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 6

Виконав студент ІП-13 Вдовиченко Станіслав Юрійович
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Лабораторна робота 9
Дослідження алгоритмів обходу масивів

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

Варіант 6:

6	Задано матрицю дійсних чисел A[m,n]. При обході матриці по стовпчиках знайти в ній перший мінімальний елемент і його місцезнаходження. Обміняти знайдене значення X з елементом побічної діагоналі.

Постановка задачі: треба задати матрицю (двовимірний масив) дійсних чисел (розмірність визначає користувач), заповнити її випадковими числами, обійти матрицю по стовпчиках, знайти перший мінімальний елемент та запам’ятати його місцезнаходження, визначити елемент побічної діагоналі та обміняти його з мінімумом.

Математична модель:

Змінна	**Тип**	**Ім’я**	**Призначення**
Розмір матриці	Цілочисельний	size	Вхідні дані
Матриця	Дійсний	matrix	Вихідні дані
Мінімальний елемент	Дійсний	min	Проміжні дані
Індекс рядка мінімального елемента	Цілочисельний	x	Проміжні дані
Індекс стовпчика мінімального елемента	Цілочисельний	y	Проміжні дані

Підпрограма fillMatrix

Змінна	**Тип**	**Ім’я**	**Призначення**
Розмір матриці	Цілочисельний	n	Вхідні дані

Підпрограма findMinimum

Змінна	**Тип**	**Ім’я**	**Призначення**
Розмір матриці	Цілочисельний	size	Вхідні дані
Матриця	Дійсний	matrix	Проміжні дані
Мінімальний елемент	Дійсний	minimum	Вихідні дані

Індекс рядка мінімального елемента	Цілочисельний	x	Проміжні дані
Індекс стовпчика мінімального елемента	Цілочисельний	y	Проміжні дані

Підпрограма swapElements

Змінна	**Тип**	**Ім’я**	**Призначення**
Розмір матриці	Цілочисельний	size	Вхідні дані
Матриця	Дійсний	matrix	Проміжні дані
Мінімальний елемент	Дійсний	minimum	Проміжні дані
Індекс рядка мінімального елемента	Цілочисельний	x	Проміжні дані
Індекс стовпчика мінімального елемента	Цілочисельний	y	Проміжні дані
Тимчасова змінна для обміну елементів	Дійсний	temp	Проміжні дані

Так як нам треба визначати елементи на діагоналі, то очевидно, що ця матриця – квадратна, отже рядків і стовпчиків буде однаково, їх ми задаємо через змінну size.

Таким чином математичне формулювання задачі зводиться до створення двовимірного масиву, заповнення його випадковими числами за допомогою підпрограми fillMatrix. Далі знаходимо перший мінімальний елемент при обході за стовпцями, для цього визначаємо елемент з індексом [0][0] як мінімальний (його індекси 0, 0), та за допомогою арифметичного циклу шукаємо елемент менший за мінімум, якщо такий існує – то змінюємо значення змінної minimum на значення даного елемента, а його індекс вносимо в змінні x та y(змінюємо не значення змінних в функції, а значення глобальних змінних). Далі обмінюємо мінімум з елементом побічної діагоналі за допомогою підпрограми swapElements: вибираємо елемент побічної діагоналі такий, що має рядок як у мінімума і стовпчик size – x – 1, тобто буде знаходитись на побічній діагоналі. Якщо ж мінімум спочатку знаходився на побічній діагоналі, то залишаємо його на місці. Виводимо

отриману матрицю. Для генерування дійсних чисел використаємо функція rand, яка буде генерувати числа з діапазону [-10;10].

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.
Крок 1. Визначимо основні дії.
Крок 2. Ініціювання двовимірного масиву.
Крок 3. Заповнення двовимірного масиву, його вивід.
Крок 4. Знаходження мінімального елемента в масиві.
Крок 5. Обмін мінімуму з елементом побічної діагоналі.

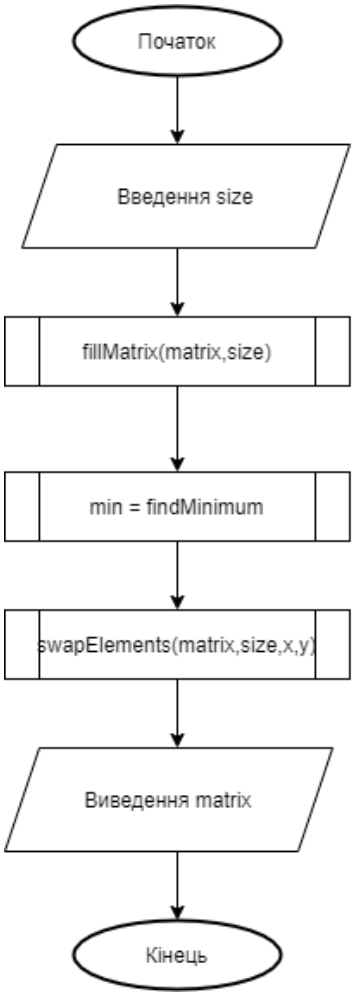
Псевдокод

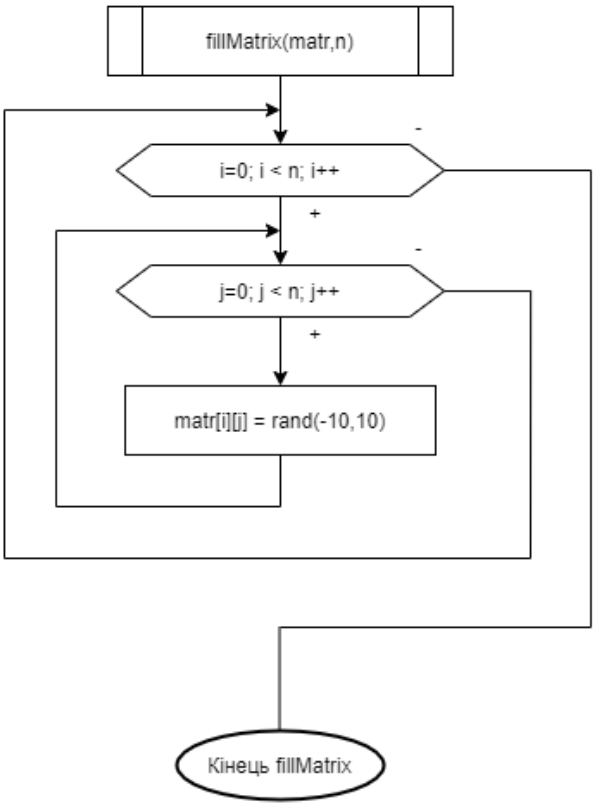
```
Процедура
    fillMatrix(matr,n)
        повторити від 0 до n з кроком 1
            повторити від 0 до n з кроком 1
                matr[i][j] = rand(-10,10)
            все повторити
        все повторити
Все процедура

Процедура
    findMinimum(matr,size,x,y)
        minimum = matr[0][0]
        x = y = 0
        повторити від 0 до size з кроком 1
            повторити від 0 до size з кроком 1
                якщо (matr[i][j] < minimum)
                    minimum = matr[i][j]
                    x = i
                    y = j
                все якщо
            все повторити
        все повторити
        return minimum
Все процедура
```

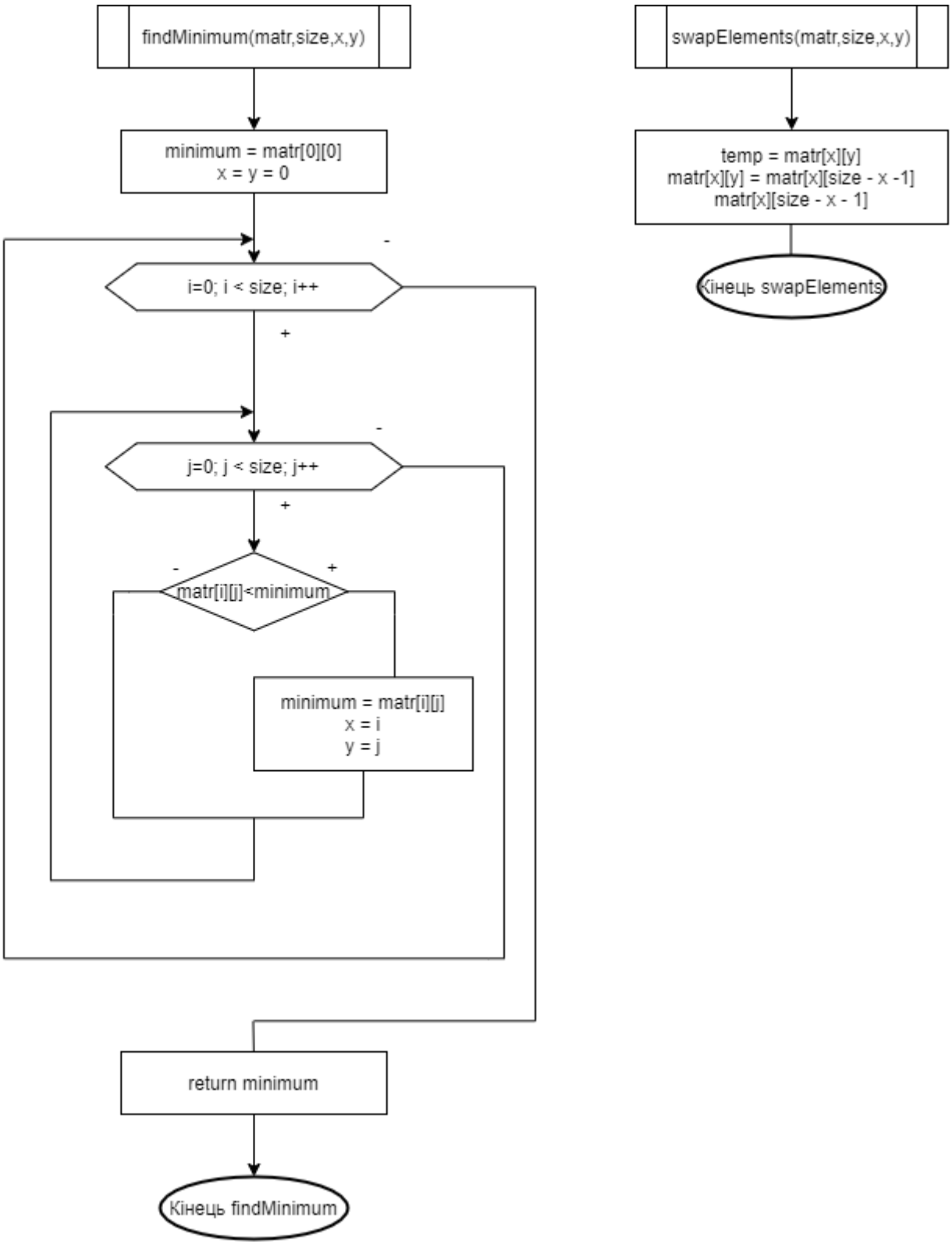
Процедура
swapElements(matr,size,x,y)
temp = matr[x][y]
matr[x][y] = matr[x][size – x – 1]
matr[x][size – x – 1] = temp
Все процедура

Початок
Введення size
fillMatrix(matrix,size)
min = findMinimum(matrix,size,x,y)
swapElements(matrix,size,x,y)
Виведення matrix
Кінець

Блок-схема


Початок
Введення size
fillMatrix(matrix,size)
min = findMinimum
swapElements(matrix,size,x,y)
Виведення matrix
Кінець


fillMatrix(matr,n)
i=0; i < n; i++
-
+
j=0; j < n; j++
-
+
matr[i][j] = rand(-10,10)
Кінець fillMatrix


findMinimum(matr,size,x,y)
minimum = matr[0][0]
x = y = 0
i=0; i < size; i++
-
+
j=0; j < size; j++
-
+
matr[i][j]<minimum
-
+
minimum = matr[i][j]
x = i
y = j
return minimum
Кінець findMinimum
swapElements(matr,size,x,y)
temp = matr[x][y]
matr[x][y] = matr[x][size - x -1]
matr[x][size - x - 1]
Кінець swapElements

Код програми

```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <iomanip>
using namespace std;

double** createMatrix(int);
void deleteMatrix(double *[],int);
void fillMatrix(double* [],int);
void showMatrix(double* [],int);
double findMinimum(double* [],int, int&, int&);
void swapElements(double* [], int, int, int);
int main() {
    srand( _Seed: time( _Time: NULL));
    int size;
    double min;
    int x;
    int y;
    double **matrix;

    cout << "Enter size of matrix (rows = columns):  ";
    cin >> size;
    matrix = createMatrix(size);
    fillMatrix(matrix,size);
    cout << "Matrix " << size << " on " << size << ":\n";
    showMatrix(matrix,size);
    min = findMinimum(matrix,size, &x, &y);
    cout << "First minimum is: " << min << " at " << "["<< x+1 <<"]" << "["<< y+1 <<"]" << endl;
    swapElements(matrix,size,x,y);
    cout << "Updated matrix: " << endl;
    showMatrix(matrix,size);
    deleteMatrix(matrix,size);
    return 0;
}
```

```cpp
double** createMatrix(int n){
    auto **matr = new double* [n];
    for (int i = 0; i < n; ++i) {
        matr[i] = new double[n];
    }
    return matr;
}
void fillMatrix(double* matr[], int n){
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            matr[i][j] = (rand()%201 - 100) / 10.0;
        }
    }
}
void showMatrix(double* matr[], int n){
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            cout << setw( n: 5) << matr[i][j];
        }
        cout << "\n";
    }
}
double findMinimum(double* matr[], int size, int &x, int &y){
    double minimum = matr[0][0];
    x = y = 0;
    for (int j = 0; j < size; ++j) {
        for (int i = 0; i < size; ++i) {
            if (matr[i][j] < minimum) {
                minimum = matr[i][j];
                x = i;
                y = j;
            }
        }
    }
    return minimum;
}
```

```cpp
void swapElements(double* matr[], int size, int x, int y){
    double temp;
    temp = matr[x][y];
    cout << "Exchange minimum with element: " << matr[x][size - x - 1] << " at [" << x + 1 << "]" << "[" << size - x << "]" << endl;
    matr[x][y] = matr[x][size - x - 1];
    matr[x][size - x - 1] = temp;
}
void deleteMatrix(double* matr[], int n){
    for (int i = 0; i < n; ++i) {
        delete []matr[i];
    }
    delete []matr;
}
```

```
Enter size of matrix (rows = columns):4
  Matrix 4 on 4:
 -7.9 -4.6  0.8  3.8
 -9.3 -6.4 -3.8   -6
   -7  3.3  2.7  2.3
 -2.6  4.6 -9.2 -2.7
First minimum is: -9.3 at [2][1]
Exchange minimum with element: -3.8 at [2][3]
Updated matrix:
 -7.9 -4.6  0.8  3.8
 -3.8 -6.4 -9.3   -6
   -7  3.3  2.7  2.3
 -2.6  4.6 -9.2 -2.7

Process finished with exit code 0
```

```
Enter size of matrix (rows = columns):10
  Matrix 10 on 10:
  4.8 -0.8   -6  9.4   -9  9.3 -7.6   -4 -2.1  9.2
  8.8 -6.8 -3.5  4.3 -3.3  9.6    0  1.7  5.7  2.9
  4.6 -2.5  6.6 -6.6  3.3 -1.6  6.8  3.6  6.7   -8
 -0.8 -3.3  1.7 -7.1 -0.7  1.3  0.4  9.6  3.4 -8.7
 -9.8 -5.3 -7.3   -7 -1.4  5.8 -6.1  4.6  2.1 -2.7
 -6.3  3.8 -5.8 -8.6  6.9  3.1    2 -6.7 -5.2   -6
    4  4.9  7.7    0  0.5  0.1  7.3 -8.3  9.5  5.8
 -1.7 -6.3  0.8  8.7  4.2 -7.6 -4.8  2.5  5.8 -8.7
  8.9    5  5.6  8.3  5.5  5.2  0.8 -0.1 -3.7 -7.3
  0.2  3.4 -4.9  2.7  1.6    7  2.3   10  5.5 -3.6
First minimum is: -9.8 at [5][1]
Exchange minimum with element: 5.8 at [5][6]
Updated matrix:
  4.8 -0.8   -6  9.4   -9  9.3 -7.6   -4 -2.1  9.2
  8.8 -6.8 -3.5  4.3 -3.3  9.6    0  1.7  5.7  2.9
  4.6 -2.5  6.6 -6.6  3.3 -1.6  6.8  3.6  6.7   -8
 -0.8 -3.3  1.7 -7.1 -0.7  1.3  0.4  9.6  3.4 -8.7
  5.8 -5.3 -7.3   -7 -1.4 -9.8 -6.1  4.6  2.1 -2.7
 -6.3  3.8 -5.8 -8.6  6.9  3.1    2 -6.7 -5.2   -6
    4  4.9  7.7    0  0.5  0.1  7.3 -8.3  9.5  5.8
 -1.7 -6.3  0.8  8.7  4.2 -7.6 -4.8  2.5  5.8 -8.7
  8.9    5  5.6  8.3  5.5  5.2  0.8 -0.1 -3.7 -7.3
  0.2  3.4 -4.9  2.7  1.6    7  2.3   10  5.5 -3.6

Process finished with exit code 0
```

Висновки

Під час виконання даної лабораторної роботи я дослідив алгоритми обходу масивів, набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Розробив алгоритм для розв’язання поставленої задачі, побудував математичну модель, псевдокод, блок-схему. Написав код програми, протестував алгоритм і переконався в його правильності.