**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ   
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»**

**ХАБАРОВСКИЙ ИНСТИТУТ ИНФОКОММУНИКАЦИЙ (ФИЛИАЛ)**

**СРЕДНЕЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАНИЕ**

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

по МДК 01.01 Системное программирование

на тему

Разработка консольных приложений с использованием двумерных массивов. Методы для работы с двумерными массивами.

Выполнил: А.В. Медведев

Проверил: Н.А. Насонова

Группа: ПКС-320

2023

Составление программ.

Для начала работы подключим библиотеки для работы с основными функциями. И подключим пространство имён std.

#include <iostream>

#include <cmath>

#include <cstdlib>

#include <ctime>

#include <algorithm>

#include <vector>

#include <iomanip>

using namespace std;

iostream — заголовочный файл с классами, функциями и переменными для организации ввода-вывода.

cmath – заголовочный файл стандартной библиотеки разработанный для выполнения простых математических операций.

cstdlib – заголовочный файл, который содержит в себе функции, занимающиеся выделением памяти, контролем процесса выполнения программы, преобразованием типов и другие.

ctime – включает в себя фунцкии времени.

algorithm – заголовочный файл в стандартной библиотеке языка программирования C++, включающий набор функций для выполнения алгоритмических операций.

vector – стандартный шаблон обобщённого программирования языка C++, реализующий динамический массив.

iomanip – библиотека, в которой содержится модификатор setw.

В методе main опишем использования русского алфавита.

setlocale(LC\_ALL, "Russian");

Дана вещественная матрица размером NxM. Переставляя ее строки и столбцы, добиться того, чтобы наибольший элемент (один из них) оказал-ся в верхнем левом углу.

#include<iostream>

using namespace std;

int main()

{

int i, j, n, m;

double myArr[3][4], max, cur;

cout << "Write elemets 3 X 4:" << endl;

for (i = 0; i < 3; i++)

for (j = 0; j < 4; j++)

cin >> myArr[i][j];

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 4; j++)

cout << myArr[i][j] << "\t";

cout << endl;

}

max = myArr[0][0];

for (i = 0; i < 3; i++)

for (j = 0; j < 4; j++)

if (myArr[i][j] > max)

{

max = myArr[i][j];

n = i, m = j;

}

cout << "Max = myArr" << "[" << n << "][" << m << "] = " << max << endl;

//обмен верхнего левого и найденного максимального элементов

cur = myArr[0][0];

myArr[0][0] = max;

myArr[n][m] = cur;

cout << "Result:\n";

for (int i = 0; i < 7; i++)

{

for (int j = 0; j < 4; j++)

cout << myArr[i][j] << "\t";

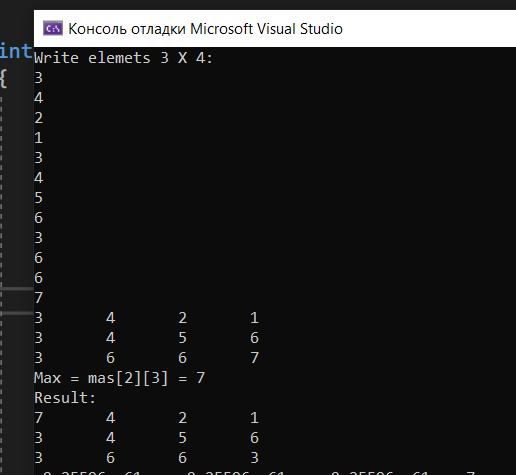
cout << endl;

}

system("pause");

return 0;

}



Дана вещественная матрица размером NxM. Упорядочить ее строки по возрастанию их первых элементов.

#include "iostream"

#include <iomanip>

using namespace std;

int main()

{

double\*\* a, b;

int i, j, n, m, k;

cout << "Vvedite razmer: n,m" << endl;

cin >> n >> m;

a = new double\* [n];

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

a[i] = new double[m];

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

for (j = 0; j < m; j++)

{

cout << "Vvedite a[" << i << "][" << j << "]:";

cin >> a[i][j];

}

cout << "MASSIVE:" << endl;

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

{

for (j = 0; j < m; j++)

cout << setw(10) << a[i][j] << " ";

cout << endl;

}

for (i = 0; i < n - 1; i++) {

for (k = 0; k < n - 1; k++) {

if (a[k][0] > a[k + 1][0]) {

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

b = a[k][j];

a[k][j] = a[k + 1][j];

a[k + 1][j] = b;

}

}

}

}

cout << "Result:" << endl;

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

{

for (j = 0; j < m; j++)

cout << setw(10) << a[i][j] << " ";

cout << endl;

}

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

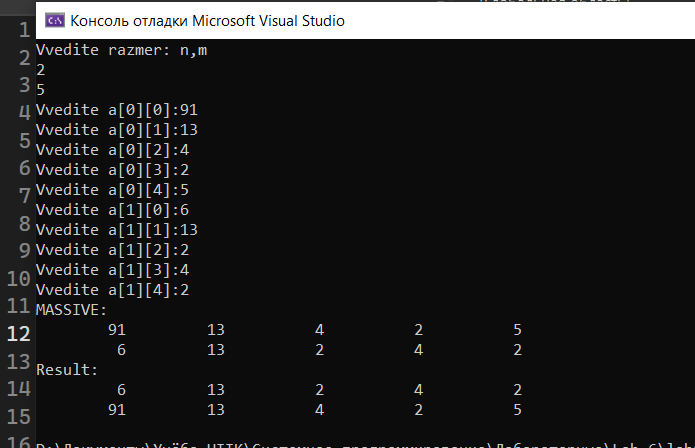
delete[] a[i];

delete[]a;

a = NULL;

return 0;

}



Дана вещественная матрица размером NхM. Упорядочить ее строки по возрастанию суммы их элементов.

#include <iostream>

#include <algorithm>

using namespace std;

bool compareRows(int\* a, int\* b, int m) {

int sumA = 0, sumB = 0;

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

sumA += a[i];

sumB += b[i];

}

return sumA < sumB;

}

int main() {

int n, m;

cout << "Vvedite razmer: n,m" << endl;

cin >> n >> m;

int\*\* myArr = new int\* [n];

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

myArr[i] = new int[m];

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

cout << "Vvedite a[" << i << "][" << j << "]:";

cin >> myArr[i][j];

}

}

sort(myArr, myArr + n, [m](int\* a, int\* b) { return compareRows(a, b, m); });

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

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

cout << myArr[i][j] << " ";

}

cout << endl;

}

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

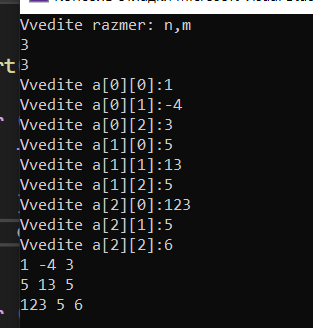
delete[] myArr[i];

}

delete[] myArr;

return 0;

}



Дана вещественная матрица размером NхM. Упорядочить ее строки по возрастанию наибольших элементов.

#include <iostream>

#include <algorithm>

using namespace std;

bool compareRows(int\* a, int\* b, int m) {

int maxA = a[0], maxB = b[0];

for (int i = 1; i < m; i++) {

maxA = max(maxA, a[i]);

maxB = max(maxB, b[i]);

}

return maxA < maxB;

}

int main() {

int n, m;

cout << "Vvedite razmer: n,m" << endl;

cin >> n >> m;

int\*\* matrix = new int\* [n];

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

matrix[i] = new int[m];

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

cout << "Vvedite a[" << i << "][" << j << "]:";

cin >> matrix[i][j];

}

}

sort(matrix, matrix + n, [m](int\* a, int\* b) { return compareRows(a, b, m); });

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

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

cout << matrix[i][j] << " ";

}

cout << endl;

}

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

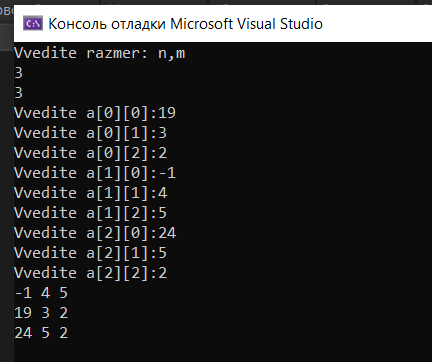
delete[] matrix[i];

}

delete[] matrix;

return 0;

}



Определить, является ли заданная целочисленная квадратная матрица по-рядка N симметричной относительно главной диагонали.

#include <iostream>

#include <vector>

using namespace std;

bool isSymmetric(vector<vector<int>> matrix, int n) {

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

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

if (matrix[i][j] != matrix[j][i]) {

return false;

}

}

}

return true;

}

int main() {

int n;

cout << "Vvedite razmer: n,n" << endl;

cin >> n;

vector<vector<int>> myArr(n, vector<int>(n));

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

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

cout << "Vvedite myArr[" << i << "][" << j << "]:";

cin >> myArr[i][j];

}

}

if (isSymmetric(myArr, n)) {

cout << "Matrix is symmetric" << endl;

}

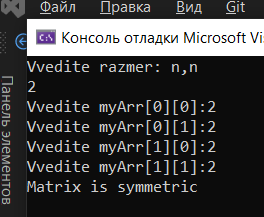
else {

cout << "Matrix is not symmetric" << endl;

}

return 0;

}



Среди тех строк целочисленной матрицы, которые содержат только нечетные элементы, найти строку с максимальной суммой модулей элементов.

#include <iostream>

#include <cmath>

#include <vector>

using namespace std;

int main() {

int n, m;

cout << "Vvedite razmer: n,m" << endl;

cin >> n >> m;

vector<vector<int>> matrix(n, vector<int>(m));

int max\_sum = 0;

int max\_sum\_index = 0;

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

int sum = 0;

bool only\_odd = true;

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

cout << "Vvedite myArr[" << i << "][" << j << "]:";

cin >> matrix[i][j];

if (matrix[i][j] % 2 == 0) {

only\_odd = false;

break;

}

sum += abs(matrix[i][j]);

}

if (only\_odd && sum > max\_sum) {

max\_sum = sum;

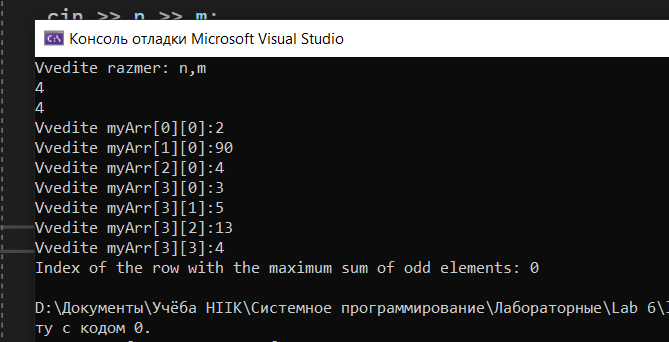
max\_sum\_index = i;

}

}

cout << "Index of the row with the maximum sum of odd elements: " << max\_sum\_index << endl;

return 0;

}

Среди тех столбцов целочисленной матрицы, которые содержат только такие элементы, значения которых по модулю не превышают 10, найти столбец с минимальным произведением элементов.

#include <iostream>

#include <climits>

using namespace std;

const int N = 100;

int myArr[N][N];

int main() {

int m, n;

cout << "Vvedite razmer: n,m" << endl;

cin >> m >> n;

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

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

cout << "Vvedite myArr[" << i << "][" << j << "]:";

cin >> myArr[i][j];

}

int min\_col = INT\_MAX;

int min\_product = INT\_MAX;

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

int product = 1;

bool flag = true;

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

if (abs(myArr[i][j]) > 10) {

flag = false;

break;

}

product \*= myArr[i][j];

}

if (flag) {

if (product < min\_product) {

min\_product = product;

min\_col = j;

}

}

}

if (min\_col == INT\_MAX) {

cout << "No such column found" << endl;

}

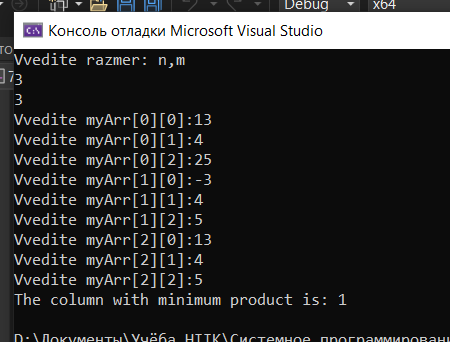
else {

cout << "The column with minimum product is: " << min\_col << endl;

}

return 0;

}



Даны целые числа А1,...,А10, целочисленная квадратная матрица порядка N. Заменить нулями в матрице те элементы, для которых имеются равные числа среди А1,...,А10.

#include <iostream>

using namespace std;

int main()

{

const int N = 20;

int myArr[N][N] = { { 0 } };

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

{

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

{

myArr[row][col] = rand() % 9 + 1;

cout << myArr[row][col] << " ";

}

cout << endl;

}

cout << endl;

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

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

if (((row + col) % 2) == 0)

{

int r\_fwd = 0, c\_fwd = 0;

while (r\_fwd < N)

{

if (c\_fwd >= N) { c\_fwd = 0; r\_fwd++; }

if (myArr[r\_fwd][c\_fwd] == myArr[row][col] && r\_fwd != row && c\_fwd != col)

myArr[r\_fwd][c\_fwd] = 0;

c\_fwd++;

}

}

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

{

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

cout << myArr[row][col] << " ";

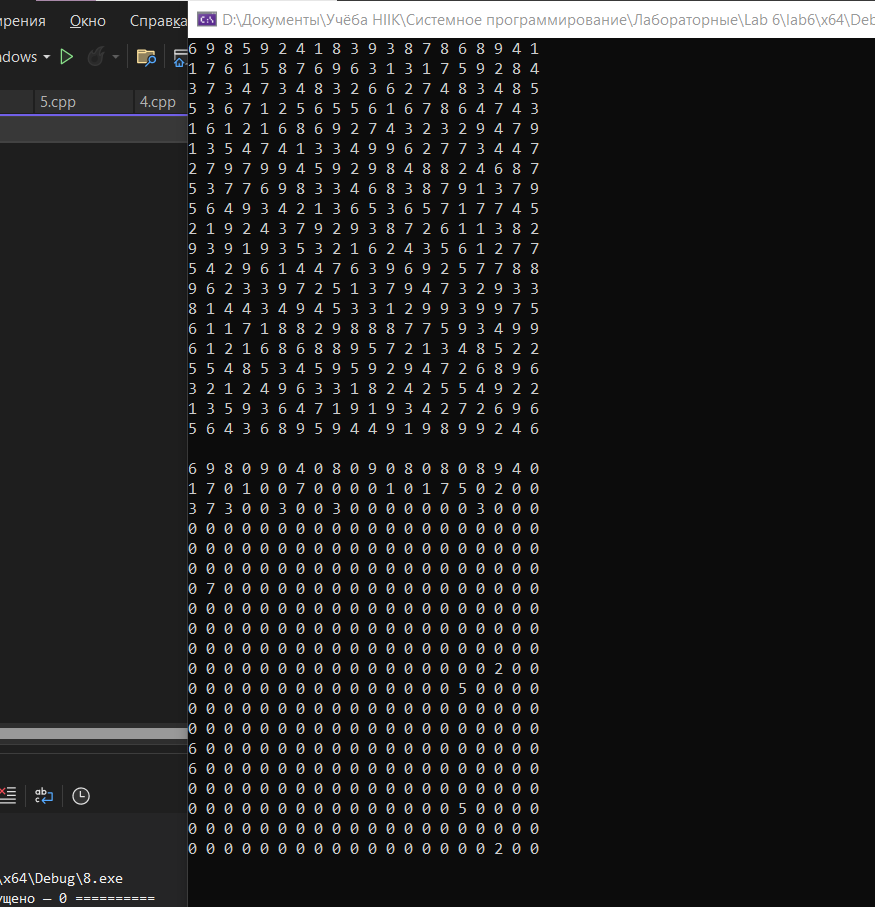
cout << endl;

}

cout << endl << endl;

cin.get();

}



В двумерном массиве целых чисел поменять местами строки, симметричные относительно середины массива (горизонтальной линии).

#include <iostream>

#include <algorithm>

using namespace std;

const int N = 10;

void swap(int a[], int b[], int size) {

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

swap(a[i], b[i]);

}

}

int main() {

int myArr[N][N];

int n, m;

cout << "Vvedite razmer: n,m" << endl;

cin >> n >> m;

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

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

cout << "Vvedite myArr[" << i << "][" << j << "]:";

cin >> myArr[i][j];

}

}

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

swap(myArr[i], myArr[n - i - 1], m);

}

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

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

cout << myArr[i][j] << " ";

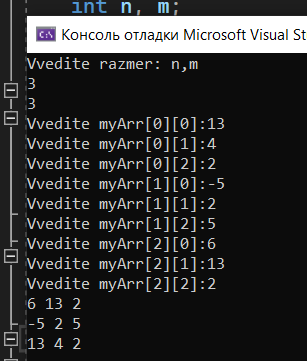
}

cout << endl;

}

return 0;

}



В двумерном массиве целых чисел поменять местами столбцы, симмет-ричные относительно середины массива (вертикальной линии).

#include<iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

int main()

{

int myArr[100][100], i, j, t, n, m;

cout << "n= "; cin >> n;

cout << "m= "; cin >> m;

cout << "Matr:" << endl;

srand(time(NULL));

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

{

for (j = 0; j < m; j++)

{

myArr[i][j] = rand() % 10;

cout << myArr[i][j] << " ";

}

cout << endl;

}

for (j = 0; j < m / 2; j++)

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

{

t = myArr[i][j]; myArr[i][j] = myArr[i][m - 1 - j]; myArr[i][m - 1 - j] = t;

}

cout << endl << "New matr:" << endl;

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

{

for (j = 0; j < m; j++)

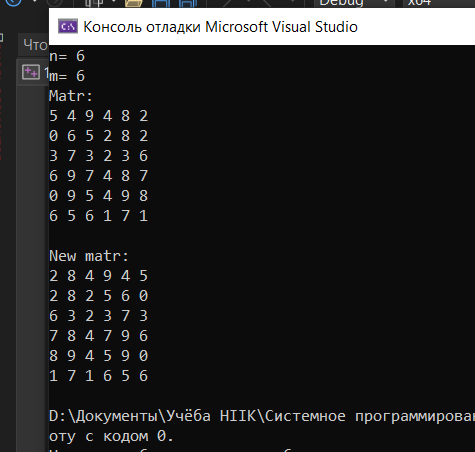
cout << myArr[i][j] << " ";

cout << endl;

}

return 0;

}



Даны две действительные квадратные матрицы порядка N. Получить новую матрицу прибавлением к элементам каждого столбца первой матрицы минимального элемента соответствующего столбца второй матрицы.

#include <iostream>

#include <cmath>

using namespace std;

const int N = 5;

void add\_min\_col(double a[][N], double b[][N], double c[][N]) {

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

double min\_val = b[0][j];

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

if (b[i][j] < min\_val) {

min\_val = b[i][j];

}

}

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

c[i][j] = a[i][j] + min\_val;

}

}

}

int main() {

double a[N][N] = { {1, 2, 3, 4, 5},

{6, 7, 8, 9, 10},

{11, 12, 13, 14, 15},

{16, 17, 18, 19, 20},

{21, 22, 23, 24, 25} };

double b[N][N] = { {25, 24, 23, 22, 21},

{20, 19, 18, 17, 16},

{15, 14, 13, 12, 11},

{10, 9, 8, 7, 6},

{5, 4, 3, 2, 1} };

double c[N][N];

add\_min\_col(a, b, c);

cout << "Result matrix:" << endl;

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

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

cout << c[i][j] << " ";

}

cout << endl;

}

return 0;

}

