**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 |  | Васильев Ю.А. |
| 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.**

Двумерный массив — это “массив массивов”, т.е. в каждом элементе хранится ещё один массив.

Массив же представляет собой индексированную последовательность однотипных элементов с заранее определенным количеством элементов.

Наглядно двумерный массив можно представить, как матрицу элементов. Индексация массива и его “подмассивов” начинаются с нуля.

Рассмотрим пример. Объявить двумерный массив размером 25 можно так:

int a[5] [5] {};

В данном случае его можно представить как квадратную матрицу 5го порядка:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |

Теперь рассмотрим строение двумерного массива в памяти. Все его “подмассивы” в памяти расположены последовательно. 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.

Так же существуют и другие способы получить этот же элемент:

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

В примере выше мы явно приводим arr к указателю (int\*), затем используем арифметику указателей.

int element = \*(\*(arr + 2) + 1);

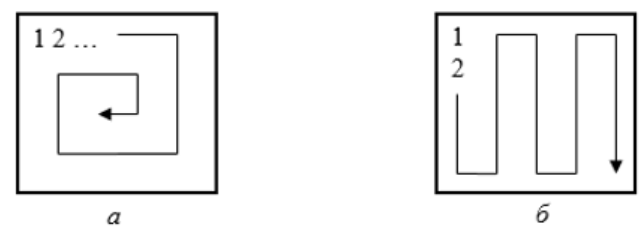
Пример выше практически не отличается от первого, но в разыменовывании указателя arr мы прибавляем 2, тем самым выбирая указатель на 3ю строку, после чего получаем нужный элемент. Эта запись эквивалентна следующей:

int element = arr[2][1];

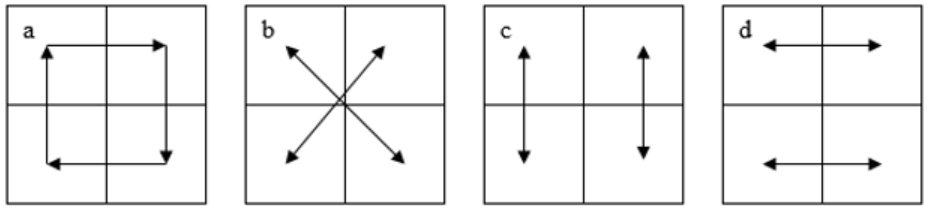
**Постановка задачи.**

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):



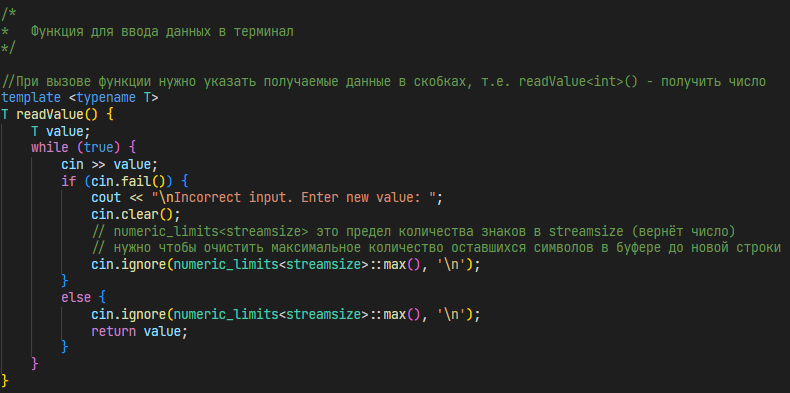
1. Необходимо получить новую матрицу, применяя преобразования к матрице из шага 1:



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

**Выполнение работы.**

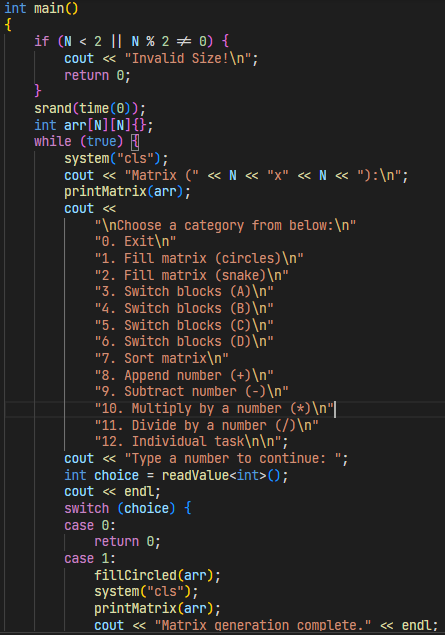
Вспомогательная функция ввода в терминал и получения из него данных в переменную (принимает только указанный тип, иначе выводит ошибку):

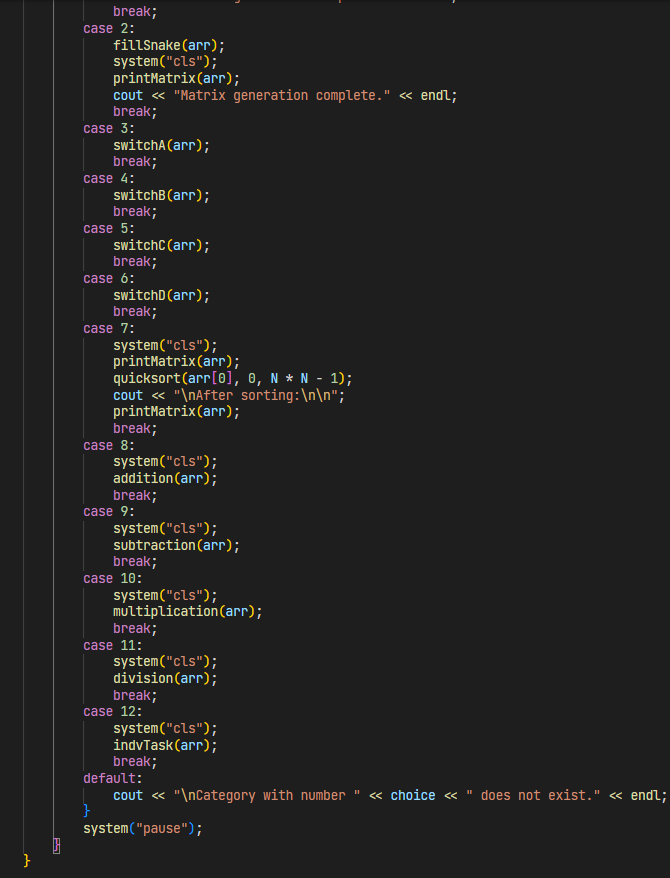


Возможность выбора размера матрицы (кратно 2):



Меню выбора функционала программы. Для перехода нужно указать число 0-12:





Вспомогательные функции (вывод матрицы, обнуление матрицы, промежуточный этап заполнения, получение случайного числа от 1 до N\*N):

Метод “вывод матрицы” выполняет проход по всем элементам матрицы, выводит их на экран.

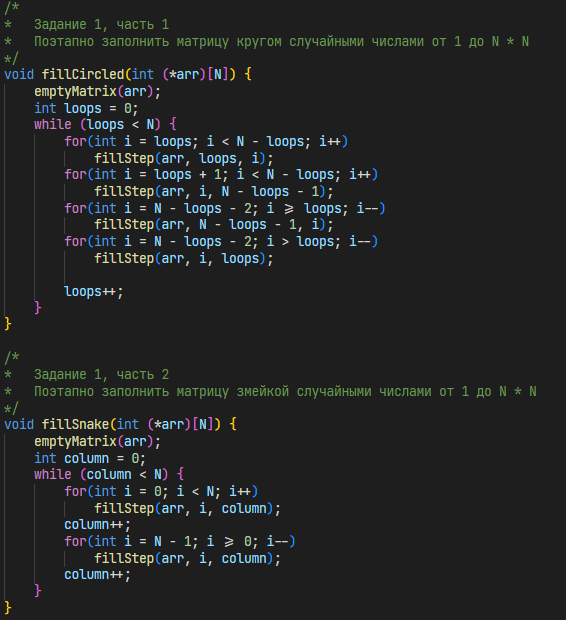
Метод “обнуление матрицы” устанавливает все элементы матрицы на 0 (исходное значение).

Промежуточный этап заполнения нужен для остановки выполнения программы до нажатия клавиши и вывода изменений на экран.

Метод “Получение случайного числа” выдаёт случайное число от 1 до N\*N.



Задание 1, заполнение матрицы:

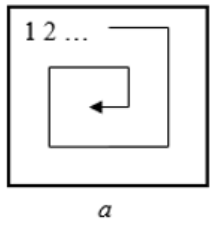


*fillCircled – заполнение “по кругу”:*

Вводится переменная loops, от которой зависят пределы циклов for.

В цикле while поэтапно вводятся изменения в матрицу.

После выполнения одного цикла loops увеличивается на 1, тем самым сужая область заполнения.

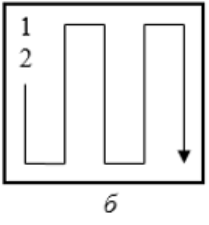


*fillSnake – заполнение змейкой:*

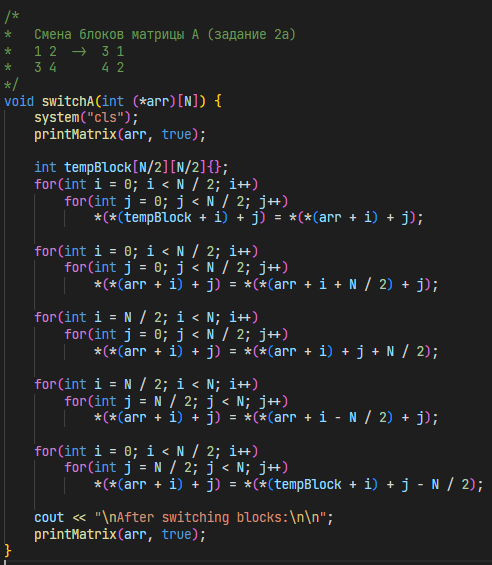
Вводится переменная column, определяющая стадию заполнения и пределы циклов for.

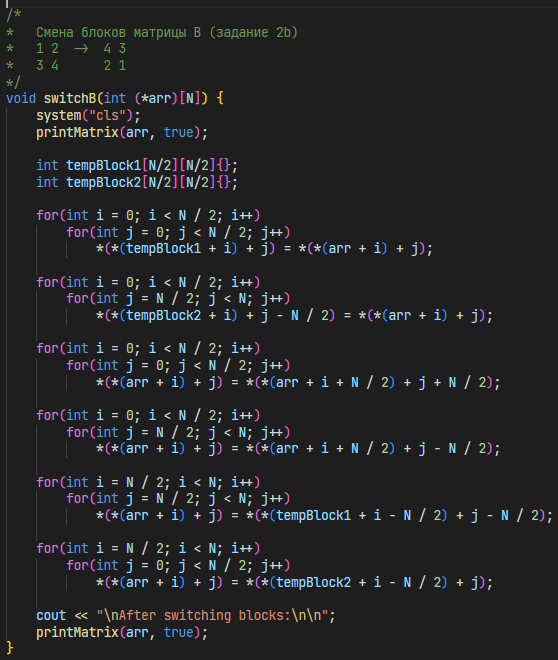
В цикле while поэтапно вводятся изменения в матрицу.

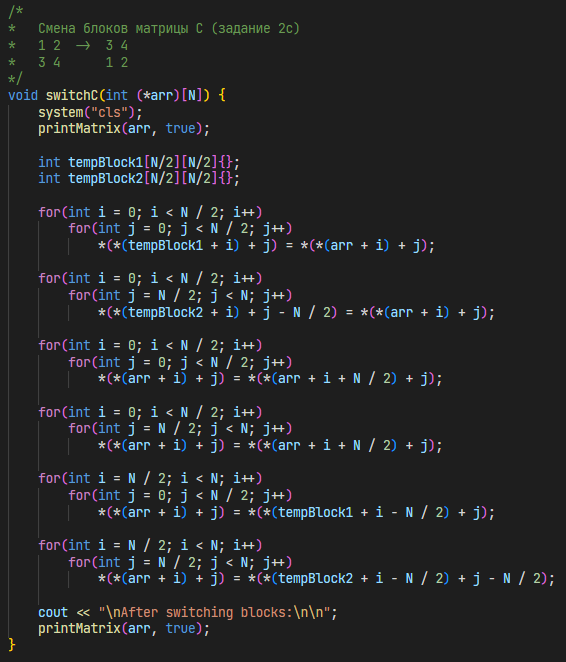
После выполнения одного цикла column увеличивается на 1, тем самым сужая область заполнения.

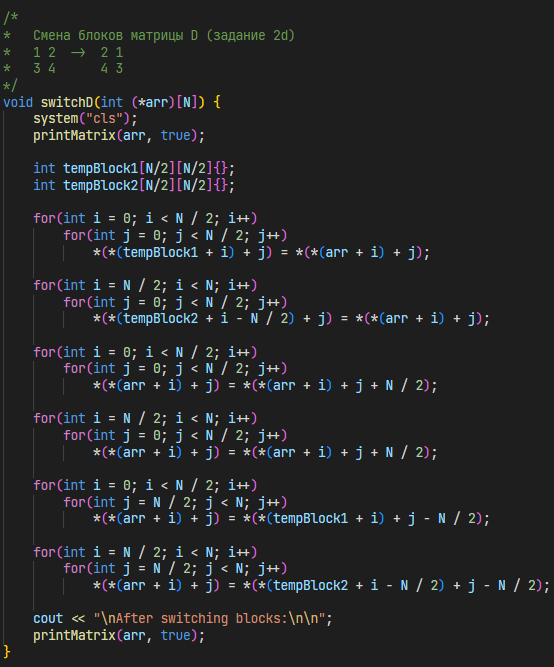


Задание 2, смена блоков в матрице:



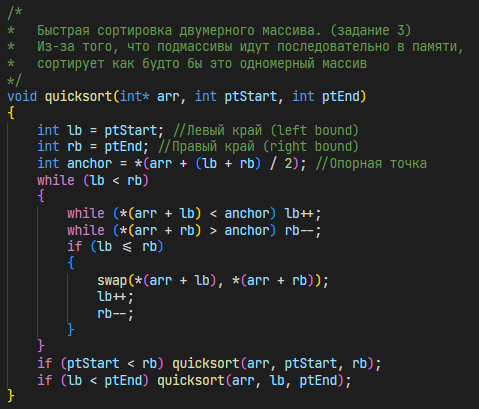






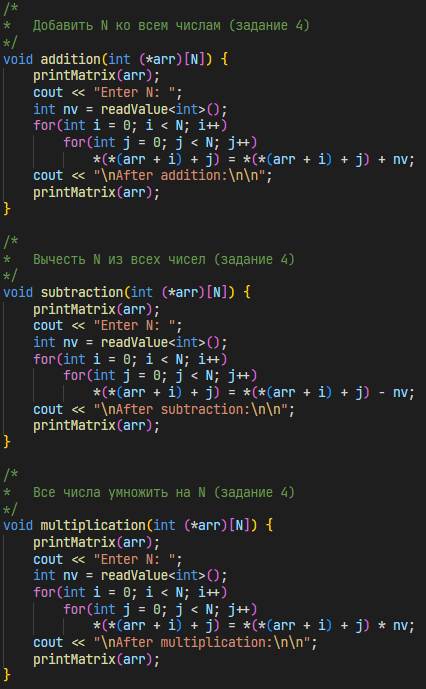
Вводятся временные блоки tempBlock, куда сохраняются блоки, которые будут перезаписаны. Далее выполняются смены блоков, а после этого временный блок записывается на своё место.

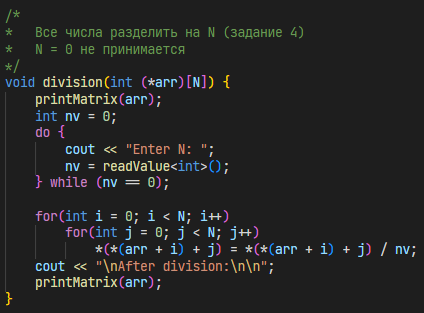
Задание 3, сортировка элементов двумерного массива. Я выбрал быструю сортировку. На вход подаётся указатель на его первый элемент:



Сложение (addition), Вычитание (subtraction), Умножение (multiplication), Деление (division) элементов матрицы и числа. Ввод числа и простой проход по всем элементам матрицы, выполнение действия.

В делении выполняется проверка на неравность нулю введённому числу.





**Выводы.**

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