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 15

Виконав студент ІП-15, Костін Вадим Анатолійович
(шифр, прізвище, ім'я, по батькові)

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

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 15

Задача

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).

15	Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках визначити в ній присутність заданого дійсного числа X і його місцезнаходження. Знайти кількість елементів, більших за X, під головною діагоналлю.

Постановка задачі

За умовою задачі дано двовимірний масив розмірністю m x n. Масив обходимо змійкою та шукаємо в ньому елемент X. Потім під головною діагоналлю знаходимо елементи більші за X.

Математична модель

Генерувати двовимірний масив A будемо за допомогою підпрограми inputA(), яка надає кожному елементу матриці рандомне число від -100 до 100. Перевіряти наявність елементу X в масиві та виведення його координат буде відбуватися за допомогою підпрограми findX(). За допомогою підпрограми findUnderDiagonalA() знайдемо кількість елементів більших за X під головною діагоналлю.

Змінна	Тип	Ім’я	Призначення
Матриця	Дійсний	A	Початкові дані

Елемент, який буде шукати програма	Дійсний	X	Початкові дані
Підпрограма що генерує масив A	Процедура	inputA()	Проміжні дані
Підпрограма що знаходить елемент X в матриці A, обходячи її по рядках	Процедура	findX()	Результат
Підпрограма що знаходить елементи більші за X під головною діагоналлю, повертає ціле значення	Процедура	findUnderDiagonalA()	Результат
Лічильник	Цілочисельний	i	Проміжні дані
Лічильник	Цілочисельний	j	Проміжні дані
Кількість елементів масиву, рівних X	Цілочисельний	count	Проміжні дані
Кількість елементів більших за X під головною діагоналлю	Цілочисельний	amount	Проміжні дані

Для виразу x = x + y будемо використовувати x += y

Для виразу x = x + 1 будемо використовувати x++

Для надання рандомного значення елементу будемо використовувати random(-100, 100)

Крок 1 Деталізуємо основні дії

Крок 2 Деталізуємо дію генерації масиву A

Крок 3 Деталізуємо дію перевірки наявності елементу X в масиві A

Крок 4 Деталізуємо дію знаходження кількості елементів більших за X під головною діагоналлю

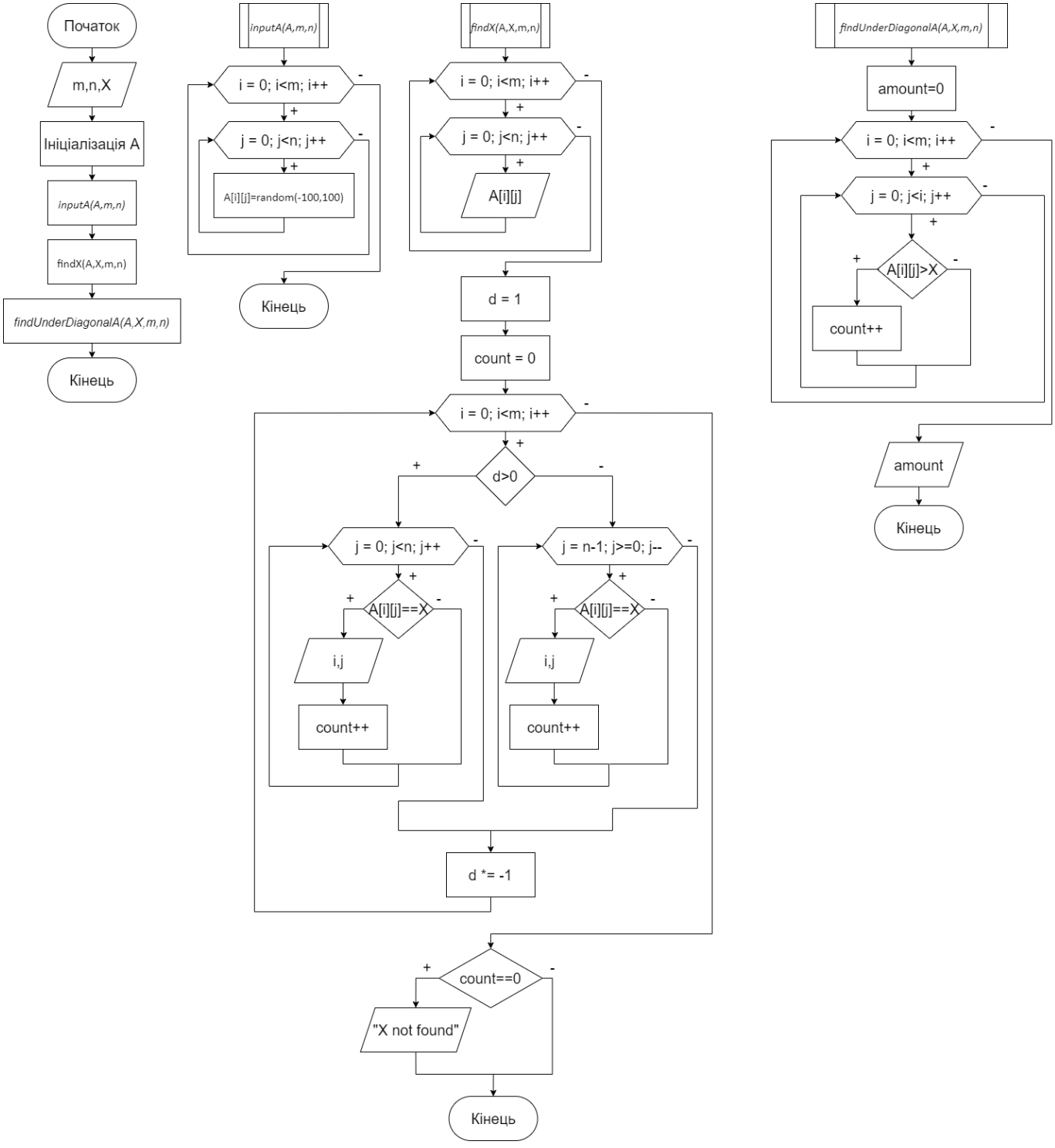
Псевдокод

Крок 1 **Початок** Ініціалізація A,m,n,X Деталізуємо дію генерації масиву A Деталізуємо дію перевірки наявності елементу X в масиві A Деталізуємо дію знаходження кількості елементів більших за X під головною діагоналлю **Кінець**	*Підпрограма inputA()* *Підпрограма findX()* *Підпрограма findUnderDiagonalA()*
Крок 2 **Початок** Ініціалізація A,m,n,X *inputA(A,m,n)* Деталізуємо дію перевірки наявності елементу X в масиві A Деталізуємо дію знаходження кількості елементів більших за X під головною діагоналлю **Кінець**	*Підпрограма inputA(A,m,n)* **Початок** **Повторити** для i від 0 до m **Повторити** для j від 0 до n A[i][j]=random(-100,100) **Все повторити** **Все повторити** **Кінець** *Підпрограма findX()* *Підпрограма findUnderDiagonalA()*
Крок 3 **Початок** Ініціалізація A,m,n,X *inputA(A,m,n)* findX(A,X,m,n) Деталізуємо дію знаходження кількості елементів більших за X під головною діагоналлю **Кінець**	*Підпрограма inputA(A,m,n)* **Початок** **Повторити** для i від 0 до m **Повторити** для j від 0 до n A[i][j]=random(-100,100) **Все повторити** **Все повторити** **Кінець** *Підпрограма findX(*A,X,m,n*)* **Початок** **Повторити** для i від 0 до m **Повторити** для j від 0 до n Виведення A[i][j] **Все повторити** **Все повторити** d=1 count=0 **Повторити** для i від 0 до m **Якщо** d>1 **Повторити** для j від 0 до n **Якщо** A[i][j]==X Виведення i, j count++ **Все якщо** **Все повторити** **Інакше**

	Повторити для j від n-1 до 0 **Якщо** A[i][j]==X Виведення i, j count++ **Все якщо** **Все повторити** **Все повторити** **Якщо** count==0 Виведення “X not found” **Все якщо** **Кінець** *Підпрограма findUnderDiagonalA()*
Крок 4 **Початок** Ініціалізація A,m,n,X *inputA(A,m,n)* findX(A,X,m,n) *findUnderDiagonalA(A,X,m,n)* **Кінець**	*Підпрограма inputA(A,m,n)* **Початок** **Повторити** для i від 0 до m **Повторити** для j від 0 до n A[i][j]=random(-100,100) **Все повторити** **Все повторити** **Кінець** *Підпрограма findX(*A,X,m,n*)* **Початок** **Повторити** для i від 0 до m **Повторити** для j від 0 до n Виведення A[i][j] **Все повторити** **Все повторити** d=1 count=0 **Повторити** для i від 0 до m **Якщо** d>1 **Повторити** для j від 0 до n **Якщо** A[i][j]==X Виведення i, j count++ **Все якщо** **Все повторити** **Інакше** **Повторити** для j від n-1 до 0 **Якщо** A[i][j]==X Виведення i, j count++ **Все якщо** **Все повторити** **Все повторити** **Якщо** count==0 Виведення “X not found” **Все якщо** **Кінець** *Підпрограма findUnderDiagonalA(A,X,m,n)* **Початок** amount = 0

	Повторити для i від 0 до m **Повторити** для j від 0 до i **Якщо** A[i][j]>X amount++ **Все повторити** **Все повторити** Виведення amount **Кінець**

Блок-схема



Код програми

```
#include <iostream>
#include <time.h>
#include <iomanip>

using namespace std;

void inputA(float *a[],int m,int n);

void findX(float *a[], float x, int m, int n);

void findUnderDiagonalA(float *a[], float x, int m, int n);

int main(){
    srand(time(NULL));
    int m,n;
    float X;

    cout << "Enter m and n: ";
    cin >> m >> n;

    cout << "Enter X: ";
    cin >> X;

    float A[m][n];
    float *p[m];
    for (int i = 0; i < n; i++)
    {
        p[i] = A[i];
    }

    inputA(p,m,n);

    findX(p,X,m,n);

    findUnderDiagonalA(p,X,m,n);

    system("pause");
}

void inputA(float *a[],int m,int n){
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            a[i][j] = (float)(rand()%200 - 100);
        }
    }
}

void findX(float *a[], float x, int m, int n){
    for (int i = 0; i < m; i++)
    {
```

```
        for (int j = 0; j < n; j++)
        {
            cout << setw(4) << a[i][j];
        }
        cout << endl;
    }
    cout << endl;

    int d = 1, count = 0;
    for (int i = 0; i < m; i++)

    {
        if(d>0){
            for (int j = 0; j < n; j++)
            {
                if(a[i][j]==x){
                    cout <<"X found at A[" << i+1 <<"]["<< j+1 <<"]\n";
                    count++;
                }
            }
        } else {
            for (int j = n-1; j >= 0; j--)
            {
                if(a[i][j]==x){
                    cout <<"X found at A[" << i+1 <<"]["<< j+1 <<"]\n";
                    count++;
                }
            }
        }
        d *= -1;
    }
    if(count==0)
        cout << "X not found\n\n";
}

void findUnderDiagonalA(float *a[], float x, int m, int n){
    int amount = 0;
    cout << "\nElements bigger then X under the diagonal: \n";
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < i; j++)
        {
            if(a[i][j]>x){
                cout << a[i][j] << " ";
                amount++;
            }
        }
    }
    cout <<"\nAmount: " << amount << endl;

}
```

Результат роботи програми

```
Enter m and n: 10 10
Enter X: 47
  78  37  50 -28 -79  74  52 -57  18 -94
 -78  83 -17  89  56 -26 -61 -13  68  82
 -62 -26 -59 -26  29 -42-100  53 -57  59
 -95 -27  46  -3 -39  -5 -10  35  46 -76
 -53  96  48   7  13  -9 -69  42 -71 -23
  33  90  30 -91  89 -57 -20 -73  16 -49
  36  11  39   9 -84 -92  76  69  10  56
   6 -78  19 -18  12  88 -36  44  45  24
  65  18  13  41 -92  26  46  60 -40 -68
  29  97 -74 -83  47  75  88 -72 -92 -46

X found at A[10][5]

Elements bigger then X under the diagonal:
96 48 90 89 88 65 60 97 75 88
Amount: 10
Press any key to continue . . .
```

```
Enter m and n: 5 5
Enter X: 50
 -33 -80  82 -53 -61
  28  46  52 -28 -87
  20  -6 -11 -48  -5
 -58  23 -74 -14  67
 -92 -31 -92  13  31

X not found


Elements bigger then X under the diagonal:

Amount: 0
Press any key to continue . . .
```

Висновки

Протягом дев’ятої лабораторної роботи ми дослідили алгоритми обходу масивів та набули практичних навичок використання цих алгоритмів під час складання програмних специфікацій.