

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

і

Звіт

з лабораторної роботи № 9 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

«Дослідження алгоритмів обходу масивів»

Варіант 23

Виконав студент ІП-15, Мочалов Дмитро Юрійович

Перевірив Вечерковська Анастасія Сергіївна

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.

Варіант 23

Задача.

Завдання

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

1. 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).

23	Задано матрицю дійсних чисел $A[n,n]$, ініціалізувати матрицю обходом по стовбцям. Знайти середньоарифметичне значення P елементів побічної діагоналі матриці. Елементи, розташовані вище головної діагоналі і є меншими за P , обнулити.
----	--

Мат. модель

Змінна	Тип	Ім'я	Призначення
Розмірність масиву	цілий	n	Проміжні данні
Початковий масив	дійсний	arr	Проміжні данні, результат
лічильник	цілий	i	Проміжні данні
лічильник	цілий	j	Проміжні данні
Середнє арифметичне елементів побічної діагоналі	дійсний	P	Проміжні данні
Напрямок обходу масиву	цілий	dir	Проміжні данні
Заповнювач масиву	дійсний	a	Проміжні данні
Середнє арифметичне у функції	дійсний	$result$	Проміжні

Таким чином, математичне модулювання зводиться до заповнення масиву по стовпцям у функції `sett_arr`, у якій за допомогою змінної `dir` ми визначаємо напрям обходу. Якщо менше нуля, то вниз по стовпцю, якщо більше нуля – вгору по стовпцю. Заповнюємо масив значеннями змінної `a`. У функції `average` визначаємо середнє арифметичне елементів побічної діагоналі масиву `arr`. У функції `zeroing` обнуляємо значення елементів масиву які знаходяться вище головної діагоналі і менші за середнє значення елементів побічної діагоналі.

В розгалуженні “i” позначимо як &&. ++ означає збільшення змінної на 1.

Крок1: визначитись з алгоритмом

Крок2: деталізуємо алгоритм задання значень двовимірного масиву за допомогою підпрограми

Крок2: деталізуємо алгоритм обнулення елементів масиву за допомогою підпрограми.

Псевдокод

Початок

n = 5

set_arr(arr)

P = average(arr)

zeroing(arr,P)

кінець

Підпрограми

set_arr(arr)

початок

a = 1.2

dir = -1

повторити для i від 0 до n

якщо dir < 0

повторити для j від 0 до n

arr[i][j] = a++

все повторити

інакше

повторити для j від 0 до n

arr[i][j] = a++

все повторити

все інакше

все якщо

dir = dir * (-1)

все повторити

кінець

average(arr)

початок

sum = 0

повторити для i від 0 до n

sum = sum + arr[i,n-1-i]

все повторити

result = sum/n

повернути result

кінець

zeroing(arr,ave)

початок

повторити для i від 0 до n

повторити для j від 0 до n

якщо arr[i,j] < ave && i < j то

arr[i,j] = 0

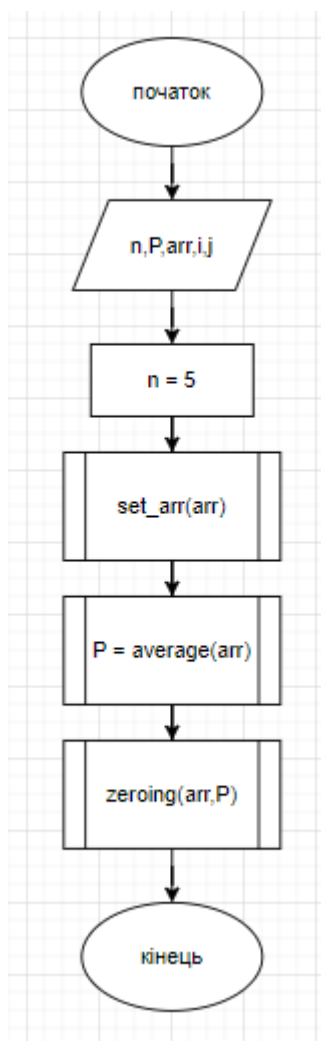
все якщо

все повторити

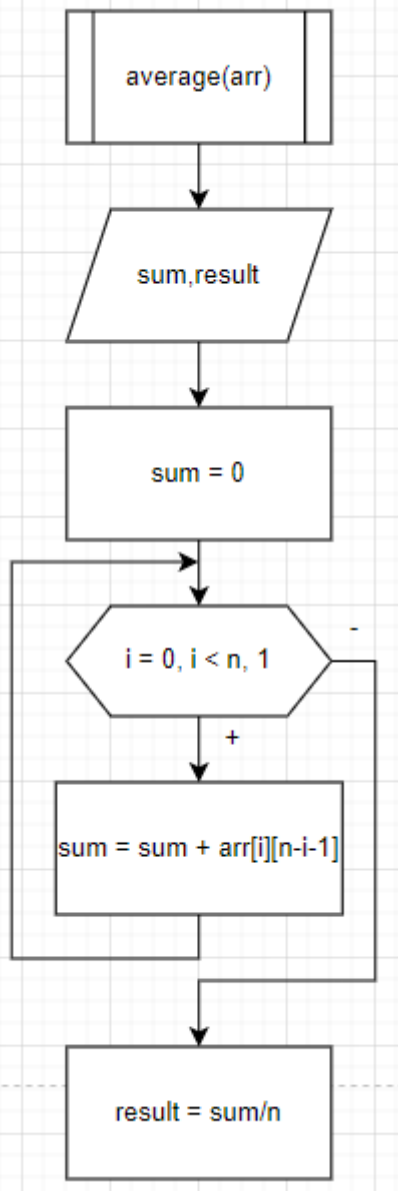
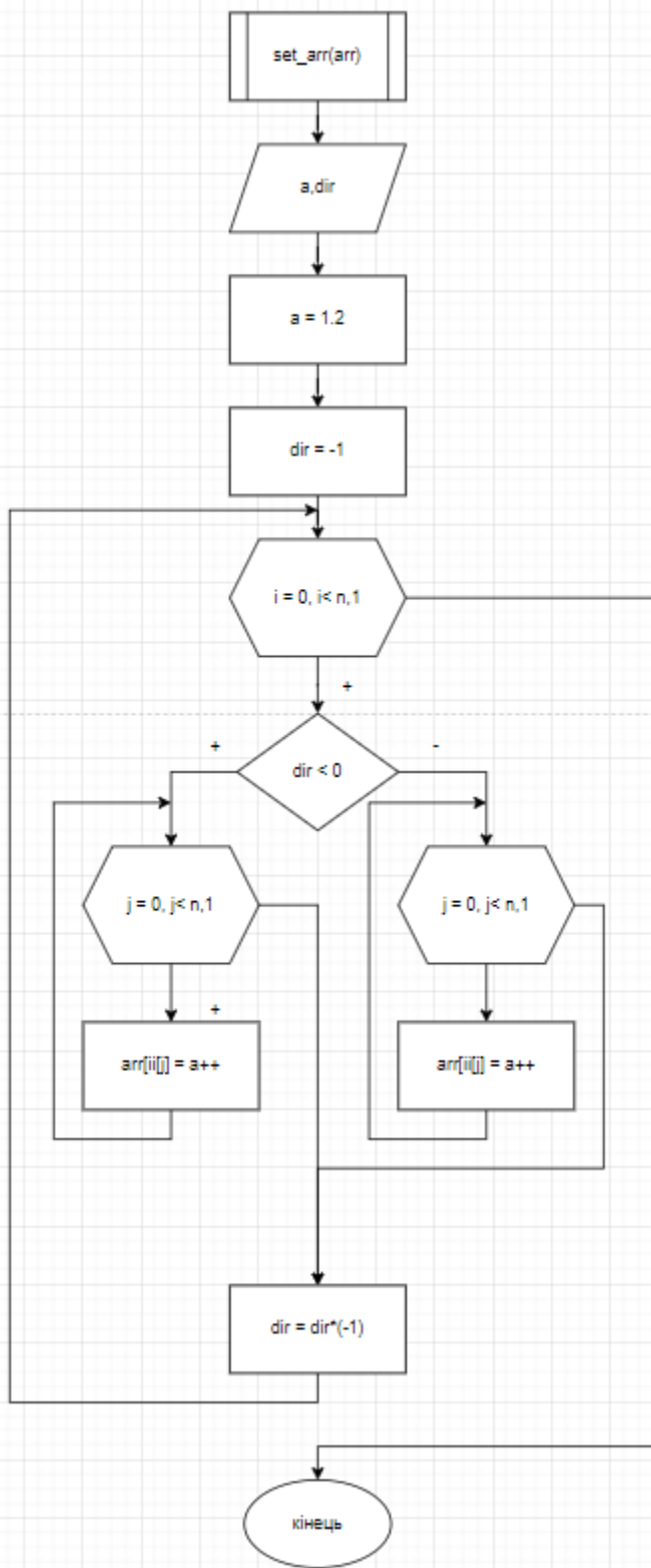
все повторити

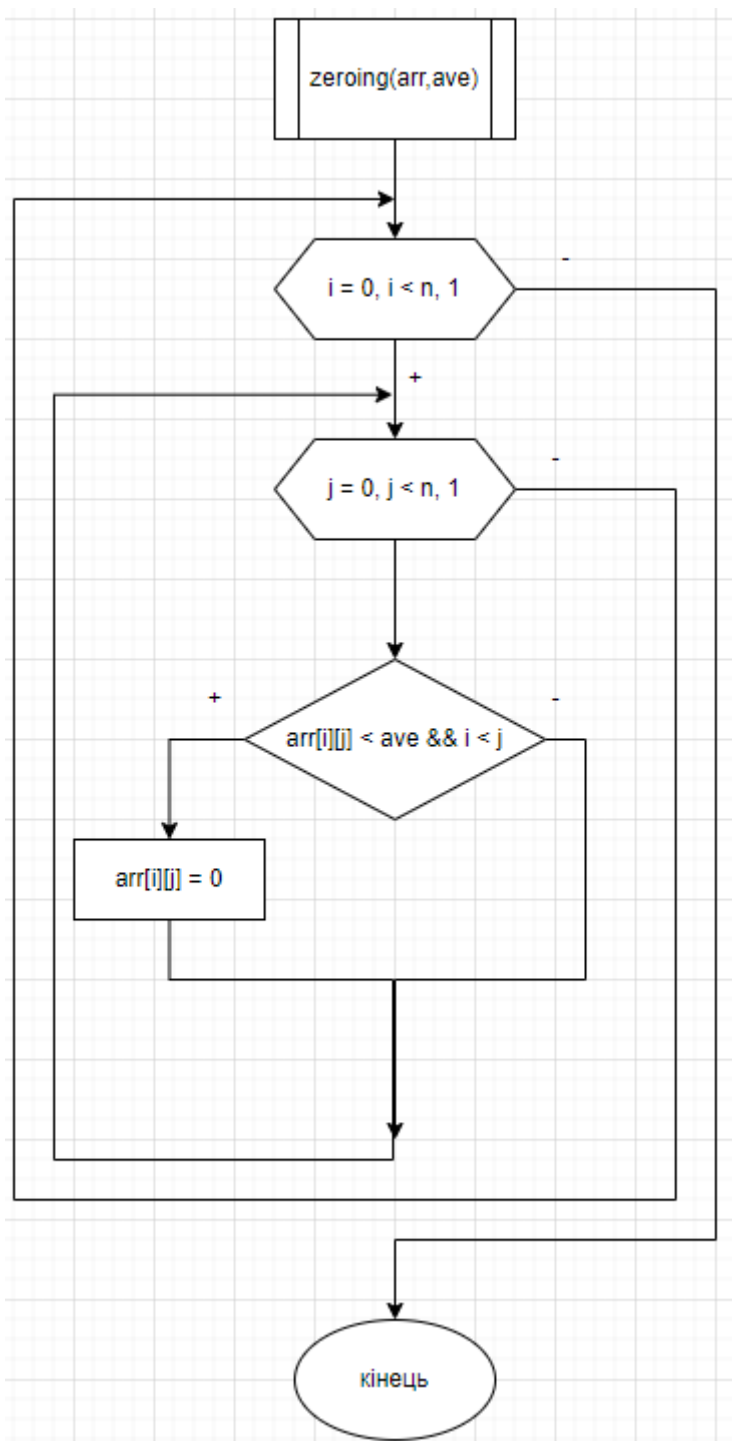
кінець

Блок-схема



Підпрограми





Код

```
1      using System;
2
3      namespace Lab9
4      {
5          internal class Program
6          {
7              public static void Main(string[] args)
8              {
9                  int n = 5;
10                 double[,] arr = new double[n, n];
11                 set_arr(arr);
12                 Console.WriteLine("Вхідний масив:");
13                 for (int i = 0; i < n; i++)
14                 {
15                     for (int j = 0; j < n; j++)
16                     {
17                         Console.Write(arr[i, j] + "\t");
18                     }
19                     Console.WriteLine();
20                 }
21
22                 double p = average(arr);
23                 Console.WriteLine("\nСереднє арифметичне: " + p);
24                 zeroing(arr, p);
25                 Console.WriteLine("\nРезультуючий масив:");
26                 for (int i = 0; i < n; i++)
27                 {
28                     for (int j = 0; j < n; j++)
29                     {
30                         Console.Write(arr[i, j] + "\t");
31                     }
32                     Console.WriteLine();
33                 }
34             }
35         }
```

```

static void set_arr(double[,] arr)
{
    double a = 1.2;
    int dir = -1;
    for (int i = 0; i < arr.GetLength(dimension: 0); i++)
    {
        if (dir < 0)
        {
            for (int j = 0; j < arr.GetLength(dimension: 1); j++)
            {
                arr[j, i] = a++;
            }
        }
        else
        {
            for (int j = arr.GetLength(dimension: 1) - 1; j >= 0 ; j--)
            {
                arr[j, i] = a++;
            }
        }

        dir *= (-1);
    }
}

```

1 usage

```

static double average(double[,] arr)
{
    double sum = 0;
    double result;
    int n = arr.GetLength(dimension: 0);
    for (int i = 0; i < n; i++)
    {
        sum += arr[i, n - 1 - i];
    }

    result = sum / n;
    return result;
}

```

```

static void zeroing(double[,] arr, double ave)
{
    for (int i = 0; i < arr.GetLength( dimension: 0); i++)
    {
        for (int j = 0; j < arr.GetLength( dimension: 1); j++)
        {
            if (arr[i, j] < ave && i < j)
            {
                arr[i, j] = 0;
            }
        }
    }
}

```

Тестування

Вхідний масив:

1,2	10,2	11,2	20,2	21,2
2,2	9,2	12,2	19,2	22,2
3,2	8,2	13,2	18,2	23,2
4,2	7,2	14,2	17,2	24,2
5,2	6,2	15,2	16,2	25,2

Середнє арифметичне: 13,2

Результуючий масив:

1,2	0	0	20,2	21,2
2,2	9,2	0	19,2	22,2
3,2	8,2	13,2	18,2	23,2
4,2	7,2	14,2	17,2	24,2
5,2	6,2	15,2	16,2	25,2

Висновок: Ми дослідили алгоритми обходу масивів, набули практичних навичок використання цих алгоритмів під час складання програмних специфікацій.