microsoft Visual Studio
Please, enter the count of rows: 3
Please, enter the count of columns: 10
Here is a matrix:
    0 -56.7742  10.9677    42  48.871    0    0    0    0    0
    0 -62.5806   940 -2.09677    0    0   672    0 -39.5161  356
  63.871   470   718    0    0 -79.5161  622  9.83871    0 -882

Min. element of row 1 in position 2: -56.7742

Here is a sequence:
0 0 10.9677 42 48.871 -56.7742 0 0 0 0

Min. element of row 2 in position 2: -62.5806

Here is a sequence:
0 0 940 -2.09677 0 -62.5806 672 0 -39.5161 356

Min. element of row 3 in position 10: -882

Here is a sequence:
63.871 470 718 0 0 -882 622 9.83871 0 -79.5161

The final matrix is:
Here is a matrix:
    0    0  10.9677    42  48.871 -56.7742    0    0    0    0
    0    0   940 -2.09677    0 -62.5806   672    0 -39.5161  356
  63.871   470   718    0    0   -882   622  9.83871    0 -79.5161
```

## **6. В и с н о в о к**

На цій лабораторній роботі я дослідив алгоритми обходу масивів та набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Мені довелося працювати з елементами матриці: використання пошуку, порівнянь та перестановок.