**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. 2373 |  | Красножён Е. Д. |
| 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);

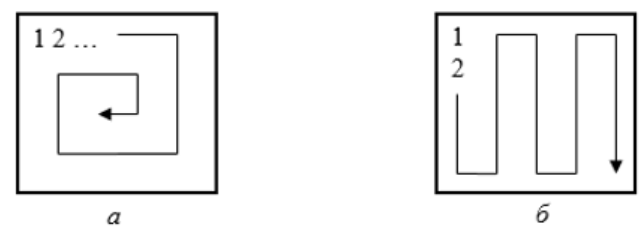
Пример выше практически не отличается от первого, но в разыменовывании указателя arr мы прибавляем 2, тем самым выбирая указатель на 3ю строку, после чего получаем нужный элемент. Эта запись эквивалентна следующей:

int element = arr[2][1];

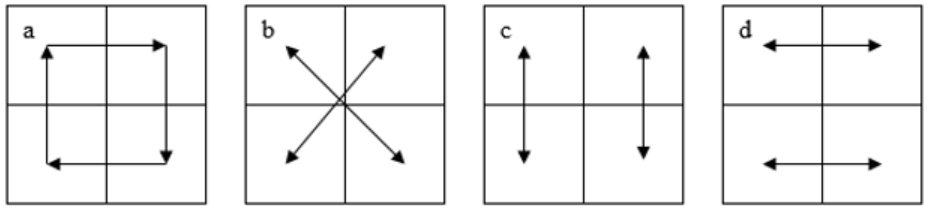
**Постановка задачи.**

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.**

Код программы представлен в приложении А.

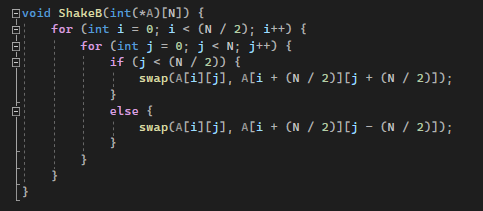
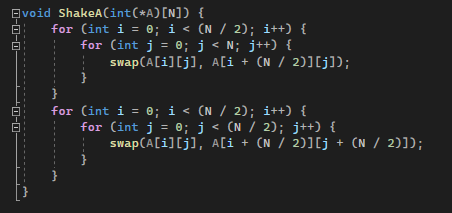
**Блок описания кода и использованных алгоритмов:**

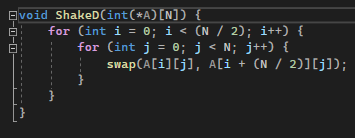
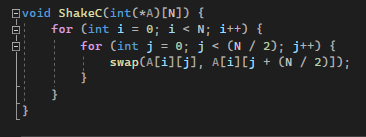
1. Создаём двумерный массив и заполняем его, как указано на схеме а.

Изображение выглядит как текст

Автоматически созданное описание

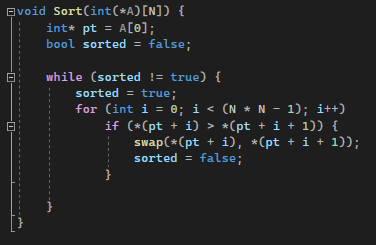
1. Создаём функции по перемешиванию массива, согласно схемам. Под каждый тип перемешивания делаем отдельную функцию:





Далее просто вызываем их поочерёдно.

1. С помощью арифметики указателей реализовываем функцию сортировки. Для простоты был выбран «Bubble sort».

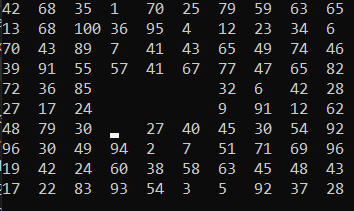


1. Делаем небольшое меню, в котором пользователь выбирает желаемое действие.

Изображение выглядит как текст

Автоматически созданное описание

**Блок скриншотов работы программы**

****

**Изображение выглядит как текст, электроника, клавиатура

Автоматически созданное описание**

**Выводы.**

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

Приложение А

рабочий код

#include <iostream>

#include <ctime>

#include <iomanip>

#include <Windows.h>

using namespace std;

const int N = 6;

void Show(int(\*A)[N]) {

HANDLE hStdout;

COORD destCoord;

hStdout = GetStdHandle(STD\_OUTPUT\_HANDLE);

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

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

{

destCoord.X = i \* 4;

destCoord.Y = j;

SetConsoleCursorPosition(hStdout, destCoord);

cout << A[i][j];

}

}

void ShakeA(int(\*A)[N]) {

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

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

swap(A[i][j], A[i + (N / 2)][j]);

}

}

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

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

swap(A[i][j], A[i + (N / 2)][j + (N / 2)]);

}

}

}

void ShakeB(int(\*A)[N]) {

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

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

if (j < (N / 2)) {

swap(A[i][j], A[i + (N / 2)][j + (N / 2)]);

}

else {

swap(A[i][j], A[i + (N / 2)][j - (N / 2)]);

}

}

}

}

void ShakeC(int(\*A)[N]) {

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

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

swap(A[i][j], A[i][j + (N / 2)]);

}

}

}

void ShakeD(int(\*A)[N]) {

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

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

swap(A[i][j], A[i + (N / 2)][j]);

}

}

}

void Sort(int(\*A)[N]) {

int\* pt = A[0];

bool sorted = false;

while (sorted != true) {

sorted = true;

for (int i = 0; i < (N \* N - 1); i++)

if (\*(pt + i) > \*(pt + i + 1)) {

swap(\*(pt + i), \*(pt + i + 1));

sorted = false;

}

}

}

int main()

{

setlocale(0, "");

srand(time(0));

int A[N][N];

/\* Задание 1 \*/ {

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

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

A[i][j] = 0;

}

Show(A);

int\* end = A[0] + N \* N - 1; // указ. на последний элемент

HANDLE hStdout;

COORD destCoord;

hStdout = GetStdHandle(STD\_OUTPUT\_HANDLE);

for (int\* pt = A[0], i = 0, x = 0, y = 0; pt <= end + (N - 1); pt++, i++, y++) {

if (i % N == 0 && i != 0) {

x++;

y = 0;

if (x % 2 == 0)

pt = &A[x][y];

else

pt = &A[x][y + N - 1];

}

if (x % 2 == 0) {

destCoord.X = x \* 4;

destCoord.Y = y;

SetConsoleCursorPosition(hStdout, destCoord);

\*pt = 1 + rand() % (N \* N);

cout << \*pt;

Sleep(50);

}

else {

destCoord.X = x \* 4;

destCoord.Y = N - y - 1;

SetConsoleCursorPosition(hStdout, destCoord);

\*(pt - y - y) = 1 + rand() % (N \* N);

cout << \*(pt - y - y);

Sleep(50);

}

}

cout << "\r";

for (int i = 1; i <= N; i++)

cout << "\n";

system("pause");

}

/\* Задание 2 \*/ {

system("cls");

ShakeA(A);

Show(A);

Sleep(2000);

system("cls");

ShakeB(A);

Show(A);

Sleep(2000);

system("cls");

ShakeC(A);

Show(A);

Sleep(2000);

system("cls");

ShakeD(A);

Show(A);

cout << "\n";

system("pause");

}

/\* Задание 3 \*/ {

system("cls");

Sort(A);

Show(A);

cout << "\n";

system("pause");

}

/\* Задание 4 \*/ {

int input;

string sign;

Menu: {

cout << "Напишите через пробел действие ('+', '-', '\*', '/') и на сколько вы хотите его выполнить\n";

cout << "Пример: '+ 5'\n";

cin >> sign >> input;

}

if (sign == "+") {

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

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

A[i][j] += input;

}

else if (sign == "-") {

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

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

A[i][j] -= input;

}

else if (sign == "\*") {

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

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

A[i][j] \*= input;

}

else if (sign == "/") {

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

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

A[i][j] /= input;

}

else {

goto Menu;

}

system("cls");

Show(A);

cout << "\n";

}

}