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 28

Виконав студент ІП-12, Сімчук Андрій Володимирович

(шифр, прізвище, ім'я, по батькові)

Перевірив \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

( 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.

**I. Задача.**

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

1. Опису та ініціювання змінної індексованого типу (двовимірний масив) m х n (дійсний тип даних).

2. Для кожного стовпчика матриці знайти перший мінімальний елемент Х і його місцезнаходження.

3. Обміняти знайдене значення Х з елементом першого рядка.

**II. Розв’язання.**

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

*Крок 1.* Визначимо основні дії.

*Крок 2.* Деталізація ініціалізації змінної індексованого типу (двовимірного масиву).

*Крок 3.* Деталізуємо дію заповнення двовимірного масиву.

*Крок 4.* Деталізуємо дії знаходження першого мінімального елемента Х і його місцезнаходження та обміну знайденого значення Х з елементом першого рядка.

*Крок 5.* Деталізуємо дію виводу зміненої матриці.

***Побудова математичної моделі***

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім’я** | **Призначення** |
| Двовимірний масив розміром  m х n | Індексований | A | Початкове значення, результат |
| Кількість рядків двовимірного масиву | Натуральний | m | Початкове значення |
| Кількість стовпчиків двовимірного масиву | Натуральний | n | Початкове значення |
| Додаткова змінна | Дійсний | additional | Проміжне значення |
| Показник лічильнику | Цілий | i | Проміжне значення |
| Показник лічильнику | Цілий | j | Проміжне значення |
| Перший мінімальний елемент стовпчика двовимірного масиву Х | Дійсний | X | Проміжне значення, результат |

***Функції:***

fill\_matrix() – функція заповнення двовимірного масиву;

first\_minimum\_and\_exchange() – функція знаходження першого мінімального елемента Х і його місцезнаходження та обміну знайденого значення Х з елементом першого рядка;

output\_matrix() – функція виводу матриці;

***Псевдокод (основна програма):***

*Крок 1.*

**Початок**

**Введення m, n**

ініціалізація змінної індексованого типу (двовимірного масиву)

заповнення двовимірного масиву

знаходження першого мінімального елемента Х і його місцезнаходження та

обмін знайденого значення Х з елементом першого рядка

виведення зміненої матриці

**Кінець**

*Крок 2.*

**Початок**

**Введення m, n**

А[m][n]

заповнення двовимірного масиву

знаходження першого мінімального елемента Х і його місцезнаходження та

обмін знайденого значення Х з елементом першого рядка

виведення зміненої матриці

**Кінець**

*Крок 3.*

**Початок**

**Введення m, n**

А[m][n]

fill\_matrix(A[][], m, n)

знаходження першого мінімального елемента Х і його місцезнаходження та

обмін знайденого значення Х з елементом першого рядка

виведення зміненої матриці

**Кінець**

*Крок 4.*

**Початок**

**Введення m, n**

А[m][n]

fill\_matrix(A[][], m, n)

first\_minimum\_and\_exchange(A[][], m, n)

виведення зміненої матриці

**Кінець**

*Крок 5.*

**Початок**

**Введення m, n**

А[m][n]

fill\_matrix(A[][], m, n)

first\_minimum\_and\_exchange(A[][], m, n)

output\_matrix(A[][], m, n)

**Кінець**

***Псевдокод (функція fill\_matrix):***

**fill\_matrix(A[][], m, n)**

**повторити**

**для** і **від** 0 **до** m **із кроком** 1

**повторити**

**для** j **від** 0 **до** n **із кроком** 1

**Введення A[i][j]**

**все повторити**

**все повторити**

**Кінець**

***Псевдокод (функція*** ***first\_minimum\_and\_exchange):***

**first\_minimum\_and\_exchange(A[][], m, n)**

X = 1000000

additional = 0

**повторити**

**для** j **від** 0 **до** n **із кроком** 1

**повторити**

**для** i **від** 0 **до** m **із кроком** 1

**якщо** А[i][j] < X

**то**

X = A[i][j]

**все якщо**

**все повторити**

**повторити**

**для** i **від** 0 **до** m **із кроком** 1

**якщо** А[i][j] = X

**то**

**Виведення A[i][j], i, j**

additional = A[0][j]

A[0][j] = A[i][j]

A[i][j] = additional

**все якщо**

**все повторити**

X = 1000000

**все повторити**

**Кінець**

***Псевдокод (функція*** ***output\_matrix):***

**output\_matrix(A[][], m, n)**

**повторити**

**для** і **від** 0 **до** m **із кроком** 1

**повторити**

**для** j **від** 0 **до** n **із кроком** 1

**Виведення A[i][j]**

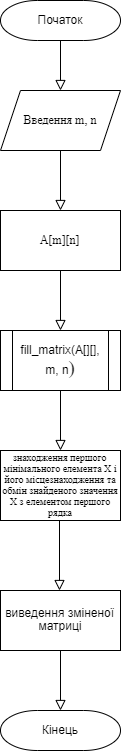
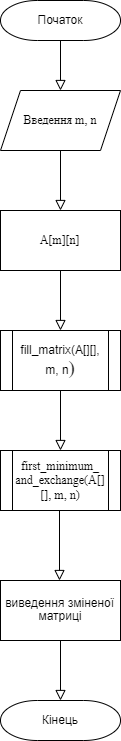
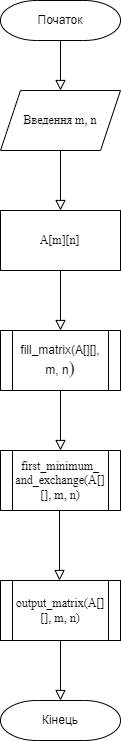
**все повторити**

**все повторити**

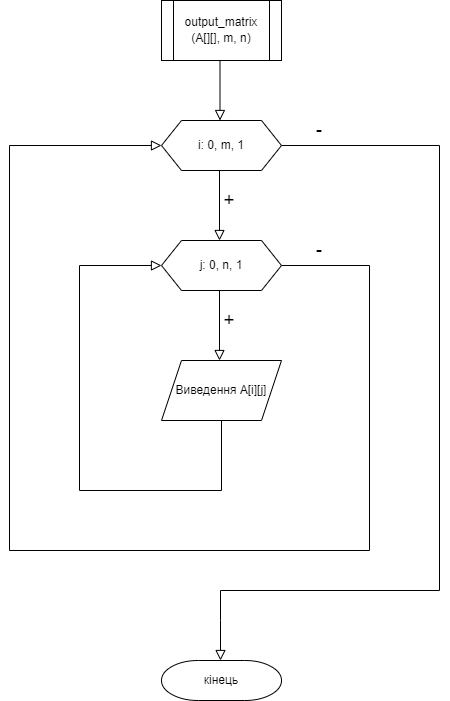
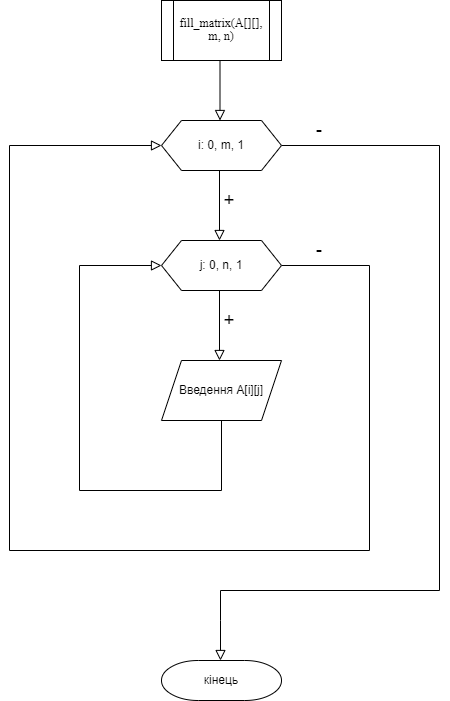
**Кінець**

***Блок-схема (Основна програма):***

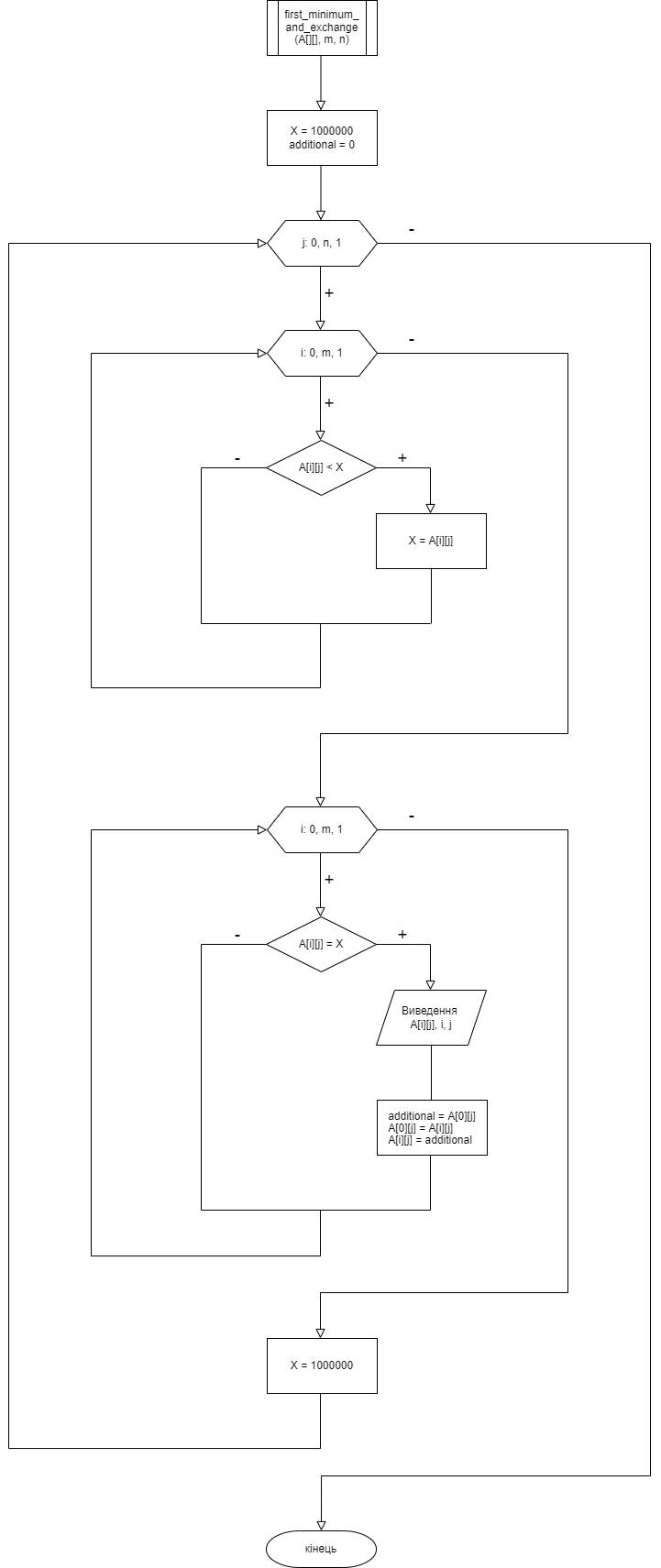
*Крок 1. Крок 2. Крок 3. Крок4. Крок 5.*



***Блок-схема (функція fill\_matrix): Блок-схема (функція output\_matrix):***



***Блок-схема (функція first\_minimum\_and\_exchange):***



***Код програми на мові С++***

#include <iostream>

#include <math.h>

#include <stdio.h>

#include <ctime>

#include <iomanip>

using namespace std;

void fill\_matrix(double\*\*, int, int);

void first\_minimum\_and\_exchange(double\*\*, int, int);

void output\_matrix(double\*\*, int, int);

int main() {

srand((double)time(NULL));

int m, n;

cout << "Enter the value of m: ";

cin >> m;

cout << "Enter the value of n: ";

cin >> n;

double\*\* A = new double\* [m];

for (int i = 0; i < m; i++) A[i] = new double[n];

fill\_matrix(A, m, n);

cout << "Your matrix:" << endl;

output\_matrix(A, m, n);

first\_minimum\_and\_exchange(A, m, n);

cout << "New matrix:" << endl;

output\_matrix(A, m, n);

for (int i = 0; i < m; i++) {

delete[] A[i];

}

delete[] A;

system("pause");

return 0;

}

void fill\_matrix(double\*\* A, int m, int n) {

for (int i = 0; i < m; i++) {

for (int j = 0; j < n; j++) {

A[i][j] = (rand() % 1000) \* pow(-1, rand() % 1000);

A[i][j] += (rand() % 1000) / 1000.000;

}

}

}

void first\_minimum\_and\_exchange(double\*\* A, int m, int n) {

double X = 1000000.000, additional = 0;

for (int j = 0; j < n; j++) {

for (int i = 0; i < m; i++) {

if (A[i][j] < X) {

X = A[i][j];

}

}

for (int i = 0; i < m; i++) {

if (A[i][j] == X) {

cout << "First minimum element of the column " << j + 1 << ": " << A[i][j] << endl;

cout << "Its location: " << i << " " << j << endl;

additional = A[0][j];

A[0][j] = A[i][j];

A[i][j] = additional;

}

}

X = 10000.000;

}

cout << endl;

}

void output\_matrix(double\*\* A, int m, int n) {

for (int i = 0; i < m; i++) {

for (int j = 0; j < n; j++) {

cout << setw(9) << A[i][j];

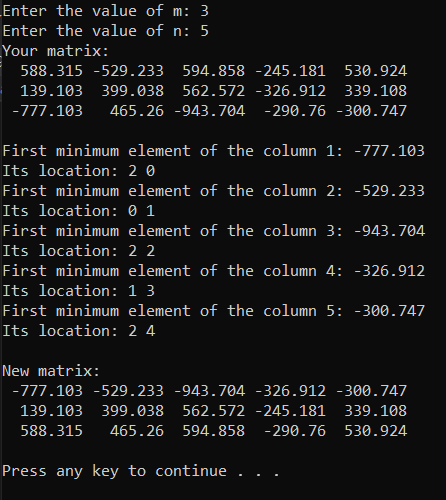
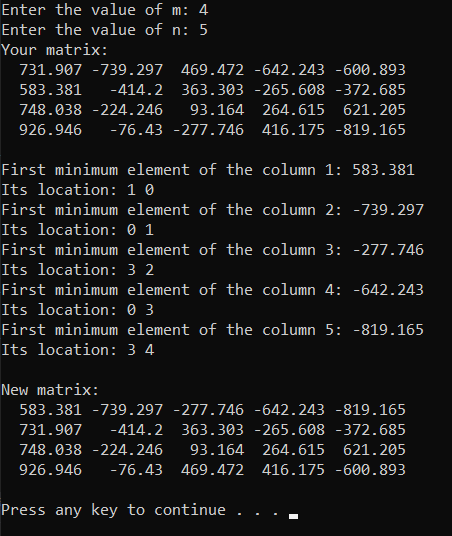
}

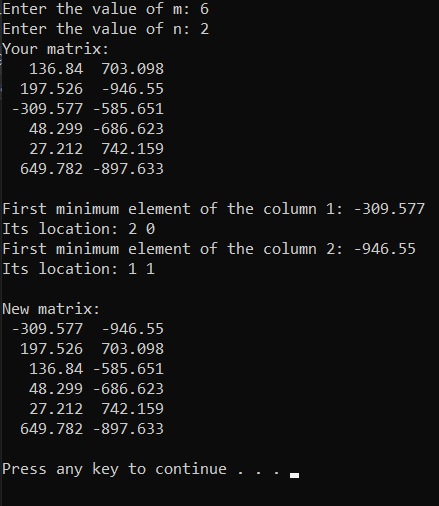
cout << endl;

}

cout << endl;

}





**IV. Висновки.**

Було досліджено алгоритми обходу масивів (зокрема алгоритм обходу матриці по стовпцям) та було набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій, створивши алгоритм для пошуку мінімального елемента Х кожного стовпчика (та його місцезнаходження) й обміну його з відповідним елементом першого рядка матриці.