

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО «МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| **Институт**  **информационных**  **технологий** | **Кафедра**  **информационных систем** |

**Основная образовательная программа 09.03.02  
«Информационные системы и технологии»**

**Отчет по дисциплине «Структурное программирование»**

**по лабораторной работе № 2**

**по теме: «Программирование с использованием функций»**

|  |  |
| --- | --- |
| Студент  группы ИДБ-22-06 | Мустафаева П.М. |
|  |  |
| Преподаватель | Репин С.В. |

**ЗАДАНИЕ**

**Описание**

Все массивы – динамические. Размерность массивов n строк и m столбцов, если в задании не указано, что дана квадратная матрица (в этом случае размерность n строк и n столбцов) или что-то иное. В задании необходимо использовать функции (не менее двух, кроме главной). Название функции должно быть с большой буквы и должно отображать то, что данная функция выполняет: например, функция CreateArray создает новый массив. Нельзя использовать глобальные переменные, кроме размерности массивов. Код всех программ оформляется в определенном стиле (объясняется устно на занятии).

**ЗАДАНИЕ 1(ВАРИАНТ 12)**

**Описание**

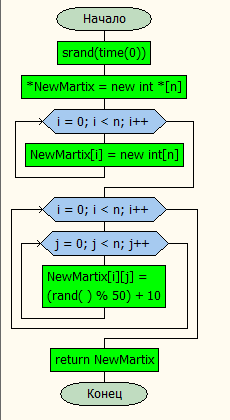
Если максимальный элемент квадратной матрицы находится выше главной диагонали, транспонировать матрицу, иначе определить сумму элементов строки и столбца с номерами, равными индексам максимального элемента.

**Использованные переменные**

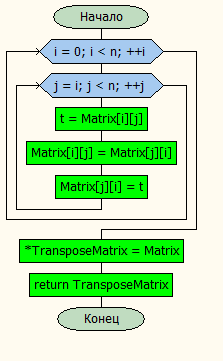
|  |  |  |  |
| --- | --- | --- | --- |
| **Имя переменной** | **Класс** | **Тип** | **Смысл** |
| n | Локальная | Целочисленный (int) | Количество строк и столбцов матрицы |
| i | Локальная | Целочисленный (int) | Итерации цикла |
| j | Локальная | Целочисленный (int) | Итерации цикла |
| NewMatrix | Локальная | Указатель второго порядка (int\*\*) | Двумерный динамический массив |
| Matrix | Локальная | Указатель второго порядка (int) | Двумерный динамический массив |
| sum | Локальная | Целочисленный (int) | Сумма элементов с номерам равными индексу макс. элемента |
| max | Локальная | Целочисленный (int) | Максимальный элемент |
| indexI | Локальная | Целочисленный (int) | Номер строк с макс. элементом |
| indexJ | Локальная | Целочисленный (int) | Номер столбца с макс. элементом |

**Блок-схема**

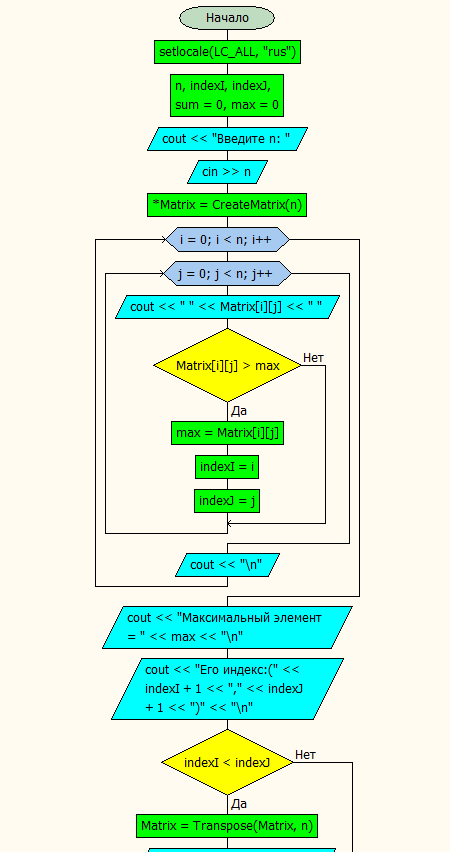
**CreateMatrix**

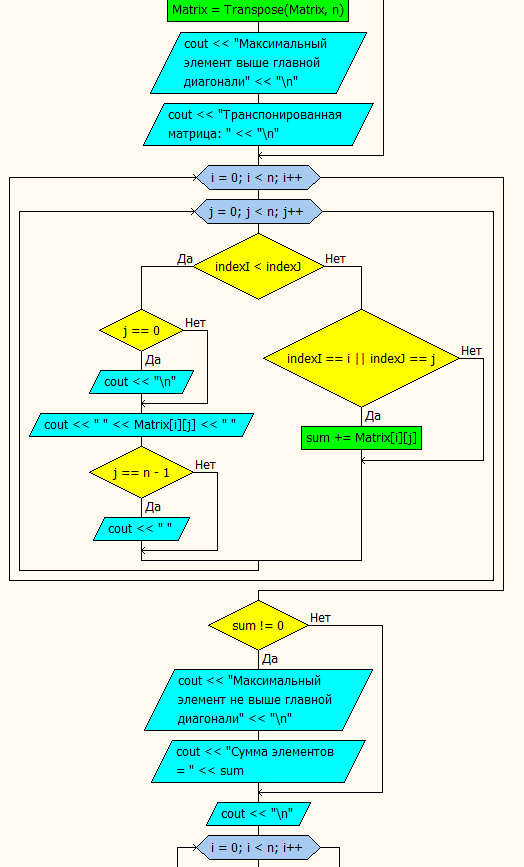
****

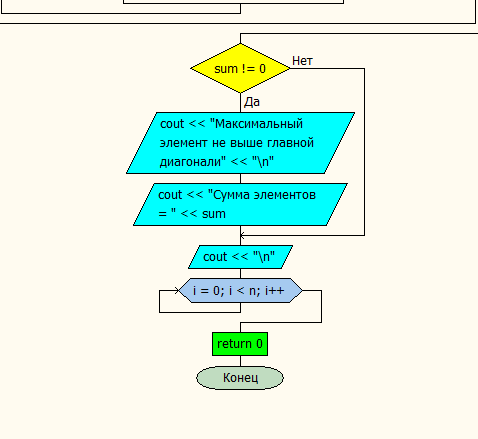
**Transpose**

****

**Main**

****

****

****

**Программа**

#include <iostream>

#include<ctime>

#include<cstdlib>

using namespace std;

int\*\* CreateMatrix(int n)

{

srand(time(0));

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

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

NewMartix[i] = new int[n];

}

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

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

NewMartix[i][j] = (rand() % 50) + 10;

}

}

return NewMartix;

}

int\*\* Transpose(int\*\* Matrix, int n)

{

int t;

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

{

for (int j = i; j < n; ++j)

{

t = Matrix[i][j];

Matrix[i][j] = Matrix[j][i];

Matrix[j][i] = t;

}

}

int\*\* TransposeMatrix = Matrix;

return TransposeMatrix;

}

int main()

{

setlocale(LC\_ALL, "rus");

int n, indexI, indexJ, sum = 0, max = 0;

cout << "Введите n: ";

cin >> n;

int\*\* Matrix = CreateMatrix(n);

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

{

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

{

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

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

{

max = Matrix[i][j];

indexI = i;

indexJ = j;

}

}

cout << "\n";

}

cout << "Максимальный элемент = " << max << "\n";

cout << "Его индекс:(" << indexI + 1 << "," << indexJ + 1<< ")" << "\n";

if (indexI < indexJ)

{

Matrix = Transpose(Matrix, n);

cout << "Максимальный элемент выше главной диагонали" << "\n";

cout << "Транспонированная матрица: " << "\n";

}

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

{

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

{

if (indexI < indexJ)

{

if (j == 0)

{

cout << "\n";

}

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

if (j == n - 1)

{

cout << " ";

}

}

else

{

if (indexI == i || indexJ == j)

{

sum += Matrix[i][j];

}

}

}

}

if (sum != 0)

{

cout << "Максимальный элемент не выше главной диагонали" << "\n";

cout << "Сумма элементов = " << sum;

}

cout << "\n";

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

{

delete[] Matrix[i];

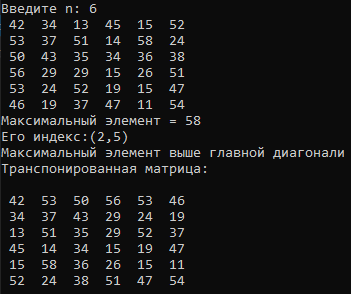
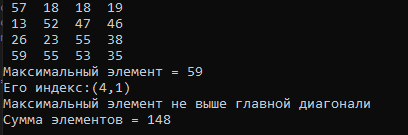
}

delete[] Matrix;

return 0;

}

**Примеры работы программы**



**ЗАДАНИЕ 2(ВАРИАНТ 3)**

**Описание**

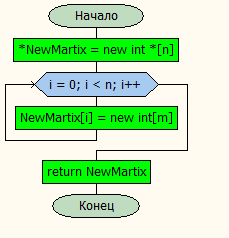
Найти максимальный среди отрицательных элементов прямоугольной матрицы и минимальный среди положительных. Если они отличаются по модулю меньше, чем на заданную величину, заменить все отрицательные элементы матрицы их модулями.

**Использованные переменные**

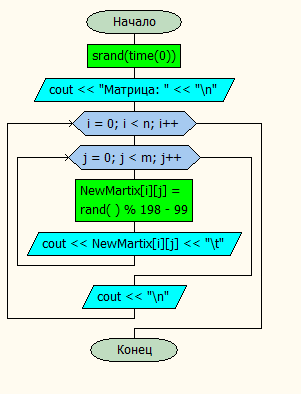
|  |  |  |  |
| --- | --- | --- | --- |
| n | Локальная | Целочисленный(int) | Количество строк массива |
| m | Локальная | Целочисленный(int) | Количество столбцов массива |
| i | Локальная | Целочисленный(int) | Итерации цикла |
| j | Локальная | Целочисленный(int) | Итерации цикла |
| NewMatrix | Локальная | Указатель второго порядка(int\*\*) | Двумерный динамический массив |
| Matrix | Локальная | Указатель второго порядка(int\*\*) | Двумерный динамический массив |
| val | Локальная | Целочисленный(int) | Заданное значение для сравнения результатов |
| min | Локальная | Целочисленный(int) | Минимальный положительный элемент |
| max | Локальная | Целочисленный(int) | Максимальный отрицательный элемент |
| razn | Локальная | Целочисленный(int) | Разница значений(по модулю) |

**Блок-схема**

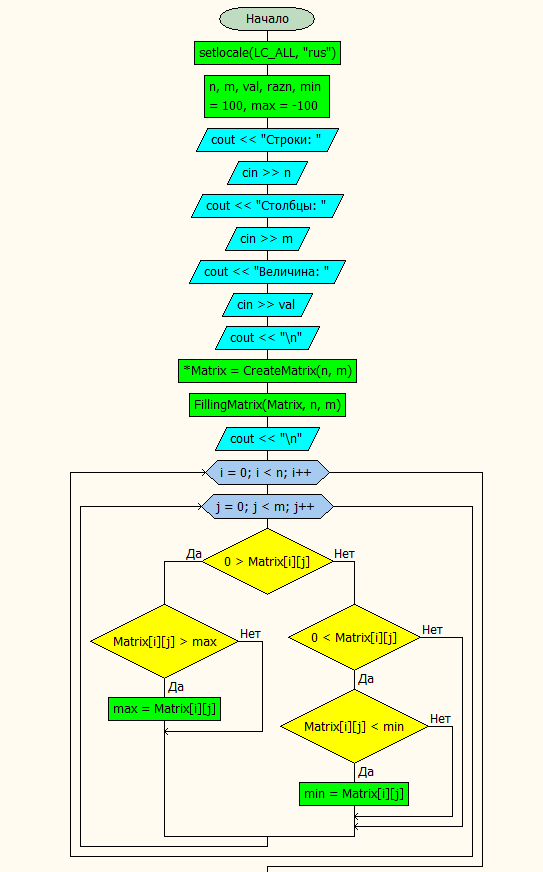
**CreateMatrix**

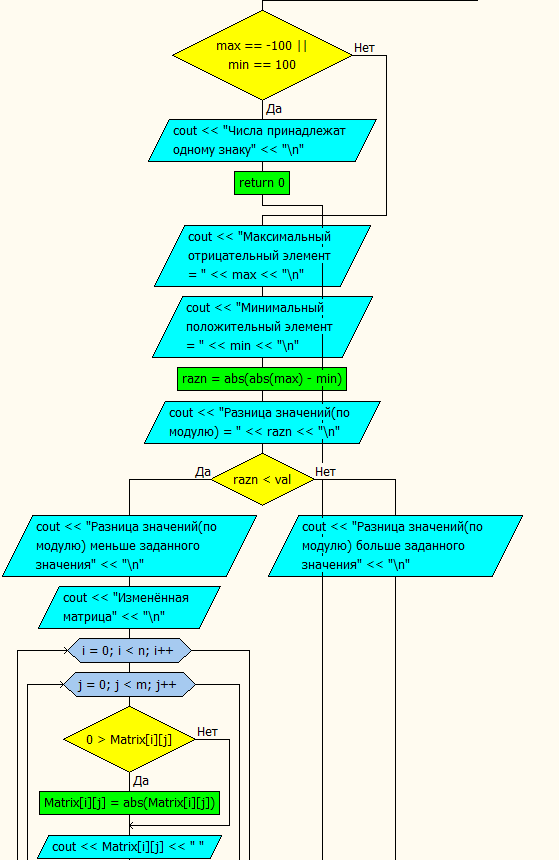
****

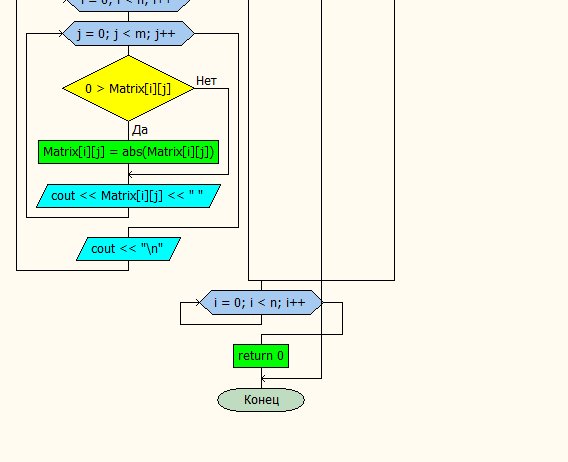
**FillingMatrix**

****

**Main**

****

****

****

**Программа**

#include<iostream>

#include<ctime>

#include<cstdlib>

#include<cmath>

using namespace std;

int\*\* CreateMatrix(int n, int m)

{

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

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

{

NewMartix[i] = new int[m];

}

return NewMartix;

}

void FillingMatrix(int\*\* NewMartix, int n, int m)

{

srand(time(0));

cout << "Mатрица: " << "\n";

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

{

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

{

NewMartix[i][j] = rand() % 198 - 99;

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

}

cout << "\n";

}

}

int main()

{

setlocale(LC\_ALL, "rus");

int n, m, val, razn, min = 100, max = -100;

cout << "Строки: ";

cin >> n;

cout << "Столбцы: ";

cin >> m;

cout << "Величина: ";

cin >> val;

cout << "\n";

int\*\* Matrix = CreateMatrix(n, m);

FillingMatrix(Matrix, n, m);

cout << "\n";

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

{

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

{

if (0 > Matrix[i][j])

{

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

{

max = Matrix[i][j];

}

}

else if (0 < Matrix[i][j])

{

if (Matrix[i][j] < min)

{

min = Matrix[i][j];

}

}

}

}

if (max == -100 || min == 100)

{

cout << "Числа принадлежат одному знаку" << "\n";

return 0;

}

cout << "Максимальный отрицательный элемент = " << max << "\n";

cout << "Минимальный положительный элемент = " << min << "\n";

razn = abs(abs(max) - min);

cout << "Разница значений(по модулю) = " << razn << "\n";

if (razn < val)

{

cout << "Разница значений(по модулю) меньше заданного значения" << "\n";

cout << "Изменённая матрица" << "\n";

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

{

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

{

if (0 > Matrix[i][j])

{

Matrix[i][j] = abs(Matrix[i][j]);

}

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

}

cout << "\n";

}

}

else {

cout << "Разница значений(по модулю) больше заданного значения" << "\n";

}

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

{

delete[] Matrix[i];

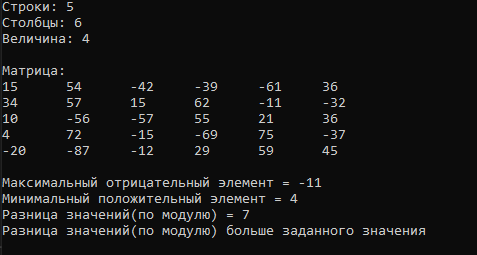
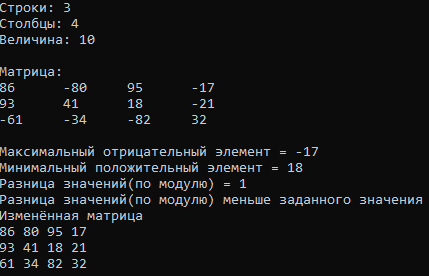
}

delete[] Matrix;

return 0;

}

**Примеры работы программы**

****

**Вывод**

**Вывод**

В данной лабораторной работе были написаны программы с использованием функций и двумерных динамических массивов.