Додаток 1

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

Звіт

з лабораторної роботи № 9

з дисципліни «Алгоритми та структури даних-1. Основи алгоритмізації»

«ДОСЛІДЖЕННЯ АЛГОРИТМІВ ОБХОДУ МАСИВІВ»

Варіант 8

Виконав ІП-15, Дацьо Іван Іванович

студент (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.

Індивідуальне завдання :

Варіант 8

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).

8	Задано матрицю дійсних чисел A[n,n], ініціалізувати матрицю обходом по рядках. На головній діагоналі матриці знайти перший від'ємний і останній додатний елементи, та поміняти їх місцями з елементами побічної діагоналі .

1. Постановка задачі.

Створюємо масив дійсного типу розміром nxn . Ініціалізуємо матрицю обходом по рядках. Знаходимо серед елементів головної діагоналі перший від'ємний та останній додатній та міняємо їх на відповідні елементи побічної діагоналі.

2.Побудова математичної моделі

Змінна	*Тип*	*Ім'я*	*Призначення*

Кількість рядків/ стовпців матриці	Цілий	n	Початкові дані
Лічильник1 циклу	Цілий	i	Параметр циклу
Лічильник2 циклу	Цілий	j	Параметр циклу
Лічильник3 циклу	Цілий	k	Параметр циклу
Двовимірний масив	Дійсний	a	Проміжні дані
Індекс першого від'ємного елемента гол.діагоналі	Цілий	ivid	Проміжні дані
Індекс останнього додатного елемента гол.діагоналі	Цілий	ios	Проміжні дані
Індекс елемента побічної діагоналі	Цілий	z	Проміжні дані
Змінна зміни розташування	Цілий	x	Проміжні дані
Присвоєння значень двовимірномому масиву	Процедура	input	Процедура
Вивід двовимірного масиву	Процедура	output	Процедура
Пошук індексу першого від'ємного ел. Головної діагоналі	Процедура	elem1	Процедура
Пошук індексу останнього додатного ел. Головної діагоналі	Процедура	elem2	Процедура
Зміна місцями пер. від'єм. та елемента побічної діагоналі	Процедура	replace1	Процедура
Зміна місцями ост. додат. та елемента побічної діагоналі	Процедура	replace1	Процедура

Вивід матриці	Процедура	result	Процедура

Індексацію масиву розпочинаємо із 0 .

Генерацію випадкових чисел позначаємо rand , для генерації дійсних чисел використовуємо формулу :

(float)(rand()%200 + 1)/10 -10

Для позначення збільшення на 1 використовуємо знаки : ++

Математичне формулювання задачі зводиться до таких дій :

Випадковим чином генеруємо значення масиву a в функці **input** обходом по рядках.Для цього створюємо умову для генеруваня парних та не парних рядків .Генерування випадкових значення відбуватиметься за допомогою класу Rand . Наступною дією буде виведення цього двовимірного масиву , для виразності в функції **output** за допомогою двох циклів .Наступною дією буде пошук першого від'ємного елемента головної діагоналі у функції **elem1**. Для цього створюємо параметр циклу k і присвоюємо йому 0. В циклі while перевіряємо чи значення більше або рівне 0.В разі виконання умови збільшуємо параметр на 1. Присвоюємо значення індексу k. Виводимо його для виразності . В функції **elem2** за допомогою шукаємо елементи на гол .діагоналі більші за 0 , присвоюємо змінній ios значення i. Наступною дією у функції **replace1** створюємо змінну заміни і міняємо місцями перший від'ємний елемент головної та відповідний йому елемент побічної діагоналі. .У функції **replace2** створюємо змінну заміни і міняємо місцями останній додатній елемент головної та відповідний йому елемент побічної діагоналі.

Функцією **result** виводимо нову матрицю .

3.Розв'язання

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

Крок 1. Визначимо основні дії .

Крок 2.Деталізуємо дію генерування двовимірного масиву за допомогою функції.

Крок 3. Деталізуємо дію виведення матриці для виразності за допомогою фунцкії.

Крок 4. Деталізуємо дію пошуку індексу першого від'ємного елемента головної діагоналі за допомогою функції.

Крок 5. Деталізуємо дію пошуку індексу останнього додатного елемента головної діагоналі за допомогою функції .

Крок 6. Деталізуємо дію зміни місцями першого від'ємного елемента та елемента побічної діагоналі за допомогою функції .

Крок 7. Деталізуємо дію зміни місцями останнього додатного елемента головної діагоналі та елемента побічної діагоналі за допомогою функції .

Крок 8. Деталізуємо дію виведення кінцевої матриці для виразності за допомогою функції .

4. Псевдокод

Основна програма :

початок

n = 6

ivid , ios

a [n][n]

input (n , a)

output (n , a)

elem1 (n , a ,ivid)

elem2 (n , a ,ios)

replace1 (n , a ,ivid)

replace2 (n , a ,ios)

result (n , a)

кінець

Підпрограми :

input (n , a)

для i **від** 0 **до** n **повторити**

якщо i % 2 == 0

то для j **від** 0 **до** n **повторити**

a[i][j] = (rand()%200 + 1)/10 -10

все повторити

інакше

для j **від** n-1 **до -1 повторити**

a [i][j] =(rand()%200 + 1)/10 -10

все повторити

все якщо

кінець

output (n , a)

для i **від** 0 **до** n **повторити**

для j **від** 0 **до** n **повторити**

Вивести a[i][j]

все повторити

все повторити

кінець

elem1 (n , a , ivid)

k = 0

Поки a[k][k] >= 0 **повторити**

k = k + 1

все повторити

ivid = k

Вивести ivid

Кінець

elem2 (n , a , ios)

для i **від** 0 **до** n **повторити**

якщо a [i][i] >= 0

то ios = i

все якщо

все повторити

Кінець

replace1 (n , a ,ivid)

```
z = n-1 - ivid
x = a[ivid][ivid]
a[ivid][ivid] = a[ivid][z]
a[ivid][z] = x
```

кінець

replace2 (n , a ,ios)

```
z = n-1 - ios
x = a[ios][ios]
a[ios][ios] = a[ios][z]
a[ios][z] = x
```

кінець

result (n , a)

для i **від** 0 **до** n **повторити**

для j **від** 0 **до** n **повторити**

Вивести a[i][j]

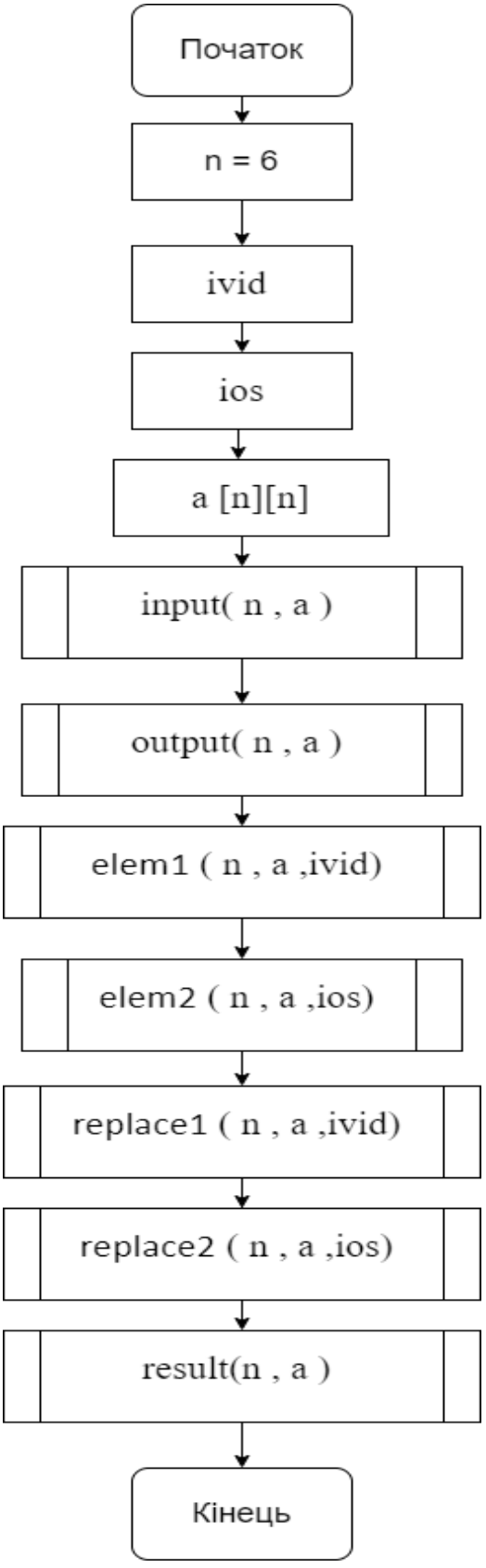
все повторити

все повторити

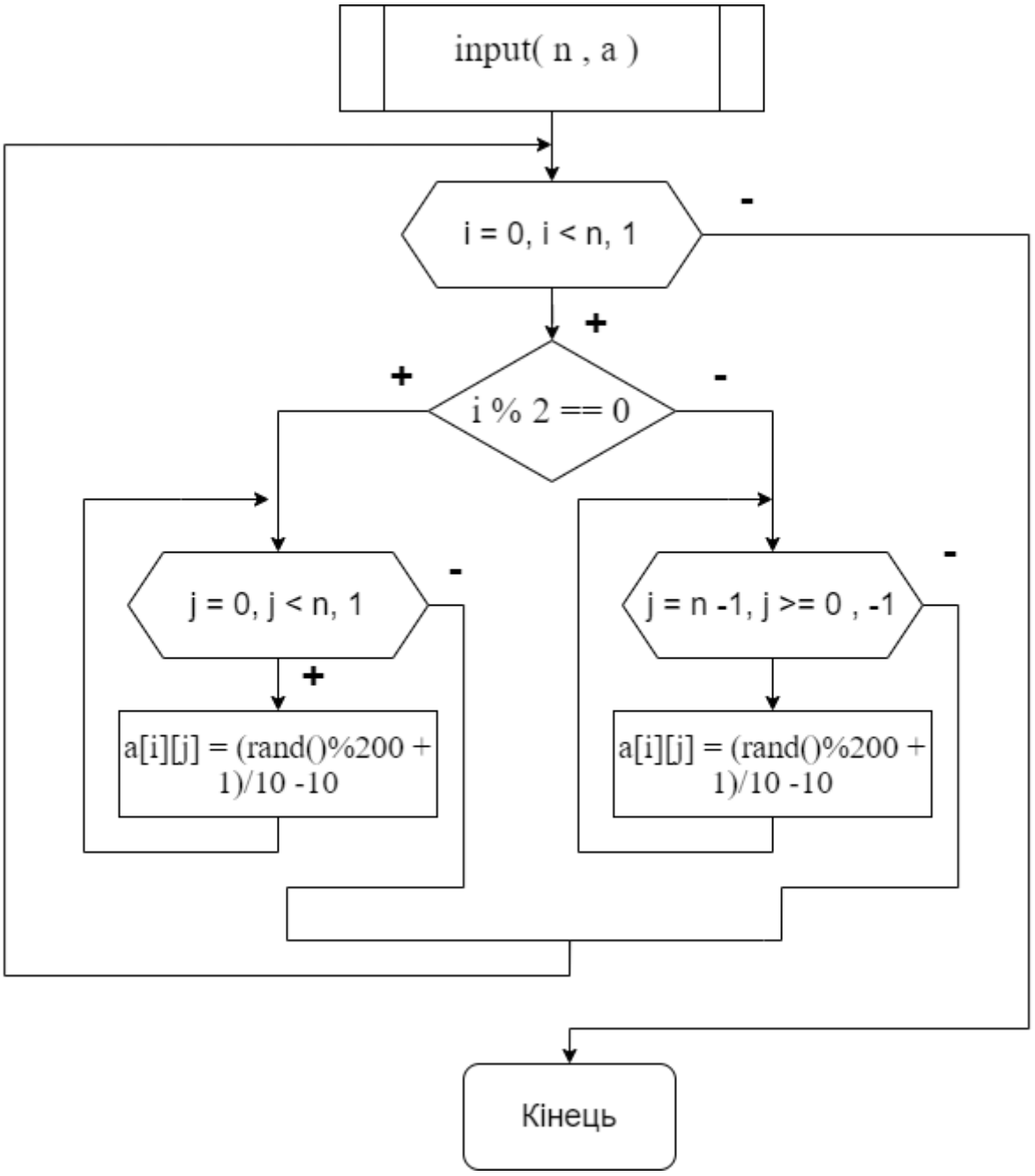
кінець

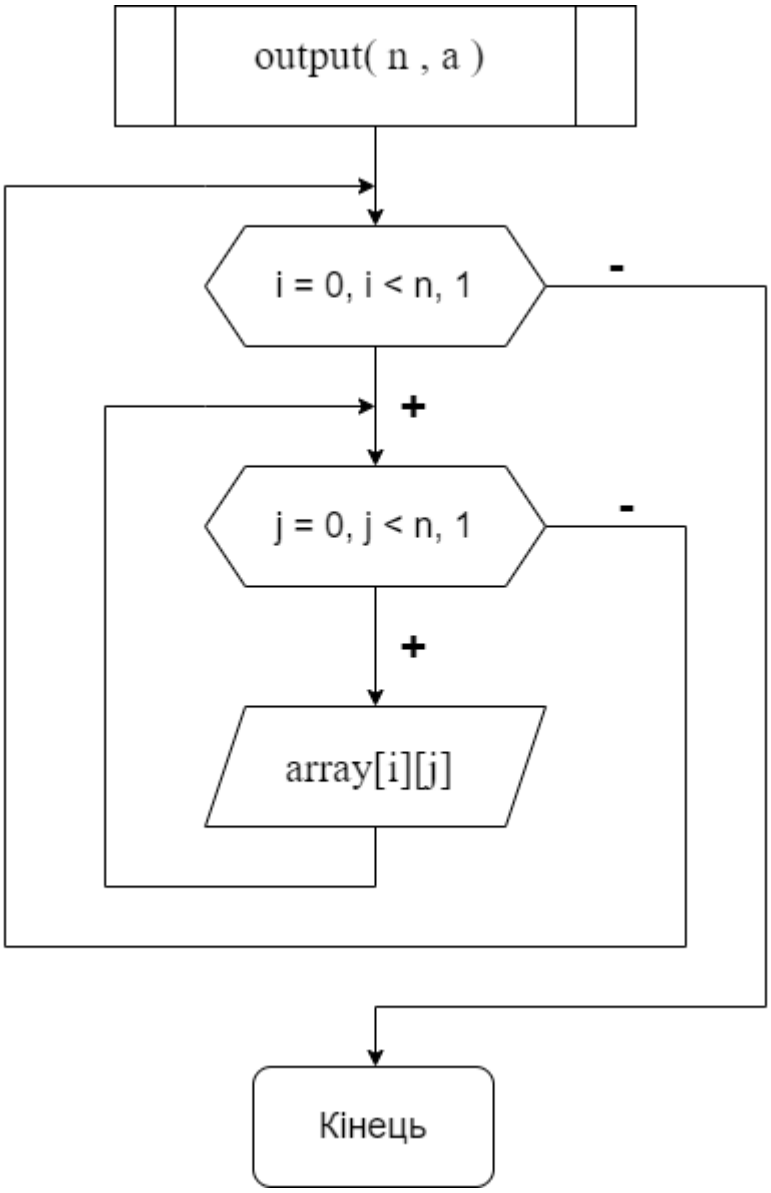
5. Блок-схема алгоритму

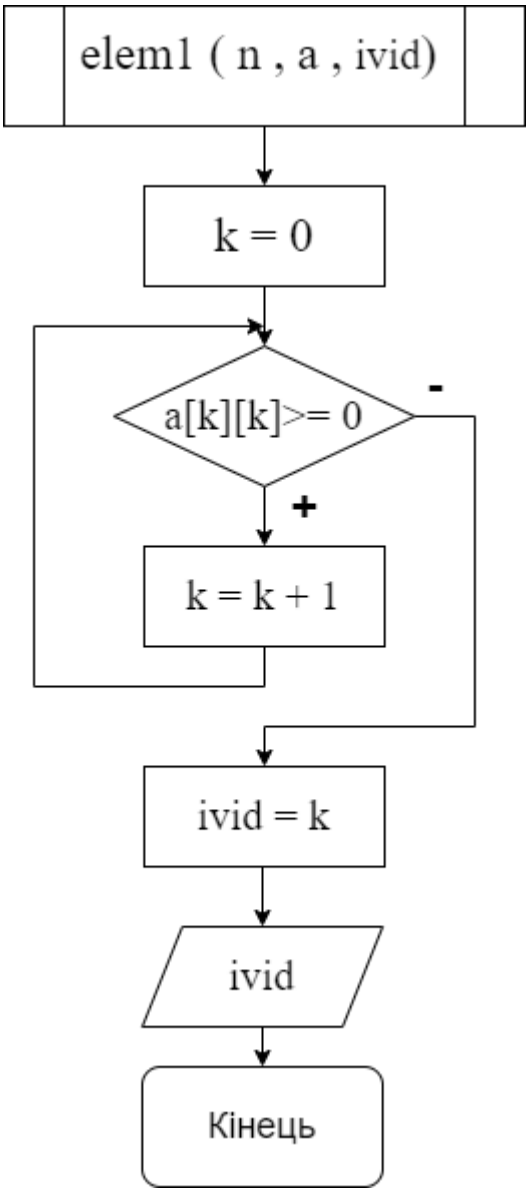
Основна програма :

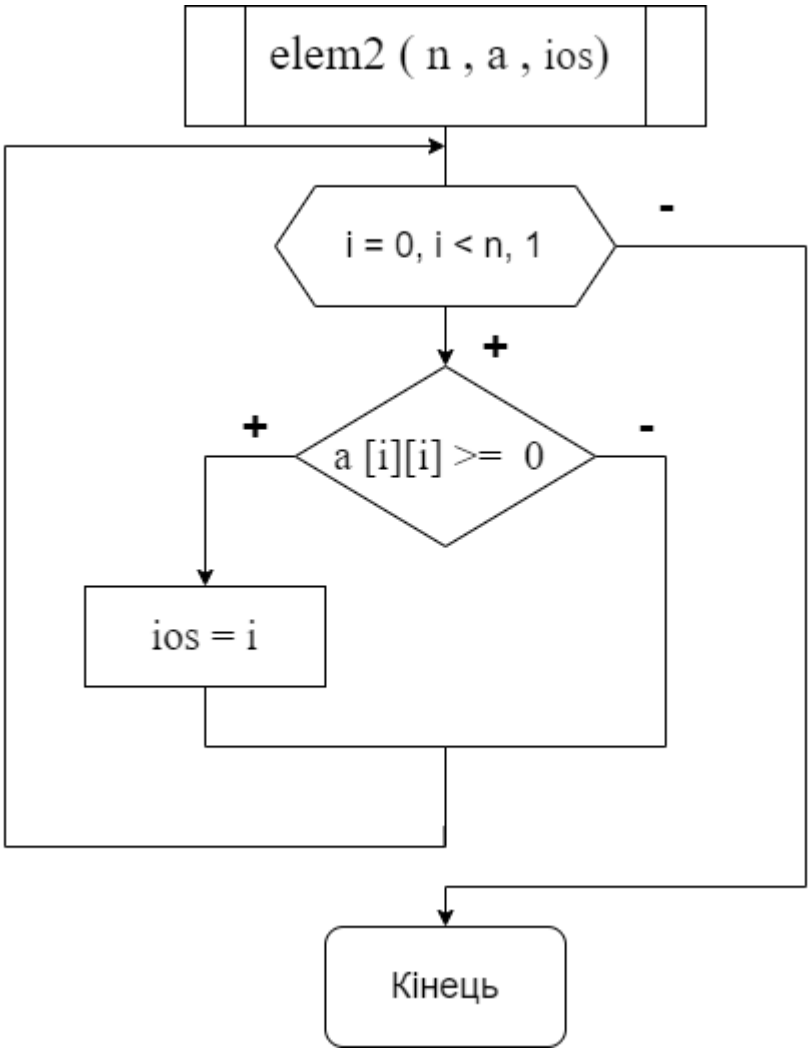


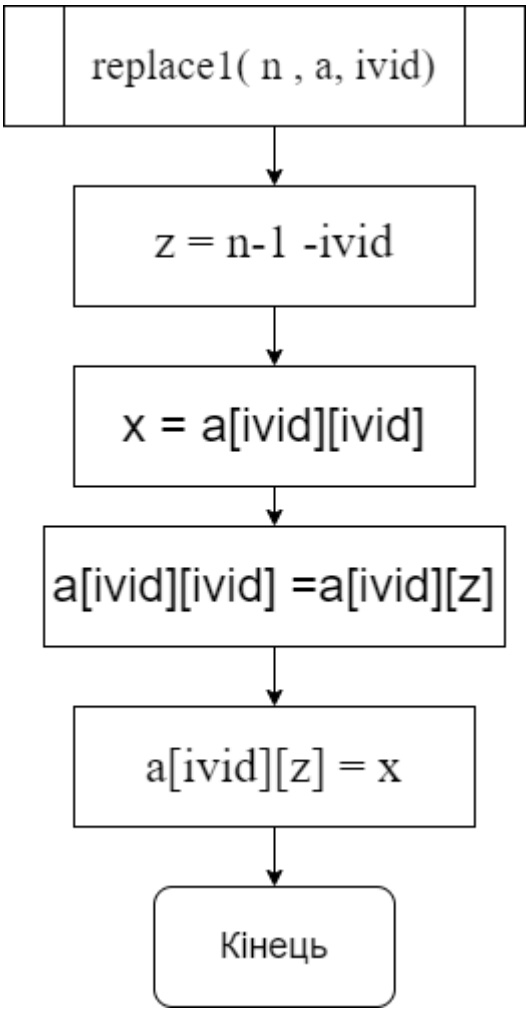
Підпрограми :

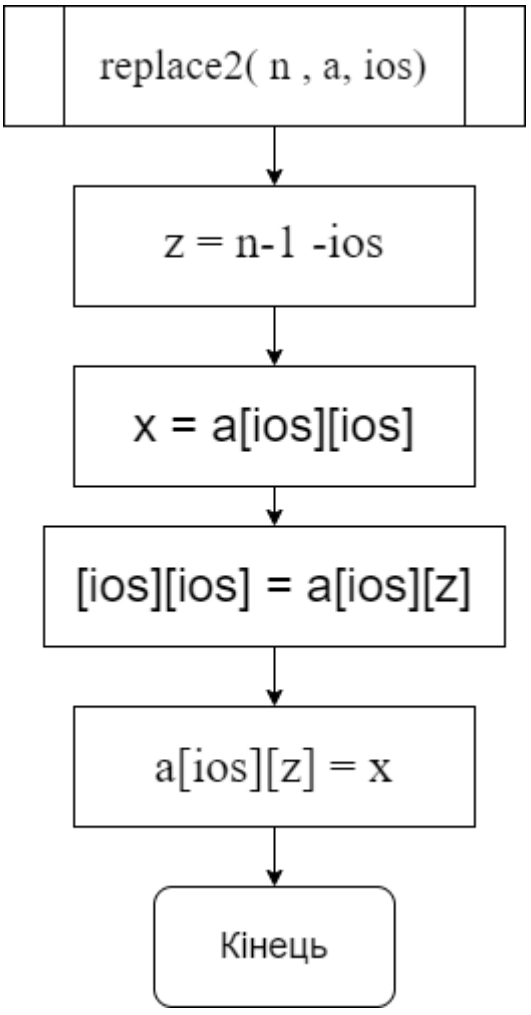

input(n , a)
i = 0, i < n, 1
-
+
i % 2 == 0
+
-
j = 0, j < n, 1
-
+
a[i][j] = (rand()%200 + 1)/10 -10
j = n -1, j >= 0 , -1
-
a[i][j] = (rand()%200 + 1)/10 -10
Кінець

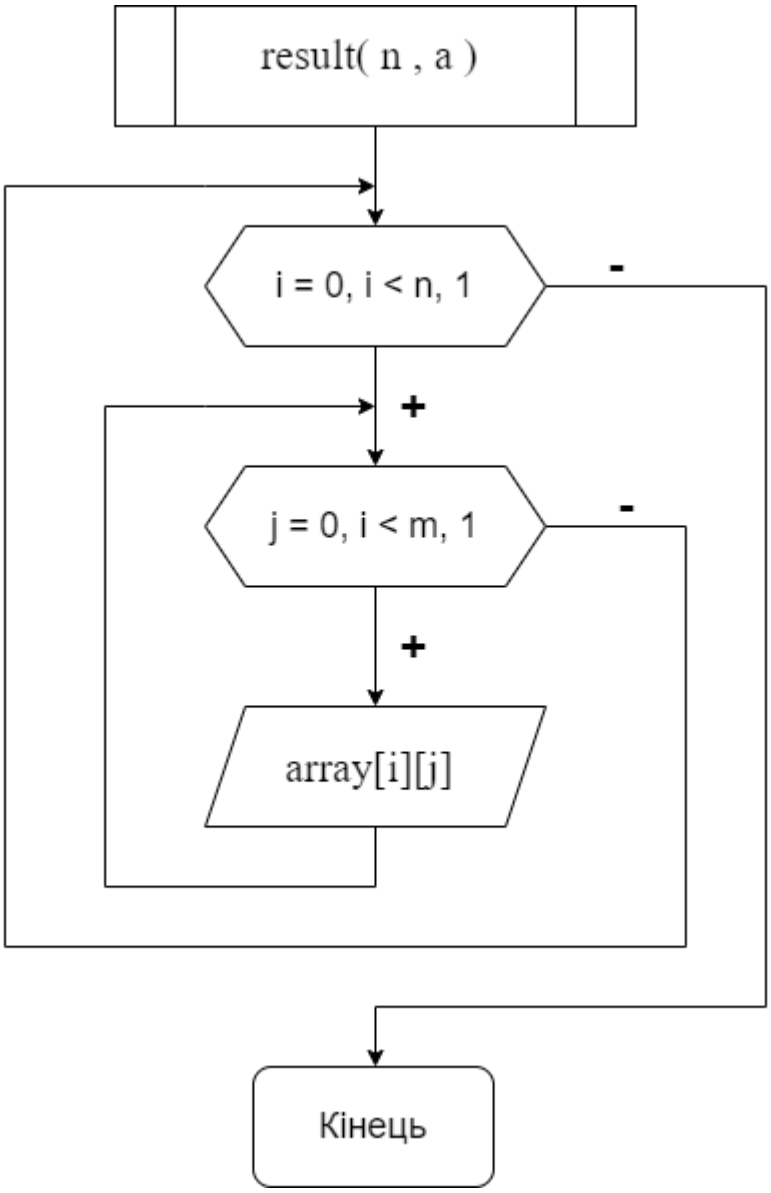

output(n , a)
i = 0, i < n, 1
-
+
j = 0, j < n, 1
-
+
array[i][j]
Кінець


elem1 (n , a , ivid)
k = 0
a[k][k]>= 0
-
+
k = k + 1
ivid = k
ivid
Кінець


elem2 (n , a , ios)
i = 0, i < n, 1
-
+
+
a [i][i] >= 0
-
ios = i
Кінець


replace1(n , a, ivid)
z = n-1 -ivid
x = a[ivid][ivid]
a[ivid][ivid] =a[ivid][z]
a[ivid][z] = x
Кінець


replace2(n , a, ios)
z = n-1 -ios
x = a[ios][ios]
[ios][ios] = a[ios][z]
a[ios][z] = x
Кінець



6. Код програми(c++)

```cpp
#include <iostream>
#include <stdlib.h>
using namespace std ;
void input (int n ,double a[6][6]){
    for ( int i = 0; i < n; i++){
        if (i % 2 == 0 ){
            for ( int j = 0 ; j < n; j++){
                a[i][j] = (float)(rand()%200 + 1)/10 -10;
            }

        }
        else{
            for( int j = n-1; j >= 0 ; j--){
                a[i][j] = (float)(rand()%200 + 1)/10 -10;
            }
        }
    }
}
void output ( int n, double a[6][6]){
    cout << "Given matrix: "<< endl;
    for ( int i = 0; i < n; i++){
        cout<< endl;
        for ( int j = 0 ; j < n; j++){
            cout << a[i][j] <<"\t";
        }
    }
    cout << endl;
}
```

```
29  void elem1 ( int n , double a[6][6] , int &ivid ){
30      int k = 0;
31      while (a[k][k] >=0){
32          k++;
33      }
34      ivid = k;
35      cout <<"index =  " <<ivid << endl;
36  }
37  void elem2(int n , double a[6][6] ,int &ios){
38      for (int i = 0; i < n; i++){
39          if (a[i][i] >= 0){
40              ios = i ;
41          }
42      }
43  }
44  void replace1 (int n , double a[6][6] , int &ivid){
45      int z = n-1 - ivid;
46      float x = a[ivid][ivid];
47      a[ivid][ivid] = a[ivid][z];
48      a[ivid][z] = x;
49  }
50  void replace2 (int n , double a[6][6] , int &ios ){
51      int z = n-1 - ios;
52      float x = a[ios][ios];
53      a[ios][ios] = a[ios][z];
54      a[ios][z] = x;
55  }
```

```
56  void result(int n , double a[6][6]){
57      cout << "Final matrix: "<<endl;
58      for( int i = 0 ; i < n ; i++){
59          for ( int j = 0; j < n; j++){
60              cout << a[i][j] <<"\t";
61          }
62          cout <<endl;
63      }
64  }
65  int main(){
66      srand(time(NULL));
67      const int n = 6 ;
68      int ivid , ios ;
69      double a[n][n];
70      input( n, a );
71      output( n, a);
72      elem1 ( n, a , ivid);
73      elem2 ( n, a , ios);
74      replace1 (n , a, ivid);
75      replace2 (n , a, ios);
76      result( n, a);
77  }
```

Випробування :

```
Given matrix:

-6.2    5.1     9.1     -6.5    -8.1    8.2
-5.8    -9.1    -0.3    -3      -2.9    8.6
0.8     -1.1    7.2     -8.6    -8.4    2
-2      -8.2    4.7     -7.6    3.8     -6.7
6.1     9.3     -6.9    -2.4    8.7     -4.7
7.3     -9      4.4     5.5     -7.6    5.4
index =  0
Final matrix:
8.2     5.1     9.1     -6.5    -8.1    -6.2
-5.8    -9.1    -0.3    -3      -2.9    8.6
0.8     -1.1    7.2     -8.6    -8.4    2
-2      -8.2    4.7     -7.6    3.8     -6.7
6.1     9.3     -6.9    -2.4    8.7     -4.7
5.4     -9      4.4     5.5     -7.6    7.3
```

7.Висновок

Ми дослідили алгоритми обходу масивів , набули практичних навичок використання алгоритмів під час складання програмних специфікацій .В результаті виконання лабораторної роботи ми отримали програму для заповнення двовимірного масиву методом обходу рядків , знаходження першого від'ємного елемента головної діагоналі , пошуку останнього додатного елемента головної діагоналі та обміні цих елементів із відповідними побічної діагоналі розділивши задачу на 8 кроків.