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\_**20**\_\_

Виконав студент: **ІП-11 Лошак Віктор Іванович**

(шифр, прізвище, ім'я, по батькові)

Перевірив: **Мартинова О.П.**

Київ 2021

**Лабораторна робота**

**Дослідження алгоритмів обходу масивів**

**Мета:**

дослідити алгоритми обходу масивів, набути практичних

навичок використання цих алгоритмів під час складання програмних

специфікацій.

**Варіант №20**

**Задача:**

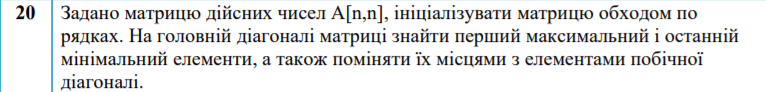
Розробити алгоритм та написати програму, яка складається з наступних дій:

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).

**Варіант**:



**Розв’язок:**

1)Постановка задачі

Результатом розв’язку є видозмінена матриця значення яка отримується

шляхом перестановки знайдених елементів матриці що відповідають заданій умові( є найменшим і найбільшим елементом головної діагоналі) . Для ініціалізації матриці випадковими значеннями використаємо підпрограми що будуть реалізувати присвоєння значень елементам послідовностей всередині арифметичного циклу, що здійснюватиме обхід змійкою по рядках. Для пошуку заданих елементів використаємо алгоритм лінійного пошуку в матриці. Даний алгоритм можна реалізувати використовуючи арифметичний цикл, що йтиме по головній діагоналі заданої матриці, перевіряючи відповідність за допомогою логічного оператора, і у випадку успіху записуючи значення та індекс шуканого елемента в змінну . Попередньо ініціалізуємо arr3 нулями при створенні змінної для забезпечення правильної роботи алгоритму. Для реалізації алгоритму перестановки елементів головної та побічної діагоналі використаємо підпрограму що буде користуватись додаткової змінною для виконання заміни елементів. Значенням n є число що вводиться користувачем за запитом на початку програми.

2)Побудова математичної моделі

Складемо таблицю імен змінних

|  |  |  |  |
| --- | --- | --- | --- |
| Змінна | Тип | Ім’я | Призначення |
| Внутрішня змінна функції main/ параметр функцій | індексований тип з покажчиків цілого типу(подвійний масив інт) | matrix | Початкове дане/ результат обчислень |
| Внутрішня змінна функції findAndSwapEl | Цілочисельний тип | ValueMax | Проміжний результат\ значення макс елемента |
| Внутрішня змінна функції findAndSwapEl | Цілочисельний тип | ValueMin | Проміжний результат\ значення мін елемента |
| Внутрішня змінна функції findAndSwapEl | Цілочисельний тип | coordMax | Проміжний результат\координата макс елемента |
| Внутрішня змінна функції findAndSwapEl | Цілочисельний тип | coordMin | Проміжний результат\координата мін елемента |
| Параметр newMatrix, | Цілочисельний тип | rowAmount | Проміжний результат |
| Параметр newMatrix, | Цілочисельний тип | columnAmount | Проміжний результат |
| Рахівник зовнішнього/ внутрішнього/простого арифметичного циклу | Цілочисельний тип | і, j | Проміжний результат |
| Внутрішня змінна функції swapEl | Цілочисельний тип | temp | Проміжний результат |
| Внутрішня змінна функції newMatrix, | індексований тип з покажчиків цілого типу(подвійний масив інт) | arr | Проміжний результат |

3)Програмні специфікації напишемо у псевдокоді та графічній формі у вигляді блок-схеми.

*Крок 1:* Визначимо основні дії.

*Крок 1:* Деталізуємо дію користувацького вводу та виклику функцій

5)Псевдокод

*Крок 1*

**Початок**

**-** ввід значення n

-ініціалізаціяmatrix

- виклик функцій для виведення матриці та знаходження і перестановки елементів місцями

**Кінець**

*Крок 2*

**Початок**

**- ввід** n

-matrix =newMatrix(n, n)

- printMatrix(matrix, n, n)

- findAndSwapEl(matrix, n)

- printMatrix(matrix, n, n)

**Кінець**

*Підпрограми:*

*Крок 3*

**Початок** newMatrix(rowAmount, columnAmount)

**- ініціалізація** arr[rowAmount] **динамічно**

**- повторити для** і=0 **з кроком** 1 **поки** і<rowAmount

**ініціалізація** arr[i]:=new int [columnAmount] **динамічно**

**-повторити для** j=0 **з кроком** 1 **поки** j<columnAmount

arr [i][j] **:** **=** (rand( ) % (rowAmount \* columnAmount)) - (rowAmount \* columnAmount / 2 - 1)

**все повторити**

**все повторити**

**-повернути** arr

**Кінець** newMatrix(rowAmount, columnAmount)

*Крок 4*

**Початок** printMatrix( matrix, rowLength, columnLength)

**- повторити для** і=0 **з кроком** 1 **поки** і<rowAmount

**-повторити для** j=0 **з кроком** 1 **поки** j<columnAmount

**Вивід** matrix[i][j]

**все повторити**

**все повторити**

**-повернути** arr

**Кінець** printMatrix( matrix, rowLength, columnLength)

*Крок 5*

**Початок** findAndSwapEl(matrix, n)

-**ініціалізація** valueMax=matrix[0][0], valueMin = matrix[0][0]

coordMax=0

coordMin=0

**- повторити для** і=0 **з кроком** 1 **поки** і< n

**- якщо** matrix[i][i] > valueMax **то**

valueMax = matrix[i][i];

coordMax = i;

**-інакше якщо** matrix[i][i] < valueMin **то**

valueMin = matrix[i][i];

coordMin = i;

**все якщо**

**все повторити**

**-виклик** swapEl(matrix, coordMax, n)

- **виклик** swapEl(matrix, coordMin, n)

**Кінець** findAndSwapEl

*Крок 6*

**Початок** swapEl(matrix, coord, n)

**- ініціалізація** temp = matrix[coord][coord]

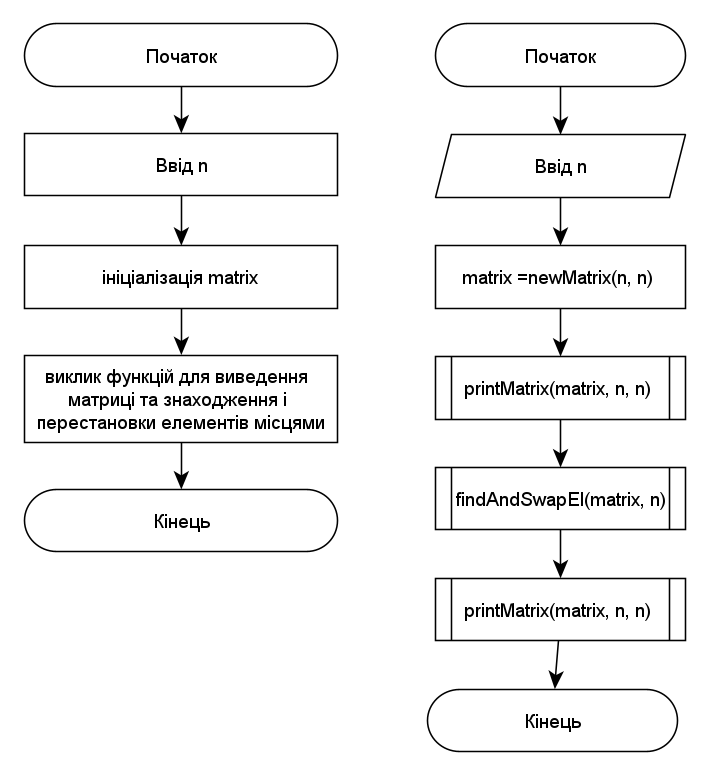
- **присвоєння** matrix[coord][coord] = matrix[coord][n - coord-1]

- **присвоєння** matrix[coord][n - coord-1] = temp

**Кінець** swapElements( arr, ind1, ind2)

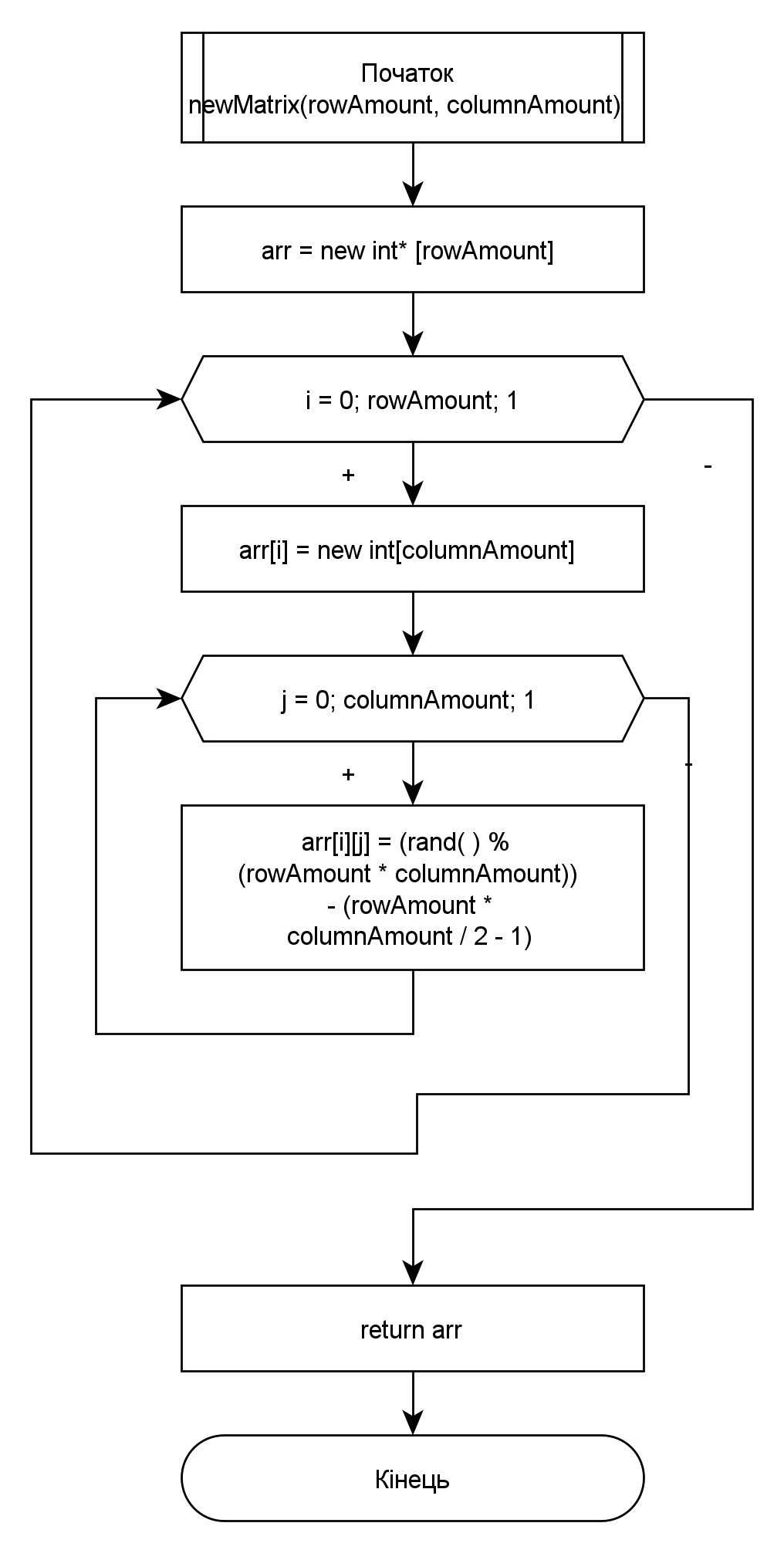
**Блок-Схема:**

**Крок 1 Крок 2**

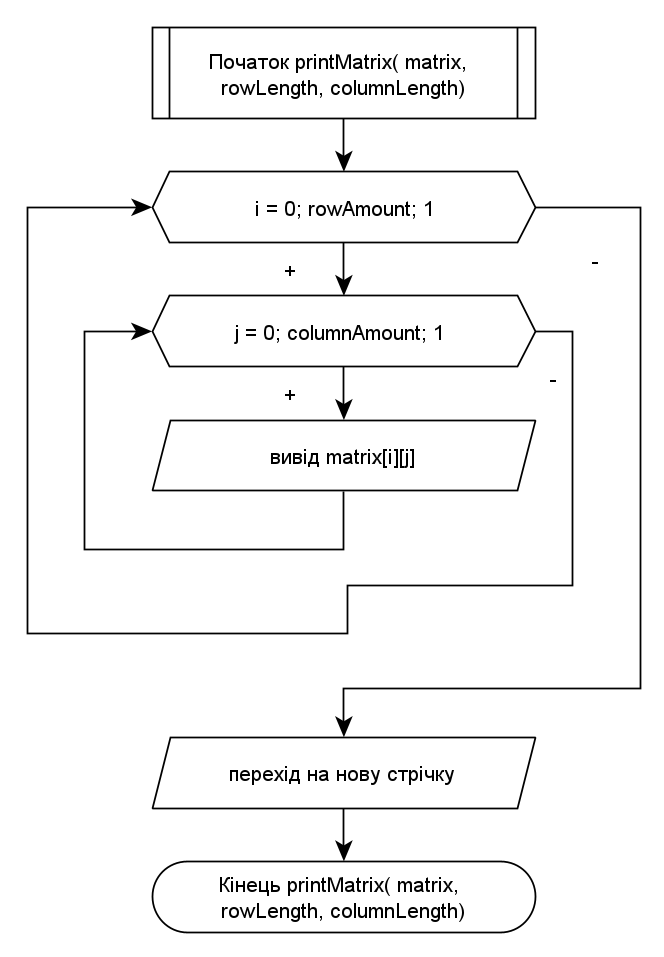
****

**Підпрограми:**

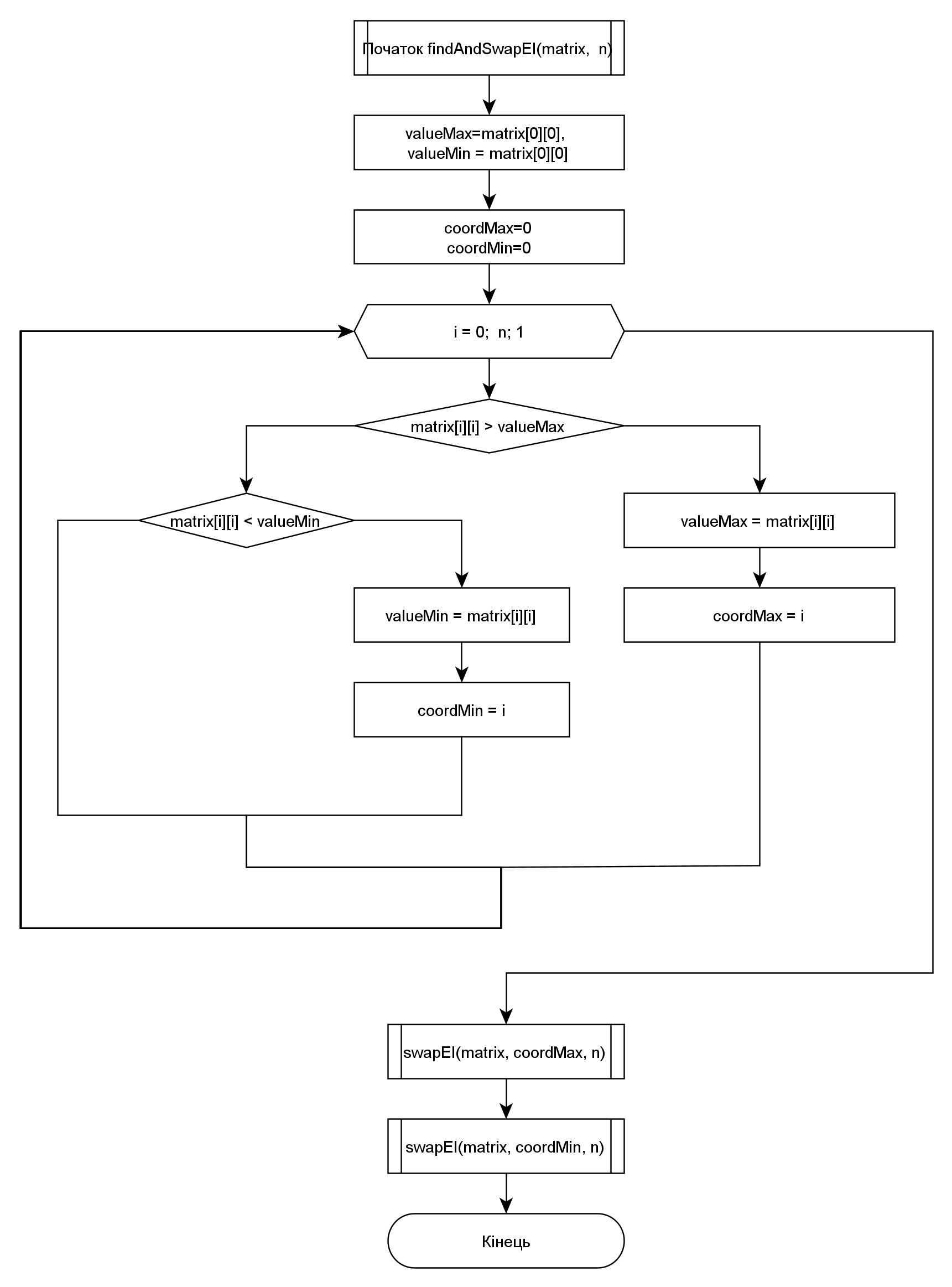
**Крок 3**

****

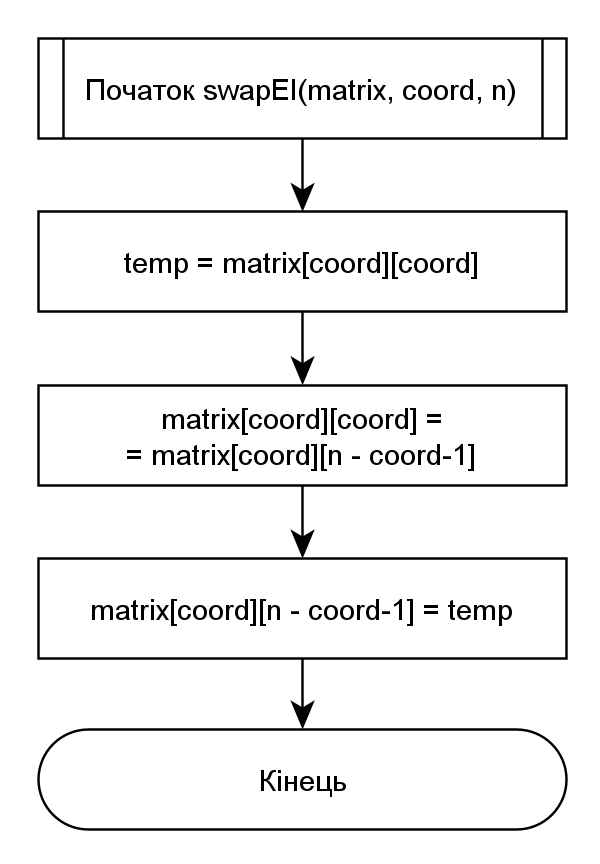
**Крок 4**



**Крок 5**

****

**Крок 6**

****

**Код програми c++:**

#include <iostream>

#include <ctime>

#include <cmath>

#include <iomanip>

using namespace std;

int\*\* newMatrix(int rowAmount, int columnAmount);

void printMatrix(int\*\* matrix, int rowLength, int columnLength);

void findAndSwapEl(int\*\* matrix, int n);

void swapEl(int\*\* matrix, int coord, int n);

int main( )

{

srand(time(0));

rand( );

int n;

std::cout << "Enter the size of square matrix: ";

cin >> n;

int\*\* matrix = newMatrix(n, n);

printMatrix(matrix, n, n);

findAndSwapEl(matrix, n);

printMatrix(matrix, n, n);

return 0;

}

void findAndSwapEl(int\*\* matrix, int n)

{

int valueMax=matrix[0][0], valueMin = matrix[0][0];

int coordMax=0;

int coordMin=0;

for (int i = 0; i < n; ++i)

{

if (matrix[i][i] > valueMax)

{

valueMax = matrix[i][i];

coordMax = i;

}

else if (matrix[i][i] < valueMin)

{

valueMin = matrix[i][i];

coordMin = i;

}

}

swapEl(matrix, coordMax, n);

swapEl(matrix, coordMin, n);

}

void swapEl(int\*\* matrix, int coord, int n)

{

int temp = matrix[coord][coord];

matrix[coord][coord] = matrix[coord][n - coord-1];

matrix[coord][n - coord-1] = temp;

}

int\*\* newMatrix(int rowAmount, int columnAmount)

{

//creating an empty matrix of given parameters

int\*\* arr = new int\* [rowAmount];

int j = 0;

for (int i = 0; i < rowAmount; ++i)

{

arr[i] = new int[columnAmount];

for (; (i % 2 == 0) ? j < columnAmount : j >= 0; (i % 2 == 0) ? ++j : --j)

{

arr[i][j] = (rand( ) % (rowAmount \* columnAmount)) - (rowAmount \* columnAmount / 2 - 1);

}

(i % 2 == 0) ? --j : ++j;

}

return arr;

}

void printMatrix(int\*\* matrix, int rowLength, int columnLength)

{

for (int i = 0; i < rowLength; ++i)

{

for (int j = 0; j < columnLength; ++j)

{

cout << setw(4) << matrix[i][j];

}

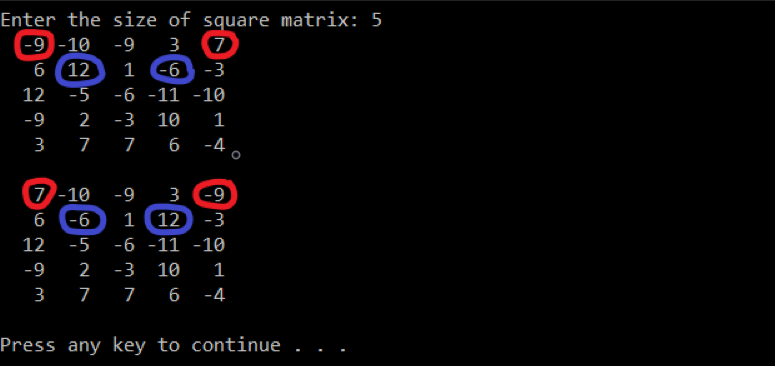
cout << '\n';

}

cout << '\n';

}

Результат запуску програми для перевірки:



|  |  |
| --- | --- |
| Блок | Дія |
|  | Початок |
| 1 | Ввід n та Ініціалізація матриці випадковими значеннями обходом по рядках змійкою всередині newMatrix |
| 2 | Перша ітерація пошуку:  valueMax=matrix[0][0]=-9,  valueMin = matrix[0][0]=-9;  coordMax=0;  coordMin=0 |
| 3 | 2 ітерація пошуку:  valueMax=matrix[0][0]=-9,  valueMin = matrix[0][0]=12;  coordMax=1;  coordMin=0 |
| 4 | 3 ітерація пошуку:  valueMax=matrix[0][0]=-9,  valueMin = matrix[0][0]=12;  coordMax=1;  coordMin=0 |
| 5 | 4 ітерація пошуку:  valueMax=matrix[0][0]=-9,  valueMin = matrix[0][0]=12;  coordMax=1;  coordMin=0 |
| 6 | 5 ітерація пошуку:  valueMax=matrix[0][0]=-9,  valueMin = matrix[0][0]=12;  coordMax=1;  coordMin=0 |
| 7 | Перестановка  matrix[0][0] = matrix[0][5 - 0-1]=7;  matrix[0][5-0-1] = -9; |
| 8 | Перестановка  matrix[0][0] = matrix[0][5 - 0-1]=7;  matrix[0][5-0-1] = -9; |
| 9 | Вивід результату обчислення(матриці результату) |
|  | Кінець |

**Висновок:**

В ході виконання даної лабораторної роботи я навчився використовувати алгоритми обходу масивів для ініціалізації і аналізу структур даних, набув практичних навичок роботи з подвійними масивами. Навчився застосовувати лінійний пошук при пошуку елементів в багатовимірних структурах даних. Навчився створювати блок-схеми алгоритмів що містить обхід змійкою та описувати програму за допомогою псевдокоду. Використовуючи отримані знання я успішно реалізував алгоритм що виконує заміну елементів в головній та побічних діагоналях, на компільованій мові с++.