Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 14

По дисциплине «Основы алгоритмизации и программирования»

На тему «Многомерные массивы»

Выполнила:

Студентка 1 курса 10 группы

Рублевская Маргарита Владимировна

Преподаватель: асс. Андронова М.В.

2023, Минск

**Вариант№ 13**

1. Выполнить задания из таблицы ниже, используя ***индексы*** для доступа к элементам массивов в первой программе и ***указатели*** − во второй программе.

1. Найти наименьший элемент главной диагонали матрицы **С(n, n)** и вывести на печать столбец, в котором он находится.

Код:

#include<iostream> // Подключаем библиотеку для работы с вводом/выводом

#include<locale> // Подключаем заголовочный файл, содержащий объявления функций для работы с локалью

int main()

{

setlocale(LC\_CTYPE, "Russian");

using namespace std;

const int Max\_n = 100; // Задаем максимальное значение для размерности матрицы

// Объявляем переменные и массив

int C[Max\_n][Max\_n];

int n, k, m, min, stolbec;

cout << "Введите количество строк и столбцов квадратной матрицы: ";

cin >> n;

srand((unsigned)time(NULL)); // Инициализация генератора случайных чисел

cout << "Матрица:" << endl;

for (k = 0; k < n; k++)

{

for (m = 0; m < n; m++)

{

C[k][m] = rand() % 51; // Присваиваем элементам массива случайного целого числа от 0 до 51

// Выводим элементы матрицы C на экран

cout << C[k][m] << ' '; // Выводим значение элемента C[k][m]

// Если значение меньше 10, добавляем пробел для выравнивания столбцов

if (C[k][m] < 10)

cout << " ";

// Выводим символ перевода строки для перехода на следующую строку матрицы

if (m == n - 1)

cout << endl;

}

}

// Находим минимальный элемент на главной диагонали матрицы C и запоминаем номер столбца, в котором он находится.

min = C[0][0];

stolbec = 0;

for (k = 0; k < n; k++)

{

if (C[k][k] < min)

{

min = C[k][k];

stolbec = k;

}

}

cout << "Наименьший элемент главной диагонали матрицы - C[" << stolbec + 1 << "; " << stolbec + 1 << "] = " << min;

cout << endl << "Столбец, в котором находится наименьший элемент главной диагонали матрицы: " << endl;

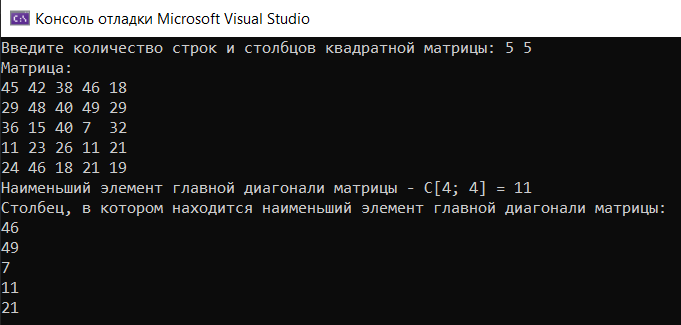
// Выводим на экран элементы столбца stolbec двумерного массива C размерности n

for (k = 0; k < n; k++)

cout << C[k][stolbec] << endl;

}

Консоль VS:



2. Задана числовая матрица размером **n**x**m**.Определить количество таких элементов матрицы, значение каждого из которых больше суммы остальных элементов своего столбца.

Код:

#include<iostream> // Подключаем библиотеку для работы с вводом/выводом

#include<iomanip> // Подключаем библиотеку для работы с setw()

int main()

{

srand(time(0)); // Инициализация генератора случайных чисел

using namespace std;

setlocale(LC\_CTYPE, "Russian");

// Объявляем переменные и массив

int A[100][100], n, m, k = 0, sum = 0;

cout << "Количество строк матрицы: "; cin >> n;

cout << "Количество столбцов матрицы: "; cin >> m;

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

{

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

{

\*(\*(A + i) + j) = rand() % 100; // Присваиваем элементам двумерного массива случайного целого числа от -99 до 99

cout << setw(4) << \*(\*(A + i) + j) << " "; // Выводим матрицу на экран

}

cout << endl; // Переходим на новую строку после заполнения столбца

}

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

{

sum = 0; // Обнуляем сумму, чтобы на каждом новом столбце она начиналась с 0

for (int j = 0; j < n; j++) // Пробегаемся по столбцу, чтобы посчитать сумму

{

sum += \*(\*(A + j) + i); // Непосредственно считаем сумму столбца

}

for (int j = 0; j < n; j++) // Пробегаемся по столбцу, чтобы найти необходимые элементы

{

if (sum - (\*(\*(A + j) + i)) < \*(\*(A + j) + i)) k++; // Делаем условие, в котором записываем, что сумма всех элементов без рассматриваемого элемента была меньше, чем сам элемент

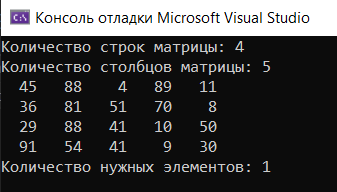
}

}

cout << "Количество нужных элементов: " << k << endl;

}

Консоль VS:



**Дополнительные задания**

1. Дана квадратная матрица порядка **2n**, элементы которой формируются случайным образом и находятся в пределах от −10 до 10. Получить новую матрицу, переставляя ее блоки размера **n×n** в соответствии со схемой.

Код:

#include<iostream> // Подключаем библиотеку для работы с вводом/выводом.

#include<ctime> // Подключаем библиотеку для работы с случайными числами.

//Функция принимает минимум и максимум и возвращает случайное число в этих пределах.

int randomNumberFunction(int rmn, int rmx);

int main()

{

setlocale(LC\_CTYPE, "Russian");

srand((unsigned)time(NULL)); // Инициализация генератора случайных чисел

const int N = 4; // Инициализация константы

int matrix[N][N]{}, intermediateVariable{}; // Объявляем переменные и массив

// Заполняем двумерный массив matrix случайными числами в диапазоне от -10 до 10 и выводим его на экран в виде матрицы

std::cout << "Матрица:" << std::endl;

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

{

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

{

matrix[i][j] = randomNumberFunction(-10, 10);

std::cout << matrix[i][j] << ' ';

}

std::cout << std::endl;

}

//Меняем блоки по схеме

intermediateVariable = matrix[0][0];

matrix[0][0] = matrix[N - 1][N - 1];

matrix[N - 1][N - 1] = intermediateVariable;

intermediateVariable = matrix[N - 1][0];

matrix[N - 1][0] = matrix[0][N - 1];

matrix[0][N - 1] = intermediateVariable;

//Выводим результат на экран

std::cout << "Матрица(после преобразования):" << std::endl;

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

{

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

{

std::cout << matrix[i][j] << ' ';

}

std::cout << std::endl;

}

return 0; }

// Генерируем случайное целое число в заданном диапазоне от rmn до rmx

int randomNumberFunction(int rmn, int rmx)

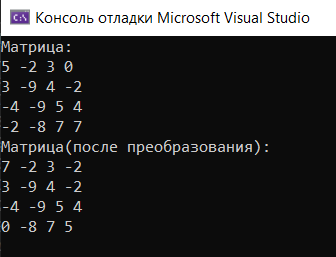
{

int number = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);

return number;

}

Консоль VS:



2. Латинским квадратом порядка **n** называется квадратная таблица размером **nхn**, каждая строка и каждый столбец которой содержат все числа от 1 до **n**. Для заданного **n** в матрице **L(n, n**) построить латинский квадрат порядка **n**.

Код:

#include<iostream> // Подключаем библиотеку для работы с вводом/выводом.

int main()

{

setlocale(LC\_CTYPE, "Russian");

const int n = 7; // Инициализация константы

int matrix[n][n]{}; // Объявляем переменные и массив

//Формируем латинский квадрат.

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

{

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

{

matrix[i][j] = (i + j) % n + 1;

}

}

//Выводим результат на экран

std::cout << "Латинский квадрат:" << std::endl;

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

{

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

{

std::cout << matrix[i][j] << ' ';

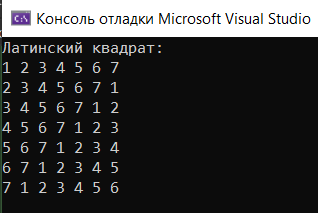
}

std::cout << std::endl;

}

return 0; }

Консоль VS:



3. Путем перестановки элементов квадратной вещественной матрицы добиться того, чтобы ее максимальный элемент находился в левом верхнем углу, следующий по величине − в позиции (2, 2), следующий − в позиции (3, 3) и т. д., заполнив таким образом всю главную диагональ.

Код:

#include<iostream> // Подключаем библиотеку для работы с вводом/выводом.

#include<ctime> // Подключаем библиотеку для работы с случайными числами.

//Функция принимает минимум и максимум и возвращает случайное число в этих пределах.

int randomNumberFunction(int rmn, int rmx);

int main(){

setlocale(LC\_CTYPE, "Russian");

srand((unsigned)time(NULL)); // Инициализация генератора случайных чисел const int SIZE\_MATRIX = 5; //Инициализация константы

// Объявляем массивы

int matrix[SIZE\_MATRIX][SIZE\_MATRIX]{}, MainDiagonalAndCoordinates[SIZE\_MATRIX][3]{}; //Инициализация двумерных массивов.

int maxElement{}, lineMaxElement{}, columnMaxElement{};

// Выводим на экран исходную матрицу, заполненную случайными числами std::cout << "Исходная матрица:" << std::endl;

for (int i = 0; i < SIZE\_MATRIX; i++)

{

for (int j = 0; j < SIZE\_MATRIX; j++)

{

matrix[i][j] = randomNumberFunction(10, 99);

std::cout << matrix[i][j] << ' ';

}

std::cout << std::endl;

}

//Ищем максимальные элементы в матрице и записываем их по убыванию вместе с их индексами.

for (int i = 0; i < SIZE\_MATRIX; i++)

{

maxElement = 0;

for (int j = 0; j < SIZE\_MATRIX; j++)

{

for (int g = 0; g < SIZE\_MATRIX; g++)

{

if (matrix[j][g] > maxElement)

{

maxElement = matrix[j][g];

lineMaxElement = j;

columnMaxElement = g;

}

}

}

MainDiagonalAndCoordinates[i][0] = maxElement;

MainDiagonalAndCoordinates[i][1] = lineMaxElement;

MainDiagonalAndCoordinates[i][2] = columnMaxElement;

matrix[lineMaxElement][columnMaxElement] = 0;

}

//Заменяем элементы главной диагонали.

for (int i = 0; i < SIZE\_MATRIX; i++)

{

matrix[MainDiagonalAndCoordinates[i][1]][MainDiagonalAndCoordinates[i][2]] = matrix[i][i];

matrix[i][i] = MainDiagonalAndCoordinates[i][0];

}

// Выводим матрицу на экран после ее заполнения случайными числами std::cout << "Матрица (после преобразования):" << std::endl;

for (int i = 0; i < SIZE\_MATRIX; i++)

{

for (int j = 0; j < SIZE\_MATRIX; j++)

{

std::cout << matrix[i][j] << ' ';

}

std::cout << std::endl; }

return 0; }

// Генерируем случайное целое число в заданном диапазоне от rmn до rmx

int randomNumberFunction(int rmn, int rmx) {

int number = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);

return number;}

Консоль VS:

