Алгоритми та структури даних. Основи алгоритмізації

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. Основи алгоритмізації»

«Дослідження алгоритмів обходу масивів»

Варіант 28

Виконав студент: ІП-15 Рибаков Дмитро Вадимович

Перевірив: Вечерковська Анастасія Сергіївна

Київ 2021

Алгоритми та структури даних. Основи алгоритмізації

Лабораторна 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

Постановка задачі

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

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).

Таблиця 1

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.

Побудова математичної моделі

Змінна	Тип	Ім’я	Призначення
Розмірність матриці, рядки	Цілий	m	Початкові дані
Розмірність матриці, стовпці	Цілий	n	Початкові дані
Рядки матриці	Цілий	row	Проміжні дані
Стовпці матриці	Цілий	column	Проміжні дані
Згенерована матриця A[m,n]	Дійсний	matrix	Проміжні дані
Перший мінімальний елемент кожного стовпчика	Дійсний	X	Вихідні дані
Крок циклу. Індекс числа у масиві	Цілий	i	Проміжні дані
Крок циклу. Індекс числа у масиві	Цілий	j	Проміжні дані
Індекс першого мінімального елементу кожного стовпчика у матриці	Цілий	ind	Проміжні дані
Індекс(рядок) першого мінімального елементу кожного стовпчика у матриці	Цілий	ind_i	Вихідні дані
Індекс(стовпець) першого мінімального елементу кожного стовпчика у матриці	Цілий	ind_ j	Вихідні дані
Відсортований обміном за спаданням масив. Результат	Цілий	array	Вихідні дані

Згенеруємо матрицю A[m,n] випадковими дійсними числами за допомогою функції rand().

У кожному стовпчику матриці знайдемо перший мінімальний елемент X і його місцезнаходження, тобто його індекс. Обміняємо знайдене значення X з елементом першого рядка у кожному стовпчику.

У роботі використаємо підпрограми.

У роботі використовуються наступні дії:

«=» - операція присвоєння;

«>» - більше (більше ніж);

«<» - менше (менше ніж);

«A[m][n]» - матриця A, де m – рядки, n – стовпці;

«++» - збільшення значення на 1.

Функція rand() генерує випадкове дійсне число.

Виведення потрібних даних здійснюється впродовж алгоритму.

Розв'язання

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

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію генерації матриці A[m,n] і її виведення.

Крок 3. Деталізуємо дію знаходження першого мінімального елемента X і його місцезнаходження у кожному стовпчику.

Крок 4. Деталізуємо дію обміну знайденого значення X з елементом першого рядка у кожному стовпчику.

Псевдокод

Основна програма:

крок 1

початок

генерації матриці A[m,n]

знаходження першого мінімального елемента X і його місцезнаходження у кожному стовпчику

обмін знайденого значення X з елементом першого рядка у кожному стовпчику

кінець

крок 2

початок

ввід m та n

matrix = generate_matrix(m, n)

write_matrix(matrix, m, n)

знаходження першого мінімального елемента X і його місцезнаходження у кожному стовпчику

обмін знайденого значення X з елементом першого рядка у кожному стовпчику

кінець

крок 3

початок

ввід m та n

matrix = generate_matrix(m, n)

write_matrix(matrix, m, n)

min_column(matrix, m, n)

обмін знайденого значення X з елементом першого рядка у кожному стовпчику

кінець

крок 4

початок

ввід m та n

matrix = generate_matrix(m, n)

write_matrix(matrix, m, n)

min_column(matrix, m, n)

min_column_swap(matrix, m, n)

write_matrix(matrix, m, n)

кінець

Підпрограми

generate_matrix(m, n)

початок

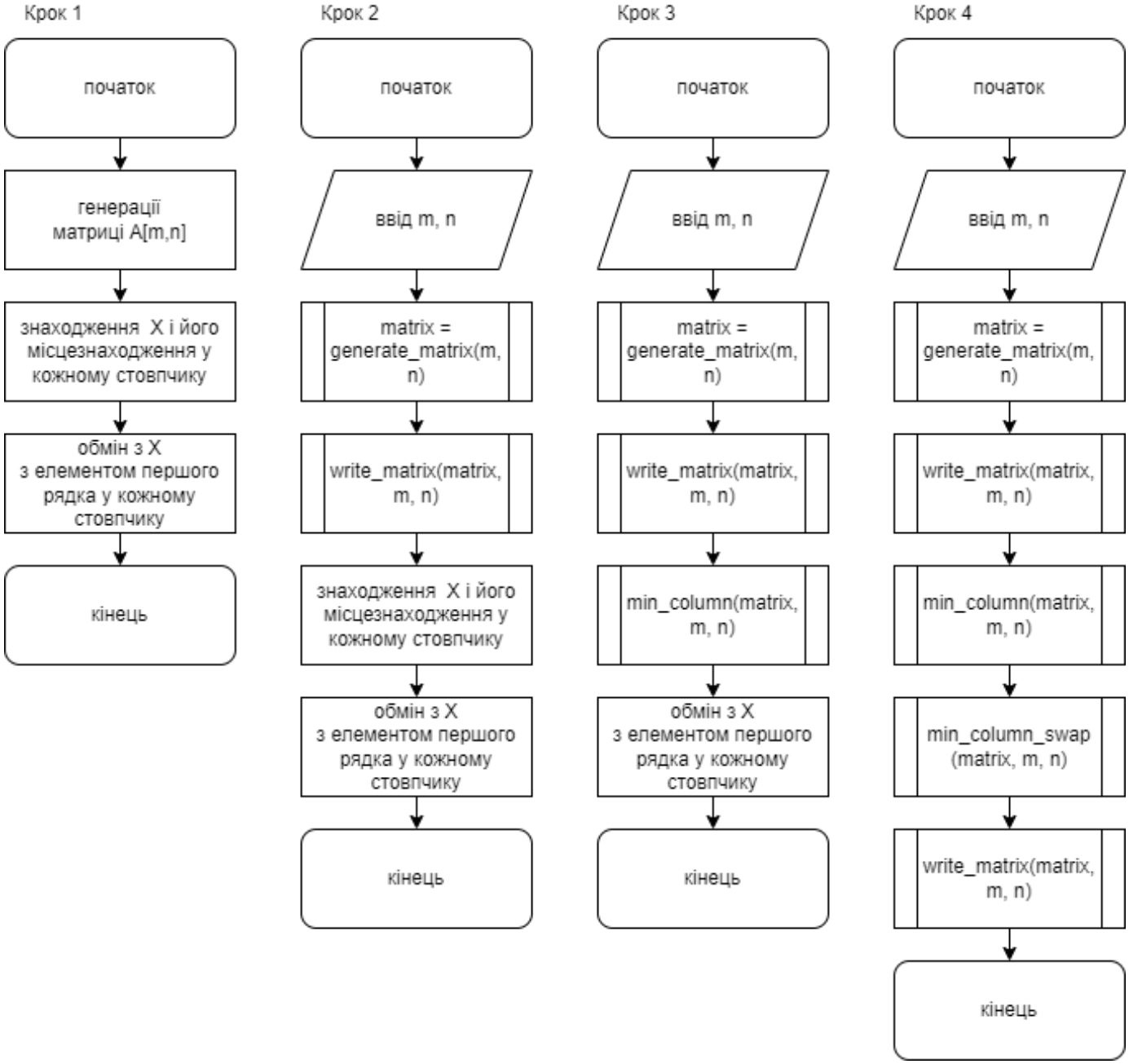
для і від 0 до row повторити

```
    для j від 0 до column повторити
        matrix[i][j] = rand() від 0 до 10
    все повторити
  все повторити
  повернути matrix
кінець

write_matrix(matrix, m, n)
початок
  для i від 0 до row повторити
    для j від 0 до column повторити
        вивід matrix[i][j]
    все повторити
  все повторити
кінець

min_column(matrix, m, n)
початок
  для j від 0 до column повторити
  X = matrix[0][j]
  ind = 0
    для i від 0 до row повторити
    якщо X > matrix[i][j]  то
       X = matrix[i][j]
       ind = i
    все якщо
    все повторити
    ind_i = ind + 1
    ind_ j = j + 1
    вивід ind_i, ind_ j, X
  все повторити
кінець
```

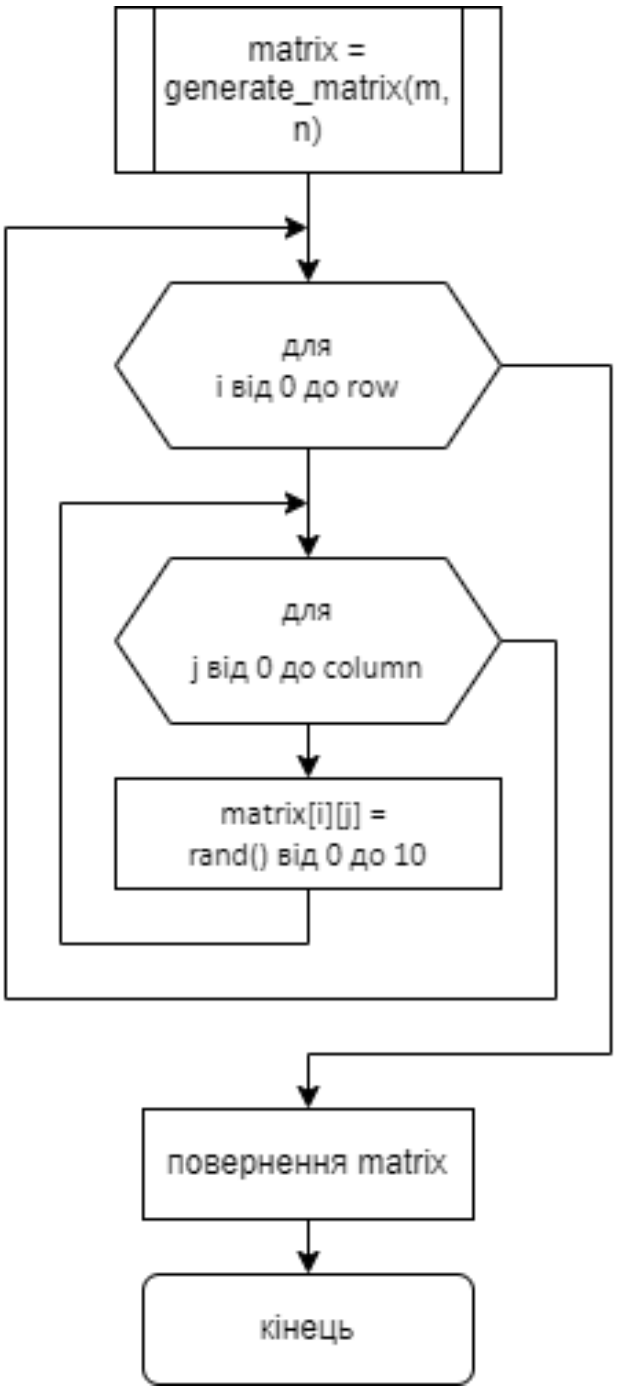
```
min_column_swap(matrix, m, n)
початок
    для j від 0 до column повторити
    X = matrix[0][j]
    ind = 0
        для i від 0 до row повторити
        якщо X > matrix[i][j]  то
            X = matrix[i][j]
            ind = i
        все якщо
        все повторити
        matrix[ind][j] = matrix[0][j]
        matrix[0][j] = X
    все повторити
кінець
```

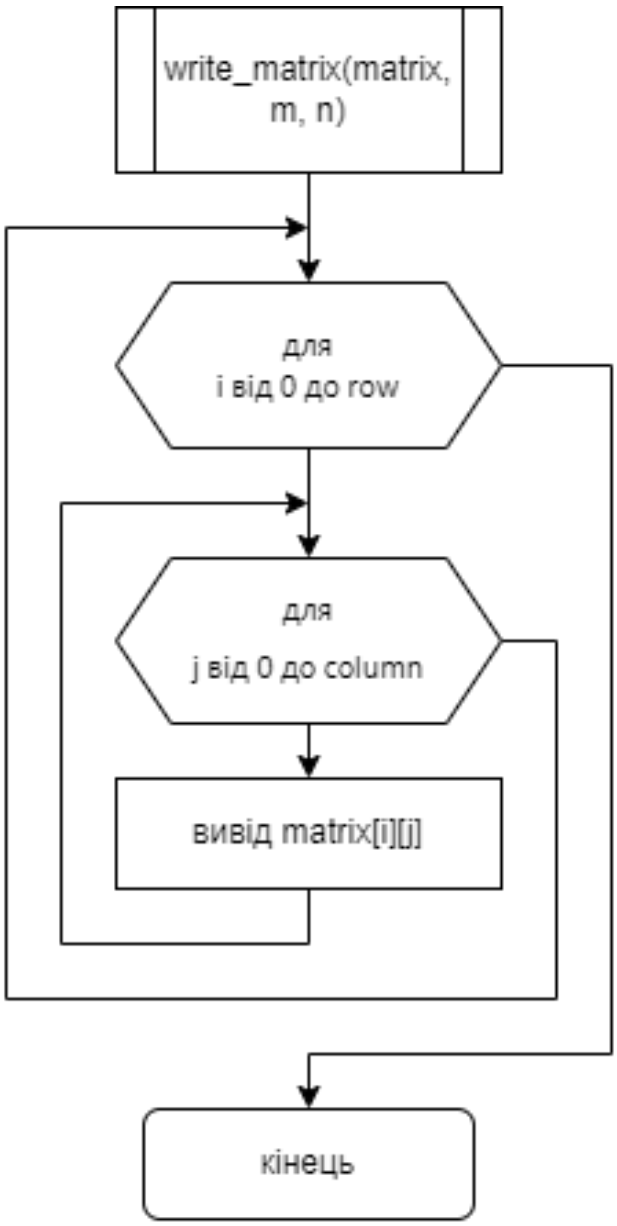
Блок-схема

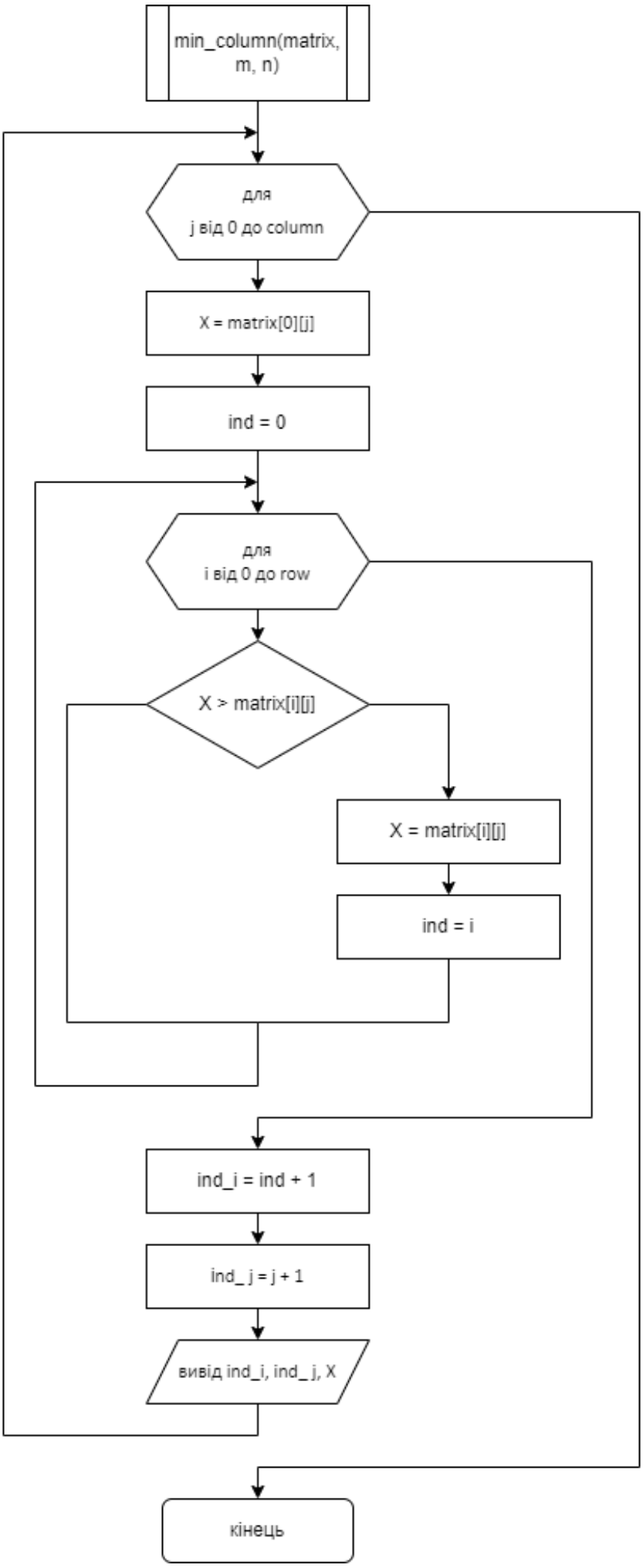
Основна програма:

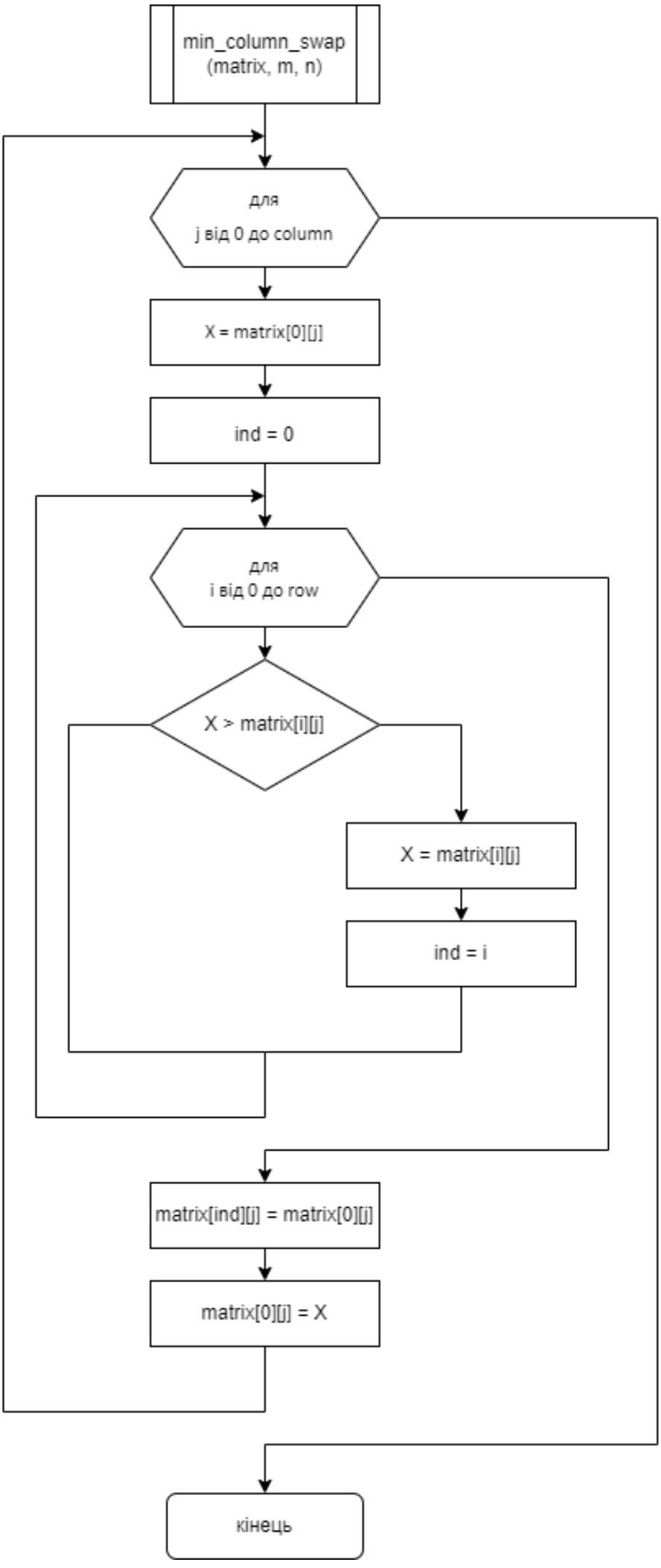


Підпрограми:






min_column(matrix, m, n)
для
j від 0 до column
X = matrix[0][j]
ind = 0
для
i від 0 до row
X > matrix[i][j]
X = matrix[i][j]
ind = i
ind_i = ind + 1
ind_ j = j + 1
вивід ind_i, ind_ j, X
кінець


min_column_swap
(matrix, m, n)
для
j від 0 до column
X = matrix[0][j]
ind = 0
для
i від 0 до row
X > matrix[i][j]
X = matrix[i][j]
ind = i
matrix[ind][j] = matrix[0][j]
matrix[0][j] = X
кінець

Код програми

```cpp
#include <iostream>
#include <iomanip>

using namespace std;

double** generate_matrix(int row, int column) {
    double** matrix = new double* [row];
    for (int i = 0; i < row; i++) {
        matrix[i] = new double[column];
    }
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < column; j++) {
            matrix[i][j] = (rand() % 101 - 0) / 10.0;
        }
    }
    return matrix;
}

void write_matrix(double** matrix, int row, int column) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < column; j++) {
            cout << setw(6) << matrix[i][j];
        }
        cout << endl;
    }
    cout << endl;
}

void min_column(double** matrix, int row, int column) {
    double X;
    int ind;
    int ind_i, ind_j;
    for (int j = 0; j < column; j++) {
        X = matrix[0][j];
        ind = 0;
        for (int i = 0; i < row; i++) {
            if (X > matrix[i][j]) {
                X = matrix[i][j];
                ind = i;
            }
        }
        ind_i = ind + 1;
        ind_j = j + 1;
        cout << "Місцезнаходження першого мінімального елемента рядків матриці m x n: " << ind_i << " x " << ind_j << ". " << "Елемент:" << " " << X << endl;
    }
```

```cpp
            }
        }
        ind_i = ind + 1;
        ind_j = j + 1;
        cout << "Місцезнаходження першого мінімального елемента рядків матриці m x n: " << ind_i << " x " << ind_j << ". " << "Елемент:" << " " << X << endl;
    }
    cout << endl;
}

void min_column_swap(double** matrix, int row, int column) {
    double X;
    int ind;
    for (int j = 0; j < column; j++) {
        X = matrix[0][j];
        ind = 0;
        for (int i = 0; i < row; i++) {
            if (X > matrix[i][j]) {
                X = matrix[i][j];
                ind = i;
            }
        }
        matrix[ind][j] = matrix[0][j];
        matrix[0][j] = X;
    }
    cout << endl;
}

int main()
{
    setlocale(LC_ALL, "Ukrainian");
    srand(time(NULL));
    double** matrix;
    cout << "Введіть розмірність матриці рядків m і стовпців n: " << endl;
    int m, n;
    cin >> m >> n;
    matrix = generate_matrix(m, n);
    cout << "Згенерована матриця m x n:" << endl;
    write_matrix(matrix, m, n);
    min_column(matrix, m, n);
    min_column_swap(matrix, m, n);
    cout << "Обработана матриця m x n:" << endl;
    write_matrix(matrix, m, n);

    return 0;
}
```

```cpp
#include <iostream>
#include <iomanip>

using namespace std;

double** generate_matrix(int row, int column) {
	double** matrix = new double* [row];
	for (int i = 0; i < row; i++) {
		matrix[i] = new double[column];
	}
	for (int i = 0; i < row; i++) {
		for (int j = 0; j < column; j++) {
			matrix[i][j] = (rand() % 101 - 0) / 10.0;
		}
	}
	return matrix;
}

void write_matrix(double** matrix, int row, int column) {
	for (int i = 0; i < row; i++) {
		for (int j = 0; j < column; j++) {
			cout << setw(6) << matrix[i][j];
		}
		cout << endl;
	}
	cout << endl;
}

void min_column(double** matrix, int row, int column) {
	double X;
	int ind;
	int ind_i, ind_j;
	for (int j = 0; j < column; j++) {
		X = matrix[0][j];
		ind = 0;
		for (int i = 0; i < row; i++) {
			if (X > matrix[i][j]) {
				X = matrix[i][j];
				ind = i;
			}
		}
		ind_i = ind + 1;
		ind_j = j + 1;
		cout << "Місцезнаходження першого мінімального елемента рядків матриці m x n: " <<
ind_i << " x " << ind_j << ". " << "Елемент:" << " " << X << endl;
	}
	cout << endl;
}

void min_column_swap(double** matrix, int row, int column) {
	double X;
	int ind;
	for (int j = 0; j < column; j++) {
		X = matrix[0][j];
		ind = 0;
		for (int i = 0; i < row; i++) {
			if (X > matrix[i][j]) {
				X = matrix[i][j];
				ind = i;
			}
		}
		matrix[ind][j] = matrix[0][j];
		matrix[0][j] = X;
	}
	cout << endl;
}

int main()
{
	setlocale(LC_ALL, "Ukrainian");
```

```
        srand(time(NULL));
        double** matrix;
        cout << "Введіть розмірність матриці рядків m і стовпців n: " << endl;
        int m, n;
        cin >> m >> n;
        matrix = generate_matrix(m, n);
        cout << "Згенерована матриця m x n:" << endl;
        write_matrix(matrix, m, n);
        min_column(matrix, m, n);
        min_column_swap(matrix, m, n);
        cout << "Обработана матриця m x n:" << endl;
        write_matrix(matrix, m, n);

        return 0;
}
```

Тестування програми

```
Консоль отладки Microsoft Visual Studio
Введіть розмірність матриці рядків m і стовпців n:
6 6
Згенерована матриця m x n:
   1.1   3.2   7.8   4.4   5.3     6
   0.5   2.9   9.5   4.4   7.6   3.5
   8.6   1.8   8.4   9.8   7.7   2.6
   9.2   9.6   5.5   1.3   2.6   1.2
   2.9   7.8   5.4   3.8   0.7   0.4
     9   9.8   0.6   9.3   1.9   6.4

Місцезнаходження першого мінімального елемента рядків матриці m x n: 2 x 1. Елемент: 0.5
Місцезнаходження першого мінімального елемента рядків матриці m x n: 3 x 2. Елемент: 1.8
Місцезнаходження першого мінімального елемента рядків матриці m x n: 6 x 3. Елемент: 0.6
Місцезнаходження першого мінімального елемента рядків матриці m x n: 4 x 4. Елемент: 1.3
Місцезнаходження першого мінімального елемента рядків матриці m x n: 5 x 5. Елемент: 0.7
Місцезнаходження першого мінімального елемента рядків матриці m x n: 5 x 6. Елемент: 0.4


Обработана матриця m x n:
   0.5   1.8   0.6   1.3   0.7   0.4
   1.1   2.9   9.5   4.4   7.6   3.5
   8.6   3.2   8.4   9.8   7.7   2.6
   9.2   9.6   5.5   4.4   2.6   1.2
   2.9   7.8   5.4   3.8   5.3     6
     9   9.8   7.8   9.3   1.9   6.4
```

```
Консоль отладки Microsoft Visual Studio
Введіть розмірність матриці рядків m і стовпців n:
5 8
Згенерована матриця m x n:
   5.2   2.8   3.9   3.2   2.8   6.7   4.7   3.8
   6.2   9.8   4.3   1.7   2.5   6.5   1.3   2.4
   6.8   2.2     7   5.7     7   4.5   0.7     4
   3.3   2.5   3.1   0.9   7.5     7   6.4   2.1
   1.4   2.6   1.8   8.5   2.9     4   6.5   5.9

Місцезнаходження першого мінімального елемента рядків матриці m x n: 5 x 1. Елемент: 1.4
Місцезнаходження першого мінімального елемента рядків матриці m x n: 3 x 2. Елемент: 2.2
Місцезнаходження першого мінімального елемента рядків матриці m x n: 5 x 3. Елемент: 1.8
Місцезнаходження першого мінімального елемента рядків матриці m x n: 4 x 4. Елемент: 0.9
Місцезнаходження першого мінімального елемента рядків матриці m x n: 2 x 5. Елемент: 2.5
Місцезнаходження першого мінімального елемента рядків матриці m x n: 5 x 6. Елемент: 4
Місцезнаходження першого мінімального елемента рядків матриці m x n: 3 x 7. Елемент: 0.7
Місцезнаходження першого мінімального елемента рядків матриці m x n: 4 x 8. Елемент: 2.1


Обработана матриця m x n:
   1.4   2.2   1.8   0.9   2.5     4   0.7   2.1
   6.2   9.8   4.3   1.7   2.8   6.5   1.3   2.4
   6.8   2.8     7   5.7     7   4.5   4.7     4
   3.3   2.5   3.1   3.2   7.5     7   6.4   3.8
   5.2   2.6   3.9   8.5   2.9   6.7   6.5   5.9
```

```
Консоль отладки Microsoft Visual Studio
4 9
Згенерована матриця m x n:
     1   8.9   8.5   9.2   4.1   0.9   2.9   9.6     3
   1.6     1   5.5   5.2   6.2   2.5   4.7   9.7    10
   5.7   2.8   5.1   9.9   4.8   1.6   3.6    10   1.8
   3.5   2.8   4.2   3.7   1.6   6.2   2.9   1.4   7.4

Місцезнаходження першого мінімального елемента рядків матриці m x n: 1 x 1. Елемент: 1
Місцезнаходження першого мінімального елемента рядків матриці m x n: 2 x 2. Елемент: 1
Місцезнаходження першого мінімального елемента рядків матриці m x n: 4 x 3. Елемент: 4.2
Місцезнаходження першого мінімального елемента рядків матриці m x n: 4 x 4. Елемент: 3.7
Місцезнаходження першого мінімального елемента рядків матриці m x n: 4 x 5. Елемент: 1.6
Місцезнаходження першого мінімального елемента рядків матриці m x n: 1 x 6. Елемент: 0.9
Місцезнаходження першого мінімального елемента рядків матриці m x n: 1 x 7. Елемент: 2.9
Місцезнаходження першого мінімального елемента рядків матриці m x n: 4 x 8. Елемент: 1.4
Місцезнаходження першого мінімального елемента рядків матриці m x n: 3 x 9. Елемент: 1.8


Обработана матриця m x n:
     1     1   4.2   3.7   1.6   0.9   2.9   1.4   1.8
   1.6   8.9   5.5   5.2   6.2   2.5   4.7   9.7    10
   5.7   2.8   5.1   9.9   4.8   1.6   3.6    10     3
   3.5   2.8   8.5   9.2   4.1   6.2   2.9   9.6   7.4
```

Висновки

На цій лабораторній роботі ми дослідили алгоритми обходу масивів, набули практичних навичок використання цих алгоритмів під час складання програмних специфікацій. В результаті виконання лабораторної роботи ми отримали алгоритм сортування обміном за спаданням, при цьому розділили виконання задачи на 4 кроки. В процесі випробування ми розглянули декілька випадків результатом яких

отримали виведення першого мінімального елемента X і його місцезнаходження, виведення обробленої матриці обміном знайденого значення X з елементом першого рядка у кожному стовпчику. Алгоритм ефективний і працює з матрицями будь-якого розміру.