**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №3**

**по дисциплине «Программирование»**

Тема: “Двумерные статические массивы. Указатели”

|  |  |  |
| --- | --- | --- |
| Студент гр. 2372 |  | Симаков Г.О. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2022

**Цель работы.**

Изучение внутреннего строения двумерного массива, понимания, как хранятся типы данных в двумерном массиве, его представление в памяти.

**Основные теоретические положения.**

### Объявление многомерных  массивов

* Многомерные массивы определяются аналогично одномерным массивам. Количество элементов по каждому измерению указывается отдельно в квадратных скобках.
* Общее число элементов в многомерном массиве определяется как произведение количества элементов по каждому измерению. Так, например, массив **D** содержит **10 \* 15 \* 3 = 450**  элементов типа **double**, а объем памяти, требующийся для этого массива, будет равен **450 \* 8 = 3600** байтам.
* Массивы с большим, чем 3, количеством измерений используются достаточно редко. Одной из причин этого является быстрый рост объема памяти, необходимой для размещения таких массивов.
* Зная, что массив в C++ это указатель на его первый элемент, мы можем получить элемент 2го столбца и 3й строки, воспользовавшись арифметикой указателей:

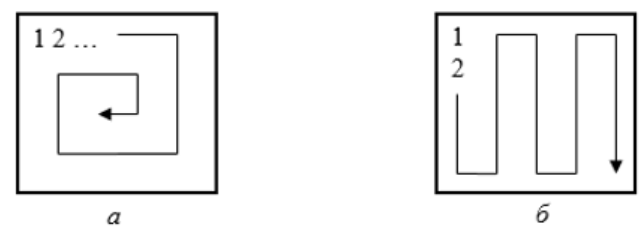
int element = \*(\*a + 11);

* Т.к. это двумерный массив, то a – указатель на первый элемент двумерного массива, \*a – это указатель на первый элемент первого подмассива. Разыменовываем указатель *a* и прибавляем к нему 11. Таким образом, получаем нужный элемент.

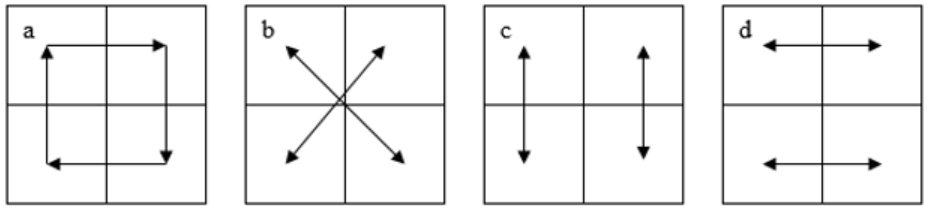
**Постановка задачи.**

Вся работа должна быть выполнена с использованием арифметики указателей.

1. Необходимо создать двумерный массив (матрицу) и заполнить его случайными числами от 1 до N\*N согласно следующим схемам (прогресс заполнения должен отображаться на экране):



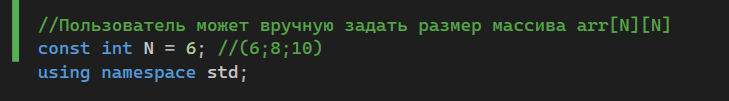
1. Необходимо получить новую матрицу, применяя преобразования к матрице из шага 1:



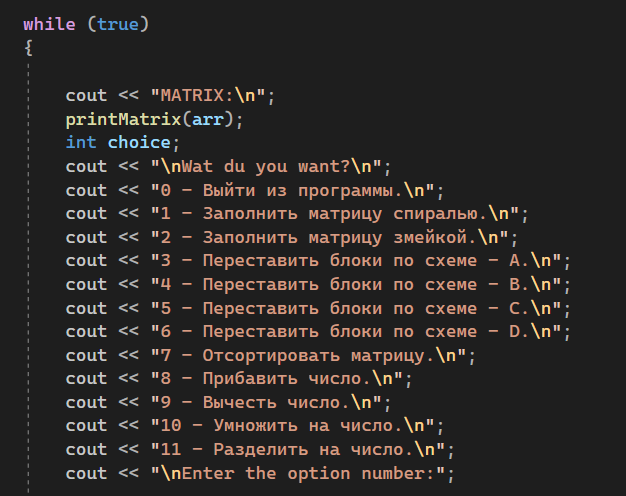
1. Отсортировать элементы матрицы, как будто это одномерный массив
2. Уменьшить, увеличить, умножить, разделить матрицу на введённое пользователем число.

**Выполнение работы.**

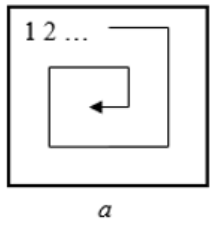
1)Выбор размера матрицы

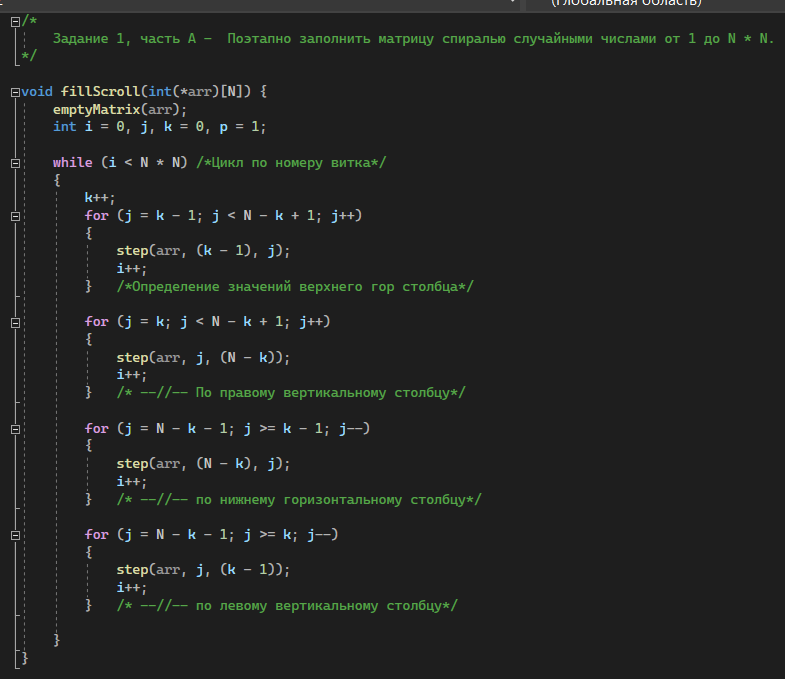


2)Выбор действия через меню:

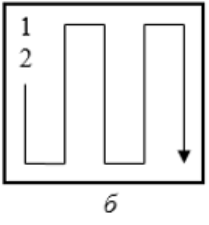


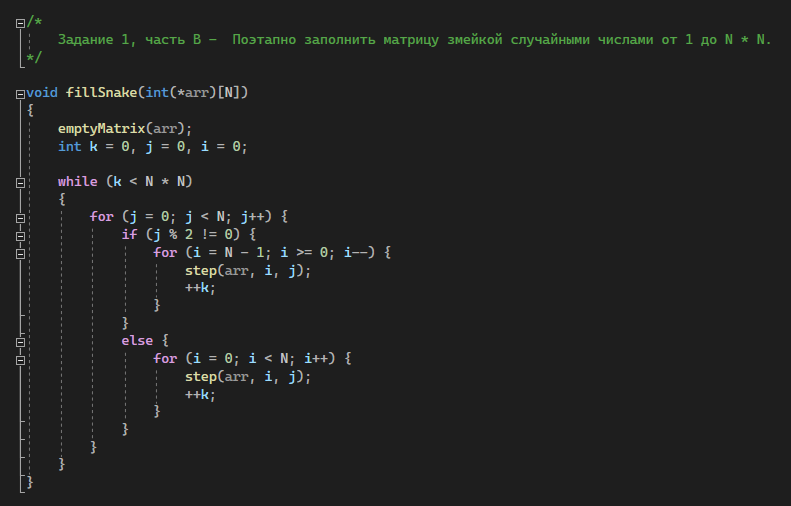
Задание 1.1 - заполнение матрицы спиралью



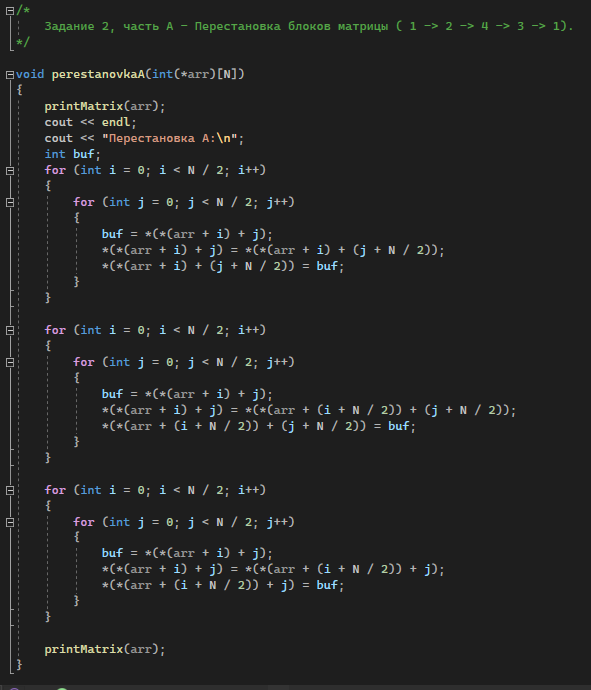


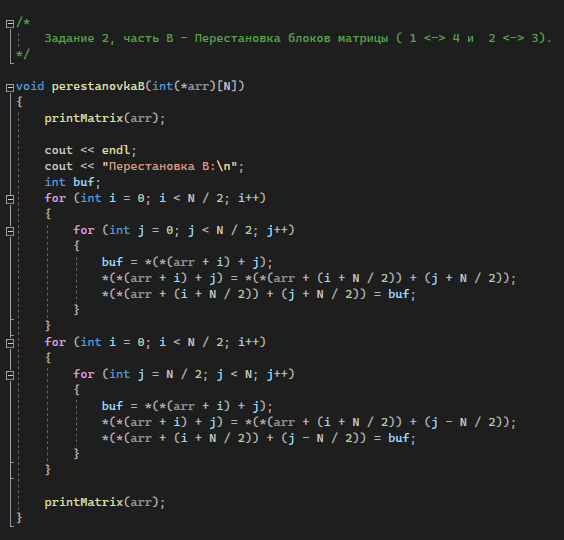
Задание 1.2 - заполнение матрицы змейкой

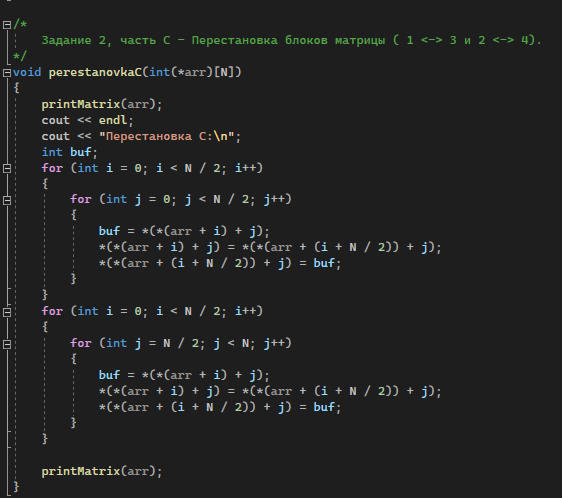


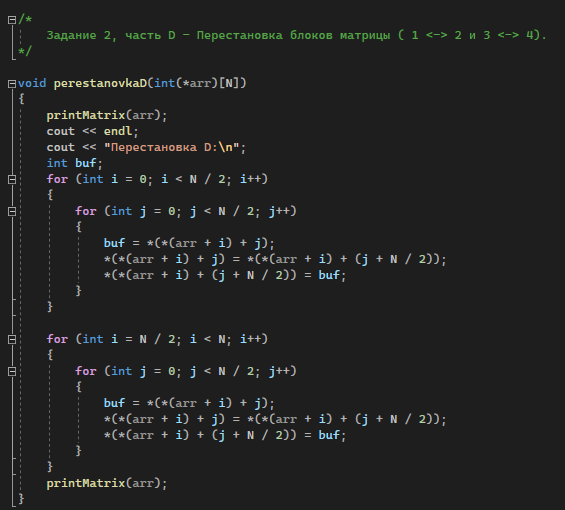


Задание 2 - смена блоков в матрице:

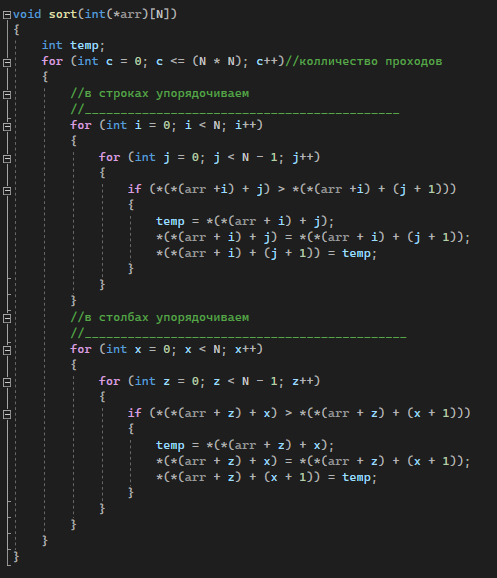






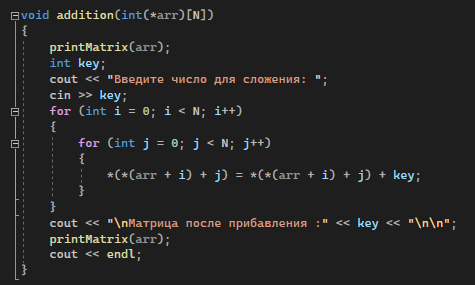


Задание 3 - сортировка элементов двумерного массива.

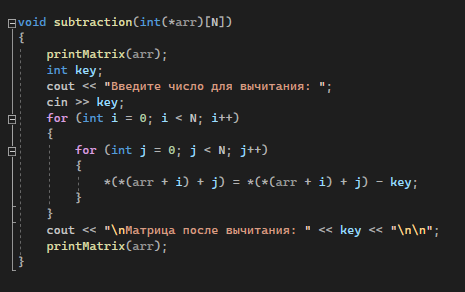


Задание 4 – сложение, вычитание, умножение, деление массива на число.

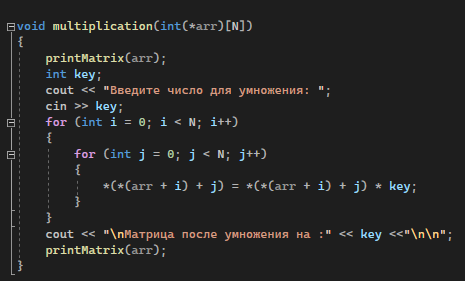
1)сложение:



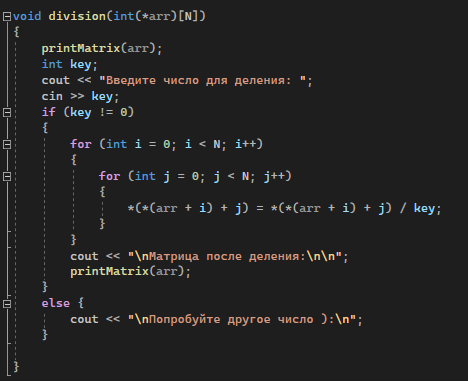
2)Вычитание:



3)Умножение:



4)Деление:



**Выводы.**

Я научился работать с двумерными массивами, изучил указатели, изучил арифметику указателей и строение двумерного массива в памяти.

Рабочий Код:

#include <iostream>

#include <ctime>

#include <Windows.h>

#include <conio.h>

#include <iomanip>

//Пользователь может вручную задать размер массива arr[N][N]

const int N = 6; //(6;8;10)

using namespace std;

/\*

Функция для вывода матрицы.

\*/

void printMatrix(int(\*arr)[N]) {

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

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

cout << setw(4) << \*(\*(arr + i) + j);

}

cout << endl;

}

}

/\*

Функция для зануления матрицы.

\*/

void emptyMatrix(int(\*arr)[N]) {

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

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

\*(\*(arr + i) + j) = 0;

}

}

}

/\*

Получить рандомное число от 1 до N \* N

\*/

int randomValue() {

return rand() % (N \* N - 1) + 1;

}

/\*

Поэтапное заполнение матрицы.

\*/

void step(int(\*arr)[N], int i, int j) {

system("cls");

int rv = randomValue();

\*(\*(arr + i) + j) = rv;

printMatrix(arr);

Sleep(250);

}

/\*

Задание 1, часть A - Поэтапно заполнить матрицу спиралью случайными числами от 1 до N \* N.

\*/

void fillScroll(int(\*arr)[N]) {

emptyMatrix(arr);

int i = 0, j, k = 0, p = 1;

while (i < N \* N) /\*Цикл по номеру витка\*/

{

k++;

for (j = k - 1; j < N - k + 1; j++)

{

step(arr, (k - 1), j);

i++;

} /\*Определение значений верхнего гор столбца\*/

for (j = k; j < N - k + 1; j++)

{

step(arr, j, (N - k));

i++;

} /\* --//-- По правому вертикальному столбцу\*/

for (j = N - k - 1; j >= k - 1; j--)

{

step(arr, (N - k), j);

i++;

} /\* --//-- по нижнему горизонтальному столбцу\*/

for (j = N - k - 1; j >= k; j--)

{

step(arr, j, (k - 1));

i++;

} /\* --//-- по левому вертикальному столбцу\*/

}

}

/\*

Задание 1, часть B - Поэтапно заполнить матрицу змейкой случайными числами от 1 до N \* N.

\*/

void fillSnake(int(\*arr)[N])

{

emptyMatrix(arr);

int k = 0, j = 0, i = 0;

while (k < N \* N)

{

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

if (j % 2 != 0) {

for (i = N - 1; i >= 0; i--) {

step(arr, i, j);

++k;

}

}

else {

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

step(arr, i, j);

++k;

}

}

}

}

}

/\*

Задание 2, часть A - Перестановка блоков матрицы ( 1 -> 2 -> 4 -> 3 -> 1).

\*/

void perestanovkaA(int(\*arr)[N])

{

printMatrix(arr);

cout << endl;

cout << "Перестановка A:\n";

int buf;

for (int i = 0; i < N / 2; i++)

{

for (int j = 0; j < N / 2; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + i) + (j + N / 2));

\*(\*(arr + i) + (j + N / 2)) = buf;

}

}

for (int i = 0; i < N / 2; i++)

{

for (int j = 0; j < N / 2; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + (i + N / 2)) + (j + N / 2));

\*(\*(arr + (i + N / 2)) + (j + N / 2)) = buf;

}

}

for (int i = 0; i < N / 2; i++)

{

for (int j = 0; j < N / 2; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + (i + N / 2)) + j);

\*(\*(arr + (i + N / 2)) + j) = buf;

}

}

printMatrix(arr);

}

/\*

Задание 2, часть B - Перестановка блоков матрицы ( 1 <-> 4 и 2 <-> 3).

\*/

void perestanovkaB(int(\*arr)[N])

{

printMatrix(arr);

cout << endl;

cout << "Перестановка B:\n";

int buf;

for (int i = 0; i < N / 2; i++)

{

for (int j = 0; j < N / 2; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + (i + N / 2)) + (j + N / 2));

\*(\*(arr + (i + N / 2)) + (j + N / 2)) = buf;

}

}

for (int i = 0; i < N / 2; i++)

{

for (int j = N / 2; j < N; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + (i + N / 2)) + (j - N / 2));

\*(\*(arr + (i + N / 2)) + (j - N / 2)) = buf;

}

}

printMatrix(arr);

}

/\*

Задание 2, часть C - Перестановка блоков матрицы ( 1 <-> 3 и 2 <-> 4).

\*/

void perestanovkaC(int(\*arr)[N])

{

printMatrix(arr);

cout << endl;

cout << "Перестановка C:\n";

int buf;

for (int i = 0; i < N / 2; i++)

{

for (int j = 0; j < N / 2; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + (i + N / 2)) + j);

\*(\*(arr + (i + N / 2)) + j) = buf;

}

}

for (int i = 0; i < N / 2; i++)

{

for (int j = N / 2; j < N; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + (i + N / 2)) + j);

\*(\*(arr + (i + N / 2)) + j) = buf;

}

}

printMatrix(arr);

}

/\*

Задание 2, часть D - Перестановка блоков матрицы ( 1 <-> 2 и 3 <-> 4).

\*/

void perestanovkaD(int(\*arr)[N])

{

printMatrix(arr);

cout << endl;

cout << "Перестановка D:\n";

int buf;

for (int i = 0; i < N / 2; i++)

{

for (int j = 0; j < N / 2; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + i) + (j + N / 2));

\*(\*(arr + i) + (j + N / 2)) = buf;

}

}

for (int i = N / 2; i < N; i++)

{

for (int j = 0; j < N / 2; j++)

{

buf = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + i) + (j + N / 2));

\*(\*(arr + i) + (j + N / 2)) = buf;

}

}

printMatrix(arr);

}

void sort(int(\*arr)[N])

{

int temp;

for (int c = 0; c <= (N \* N); c++)//колличество проходов

{

//в строках упорядочиваем

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

{

for (int j = 0; j < N - 1; j++)

{

if (\*(\*(arr +i) + j) > \*(\*(arr +i) + (j + 1)))

{

temp = \*(\*(arr + i) + j);

\*(\*(arr + i) + j) = \*(\*(arr + i) + (j + 1));

\*(\*(arr + i) + (j + 1)) = temp;

}

}

}

//в столбах упорядочиваем

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

{

for (int z = 0; z < N - 1; z++)

{

if (\*(\*(arr + z) + x) > \*(\*(arr + z) + (x + 1)))

{

temp = \*(\*(arr + z) + x);

\*(\*(arr + z) + x) = \*(\*(arr + z) + (x + 1));

\*(\*(arr + z) + (x + 1)) = temp;

}

}

}

}

}

void addition(int(\*arr)[N])

{

printMatrix(arr);

int key;

cout << "Введите число для сложения: ";

cin >> key;

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

{

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

{

\*(\*(arr + i) + j) = \*(\*(arr + i) + j) + key;

}

}

cout << "\nМатрица после прибавления :" << key << "\n\n";

printMatrix(arr);

cout << endl;

}

void subtraction(int(\*arr)[N])

{

printMatrix(arr);

int key;

cout << "Введите число для вычитания: ";

cin >> key;

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

{

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

{

\*(\*(arr + i) + j) = \*(\*(arr + i) + j) - key;

}

}

cout << "\nМатрица после вычитания: " << key << "\n\n";

printMatrix(arr);

}

void multiplication(int(\*arr)[N])

{

printMatrix(arr);

int key;

cout << "Введите число для умножения: ";

cin >> key;

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

{

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

{

\*(\*(arr + i) + j) = \*(\*(arr + i) + j) \* key;

}

}

cout << "\nМатрица после умножения на :" << key <<"\n\n";

printMatrix(arr);

}

void division(int(\*arr)[N])

{

printMatrix(arr);

int key;

cout << "Введите число для деления: ";

cin >> key;

if (key != 0)

{

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

{

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

{

\*(\*(arr + i) + j) = \*(\*(arr + i) + j) / key;

}

}

cout << "\nМатрица после деления:\n\n";

printMatrix(arr);

}

else {

cout << "\nПопробуйте другое число ):\n";

}

}

int main()

{

setlocale(0, "rus");

srand(time(0));

int arr[N][N]{};

while (true)

{

cout << "MATRIX:\n";

printMatrix(arr);

int choice;

cout << "\nWat du you want?\n";

cout << "0 - Выйти из программы.\n";

cout << "1 - Заполнить матрицу спиралью.\n";

cout << "2 - Заполнить матрицу змейкой.\n";

cout << "3 - Переставить блоки по схеме - A.\n";

cout << "4 - Переставить блоки по схеме - B.\n";

cout << "5 - Переставить блоки по схеме - C.\n";

cout << "6 - Переставить блоки по схеме - D.\n";

cout << "7 - Отсортировать матрицу.\n";

cout << "8 - Прибавить число.\n";

cout << "9 - Вычесть число.\n";

cout << "10 - Умножить на число.\n";

cout << "11 - Разделить на число.\n";

cout << "\nEnter the option number:";

cin >> choice;

switch (choice)

{

case 1:

fillScroll(arr);

printMatrix(arr);

system("cls");

break;

case 2:

fillSnake(arr);

printMatrix(arr);

system("cls");

break;

case 3:

system("cls");

perestanovkaA(arr);

cout << endl;

break;

case 4:

perestanovkaB(arr);

cout << endl;

break;

case 5:

perestanovkaC(arr);

break;

case 6:

perestanovkaD(arr);

cout << endl;

break;

case 7:

system("cls");

printMatrix(arr);

sort(arr);

cout << "Массив после сортировки: \n";

printMatrix(arr);

system("pause");

system("cls");

break;

case 8:

addition(arr);

break;

case 9:

subtraction(arr);

break;

case 10:

multiplication(arr);

break;

case 11:

division(arr);

break;

case 0:

return 0;

}

}

}