**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 |  | Коваленко М.А. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2022

**Цель работы.**

Научиться работать с двумерными статистическими массивами, закрепить навык работы с указателями, заполнять матрицу по разным схемам, перемещать блоки матрицы согласно предложенным схемам, применять разные способы сортировок для матрицы, и выполнять различные действия с числами нашей матрицы.

**Основные теоретические положения.**

Указатели и ссылки являются одними из самых важных и достаточно сложных для понимания и использования средств языка программирования. Они ориентированы на прямую работу с памятью компьютера. С помощью этих средств реализуется работа с динамической памятью и динамическими объектами, возвращение из функций измененных данных и многое другое. К использованию указателей и ссылок мы будем неоднократно возвращаться в последующих разделах.

Указатели поддерживают ряд операций: присваивание, получение адреса указателя, получение значения по указателю, некоторые арифметические операции и операции сравнения.

К указателям можно применять некоторые арифметические операции. К таким операциям относятся: +,**-**, **++**, **--**. Результаты выполнения этих операций по отношению к указателям существенно отличаются от результатов соответствующих арифметических операций, выполняющихся с обычными числовыми данными.

**Постановка задачи.**

1. Используя арифметику указателей, заполняет квадратичную целочисленную матрицу порядка N (6,8,10) случайными числами от 1 до N\*N согласно схемам, приведенным на рисунках. Пользователь должен видеть процесс заполнения квадратичной матрицы



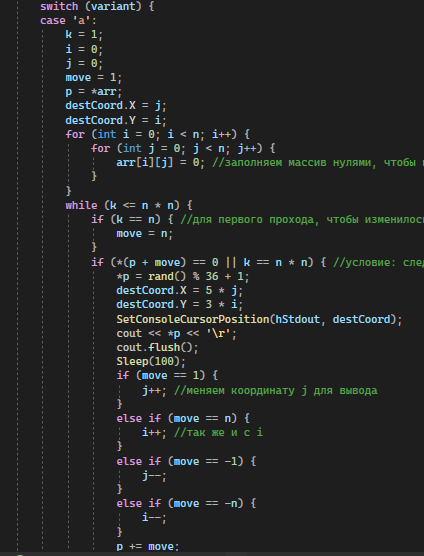
1. Получает новую матрицу, из матрицы п. 1, переставляя ее блоки в соответствии со схемами:



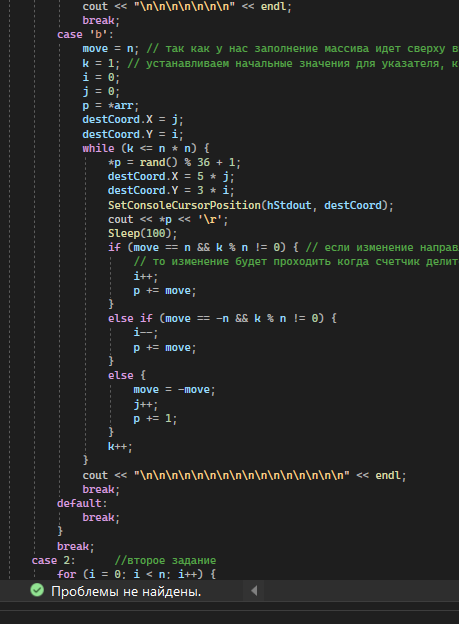
3)    Используя арифметику указателей, сортирует элементы любой сортировкой.

4)    Уменьшает, увеличивает, умножает или делит все элементы матрицы на введенное пользователем число.

**Выполнение работы.**

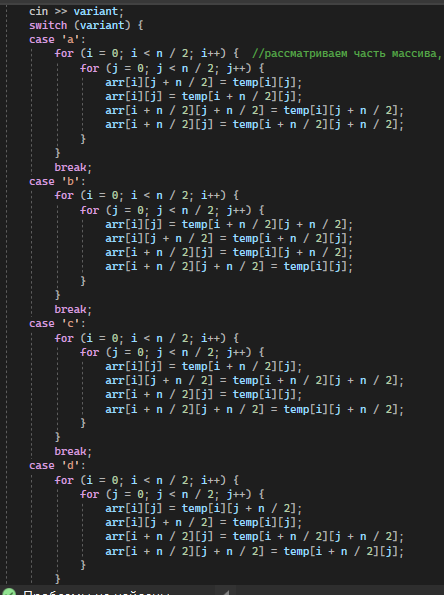
Создаём исходную матрицу размерностью 6 на 6

Далее мы даём пользователю выбор каким способом он хочет видеть заполнение данной матрицы



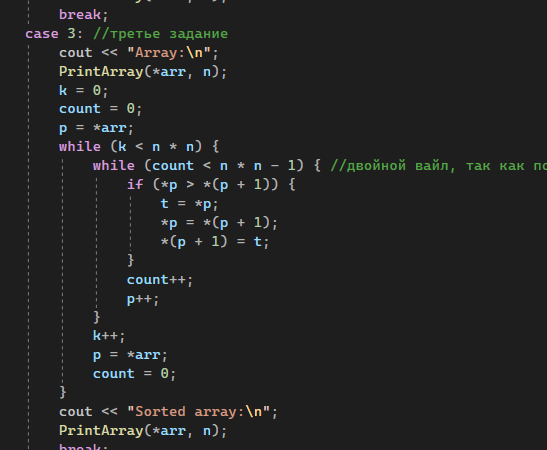
Для этого мы написали второй способ заполнения матрицы

Далее мы делаем так, чтобы пользователь мог менять местами блоки матрицы соответственно приведённым схемам

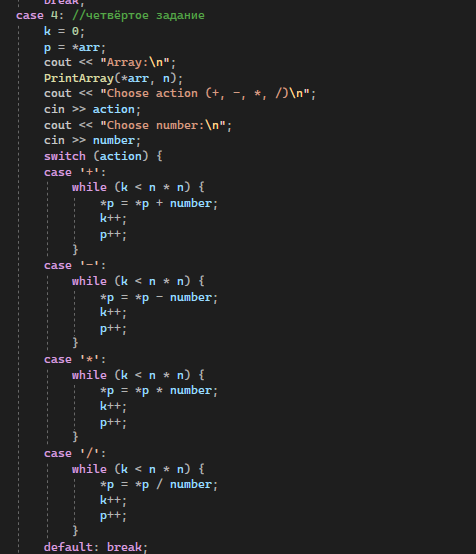


На рисунке приведены все 4 кода для выполнения 4 разных способов перестановки

Далее мы сортируем нашу матрицу любым способом



И в конце нашей работы мы выполняем любые операции с числами матрицы с числом, которое ввёл пользователь



**Выводы.**

В ходе практической работы мы усвоили, как заполнять матрицу разными способами, научились переставлять блоки матрицы соответственно схемам, приведёнными в примере, повторили как нужно применять способы сортировок, и выполняли простейшие операции с числами нашей матрицы.

Приложение А

Полный код программы

#include <iostream>

#include <windows.h>

#include <ctime>

using namespace std;

void PrintArray(int\* arr, int n) { // функция вывода массива

int\* p = arr;

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

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

cout << \*p << " ";

p++;

}

cout << endl;

}

}

int main() {

srand(time(0)); //для рандома

HANDLE hStdout; // из дискорда

COORD destCoord;

hStdout = GetStdHandle(STD\_OUTPUT\_HANDLE);

const int n = 6; // размер массива

int arr[n][n]; // массив

int\* p = \*arr;; //указатель на массив

int k = 1; //счетчик

int move = 1; // направление движения 1 - вправо

int j = 0;

int i = 0; // индексы массива для вывода

int answer = 0; // repeat?

int task; // выбор задания

char variant; //для выбора a, b, c, d

//для второго задания

int temp[n][n];

//для третьего задания

int t = 0; // так как указатели не имеют оператора функции swap

int count = 0; // счетчик

//для четвертого задания

char action; // действие

int number; // число с которым будут действовать элементы массива

do {

cout << "choose task(1-4)" << endl;

cin >> task;

switch (task) {

case 1://первое задание

cout << "a or b?" << endl;

cin >> variant;

system("cls");

switch (variant) {

case 'a':

k = 1;

i = 0;

j = 0;

move = 1;

p = \*arr;

destCoord.X = j;

destCoord.Y = i;

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

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

arr[i][j] = 0; //заполняем массив нулями, чтобы поставить условие

}

}

while (k <= n \* n) {

if (k == n) { //для первого прохода, чтобы изменилось направление заполнения

move = n;

}

if (\*(p + move) == 0 || k == n \* n) { //условие: следующий элемент массива равен нулю и если к = 36, то выполнится

\*p = rand() % 36 + 1;

destCoord.X = 5 \* j;

destCoord.Y = 3 \* i;

SetConsoleCursorPosition(hStdout, destCoord);

cout << \*p << '\r';

cout.flush();

Sleep(100);

if (move == 1) {

j++; //меняем координату j для вывода

}

else if (move == n) {

i++; //так же и с i

}

else if (move == -1) {

j--;

}

else if (move == -n) {

i--;

}

p += move;

k++;

}

else { //если наше условие не выполняется, то меняем направление

switch (move) {

case 1:

move = n; // n - вниз

break;

case n:

move = -1; // -1 - влево

break;

case -1:

move = -n; // -n - вверх

break;

case -n:

move = 1; // ну и 1 - вправо

break;

default:

break;

}

}

}

cout << "\n\n\n\n\n\n\n" << endl;

break;

case 'b':

move = n; // так как у нас заполнение массива идет сверху вниз, то первое направление движения будет вниз

k = 1; // устанавливаем начальные значения для указателя, координат, счетчика

i = 0;

j = 0;

p = \*arr;

destCoord.X = j;

destCoord.Y = i;

while (k <= n \* n) {

\*p = rand() % 36 + 1;

destCoord.X = 5 \* j;

destCoord.Y = 3 \* i;

SetConsoleCursorPosition(hStdout, destCoord);

cout << \*p << '\r';

Sleep(100);

if (move == n && k % n != 0) { // если изменение направления происходит только тогда, когда наш столбец из 6 символов заполняется

// то изменение будет проходить когда счетчик делится нацело на 6

i++;

p += move;

}

else if (move == -n && k % n != 0) {

i--;

p += move;

}

else {

move = -move;

j++;

p += 1;

}

k++;

}

cout << "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n" << endl;

break;

default:

break;

}

break;

case 2: //второе задание

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

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

temp[i][j] = arr[i][j]; //создаем такой же массив как и искомый

}

}

cout << "Array:\n";

PrintArray(\*arr, n);

cout << "choose variant (a,b,c,d)\n";

cin >> variant;

switch (variant) {

case 'a':

for (i = 0; i < n / 2; i++) { //рассматриваем часть массива, которая в два раза меньше искомого и просто меняем значения на нужные нам

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

arr[i][j + n / 2] = temp[i][j];

arr[i][j] = temp[i + n / 2][j];

arr[i + n / 2][j + n / 2] = temp[i][j + n / 2];

arr[i + n / 2][j] = temp[i + n / 2][j + n / 2];

}

}

break;

case 'b':

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

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

arr[i][j] = temp[i + n / 2][j + n / 2];

arr[i][j + n / 2] = temp[i + n / 2][j];

arr[i + n / 2][j] = temp[i][j + n / 2];

arr[i + n / 2][j + n / 2] = temp[i][j];

}

}

break;

case 'c':

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

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

arr[i][j] = temp[i + n / 2][j];

arr[i][j + n / 2] = temp[i + n / 2][j + n / 2];

arr[i + n / 2][j] = temp[i][j];

arr[i + n / 2][j + n / 2] = temp[i][j + n / 2];

}

}

break;

case 'd':

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

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

arr[i][j] = temp[i][j + n / 2];

arr[i][j + n / 2] = temp[i][j];

arr[i + n / 2][j] = temp[i + n / 2][j + n / 2];

arr[i + n / 2][j + n / 2] = temp[i + n / 2][j];

}

}

break;

}

PrintArray(\*arr, n);

break;

case 3: //третье задание

cout << "Array:\n";

PrintArray(\*arr, n);

k = 0;

count = 0;

p = \*arr;

while (k < n \* n) {

while (count < n \* n - 1) { //двойной вайл, так как после первой итерации на место встает только один элемент

if (\*p > \*(p + 1)) {

t = \*p;

\*p = \*(p + 1);

\*(p + 1) = t;

}

count++;

p++;

}

k++;

p = \*arr;

count = 0;

}

cout << "Sorted array:\n";

PrintArray(\*arr, n);

break;

case 4: //четвёртое задание

k = 0;

p = \*arr;

cout << "Array:\n";

PrintArray(\*arr, n);

cout << "Choose action (+, -, \*, /)\n";

cin >> action;

cout << "Choose number:\n";

cin >> number;

switch (action) {

case '+':

while (k < n \* n) {

\*p = \*p + number;

k++;

p++;

}

case '-':

while (k < n \* n) {

\*p = \*p - number;

k++;

p++;

}

case '\*':

while (k < n \* n) {

\*p = \*p \* number;

k++;

p++;

}

case '/':

while (k < n \* n) {

\*p = \*p / number;

k++;

p++;

}

default: break;

}

PrintArray(\*arr, n);

default:

break;

}

cout << "repeat? (0 - no, 1 - yes)" << endl;

cin >> answer;

system("cls");

} while (answer);

}