# 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 і структури даних

»

Варіант 23

Виконав студент Панченко Сергій Віталійович

(шифр, прізвище, ім'я, по батькові)

Перевірив Мартинова Оксана Петрівна

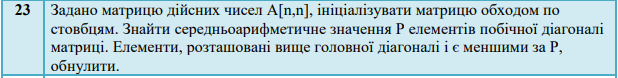
( прізвище, ім'я, по батькові)

Київ 202

## Лабораторна робота 8

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

**Постановка:**



**Математична модель**:

**Складемо таблицю імен змінних**

|  |  |  |  |
| --- | --- | --- | --- |
| Зміна | Тип | Ім’я | Призначення |
| matrix | Int\*\* | матриця | Проміжний результат |
| n | int | розряд | Вхідне значення |
| avg | double | Сер знач | результат |
| one\_d\_arr | Int\* | Рядок | Проміжний результат |
| diag\_els | Int\* | елементи побіч діаг | Проміжний результат |
| I | Int | Змінна для циклу | Проміжний результат |
| J | Int | Змінна для циклу | Проміжний результат |
| K | Int | Змінна для циклу | Проміжний результат |

Заповнивши матрицю, перейдемо по її побічній діагоналі, відберемо елементи в масив, розрахуємо середнє значення. Перейдемо по гол. Діаг., ті елементи, що вище за неї та менші сер. знач, то заміними на нуль.

**Псевдокод і блок-схеми:**

Крок 1: Визначимо основні дії

Крок 2: деталізуємо заповнення arr

Крок 3: деталізуємо розрахунок avg

Крок 4: деталізуємо анулювання елементи

Крок 1:

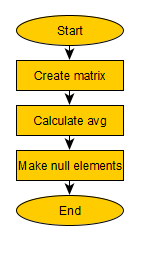
Start

Create matrix

Calculate avg

Make null elements

End



Крок 2:

Start

n = enter\_n();

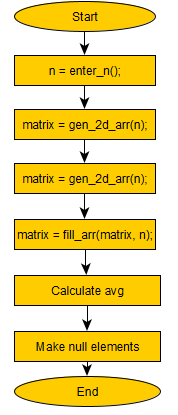
matrix = gen\_2d\_arr(n);

matrix = fill\_arr(matrix, n);

Calculate avg

Make null elements

End



Крок 3:

Start

n = enter\_n();

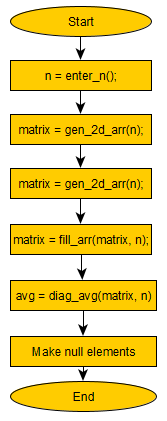
matrix = gen\_2d\_arr(n);

matrix = fill\_arr(matrix, n);

avg = diag\_avg(matrix, n)

Make null elements

End



Крок 4:

Start

n = enter\_n();

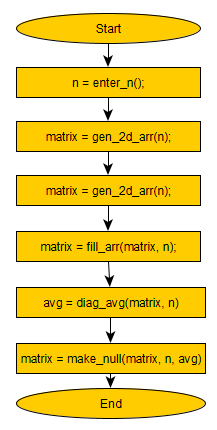
matrix = gen\_2d\_arr(n);

matrix = fill\_arr(matrix, n);

avg = diag\_avg(matrix, n)

matrix = make\_null(matrix, n, avg)

End



enter\_n():

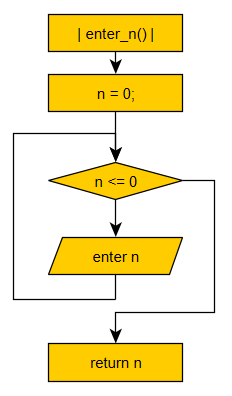
n = 0;

while (n <= 0):

enter n

while end

return n



gen\_2d\_arr(n):

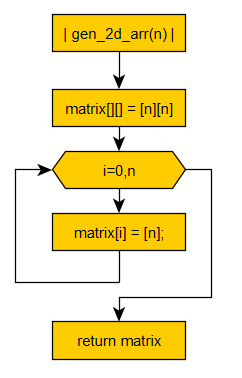
matrix[][] = [n][n]

while (i=0,n):

matrix[i] = [n];

while end

return matrix;



fill\_arr(matrix, n):

while (j = 0,n):

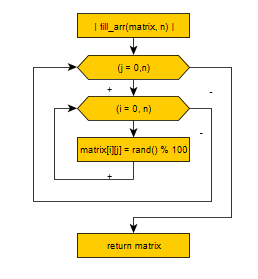
while (i = 0, n):

matrix[i][j] = rand() % 100

while end

while end

return matrix;



diag\_avg(matrix, n):

diag\_els =[n];

k=0

while (j = n-1,0,-1):

while (i = 0,n):

if (i == n-j-1):

diag\_els[k] = matrix[i][j];

k++;

if end

while end

while end

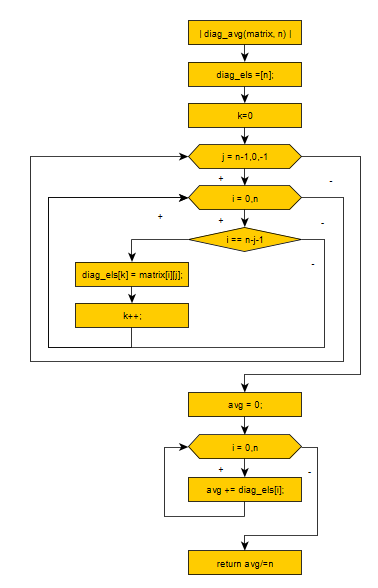
avg = 0;

while (i = 0,n):

avg += diag\_els[i];

while end

return avg /= n;



make\_null(matrix, n, avg):

while (j = 0,n)

while (i = 0,n)

if (i < j):

if(matrix[i][j] <avg):

matrix[i][j] = 0;

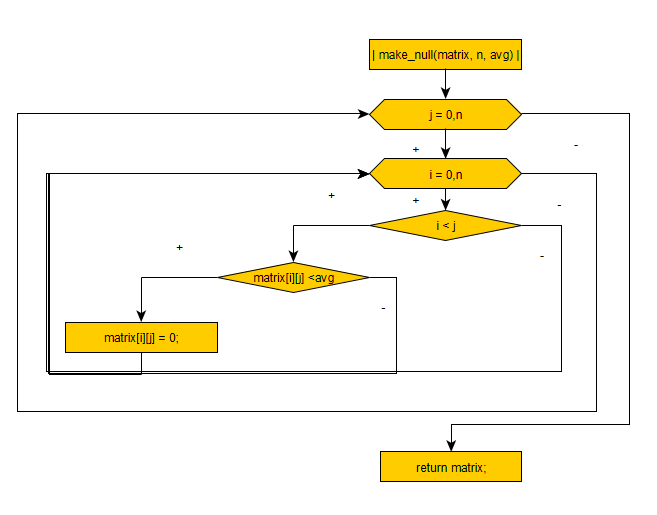
if end

if end

While end

while end

return matrix;



Код:

#include <iostream>

using namespace std;

int enter\_n();

void show\_2d\_arr(int\*\*matrix, int n);

int\*\* fill\_arr(int\*\* matrix, int n);

double diag\_avg(int\*\* matrix, int const n);

int\*\* gen\_2d\_arr(int n);

int\*\* make\_null(int\*\* matrix, int const n, double avg);

int main()

{

srand(time(NULL));

int n = enter\_n();

int\*\* matrix = gen\_2d\_arr(n);

matrix = fill\_arr(matrix, n);

cout << "Initial matrix" << endl<<endl;

show\_2d\_arr(matrix, n);

double avg = diag\_avg(matrix, n);

matrix = make\_null(matrix, n, avg);

cout <<endl<< "Final matrix" << endl<<endl;

show\_2d\_arr(matrix, n);

cout <<endl<<"Avg: " <<avg << endl << endl;

system("pause");

}

int enter\_n() {

int n = 0;

while (n <= 0) {

printf("Enter n: ");

scanf\_s("%d", &n);

if (n <= 0) {

printf("n e N");

}

}

return n;

}

int\*\* gen\_2d\_arr(int n) {

int\*\* matrix =new int\*[n];

for (int i = 0;i < n;i++) {

matrix[i] = new int[n];

}

return matrix;

}

int\*\* fill\_arr(int\*\* matrix, int n) {

for (int j = 0;j < n;j++) {

for (int i = 0;i < n;i++) {

matrix[i][j] = rand() % 100;

}

}

return matrix;

}

void show\_2d\_arr(int\*\* matrix, int n) {

for (int i = 0;i < n;i++) {

int\* one\_d\_arr = matrix[i];

for (int j = 0;j < n;j++) {

printf("%d\t", one\_d\_arr[j]);

}

printf("\n");

}

}

double diag\_avg(int\*\* matrix, int const n) {

int \*diag\_els = new int[n];

for (int j = n-1, k=0; j >=0; j--, k++)

{

for (int i = 0; i <n; i++) {

if (i == n-j-1) {

diag\_els[k] = matrix[i][j];

}

}

}

double avg = 0;

for (int i = 0; i < n; i++)

{

cout << endl << diag\_els[i] << "\t";

avg += diag\_els[i];

}

return avg /= n;

}

int\*\* make\_null(int\*\* matrix, int const n, double avg) {

for (int j = 0; j < n; j++)

{

for (int i = 0; i < n; i++) {

if (i < j) {

if(matrix[i][j] <avg)

matrix[i][j] = 0;

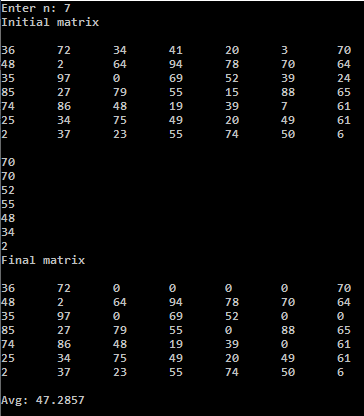
}

}

}

return matrix;

}



Отже, бачимо, що алгоритм працює.

**Висновок:**

Під час лабораторної роботи дослідили алгоритми обходу масивів, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Математична модель, блок-схеми, псевдокод, код випробовування наведені. Оскільки масив правильно перетворений, то алгоритм правильно подає результат.