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 23

Виконав студент ІП-13 Недельчев Євген Олександрович
(шифр, прізвище, ім'я, по батькові)

Перевірила Вечерковська 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 23

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.

Постановка задачі

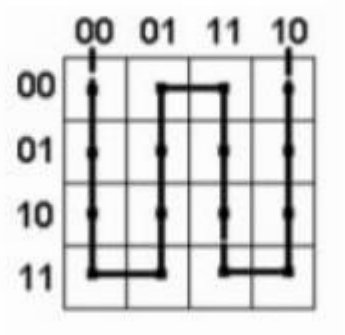
Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом.
2. Ініціювання змінної, що описана в п.1 даного завдання.
3. Обчислення змінної, що описана в п.1, згідно з варіантом.я

Побудова математичної моделі

Змінна	***Тип***	***Ім'я***	***Призначення***
Порядок квадратної матриці	Цілий	n	Вхідні дані
Верхня границя випадково генеруємих елементів матриці A	Дійсний	upLim	Вхідні дані
Нижня границя випадково генеруємих елементів матриці A	Дійсний	dnLim	Вхідні дані
Змінна індексованого типу (двовимірний масив)	Дійсний	A	Початкові дані
Середньоарифметичне елементів побічної діагоналі матриці A	Дійсний	P	Проміжний результат
Лічильник у циклах	Цілий	i	Лічильник
Лічильник у циклах	Цілий	j	Лічильник
Сума елементів побічної діагоналі	Дійсний	sum	Проміжний результат

Таким чином математичне формулювання задачі зводиться до створення двовимірного масиву A розмірнюстю n x n (квадратна матриця) та ініціалізації його обходом по стовбцям, тобто за наступним алгоритмом:



Для заповнення масиву створимо функцію fill_array(A, n), параметрами якої є масив A, який необхідно заповнити, та його розмірність n. Функція заповнює масив A випадковими елементами з діапазону [dnLim, upLim) обходом по стовбцям. Функція не повертає жодних значень.

Для пошуку середнього арифметичного елементів P побічної діагоналі матриці A створимо функцію arithmetical_mean_of_side_diagonal(A, n), параметрами якої є масив A, середнє арифметичне елементів побічної діагоналі якого необхідно обрахувати, та його розмірність n. Функція повертає значення виразу sum/n, де sum – сума елементів побічної діагоналі масиву A.

Для заміни елементів, розташованих вище головної діагоналі матриці A та які менші за значення P, на нулі створимо функцію change_elements_of_matrix(A, n, P), параметрами якої є масив A, елементи якого необхідно змінити, його розмірність n та середнє арифметичне P елементів побічної діагоналі масиву A. Функція не повертає жодних значень.

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

Розв’язання

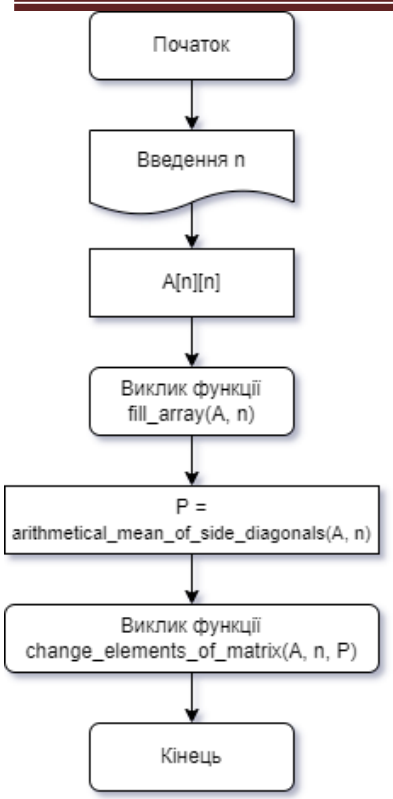
Крок 1. Визначимо основні дії.
Крок 2. Ініціалізація двовимірного масиву A[n][n] випадковими числами з діапазону [dnLim, upLim)
Крок 3. Пошук середнього арифметичного P елементів побічної діагоналі матриці A
Крок 4. Заміна елементів матриці A, розташованих вище головної діагоналі матриці A та які менші за значення P, на нулі.

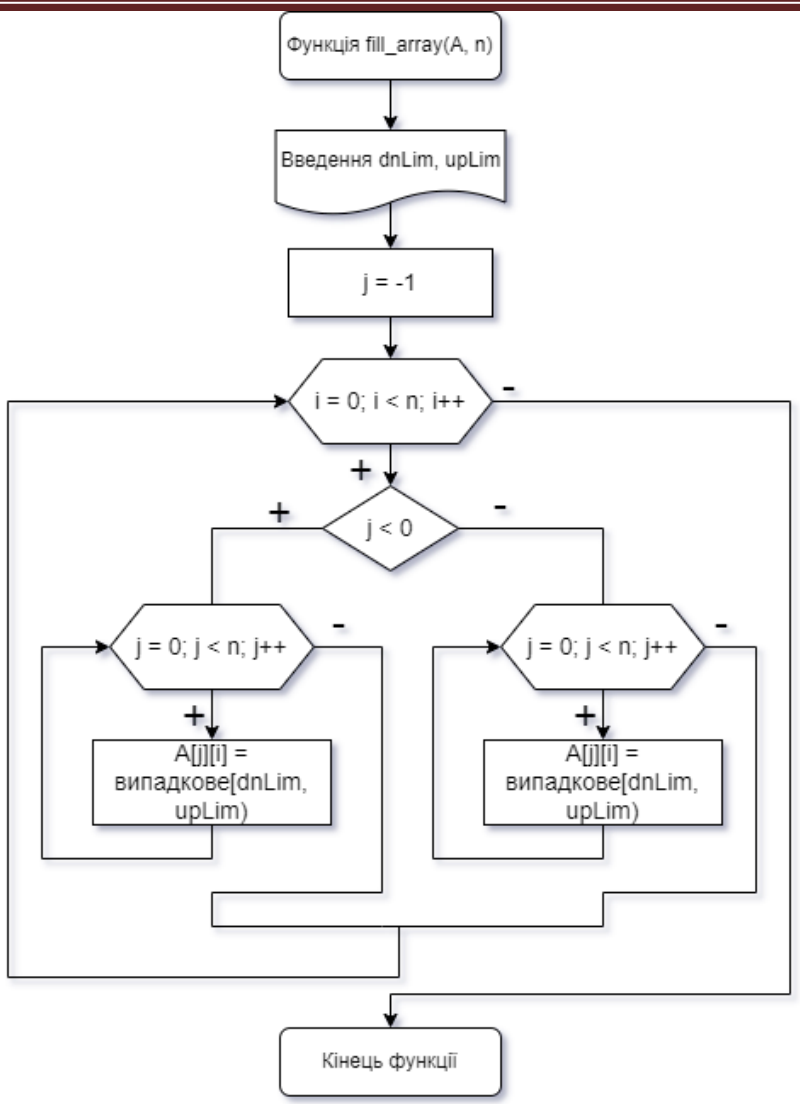
Псевдокод

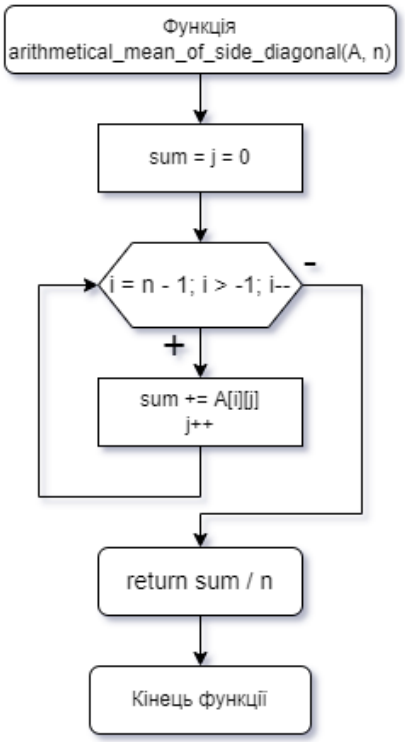
функція fill_array(A, n)

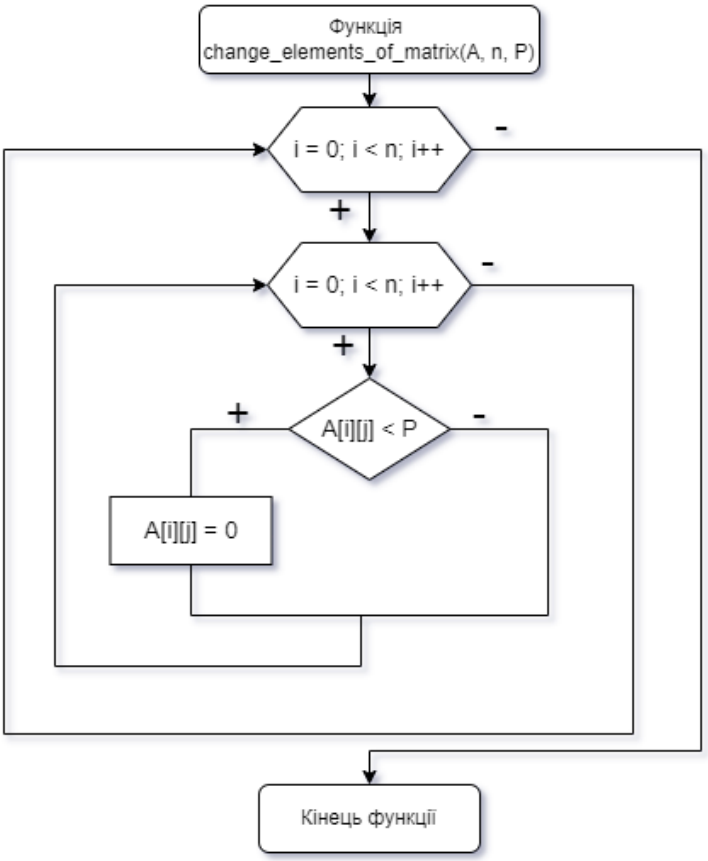
```
        Введення upLim, dnLim
        j = -1
        повторити для i від 0 до n
                якщо j < 0
                        j = 0
                        повторити для j від 0 до n
                                A[j][i] = випадкове(dnLim, upLim)
                        все повторити
                все якщо
                інакше
                        повторити для j від n-1 до -1 із кроком -1
                                A[j][i] = випадкове(dnLim, upLim)
                        все повторити
                все інакше
все функція

функція arithmetical_mean_of_side_diagonal(A, n)
        sum = j = 0
        повторити для i від n-1 до -1 із кроком -1
                sum += A[i][j]
                j++
        все повторити
        return sum / n
все функція

функція change_elements_of_matrix(A, n, P)
        повторити для i від 0 до n
                повторити для j від i + 1 до n
                        якщо A[i][j] < P
                                A[i][j] = 0
                        все якщо
                все повторити
        все повторити
все функція


початок
        Введення n
        A[n][n]
        fill_array(A, n)
        P = arithmetical_mean_of_side_diagonal(A, n)
        change_elements_of_matrix(A, n, P)
кінець
```

Блок-схема









Код програми

main.cpp*

ASD_lab9 | (Глобальная область) | main()

```cpp
#include <iostream>

using namespace std;

void fill_array(double** A, int n) {
    int dnLim, upLim;
    cout << "Enter the lower limit of generated numbers in your matrix: "; cin >> dnLim;
    cout << "Enter the upper limit of generated numbers in your matrix: "; cin >> upLim;
    int j = -1;
    for (int i = 0; i < n; i++) {
        if (j < 0) {
            for (j = 0; j < n; j++) {
                A[j][i] = (double)(rand() % (upLim * 100 - dnLim * 100 + 1) + dnLim * 100) / 100;
            }
        }
        else {
            for (j = n - 1; j > -1; j--) {
                A[j][i] = (double)(rand() % (upLim * 100 - dnLim * 100 + 1) + dnLim * 100) / 100;
            }
        }
    }
}

double arithmetical_mean_of_side_diagonal(double** A, int n) {
    double sum;
    int j;
    sum = j = 0;
    for (int i = n-1; i >= 0; i--) {
        sum += A[i][j];
        j++;
    }
    return sum / n;
}

void change_elements_of_matrix(double** A, int n, int P) {
    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            if (A[i][j] < P) A[i][j] = 0;
        }
    }
}

int main() {
    srand(time(NULL));
    cout << "Enter the order of your square matrix: ";
    int n; cin >> n;
    double** A = new double* [n];
    for (int i = 0; i < n; i++) {
        A[i] = new double[n];
    }
    fill_array(A, n);
    double P = arithmetical_mean_of_side_diagonal(A, n);
    cout << "Generated matrix: \n";
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cout << A[i][j] << ' ';
        }
        cout << endl;
    }
    cout << "The arithmetical mean of side diagonal is: " << P << endl;
    change_elements_of_matrix(A, n, P);
    cout << "Transformed matrix: \n";
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cout << A[i][j] << ' ';
        }
        cout << endl;
    }
    for (int i = 0; i < n; i++) {
        delete[] A[i];
    }
    delete[] A;
}
```

Тестування алгоритму

```
Выбрать Консоль отладки Microsoft Visual Studio
Enter the order of your square matrix: 5
Enter the lower limit of generated numbers in your matrix: -20
Enter the upper limit of generated numbers in your matrix: 20
Generated matrix:
  4.25   2.63  14.09   5.19  15.73
  1.52   9.45  19.64  16.91    0.1
  7.07   6.24   2.48   -8.6   6.77
 17.31  -7.29  12.41 -10.81 -11.25
  6.88 -16.87   9.99   9.42  13.18
The arithmetical mean of side diagonal is: 6.942
Transformed matrix:
  4.25      0  14.09      0  15.73
  1.52   9.45  19.64  16.91      0
  7.07   6.24   2.48      0   6.77
 17.31  -7.29  12.41 -10.81      0
  6.88 -16.87   9.99   9.42  13.18
```

```
Консоль отладки Microsoft Visual Studio
Enter the order of your square matrix: 7
Enter the lower limit of generated numbers in your matrix: 0
Enter the upper limit of generated numbers in your matrix: 100
Generated matrix:
 25.66  78.42    9.9  77.06  81.61  80.44  35.98
 15.84  39.51  75.82   9.71    1.6  39.15  36.68
  44.4  58.58  52.13  37.63  14.21  23.32  73.28
 58.82  10.54   9.93  26.57  78.35  67.82   10.4
 42.29  80.45  68.75   28.3  33.53  87.03  73.82
 45.62  60.52  22.73    9.6  62.77  22.75  44.73
 40.38  18.26   69.2    3.3  10.05  85.83   27.9
The arithmetical mean of side diagonal is: 40.7943
Transformed matrix:
 25.66  78.42      0  77.06  81.61  80.44      0
 15.84  39.51  75.82      0      0      0      0
  44.4  58.58  52.13      0      0      0  73.28
 58.82  10.54   9.93  26.57  78.35  67.82      0
 42.29  80.45  68.75   28.3  33.53  87.03  73.82
 45.62  60.52  22.73    9.6  62.77  22.75  44.73
 40.38  18.26   69.2    3.3  10.05  85.83   27.9
```

Висновки

Під час виконання цієї лабораторної роботи я дослідив алгоритми обходу масивів, набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій.