Додаток 1

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 29

Виконав студент ІП-15 Рибалка Ілля Сергійович
(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 29

Завдання

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.

№	Опис варіанту
29	Задано матрицю дійсних чисел A[n,n], ініціалізувати матрицю обходом по стовбцях. На побічній діагоналі матриці визначити присутність заданого дійсного числа X і його місцезнаходження. Порівняти значення X із середньоарифметичним значенням елементів під побічною діагоналлю.

1. Постановка задачі

Створити квадратну матрицю заданого розміру, заповнити її обходом по стовбцях. Знайти задане число на побічній діагоналі і порівняти його з середньоарифметичним елементів під побічною діагоналлю.

2. Побудова математичної моделі

Змінна	Тип	Ім’я	Призначення
Основна програма			
Номер рядку	Ціле	row	Початкові дані
Номер стовпця	Ціле	col	Початкові дані
Перший елемент масиву	Дійсне	num	Початкові дані
Крок	Дійсне	step	Початкові дані
Розмір матриці	Ціле	len	Вхідні дані
Число X	Дійсне	X	Вхідні дані
Матриця	Дійсне	matrix	Початкові / Проміжні дані
Середнє арифметичне	Дійсне	arthmean	Проміжні дані
Заповнення матриці числами	Підпрограма	crmatrix	Початкові дані

Пошук елемента X в матриці	Підпрограма	find	Початкові дані
Середнє арифметичне	Підпрограма	arthm	Початкові дані
Вивід матриці	Підпрограма	out	Початкові дані
Матриця	Дійсне	mtrx	Проміжні дані
Середнє арифметичне	Дійсне	res	Проміжні дані

На вхід в програму дається 2 значення - *len* і *X*, де *len* - це розмірність матриці, згідно постановки нам дана квадратна матриця, тому кількість стовпців рівна кількості рядків, а *X* - це довільне число(в умові задачі не сказано що воно обов'язково присутнє на побічній діагоналі).

Заповнюємо матрицю обходом по стовпцях, починаючи з лівого верхнього кута матриці, з кроком *step*(за умовчанням вважатимемо його рівним одному) і першим, згенерованим за рахунок функції ***rand***, елементом *num*.

Шукати місцезнаходження елемента *X* буде підпрограма *find*, підпрограма має повертати рядок і стовпець в якому знаходиться даний елемент.

Середнє арифметичне елементів під побічною діагоналлю шукатиме підпрограма *arthm*, для крайнього нижнього елемента вважатимемо, що 1 елемент цього стовпця знаходиться під ним.

Вивід усіх даних, окрім матриці, здійснюється в основній програмі, матриця виводиться за рахунок підпрограми *out*.

Розв'язання

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.
Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію заповнення матриці за рахунок підпрограми crmatrix.

Крок 3. Деталізуємо дію пошуку X в матриці за рахунок підпрограми find.

Крок 4. Деталізуємо дію знаходження середнього арифметичного за рахунок підпрограми arthm.

Крок 5. Деталізуємо дію виведення матриці за рахунок підпрограми out.

Псевдокод

Основна програма

крок 1
початок
step = 1
Введення len
num = *rand* **від** 10 **до** -5
Вивід num
Введення X
matrix(matrix, len, num, step)
out(matrix, len)
row, col = find(matrix, len, X)
якщо row = 0 **то**
    Вивід "Елемент X відсутній на побічній діагоналі"
    кінець
інакше
    Вивід "Місцезнаходження елементу X - row рядок, col стовпець"
все якщо
arthmean = arthm(matrix, len)
Вивід arthmean
якщо arthmean = X **то**
    Вивід "Елемент X рівний середньоарифметичному"
інакше якщо arthmean > X **то**
    Вивід "Елемент X менший за середньоарифметичне"
інакше
    Вивід "Елемент X більший за середньоарифметичне"
все якщо
кінець

Підпрограми

крок 2
crmatrix(mtrx, len, num, step)
mtrx[0][0] = num
для i **від** 0 **до** len **з кроком** 2 **повторити**
    mtrx[0][i] = num + step*i*len
    для j **від** 1 **до** len **повторити**
        mtrx[j][i]= mtrx[j-1][i] + step
    все повторити
    k = i + 1
    якщо k < len **то**
        mtrx[len-1][k] = num + step*k*len
        для j **від** len-2 **до** 0 **повторити**
            mtrx[j][k]= mtrx[j+1][k] + step
        все повторити
    все якщо
все повторити
кінець crmatrix

крок 3

```
find(mtrx, len, X)
j = 0
для i від len-1 до 0 повторити
        якщо mtrx[i][j] == X то
                повернути i+1, j+1
        все якщо
        j++
все повторити
повернути 0
кінець find
```

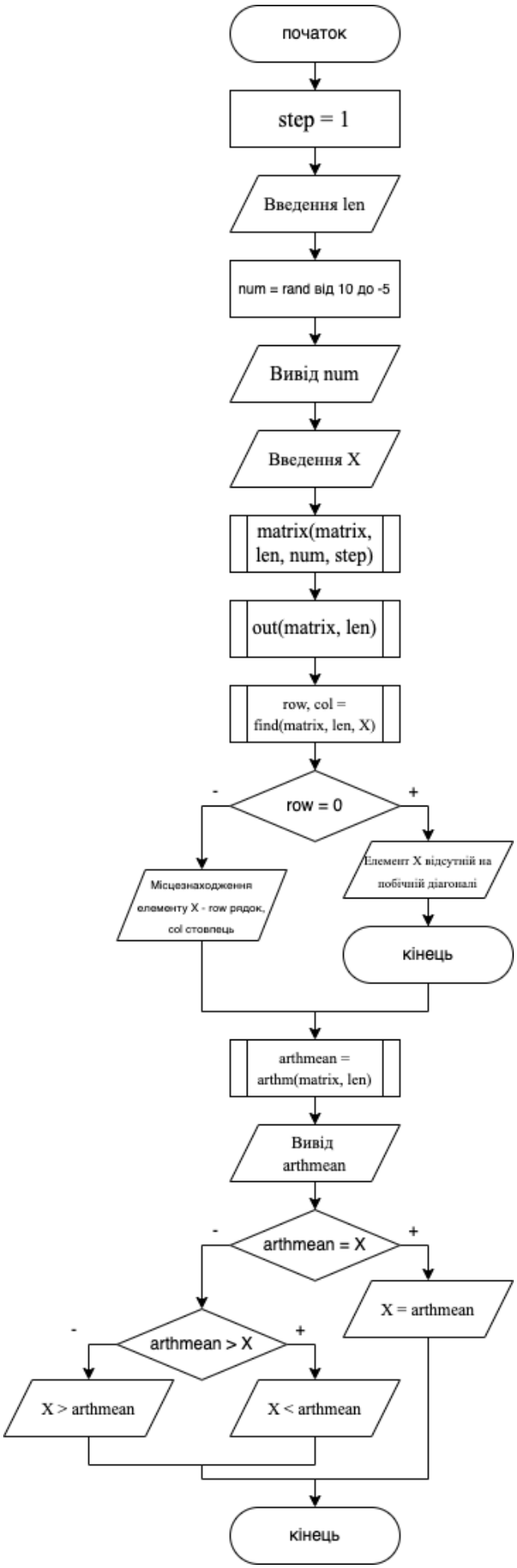
крок 4

```
arthm(mtrx, len)
j = 1
res = mtrx[0][0]
для i від len-1 до 0 повторити
        res += mtrx[i][j]
        j++
все повторити
повернути res
кінець arthm
```

крок 5

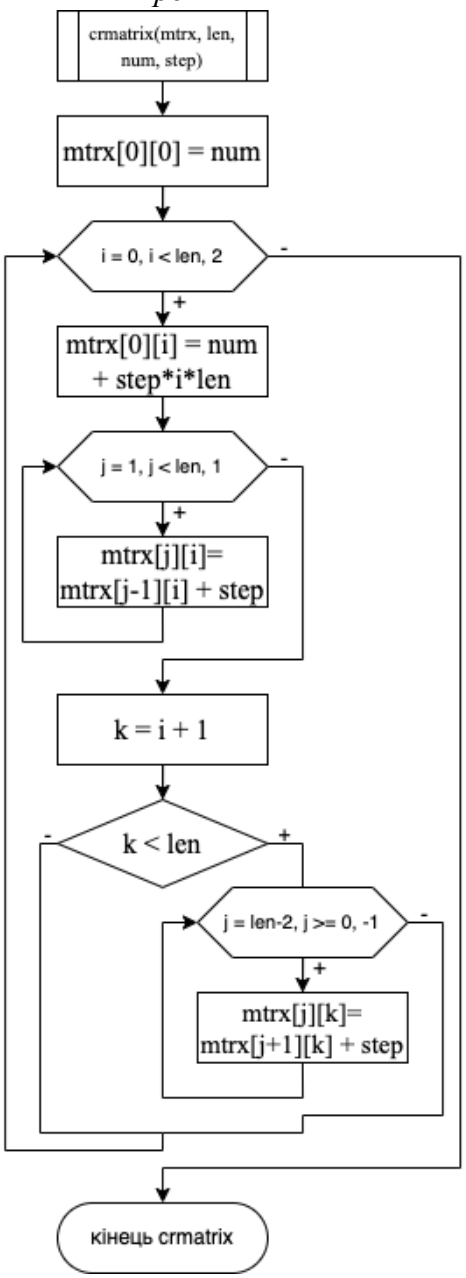
```
out(mtrx, len)
Вивід "Згенерована матриця:"
для i від 0 до len повторити
        Вивід "|"
        для j від 0 до len повторити
                Вивід matrix[i][j]
        все повторити
        Вивід "|"
все повторити
кінець out
```

Блок-Схема
Основна програма
крок 1

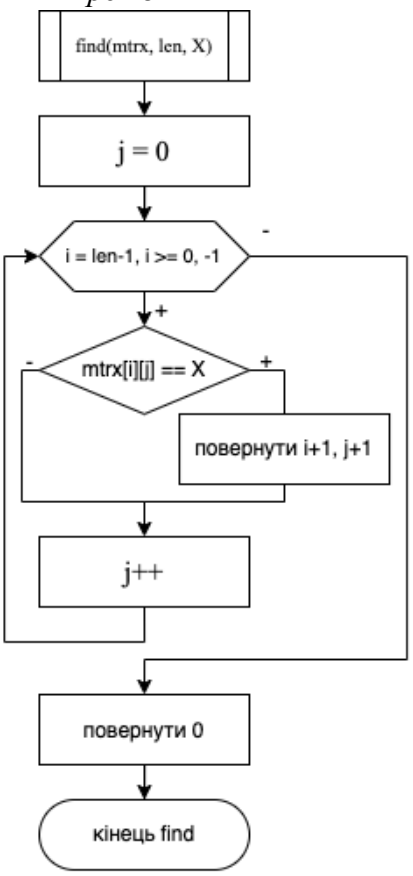


Підпрограми

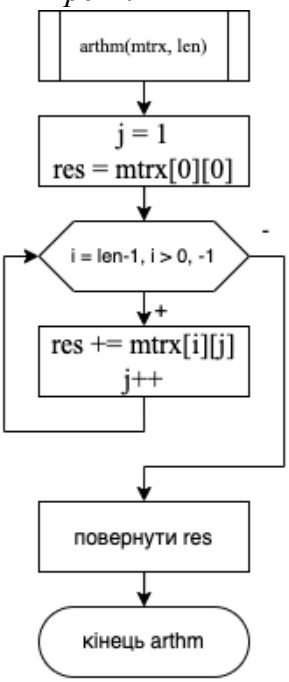
крок 2



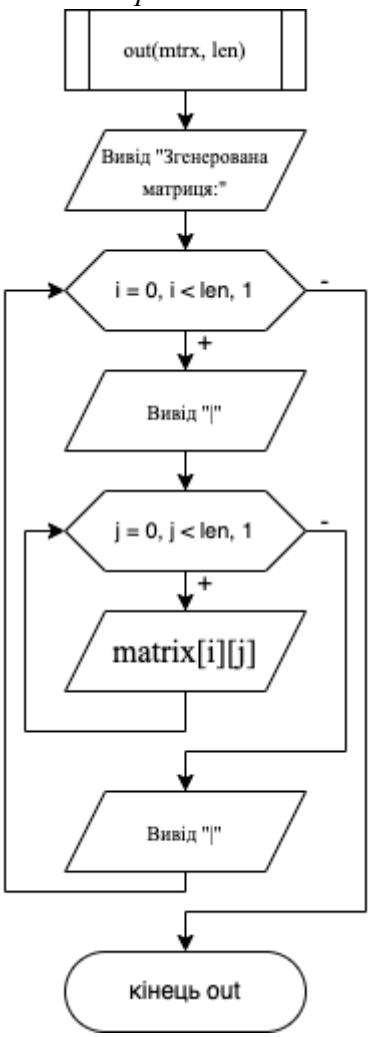
крок 3



крок 4



крок 5


out(mtrx, len)
Вивід "Згенерована матриця:"
i = 0, i < len, 1
-
+
Вивід "|"
j = 0, j < len, 1
-
+
matrix[i][j]
Вивід "|"
кінець out

Код

```cpp
#include <iostream>
#include <iomanip>
#include <ctime>

void crmatrix(double **, int, double, double); // створення матриці обходом по стовпцях
int find(double **, int, double, int *); // пошук місцезнаходження елементу X
double arthm(double **, int); // пошук середньоарифметичного елементів під побічною діагоналлю
void out(double **, int); // вивід матриці

int main()
{
   int len, row, col;
   double X, arthmean, num, step = 1;
   std::cout << "Введіть n = ";
   std::cin >> len;
   srand(time(NULL));
   num = rand()%10 + -5;
   std::cout << "Згенероване початкове число = " << num << std::endl;
   std::cout << "Введіть X = ";
   std::cin >> X;
   double **matrix = new double *[len];
   for (int i=0; i<len; i++)
   {
       matrix[i] = new double [len];
   }
   crmatrix(matrix, len, num, step);
   out(matrix, len);
   row = find(matrix, len, X, &col);
   if (row == 0)
   {
       std::cout << "Елемент X відсутній на побічній діагоналі" << std::endl;
       return 0;
   }
   else std::cout << "Месцезнаходження елементу X - " << row << " рядок, " << col << " стовпець" << std::endl;
   arthmean = arthm(matrix, len);
   std::cout << "Середньоарифметичне елементів під побічною діагоналлю = " << arthmean << std::endl;
   if (arthmean == X) std::cout << "Елемент X рівний середньоарифметичному" << std::endl;
   else if (arthmean > X) std::cout << "Елемент X менший за середньоарифметичне" << std::endl;
   else std::cout << "Елемент X більший за середньоарифметичне" << std::endl;
   for (int i=0; i<len; i++)
   {
       delete [] matrix[i];
   }
   delete [] matrix;
   return 0;
}

void crmatrix(double **mtrx, int len, double num, double step)
{
   int k;
   mtrx[0][0] = num;
   for (int i=0; i<len; i+=2)
   {
       mtrx[0][i] = num + step*i*len;
       for (int j=1; j<len; j++)
       {
           mtrx[j][i]= mtrx[j-1][i] + step;
       }
       k = i + 1;
       if (k < len)
       {
           mtrx[len-1][k] = num + step*k*len;
           for (int j=len-2; j>=0; j--)
           {
               mtrx[j][k]= mtrx[j+1][k] + step;
           }
       }
   }
}

int find(double **mtrx, int len, double X, int *col)
{
   int j = 0;
   for (int i = len - 1; i>=0; i--)
   {
       if ( mtrx[i][j] == X )
       {
           *col = j + 1;
           return i + 1;
       }
       j++;
   }
   return 0;
}

double arthm(double **mtrx, int len)
{
   int j = 1;
   double res = mtrx[0][0];
   for (int i = len - 1; i>0; i--)
   {
       res += mtrx[i][j];
       j++;
   }
   res /= len;
   return res;
}

void out(double **mtrx, int len)
{
   std::cout << "Згенерована матриця:" << std::endl;
   for (int i=0; i<len; i++)
   {
       std::cout << "|";
       for (int j=0; j<len; j++)
       {
           std::cout << std::setw(3) << mtrx[i][j];
       }
       std::cout << std::setw(2) << "|" << std::endl;
   }
}
```

Тестування

```
Введіть n = 2
Згенероване початкове число = 0
Введіть X = 1
Згенерована матриця:
|  0  3 |
|  1  2 |
Месцезнаходження елементу X - 2 рядок, 1 стовпець
Середньоарифметичне елементів під побічною діагоналлю = 1
Елемент X рівний середньоарифметичному
MacBook-Pro-Mac:ASD mac$ cd "/Volumes/files/ОП/Lab/Lab9/ASD/" && g
Введіть n = 3
Згенероване початкове число = -3
Введіть X = 1
Згенерована матриця:
| -3  2  3 |
| -2  1  4 |
| -1  0  5 |
Месцезнаходження елементу X - 2 рядок, 2 стовпець
Середньоарифметичне елементів під побічною діагоналлю = 0.333333
Елемент X більший за середньоарифметичне
MacBook-Pro-Mac:ASD mac$ cd "/Volumes/files/ОП/Lab/Lab9/ASD/" && g
Введіть n = 4
Згенероване початкове число = 4
Введіть X = 19
Згенерована матриця:
|  4 11 12 19 |
|  5 10 13 18 |
|  6  9 14 17 |
|  7  8 15 16 |
Месцезнаходження елементу X - 1 рядок, 4 стовпець
Середньоарифметичне елементів під побічною діагоналлю = 11
Елемент X більший за середньоарифметичне
MacBook-Pro-Mac:ASD mac$ cd "/Volumes/files/ОП/Lab/Lab9/ASD/" && g
Введіть n = 3
Згенероване початкове число = -2
Введіть X = -1
Згенерована матриця:
| -2  3  4 |
| -1  2  5 |
|  0  1  6 |
Елемент X відсутній на побічній діагоналі
```

Висновок

Я дослідив алгоритми обходу масивів, набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій, розробив алгоритм для ініціалізації квадратної матриці обходом по стовпцях і пошуку заданого елементу в заданих межах.