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»

агрегатних значень»

Варіант 11

Виконав студент ІП-12 Дулов Денис Валерійович

(шифр, прізвище, ім'я, по батькові)

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

( прізвище, ім'я, по батькові)

'

Київ 2021

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

**Постановка задачі:**

Задано матрицю дійсних чисел A[n,n], ініціалізувати матрицю обходом по рядках. На побічній діагоналі матриці знайти перший максимальний елемент і останній мінімальний елементи, та поміняти їх місцями з елементами головної діагоналі.

**Математична модель**

|  |  |  |  |
| --- | --- | --- | --- |
| *Значення* | *Тип даних* | *Ім'я* | *Використання* |
| Двовимірний масив розмірності nxn | *Індексований* | *arr* | *Проміжне значення* |
| Розмір arr | *Натуральний* | *n* | *Вхідне значення* |
| Перший максимальний елемент побічної діагоналі arr | *Дійсний* | *max* | *Проміжне значення* |
| Останній мінімальний елемент побічної діагоналі arr | *Дійсний* | *min* | *Проміжне значення* |
| Індекс рядка елемента max | *Натуральний* | *colMax* | *Проміжне значення* |
| Індекс рядка елемента min | *Натуральний* | *colMin* | *Проміжне значення* |
| Функція що повертає випадкове дійсне число від а до b | *Функція* | *random(a,b)* | *Допоміжна функція* |
| Функція, що ініалізує двовимірний масив обходом по рядках | *Функція* | *generate2DArray(arr, n)* | *Допоміжна функція* |
| Функція, що повертає максимальне значення побічної діагоналі двовимірного масиву | *Функція* | getMax(arr, colMax, n) | *Допоміжна функція* |
| Функція, що повертає мінімальне значення побічної діагоналі двовимірного масиву | *Функція* | getMin(arr, colMin, n) | *Допоміжна функція* |
| Функція, що міняє місцями min, max з відповідними елементами головної діагоналі (по рядках) | *Функція* | swapElements(arr, colMin, colMax, n) | *Допоміжна функція* |

**Псевдокод**

Крок 1

**початок**

Ініціалізація arr обходом по рядках

Знайти max, min та colMax, colMin

Поміняти max, min з відповідними елементами головної діагоналі (по рядках)

**кінець**

Крок 2

**початок**

generate2DArray(arr, n)

Знайти max, min та colMax, colMin

Поміняти max, min з відповідними елементами головної діагоналі (по рядках)

**кінець**

Крок 3

**початок**

generate2DArray(arr, n)

getMax(arr, colMax, n)

getMin(arr, colMin, n)

Поміняти max, min з відповідними елементами головної діагоналі (по рядках)

**кінець**

Крок 4

**початок**

generate2DArray(arr, n)

getMax(arr, colMax, n)

getMin(arr, colMin, n)

swapElements(arr, colMin, colMax, n)

**кінець**

Крок 5 (підпрограма generate2DArray(arr, n))

**початок**

j = 0

**повторити**

**для** і від 0 до n з кроком 1

**поки** j < n

arr[i][j] = random(0, 1)

j = j + 1

**все поки**

j = j – 1

**поки** j >= 0

arr[i][j] = random(0, 1)

j = j – 1

**все поки**

**все повторити**

**повернути** arr

Крок 6 (підпрограма getMax(arr, colMax, n))

**початок**

max = arr[n – 1][0]

**повторити**

**для** і від (n-1) до 0 з кроком -1

**якщо** arr[i][n – i – 1] > max

**то**

max = arr[i][n-1 - i]

colMax = i

**все якщо**

**все повторити**

**повернути** max

Крок 7 (підпрограма getMin(arr, colMin, n))

**початок**

min = arr[0][n-1]

**повторити**

**для** і від (n-1) до 0 з кроком -1

**якщо** arr[n-i-1][i] < min

**то**

min = arr[n-i-1][i]

colMin = n-i-1

**все якщо**

**все повторити**

**повернути** min

Крок 8 (підпрограма swapElements(arr, colMin, colMax, n))

**початок**

tmp = arr[colMax][n - colMax - 1]

arr[colMax][n - colMax - 1] = arr[colMax, colMax]

arr[colMax, colMax] = tmp

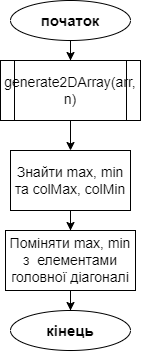
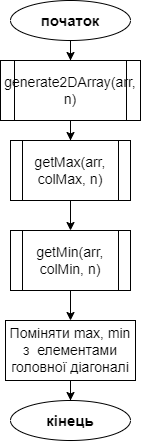
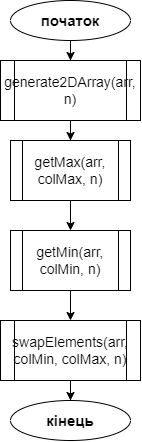
tmp = arr[colMin][n - colMin - 1];

arr[colMin][n - colMin - 1] = arr[colMin, colMin];

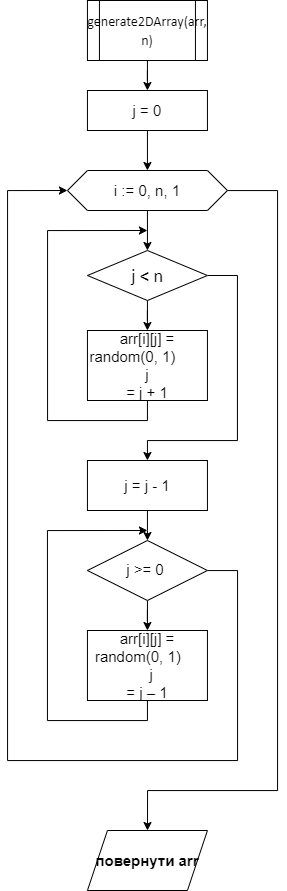
arr[colMin, colMin] = tmp;

**кінець**

Блок-схема

****Крок 1Крок 2 Крок 3 Крок 4

Крок 5 (підпрограма generate2DArray(arr, n))



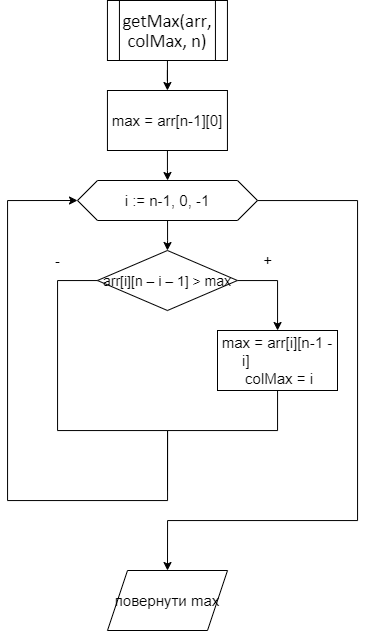
+

-

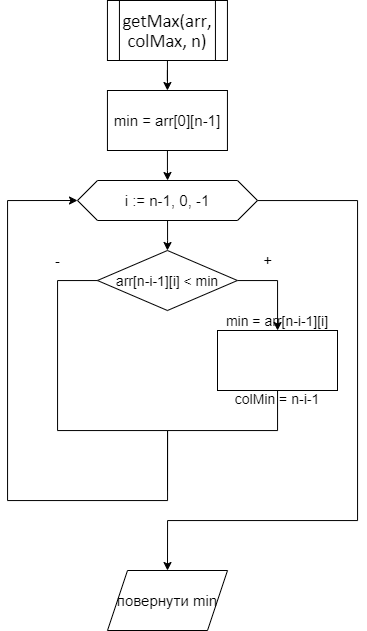
-

+

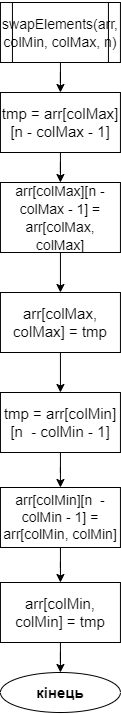
+

Крок 6(підпрограма getMax(arr, colMax, n))

Крок 7(підпрограма getMin(arr, colMin, n))



Крок 8(підпрограма swapElements(arr, colMin, colMax, n))



Код програми

using System;

namespace ASDlab9

{

class Program

{

static void Main(string[] args)

{

int n = Convert.ToInt32(Console.ReadLine());

double[,] arr = new double[n, n];

arr = generate2DArray(n);

printArray(arr);

int colMax = 0;

double max = getMax(arr, ref colMax);

Console.WriteLine("\nFirst max element of secondary diagonal is: " + max + " [" + colMax + "," + (n - 1 - colMax) + "]");

int colMin = 0;

double min = getMin(arr, ref colMin);

Console.WriteLine("\nLast min element of secondary diagonal is: " + min + " [" + colMin + "," + (n - 1 - colMin) + "]");

swapElements(ref arr, colMin, colMax);

Console.WriteLine("\nArray after swap: \n");

printArray(arr);

}

static double[,] generate2DArray(int n)

{

double[,] arr = new double[n, n];

var rand = new Random();

int j = 0;

for(int i = 0;i < arr.GetLength(0); i++)

{

while(j < arr.GetLength(1))

{

arr[i, j] = Math.Round(rand.NextDouble() % 10, 3);

j++;

}

j--;

if(i != arr.GetLength(0) - 1)

{

i++;

}

else

{

break;

}

while(j >= 0)

{

arr[i, j] = Math.Round(rand.NextDouble() % 10, 3);

j--;

}

j++;

}

return arr;

}

static void printArray(double[,] arr)

{

for(int i = 0;i < arr.GetLength(0); i++)

{

for(int j = 0;j < arr.GetLength(1); j++)

{

Console.Write(arr[i, j] + " ");

}

Console.WriteLine();

}

}

static double getMax(double[,] arr, ref int colMax)

{

int size = arr.GetLength(0);

double max = arr[size - 1, 0];

for(int i = size - 1;i >= 0; i--)

{

if(arr[i, size - 1 - i] > max)

{

max = arr[i, size -1 - i];

colMax = i;

}

}

return max;

}

static double getMin(double[,] arr, ref int colMin)

{

int size = arr.GetLength(0);

double min = arr[0, size - 1];

for (int i = size - 1; i >= 0; i--)

{

if (arr[size - 1 - i, i] < min)

{

min = arr[size - 1 - i, i];

colMin = size - 1 - i;

}

}

return min;

}

static void swapElements(ref double[,] arr, int colMin, int colMax)

{

int size = arr.GetLength(0);

double tmp = arr[colMax, size - colMax - 1];

arr[colMax, size - colMax - 1] = arr[colMax, colMax];

arr[colMax, colMax] = tmp;

tmp = arr[colMin, size - colMin - 1];

arr[colMin, size - colMin - 1] = arr[colMin, colMin];

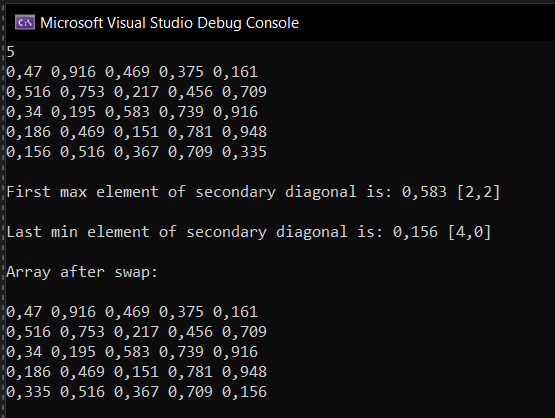
arr[colMin, colMin] = tmp;

}

}

}

Тестування



Висновок

Під час лабораторної роботи було досліджено алгоритми обходу масивів та набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій.