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-11 Сідак Кирил Ігорович
(шифр, прізвище, ім'я, по батькові)

Перевірив Мартинова Оксана Петрівна
(прізвище, ім'я, по батькові)

Київ 2021

Лабараторна робота№9

Дослідження алгоритмів обходу масивів

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

Індивідуальне завдання:

Варіант 28

Задано матрицю дійсних чисел A[m,n]. У кожному стовпчику матриці знайти перший мінімальний елемент X і його місцезнаходження. Обміняти знайдене значення X з елементом першого рядка.

Постановка задачі

За допомого підпрограми треба створити двовимірний масив заданої розмірності(m x n) та заповнити його випадковими дійсними числами із заданого проміжку. Потім за допомогою іншої підпрограми знайти мінімальний елемент кожного стовпця та його індекс й обміняти його з елементом цього ж стовпця, який належить першому рядку.

Побудова математичної моделі

Складемо таблицю змінних

Змінна	Тип	Ім’я	Призначення
Кількість рядків матриці	цілий	m	Вхідне дане
Кількість стовпців матриці	цілий	n	Вхідне дане
Задана матриця	дійсний	matrix	Вхідне дане/результат
Мінімальний елемент поточного стовпця	дійсний	temp_num	Проміжне дане(змінна підпрограми)
Індекс мінімального елемента поточного стовпця	цілий	index	Проміжне дане(змінна підпрограми)

Таким чином, формування задачі зводиться до створення двовимірного масиву matrix m x n та заповнення його випадковими дійсними числами із заданого проміжку за допомогою підпрограми generate_matrix, вивдення цбого масиву за допомогою підпрограми display_matrix та знаходження мінімального

елементу в кожному стовпці масиву й обміну його з елементом цього ж стовпця першого рядка, використовуючи підпрограму find_min_column. Результуючий масив matrix треба вивести підпрограмою display_matrix.

Розв’язання

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

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію створення та заповнення двовимірного масиву дійсних чисел.

Крок 3. Деталізуємо дію виведення двовимірного масиву.

Крок 4. Деталізуємо дію знаходження мінімального елемента стовпця та його обміну з відповідним елементом першого рядка.

Псевдокод

Основна програма

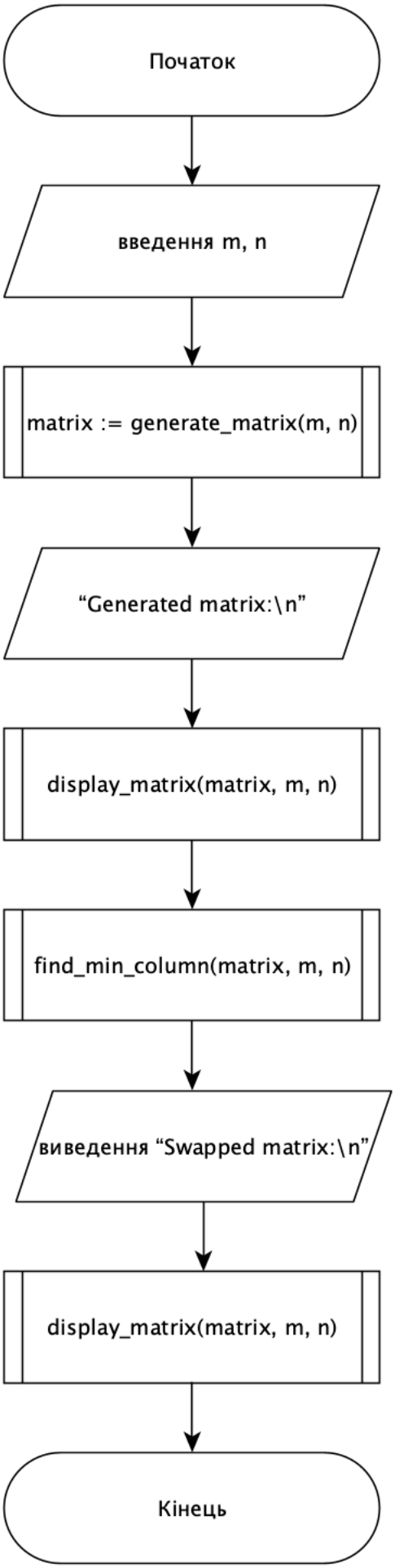
```
Початок
        введення m, n
        matrix := generate_matrix(m, n)
        виведення “Generated matrix:\n”
        display_matrix(matrix, m, n)
        find_min_column(matrix, m, n)
        виведення “Swapped matrix:\n”
        display_matrix(matrix, m, n)
Кінець
```

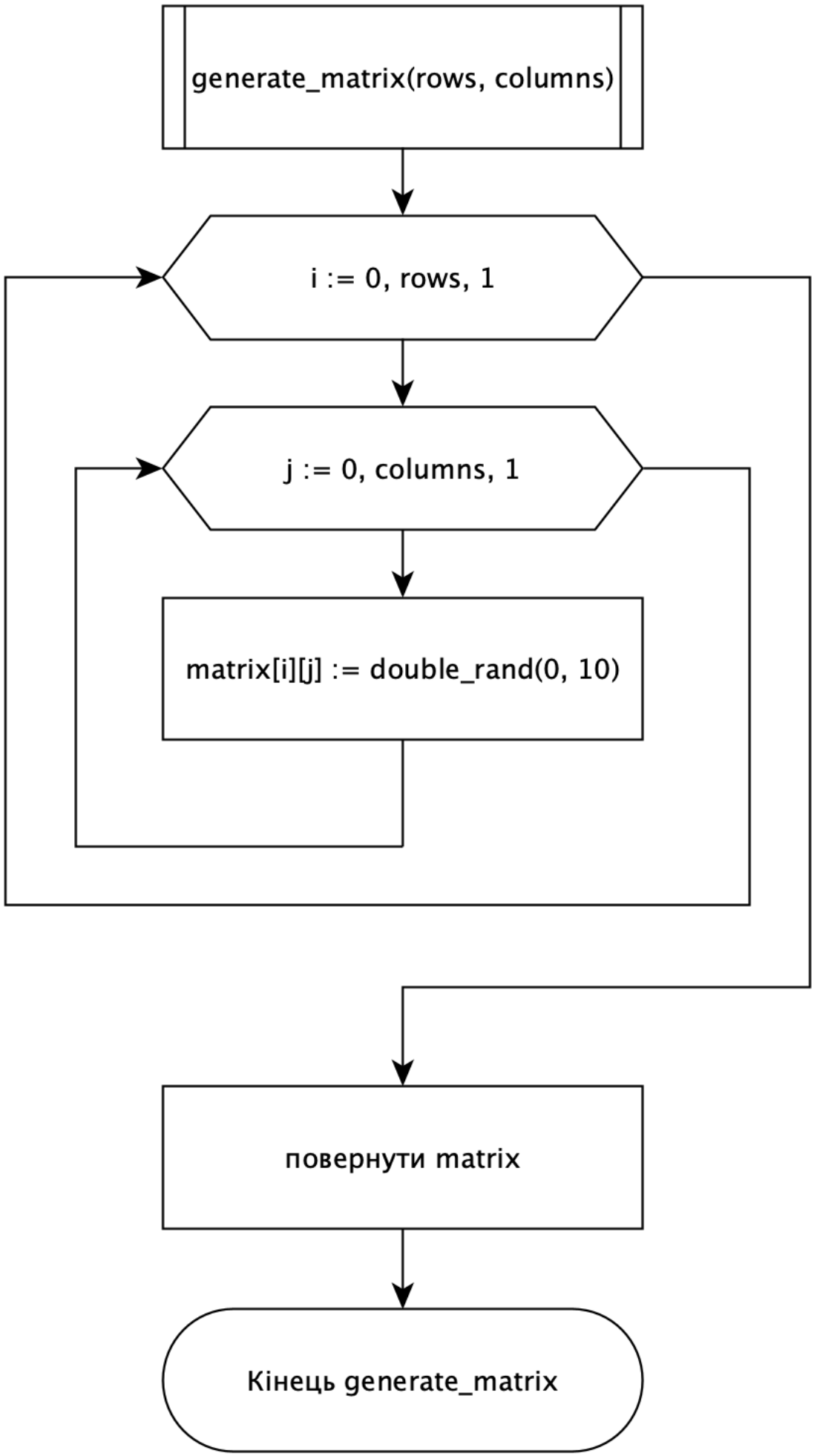
Підпрограми

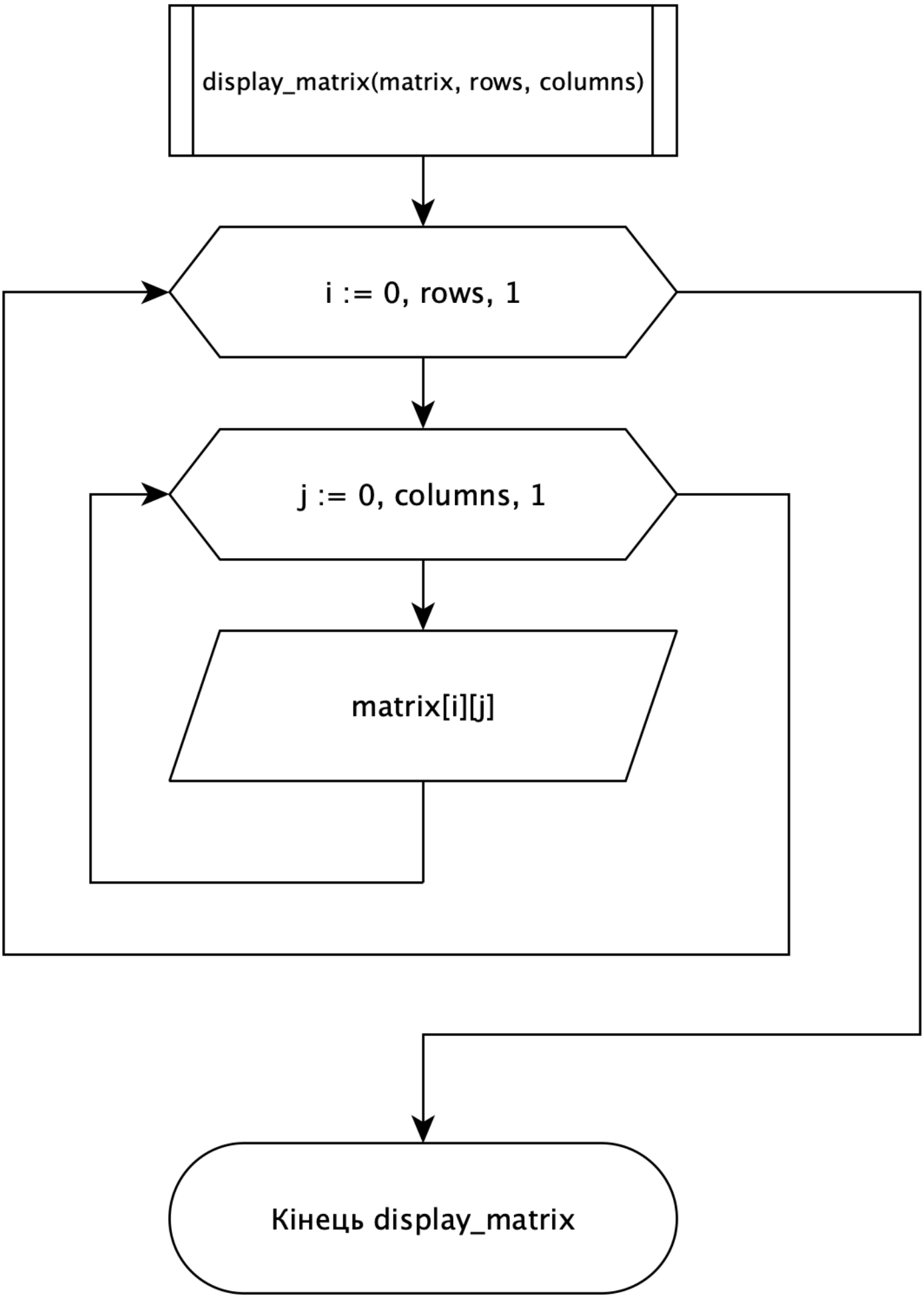
```
generate_matrix(rows, columns)
        повторити для i від 0 до rows
          повторити для j від 0 до columns
                matrix[i][j] := double_rand(0, 10)
           все повторити
```

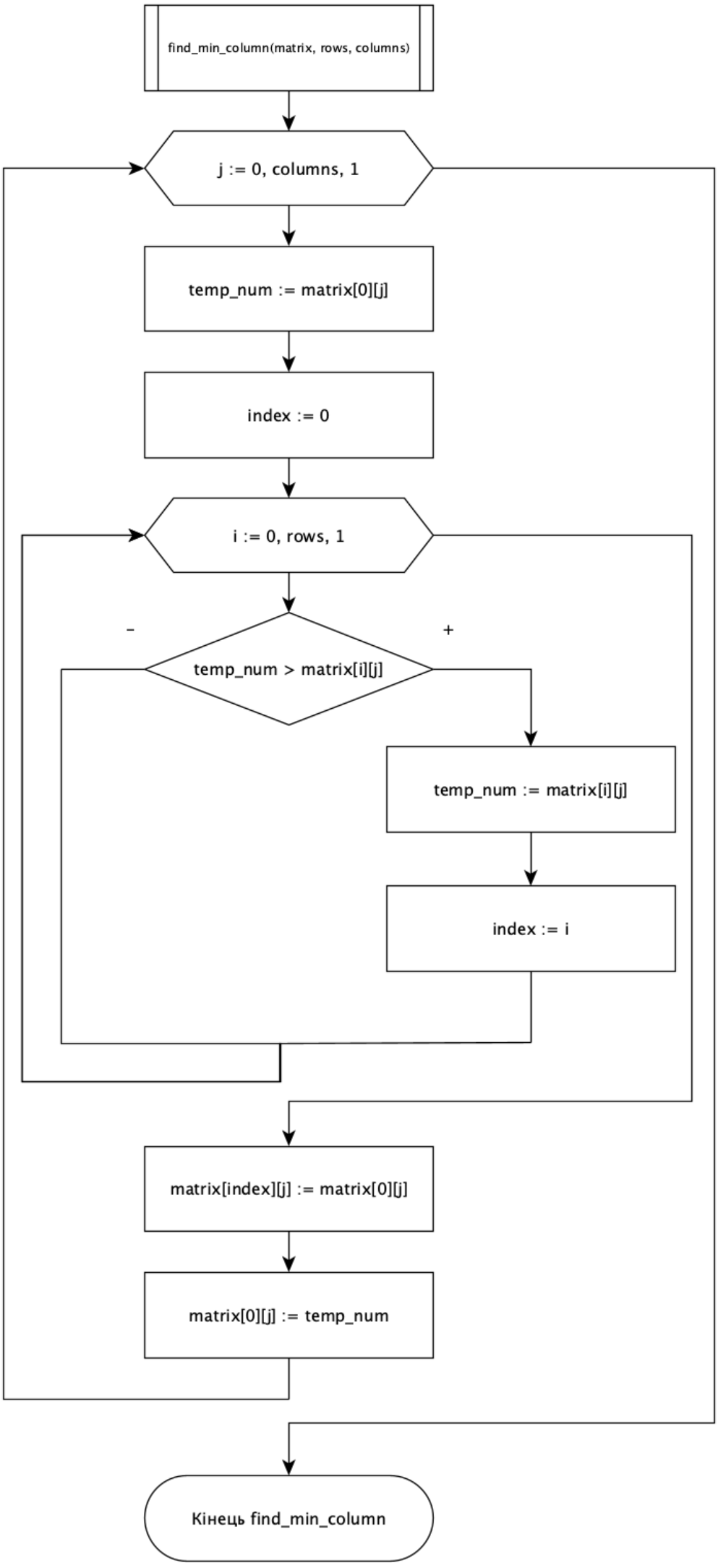
```
            все повторити
            повернути matrix
Кінець generate_matrix

 display_matrix(matrix, rows, columns)
            повторити для i від 0 до rows
              повторити для j від 0 до colums
                виведення matrix[i][j]
              все повторити
            все повторити
Кінець display_matrix

find_min_column(matrix, rows, columns)
            повторити для j від 0 до columns
              temp_num := matrix[0][j]
              index := 0
              повторити для i від 0 до rows
                    якщо temp_num > matrix[i][j]
                       то
                            temp_num := matrix[i][j]
                            index := i
                    все якщо
              все повторити
              matrix[index][j] := matrix[0][j]
              matrix[0][j] := temp_num
            все повторити
 Кінець find_min_column
```

Блок-схема


Початок
введення m, n
matrix := generate_matrix(m, n)
“Generated matrix:\n”
display_matrix(matrix, m, n)
find_min_column(matrix, m, n)
виведення “Swapped matrix:\n”
display_matrix(matrix, m, n)
Кінець


generate_matrix(rows, columns)
i := 0, rows, 1
j := 0, columns, 1
matrix[i][j] := double_rand(0, 10)
повернути matrix
Кінець generate_matrix

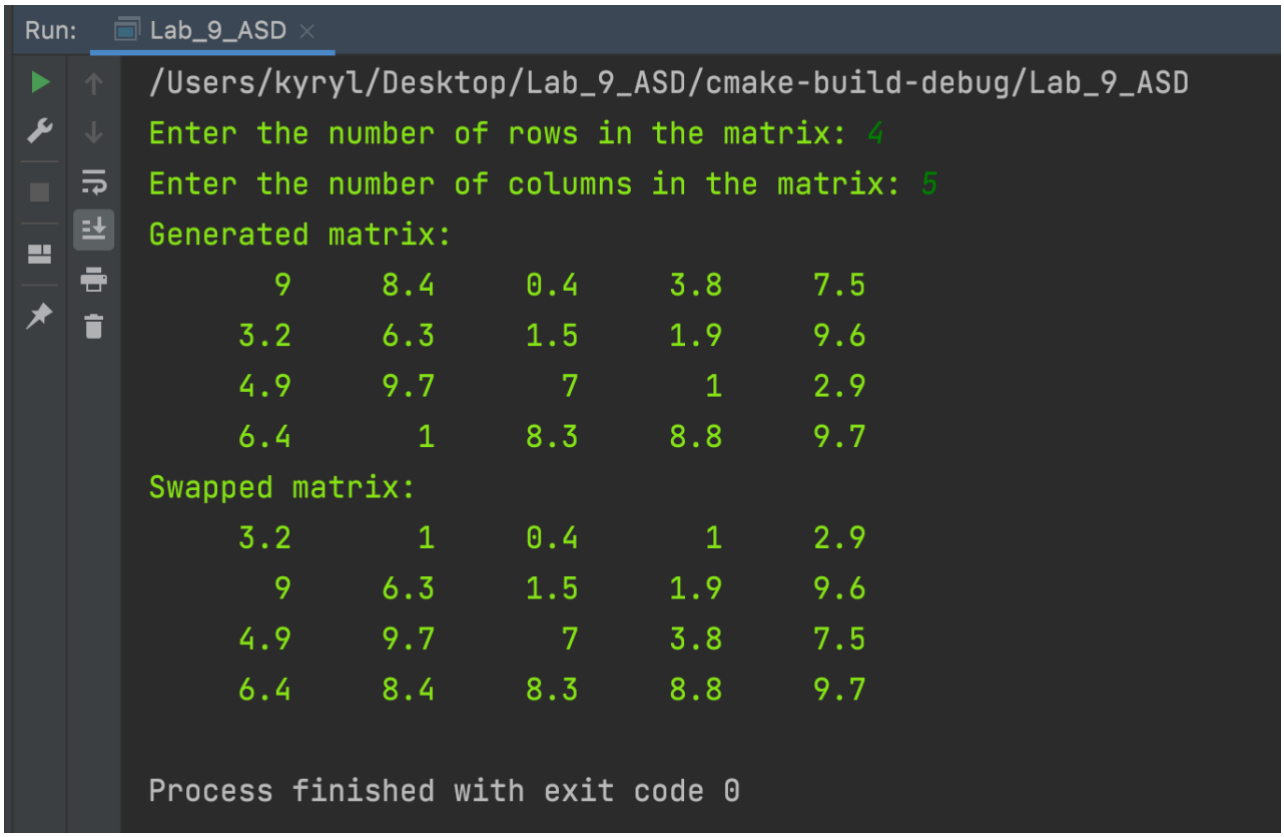

display_matrix(matrix, rows, columns)
i := 0, rows, 1
j := 0, columns, 1
matrix[i][j]
Кінець display_matrix


find_min_column(matrix, rows, columns)
j := 0, columns, 1
temp_num := matrix[0][j]
index := 0
i := 0, rows, 1
-
+
temp_num > matrix[i][j]
temp_num := matrix[i][j]
index := i
matrix[index][j] := matrix[0][j]
matrix[0][j] := temp_num
Кінець find_min_column

Програма на C++

```
#include <iostream>
#include <ctime>
#include <iomanip>
using namespace std;
double rand_double(int, int);
double** generate_matrix(int, int);
void display_matrix(double**, int, int);
void find_min_column(double**, int, int);

int main() {
    int m, n;
    double **matrix;
    srand(time(NULL));
    cout << "Enter the number of rows in the matrix: ";
    cin >> m;
    cout << "Enter the number of columns in the matrix: ";
    cin >> n;
    matrix = generate_matrix(m, n);
    cout << "Generated matrix:\n";
    display_matrix(matrix, m, n);
    find_min_column(matrix, m, n);
    cout << "Swapped matrix:\n";
    display_matrix(matrix, m, n);
    return 0;
}

double rand_double(int min, int max) {
    double fraction, rnd_max, result;
    rnd_max = rand()%(max + 1);
    fraction = (double)rnd_max / max;
    result = min + rand()%(max-min) + fraction;
    return result;
}

double** generate_matrix(int rows, int columns) {
    double **matrix = new double*[rows];
    for (int i = 0; i < rows; i++) {
        matrix[i] = new double[columns];
    }
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < columns; j++) {
            matrix[i][j] = rand_double(0, 10);
        }
    }
    return matrix;
}

void display_matrix(double** matrix, int rows, int columns) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < columns; j++) {
            cout << setw(8) << matrix[i][j];
        }
        cout << "\n";
    }
```

```
}

void find_min_column(double **matrix, int rows, int columns)  {
    double temp_num;
    int index;
    for (int j = 0; j < columns; j++) {
        temp_num = matrix[0][j];
        index = 0;
        for (int i = 0; i < rows; i++) {
            if (temp_num > matrix[i][j]) {
                temp_num = matrix[i][j];
                index = i;
            }
        }
        matrix[index][j] = matrix[0][j];
        matrix[0][j] = temp_num;
    }
}
```



Випробування алгоритму

Блок	Дія
	Початок
1	Введення m := 4, n := 5
2	Виведення Generated matrix: 9 8.4 0.4 3.8 7.5 3.2 6.3 1.5 1.9 9.6 4.9 9.7 7 1 2.9 6.4 1 8.3 8.8 9.7
3	Виведення Swapped matrix: 3.2 1 0.4 1 2.9 9 6.3 1.5 1.9 9.6 4.9 9.7 7 3.8 7.5 6.4 8.4 8.3 8.8 9.7
	Кінець

Висновок

Отже, я дослідив алгоритми обходу масивів, зокрема, алгоритм обходу матриці по стовпцях, набув практичних навичок у його використанні, створивши алгоритм для пошуку мінімального елемента кожного стовпця двовимірного масиву(матриці) й обміну його з відповідним елементом першого рядка, та отримав коректний результат.