**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 |  | Михеева Э. Д. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2022

**Цель работы.**

Изучение двумерных статических массив и основных алгоритмов работы с ними. Изучение ссылок и указателей, арифметики указателей. Написание программы, заполняющей матрицу размера N (N = 6, 8, 10) и заполняющий его паттернами «змейка» и «спираль». Программа должна также менять местами блоки (подмассивы) матрицы в соответствии с четырьмя паттернами обмена из задания. Сортировать матрицу с использованием арифметики указателей и увеличивать, уменьшать, умножать и делить все элементы матрицы на число, введенное пользователем

**Основные теоретические положения.**

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.

Добавлять к указателям или вычитать из указателей можно только целые значения.

Поскольку упомянутые арифметические операции выполняются по-разному при их применении к указателям и обычным арифметическим типам данных, а также учитывая высший приоритет операции \*, при использовании указателей в составе выражений следует внимательно обращаться со скобками.

Указатели – это очень мощное, полезное, но и очень опасное средство. Ошибки, которые возникают при неправильном использовании указателей, кроме того, что могут приводить к серьезным и непредсказуемым ошибкам в работе программы, еще и очень трудно диагностировать (обнаруживать).

**Постановка задачи.**

Разработать алгоритм и написать программу, которая:

1. Создает целочисленную матрицу размерности N\*N и заполняет её нулями.
2. Заполняет матрицу случайными числами от 1 до N^2 змейкой
3. Заполняет матрицу случайными числами от 1 до N^2 по спирали
4. Меняет местами блоки матрицы четырьмя методами, метод выбирает пользователь
5. Сортирует матрицу, используя арифметику указателей
6. Увеличивает, уменьшает, умножает или делит все элементы матрицы на число, которое вводит пользователь. Операция также выбирается пользователем.

**Выполнение работы.**

Код программы представлен в приложении А.

1. При запуске программы пользователю выводится меню доступных команд и ожидается ввод размера матрицы, а затем ожидается ввод команды с клавиатуры (рис. 1).

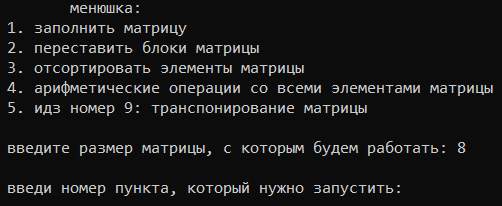


Рисунок 1. Запуск программы

1. Следующий шаг зависит от введенной команды, если пользователь ввёл:
   1. “1” – предлагается 2 способа заполнения матрицы (рис. 2). При вводе “a” массив заполняется с помощью паттерна «Змейка» (рис. 3), “b” – с помощью паттерна «Спираль» (рис. 4)

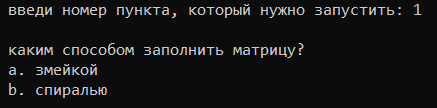


Рисунок 2. Задача 1

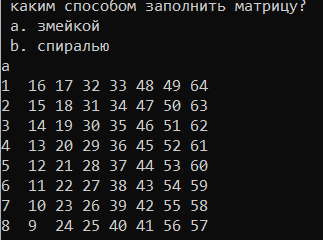


Рисунок 3. Змейка

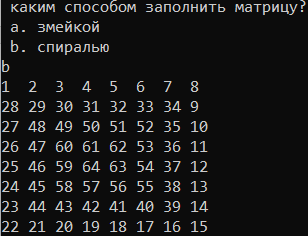


Рисунок 4. Спираль

* 1. “2” – ожидается пользовательский выбор паттерна обмена блоков матрицы. После выбора производится обмен и выводятся значения матрицы после обмена (рис. 5).

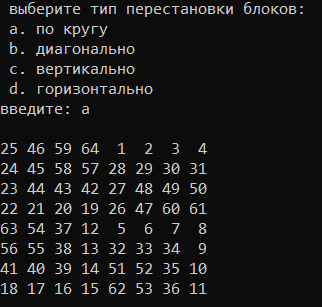
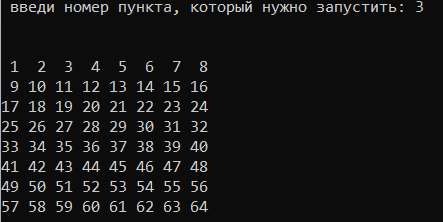


Рисунок 5. Задача 2

* 1. “3” – выполняется сортировка матрицы сортировкой вставками с использованием арифметики указателей (рис. 6).

  
 Рисунок 6. Задача 3

* 1. “4” – ожидается пользовательский выбор операции (сложение, вычитание, умножение или деление) и все элементы увеличиваются, уменьшаются, умножаются или делятся (в зависимости от выбранной операции) на число, которое также вводит пользователь (рис. 7).

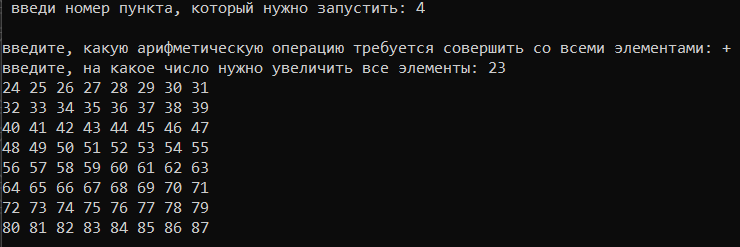


Рисунок 7. Задача 4

* 1. “5” – производится транспонирование матрицы (идз) (рис. 8).

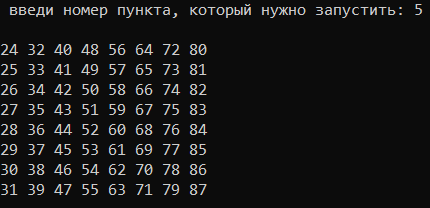


Рисунок 8. Задача 5

**Выводы.**

В ходе работы были изучены двумерные статические массивы и методы работы с ними. Изучены ссылки и указатели, арифметика указателей.

Приложение А

рабочий код

#include <iostream>  
#include <Windows.h>  
#include <thread>  
using namespace std;  
  
void XY(int x, int y){  
 COORD coord;  
 coord.X = x;  
 coord.Y = y;  
 SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord);  
 this\_thread :: sleep\_for(chrono :: milliseconds (100));  
}  
  
int getCursorPositionY(){  
 CONSOLE\_SCREEN\_BUFFER\_INFO screenBufferInfo = {};  
 GetConsoleScreenBufferInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &screenBufferInfo);  
 return screenBufferInfo.dwCursorPosition.Y;  
}  
  
void nullArr (int n, int \*arr){  
 for(int i = 1; i <= n\*n; i++) \*(arr++) = 0;  
}  
  
void arrOut (int n, int \*arr){  
 for(int \*parr = arr, i = 1; i <= n\*n; i++, parr++){  
 if (\*parr < 10) putchar(' ');  
 cout << \*parr << " ";  
 if (i % n == 0) cout << "\n";  
 }  
}  
  
void spiraling (int n, int \*arr){  
 int \*parr = arr;  
 int y0 = getCursorPositionY();  
 int qq = 1;  
 for (int prohod = 1; prohod <= n; prohod++){  
 if (prohod == 1){  
 for(int i = 1; i <= n; i++){  
 \*parr = qq;  
 XY((parr - arr) % n \* 3, y0 + (parr - arr) / n);  
 cout << \*parr;  
 parr++;  
 qq++;  
 }  
 parr--;  
 }  
 if (prohod % 2 == 0){  
 for(int i = 1; i <= n-prohod+1; i++){  
 parr+=n;  
 \*parr = qq;  
 XY((parr - arr) % n \* 3, y0 + (parr - arr) / n);  
 cout << \*parr;  
 qq++;  
 }  
 for(int j = 1; j <= n-prohod+1; j++){  
 parr--;  
 \*parr = qq;  
 XY((parr - arr) % n \* 3, y0 + (parr - arr) / n);  
 cout << \*parr;  
 qq++;  
 }  
 }  
 if ((prohod % 2 == 1) & (prohod != 1)){  
 for(int i = 1; i <= n-prohod+1; i++){  
 parr-=n;  
 \*parr = qq;  
 XY((parr - arr) % n \* 3, y0 + (parr - arr) / n);  
 cout << \*parr;  
 qq++;  
 }  
 for(int j = 1; j <= n-prohod+1; j++){  
 parr++;  
 \*parr = qq;  
 XY((parr - arr) % n \* 3, y0 + (parr - arr) / n);  
 cout << \*parr;  
 qq++;  
 }  
 }  
 }  
 for (int i = 0; i < n / 2; ++i) putchar('\n');  
}  
  
void snaking (int n, int \*arr){  
 int \*parr = arr;  
 int y0 = getCursorPositionY();  
 int qq = 0;  
 for (int prohod = 1; prohod <= n; prohod++){  
 if (prohod % 2 == 0){  
 for(int i = 1; i <= n; i++){  
 qq++;  
 parr-=n;  
 \*parr = qq;  
 XY((parr - arr) % n \* 3, y0 + (parr - arr) / n);  
 cout << \*parr;  
 }  
 }  
 if (prohod % 2 == 1){  
 for(int i = 1; i <= n; i++){  
 qq++;  
 \*parr = qq;  
 XY((parr - arr) % n \* 3,y0 + (parr - arr) / n);  
 cout << \*parr;  
 parr+=n;  
 }  
 }  
 parr++;  
 }  
 for (int i = 0; i < n; ++i) putchar('\n');  
}  
  
void splitting(int n, int \*arr, int \*mat1, int \*mat2, int \*mat3, int \*mat4){  
 int \*parr = arr, \*pm1 = mat1, \*pm2 = mat2, \*pm3 = mat3, \*pm4 = mat4;  
 for (int i = 1; i <= n; i++){  
 for (int j = 1; j <= n; j++, parr++){  
 if ((i <= n/2) & (j <= n/2)) {  
 \*pm1 = \*parr;  
 pm1++;  
 }  
 if ((i > n/2) & (j <= n/2)){  
 \*pm3 = \*parr;  
 pm3++;  
 }  
 if ((i <= n/2) & (j > n/2)){  
 \*pm2 = \*parr;  
 pm2++;  
 }  
 if ((i > n/2) & (j > n/2)){  
 \*pm4 = \*parr;  
 pm4++;  
 }  
 }  
 }  
}  
  
void transpositionA(int n, int \*arr, int \*mat1, int \*mat2, int \*mat3, int \*mat4){  
 int \*parr = arr, \*pm1 = mat1, \*pm2 = mat2, \*pm3 = mat3, \*pm4 = mat4;  
 for (int i = 1; i <= n; i++){  
 for (int j = 1; j <= n; j++, parr++){  
 if ((i <= n/2) & (j <= n/2)) {  
 \*parr = \*pm3;  
 pm3++;  
 }  
 if ((i > n/2) & (j <= n/2)){  
 \*parr = \*pm4;  
 pm4++;  
 }  
 if ((i <= n/2) & (j > n/2)) {  
 \*parr = \*pm1;  
 pm1++;  
 }  
 if ((i > n/2) & (j > n/2)) {  
 \*parr = \*pm2;  
 pm2++;  
 }  
 }  
 }  
}  
  
void transpositionB(int n, int \*arr, int \*mat1, int \*mat2, int \*mat3, int \*mat4){  
 int \*parr = arr, \*pm1 = mat1, \*pm2 = mat2, \*pm3 = mat3, \*pm4 = mat4;  
 for (int i = 1; i <= n; i++){  
 for (int j = 1; j <= n; j++, parr++){  
 if ((i <= n/2) & (j <= n/2)){  
 \*parr = \*pm4;  
 pm4++;  
 }  
 if ((i > n/2) & (j <= n/2)){  
 \*parr = \*pm2;  
 pm2++;  
 }  
 if ((i <= n/2) & (j > n/2)){  
 \*parr = \*pm3;  
 pm3++;  
 }  
 if ((i > n/2) & (j > n/2)){  
 \*parr = \*pm1;  
 pm1++;  
 }  
 }  
 }  
}  
  
void transpositionC(int n, int \*arr, int \*mat1, int \*mat2, int \*mat3, int \*mat4){  
 int \*parr = arr, \*pm1 = mat1, \*pm2 = mat2, \*pm3 = mat3, \*pm4 = mat4;  
 for (int i = 1; i <= n; i++){  
 for (int j = 1; j <= n; j++, parr++){  
 if ((i <= n/2) & (j <= n/2)){  
 \*parr = \*pm3;  
 pm3++;  
 }  
 if ((i > n/2) & (j <= n/2)){  
 \*parr = \*pm1;  
 pm1++;  
 }  
 if ((i <= n/2) & (j > n/2)){  
 \*parr = \*pm4;  
 pm4++;  
 }  
 if ((i > n/2) & (j > n/2)){  
 \*parr = \*pm2;  
 pm2++;  
 }  
 }  
 }  
}  
  
void transpositionD(int n, int \*arr, int \*mat1, int \*mat2, int \*mat3, int \*mat4){  
 int \*parr = arr, \*pm1 = mat1, \*pm2 = mat2, \*pm3 = mat3, \*pm4 = mat4;  
 for (int i = 1; i <= n; i++){  
 for (int j = 1; j <= n; j++, parr++){  
 if ((i <= n/2) & (j <= n/2)){  
 \*parr = \*pm2;  
 pm2++;  
 }  
 if ((i > n/2) & (j <= n/2)){ //3  
 \*parr = \*pm4;  
 pm4++;  
 }  
 if ((i <= n/2) & (j > n/2)){  
 \*parr = \*pm1;  
 pm1++;  
 }  
 if ((i > n/2) & (j > n/2)){  
 \*parr = \*pm3;  
 pm3++;  
 }  
 }  
 }  
}  
  
void transposition(int n, int \*arr, int \*mat1, int \*mat2, int \*mat3, int \*mat4){  
 splitting(n, arr, mat1, mat2, mat3, mat4);  
 char letter2;  
 cout << " выберите тип перестановки блоков:\n а. по кругу\n b. диагонально\n c. вертикально\n d. горизонтальнo" << endl;  
 cout << "введите: ";  
 cin >> letter2;  
 cout << "\n";  
 switch(letter2) {  
 case 'a': {  
 transpositionA(n, arr, mat1, mat2, mat3, mat4);  
 arrOut(n, arr);  
 }break;  
 case 'b': {  
 transpositionB(n, arr, mat1, mat2, mat3, mat4);  
 arrOut(n, arr);  
 }break;  
 case 'c': {  
 transpositionC(n, arr, mat1, mat2, mat3, mat4);  
 arrOut(n, arr);  
 }break;  
 case 'd': {  
 transpositionD(n, arr, mat1, mat2, mat3, mat4);  
 arrOut(n, arr);  
 }break;  
 }  
}  
  
void arrSort(int n, int \*arr){  
 for (int \*parr1 = arr+1; parr1 < arr + n\*n; ++parr1){  
 for (int \*parr2 = parr1; parr2 > arr && \*(parr2-1) > \*parr2; --parr2){  
 swap(\*parr2,\*(parr2 - 1));  
 }  
 }  
}  
  
void arithmeticOp(int \*arr, int n, char operation, int value){  
 cout << "введите, какую арифметическую операцию требуется совершить со всеми элементами: ";  
 cin >> operation;  
 if (operation == '+') cout << "введите, на какое число нужно увеличить все элементы: ";  
 if (operation == '-') cout << "введите, на какое число нужно уменьшить все элементы: ";  
 if (operation == '\*') cout << "введите, во сколько раз нужно увеличить все элементы: ";  
 if (operation == '/') cout << "введите, на какое число нужно разделить все элементы: ";  
 cin >> value;  
 for (int \*parr = arr; parr < arr + n\*n; ++parr) {  
 switch (operation) {  
 case '+': {  
 \*parr += value;  
 break;  
 }  
 case '-': {  
 \*parr -= value;  
 break;  
 }  
 case '\*': {  
 \*parr \*= value;  
 break;  
 }  
 case '/': {  
 \*parr /= value;  
 break;  
 }  
 }  
 }  
}  
  
int task1(int n, int \*arr){  
 char letter1;  
  
 cout << " каким способом заполнить матрицу?\n а. змейкой\n b. спиралью" << endl;  
 cin >> letter1;  
 nullArr(n, arr);  
 switch(letter1){  
 case 'a': {  
 snaking(n, arr);  
 break;  
 }  
 case 'b': {  
 spiraling(n, arr);  
 break;  
 }  
 }  
 return n;  
}  
  
void transposeMatrix(int \*arr, int n){  
 int size = n;  
 for (int \*pStart = arr; pStart < arr + n\*n; pStart += n + 1) {  
 for (int \*pRow = pStart, \*pCol = pStart; pCol < pStart + size; pRow += n, ++pCol)  
 swap(\*pRow, \*pCol);  
 --size;  
 }  
 arrOut(n, arr);  
}  
  
int main() {  
 SetConsoleOutputCP(CP\_UTF8);  
 cout << "\tменюшка:" << "\n 1. заполнить матрицу" << "\n 2. переставить блоки матрицы" <<  
 "\n 3. отсортировать элементы матрицы\n 4. арифметические операции со всеми элементами матрицы\n 5. идз номер 9: транспонирование матрицы" << endl;  
  
 int n, YorN = 1, operation, value;  
 cout << "\n введите размер матрицы, с которым будем работать: ";  
 cin >> n;  
 int arr[n][n];  
 int mat1[n/2][n/2], mat2[n/2][n/2], mat3[n/2][n/2], mat4[n/2][n/2];  
 int arr1[n][n], arr2[n][n], arr3[n][n];  
 int op;  
 while (YorN == 1) {  
 cout << "\n введи номер пункта, который нужно запустить: " ;  
 cin >> op;  
 cout << "\n";  
 switch (op) {  
 case 1: {  
 task1(n, &arr[0][0]);  
 }break;  
 case 2: {  
 transposition(n, &arr[0][0], &mat1[0][0], &mat2[0][0], &mat3[0][0], &mat4[0][0]);  
 }break;  
 case 3: {  
 cout << "\n";  
 arrSort(n, &arr[0][0]);  
 arrOut(n, &arr[0][0]);  
 }break;  
 case 4: {  
 arithmeticOp(&arr[0][0], n, operation, value);  
 arrOut(n, &arr[0][0]);  
 }break;  
 case 5: {  
 transposeMatrix(&arr[0][0], n);  
 }break;  
 }  
 cout << "\n\n Продолжить выполнение? (1 - да, 0 - нет)" << endl;  
 cin >> YorN;  
 cout << "\n";  
 }  
 return 0;  
}