

## ЛАБОРАТОРНАЯ РАБОТА 3

### Двумерный массив (матрица)

**Цель.** Освоение основ технологии объектно-ориентированного программирования на примере создания класса «матрица» для описания двумерного массива.

#### 1. Общие указания

По определению двумерный массив (как и одномерный) – это **последовательность элементов** памяти (то есть элементы имеют соседние адреса) **одного типа**.

Таким образом, двумерный массив определяется четверкой параметров:

- начальный адрес (логическое имя массива);
- тип элемента;
- количество элементов данного типа в строке;
- количество элементов данного типа в столбце.

Рисунок иллюстрирует двухмерный массив *a*. Массив содержит три строки и четыре столбца, так что, как говорят, - это массив три на четыре.

	Столбец 0	Столбец 1	Столбец 2	Столбец 3
Строка 0	<i>a</i> [0] [0]	<i>a</i> [0] [1]	<i>a</i> [0] [2]	<i>a</i> [0] [3]
Строка 1	<i>a</i> [1] [0]	<i>a</i> [1] [1]	<i>a</i> [1] [2]	<i>a</i> [1] [3]
Строка 2	<i>a</i> [2] [0]	<i>a</i> [2] [1]	<i>a</i> [2] [2]	<i>a</i> [2] [3]

Диаграмма с указателями:

- Имя массива: указывает на *a* в *a* [2] [0]
- Индекс строки: указывает на [2] в *a* [2] [0]
- Индекс столбца: указывает на [0] в *a* [2] [0]

Вообще, массивы с *m* строками и *n* столбцами называют массивами *m* на *n*.

Каждый элемент в массиве *a* определяется именем элемента в форме *a*[*di*]/[*dj*]; *a* - это имя массива, *di* и *dj* - индексы, которые однозначно определяют каждый элемент в *a*.

**Память класса** имеет вид

```
[private: ]
<тип> *<имя>;
int dm_l; // число строк
int dm_c; // число столбцов
```

**Методы класса**

1. Обязательными есть конструктор[ы], деструктор;
2. Необходимыми есть ввод, вывод элементов массива;
3. 3 функции обработки, в соответствии с вариантом задания.

#### 2. Пример реализации 1

**ЗАДАНИЕ 1.** Создать класс «матрица» для описания двумерного массива и реализовать следующие операции:

- вычислить сумму его элементов;
- заменить отрицательные элементы нулевыми значениями.

```
// Файл "matr.hpp" - класс matrix

#include<stdio.h>
class matrix {
    int *m; // адрес матрицы
    int dm_l; // количество строк - 1-я размерность
    int dm_c; // количество столбцов - 2-я размерность
public: //функции
    matrix(int,int); // конструктор

    // получить размерности
    int dmen_l()
        { return dm_l; }
    int dmen_c()
        { return dm_c; }

    // ВВОД
    void in_val_a(FILE *fin);
    void in_val();

    //ВЫВОД
    void display(FILE *fout);
    void screen();

    // сумма
    int sum();

    // замена отрицательных элементов нулями
    void edit();
}; // конец описания класса
```

```
// Файл "matr.cpp" - реализация интерфейса класса matrix

#include "matr.hpp"
matrix::matrix(int d_l, int d_c) {
    if ( (d_l<=0) || (d_c<=0) )
        printf("\n Size error:");
    else {
        dm_l = d_l;
        dm_c = d_c;
        m = new int[d_l*d_c];
    };
}

void matrix:: in_val_a(FILE *fin) {
    int i,j;
    for (i=0; i<dm_l; i++)
        for (j=0; j<dm_c; j++) {
            fscanf(fin,"%d",m+i*dm_c+j);
        }
}

void matrix:: in_val(){
    int i, j;
    for (i=0; i<dm_l; i++)
        for (j=0; j<dm_c; j++) {
            printf("\n Input value:");
            scanf("%d", (m+i*dm_c+j));
        }
}
```

```
int matrix::sum() {
    int i, j, s=0;
    for (i=0; i<dm_l; i++)
        for (j=0; j<dm_c; j++) {
            s=s+(m+i*dm_c+j);
        }
    return s;
}

void matrix::edit() {
    int i,j;
    for (i=0; i<dm_l; i++)
        for (j=0; j<dm_c; j++) {
            if (*(m+i*dm_c+j)<0)
                *(m+i*dm_c+j)=0;
        }
}

void matrix:: display(FILE *fout){
    int i,j;
    for (i=0; i<dm_l; i++) {
        fprintf(fout, "\n");
        for (j=0; j<dm_c; j++)
            fprintf(fout, " %d ", *(m+i*dm_c+j));
    }
}

void matrix::screen() {
    int i, j;
    for (i=0; i<dm_l; i++) {
        printf("\n");
        for (j=0; j<dm_c; j++)
            printf(" %d ", *(m+i*dm_c+j));
    }
}
```

```
// Главная программа - файл "Matrix_m.cpp"

#include <stdio.h>
#include "matr.cpp"
void main() {
    FILE *ff, *fin;
    int i,n;
    int sx;

    ff = fopen("m_rez.dat", "w+");
    fin = fopen("m_init.dat", "r");

    matrix x(4,4);
    printf("\n Dimension: %d %d",x.dmen_l(),x.dmen_c());
    x.in_val_a(fin);
    x.display(ff);

    sx = x.sum();
    fprintf(ff, "\n sum=%d", sx);

    x.edit();
    printf("\n Dimension: %d %d",x.dmen_l(),x.dmen_c());
    x.display(ff);

    sx = x.sum();
    fprintf(ff, "\n New sum=%d", sx);
```

```

    fclose(ff);
    fclose(fin);
}

```

Data-file "M-init.dat"	Result file "M_res.dat"
-1 -2 -3 -4	0 0 0 0
-2 -3 3 4	0 0 3 4
-5 -3 -4 -5	0 0 0 0
-6 -4 -5 -6 Sum=-46	0 0 0 0 Sum=7

### 3. Пример реализации №2

Класс *Matrix* может быть реализован и по-другому:

```

#include <iostream>
#include <cstdlib>
#include <ctime>
#include <iomanip>

class QMatrix {
private:
    int N;
    int M;
    double** matrix;
public:
    QMatrix(); // конструктор по умолчанию
    QMatrix(int n):Matrix(n,n) // конструктор с параметрами
    QMatrix(const Matrix& ob) // конструктор копии
    ~QMatrix() // деструктор
    void Enter()
    void Show()
    QMatrix operator + (const Matrix&)
    QMatrix operator = (const Matrix&)
};

QMatrix::QMatrix() // реализация конструктора по умолчанию
{
    N = 0;
    M = 0;
    matrix = new double*[N];
    for (int i = 0; i < N; ++i)
        matrix[i] = new double[M];
}

QMatrix::QMatrix(int n1) // реализация конструктора с параметрами
{
    N = n1;
    M = n1;
    a = new double* [M];
    for (int i=0; i<M; i++)
        a[i] = new double [N];
    for (int i=0; i<M; i++)
        for (int j=0; j<N; j++)
            a[i][j] = 0.0;
}

QMatrix::QMatrix (const Matrix &ob) // реализация конструктора копии
{
    N = ob.N;
    M = ob.M;
    matrix = new double*[N];
    for (int i = 0; i < N; ++i)
        matrix[i] = new double[M];
}

```

```
    for ( int i = 0; i < N; ++i)
        for ( int j = 0; j < M; ++j)
            matrix[i][j] = ob.matrix[i][j];
}

QMatrix::~QMatrix() // деструктор
{
    for ( int i = 0; i < N; ++i)
        delete[] matrix[i];
    delete[] matrix;
}

void QMatrix::Enter() // ввод матрицы
{
    int i, j;
    for ( i = 0; i < N; ++i)
        for ( j = 0; j < M; ++j)
            matrix[i][j] = 0.01 * (rand() % 1001);
}

void Matrix::Show() // вывод матрицы
{
    int i, j;
    for ( i = 0; i < N; ++i)
        for ( j = 0; j < M; ++j)
        {
            if( j % M == 0)
                std::cout << std::endl;
            std::cout << std::setw(10) << matrix[i][j];
        }
    std::cout << std::endl;
}

QMatrix QMatrix :: operator + ( const Matrix& ob) // сумма матриц
{
    QMatrix temp; // реализовано с помощью временного объекта
    temp.N = this->N;
    temp.M = this->M;
    int i, j;
    temp.matrix = new double*[temp.N];
    for ( i = 0; i < temp.N; ++i)
        temp.matrix[i] = new double[temp.M];
    for ( i = 0; i < temp.N; ++i)
        for ( j = 0; j < temp.M; ++j)
            temp.matrix[i][j] = this->matrix[i][j] + ob.matrix[i][j];
    return temp;
}

int main()
{
    srand((unsigned) time(0));

    QMatrix m1(3, 3);
    m1.Enter();
    m1.Show();

    QMatrix m2(3, 3);
    m2.Enter();
    m2.Show();

    QMatrix m3(3,3);
    m3 = m1 + m2;
    m3.Show();
}
```

## 4. Варианты задания

Создать класс «матрица» для описания двумерного массива и произвести с ним следующие операции, согласно варианту:

№	Задание
1.	<p>Задана квадратная матрица. Переставить в обратном порядке элементы тех столбцов матрицы, которые расположены ниже ее главной диагонали.</p> <p>В каждой строке матрицы отрицательным элементам присвоить нулевое значение, после чего перенести все положительные элементы в начало строки в порядке их исходного относительного расположения.</p>
2.	<p>Каждую элемент строки прямоугольной матрицы заменить суммой элементов этого столбца, за вычетом текущего элемента.</p> <p>Определить значение и местоположение максимального элемента матрицы до и после ее преобразования.</p> <p>Определить значение и местоположение минимального элемента матрицы до и после ее преобразования.</p>
3.	<p>Для каждого столбца прямоугольной целочисленной матрицы подсчитать сумму входящих в него элементов и определить, имеются ли столбцы с одинаковой суммой. Подсчитать количество пар таких столбцов.</p> <p>каждый нулевой элемент заменить средним арифметическим значением ненулевых элементов той строки, в которой расположен этот элемент.</p>
4.	<p>Выполнить текущее сглаживание каждой строки прямоугольной матрицы и определить максимальное отклонение ее элементов от среднего арифметического значения данной строки до и после сглаживания.</p> <p>Примечание. При текущем сглаживании массива <math>x_1, x_2, \dots, x_n</math> каждый <math>j</math>-ый элемент массива (<math>j=2, 3, \dots, n-1</math>) заменяется средним арифметическим значением элементов с индексами <math>j-1, j, j+1</math>.</p>
5.	<p>В каждой строке прямоугольной матрицы перенести максимальный элемент в последнюю позицию строки, сдвинув при этом влево расположенные после него элементы. Учесть частный случай, когда максимальный элемент уже находится в последней позиции строки.</p> <p>Пример. Строка 5 18 21 12 10 24 13 17 8 10 после преобразования будет иметь вид 5 18 21 12 10 13 17 8 10 24 .</p> <p>В каждом столбце прямоугольной матрицы найти минимальный по модулю элемент и, если он не является диагональным, поменять его местами с диагональным элементом.</p> <p>Подсчитать количество таких перестановок.</p>
6.	<p>Дана квадратная матрица. Если в ее треугольной части, расположенной выше диагонали, имеются нулевые элементы, то заменить каждый из них минимальным, но отличным от нуля, значением элементов столбца, в котором расположен нулевой элемент.</p> <p>Для каждой строки матрицы определить сумму ее положительных элементов, а затем сгруппировать строки в порядке убывания этих сумм.</p>
7.	<p>В каждой строке прямоугольной матрицы подсчитать количество изолированных положительных элементов, т.е. элементов, окруженных слева и справа хотя бы одним неположительным элементом. Первый и последний элементы строки не учитывать.</p> <p>Определить номер строки, имеющей максимальное количество таких элементов.</p> <p>В каждом столбце матрицы найти минимальный элемент и определить номер столбца, имеющего максимальное по модулю значение такого элемента.</p>
8.	<p>В прямоугольной матрице определить значение и местоположение элемента, имеющего максимальное превышение по отношению к среднему значению соседних элементов. Элементы, расположенные на периметре матрицы, не анализировать.</p> <p>Указание. Для элемента с индексами <math>(i, j)</math>, <math>i=2..(m-1), j=2..(n-1)</math> соседними элементами считать элементы, смежные с ним по вертикали и по горизонтали.</p> <p>определить номер столбца, сумма элементов которого максимальна.</p>

№	Задание
	Элементы каждой строки прямоугольной матрицы заменить их дополнениями до максимального элемента этой же строки. Определить, насколько при этом изменится общая сумма элементов матрицы.
9.	<p>Определить, имеются ли в прямоугольной матрице линейно зависимые строки и подсчитать количество пар таких строк.</p> <p><i>Примечание. Две строки матрицы линейно зависимы, если одну из них можно получить из другой умножением на постоянный коэффициент.</i></p> <p>каждый нулевой элемент заменить средним арифметическим значением ненулевых элементов того столбца, в котором расположен этот элемент.</p>
10.	<p>В прямоугольной матрице найти два элемента, которые в наименьшей и в наибольшей степени отличаются от среднего арифметического значения элементов данной матрицы, после чего обменять их местами.</p> <p>определить количество строк, элементы которых полностью упорядочены по убыванию</p> <p>определить, имеются ли в прямоугольной матрице линейно зависимые строки</p>
11.	<p>Элементами квадратной матрицы могут быть только числа -1, 0 или 1. Для каждого из столбцов матрицы выполнить следующее: если сумма элементов столбца не равна нулю, то заменить часть нулевых элементов значением +1 или -1 таким образом, чтобы указанная сумма как можно меньше отличалась от нуля.</p> <p>проверить, имеет ли место совпадение k-ой строки и k-го столбца (<math>k=1..n</math>). Отпечатать номера совпадающих строк и столбцов.</p>
12.	<p>К каждому элементу главной диагонали квадратной матрицы прибавить такое значение, чтобы сумма элементов в соответствующем столбце матрицы была нулевой.</p> <p>Сформировать вектор-строку, i-ая компонента которого равна приращению значения соответствующего элемента главной диагонали,</p> <p>после чего сгруппировать столбцы матрицы в порядке возрастания компонент этого вектора.</p>
13.	<p>Среди диагоналей квадратной матрицы, параллельных главной диагонали и расположенных ниже нее, найти такую, сумма модулей элементов которой максимальна по сравнению с другими диагоналями.</p> <p>каждый нулевой элемент заменить средним арифметическим значением ненулевых элементов того столбца, в котором расположен этот элемент.</p> <p>определить количество строк, элементы которых полностью упорядочены по возрастанию</p>
14.	<p>В прямоугольной матрице рассмотреть квадратные подматрицы размерностью 1,2,3,..., причем для всех подматриц левым верхним элементом является элемент исходной матрицы с индексами (1,1). Определить номер подматрицы, среднее арифметическое элементов которой имеет наибольшее значение.</p> <p>найти номера столбцов, содержащих соответственно максимальное и минимальное количество отрицательных элементов, после чего обменять их местами. Учесть частные случаи (в матрице нет отрицательных элементов; лишь один столбец матрицы содержит такие элементы; лишь два столбца содержат отрицательные элементы, но их количество одинаково).</p> <p>определить значение и местоположение максимального по модулю и минимального по модулю четных элементов, после чего обменять их местами.</p>
15.	<p>Обнулить элементы k-ой строки и l-го столбца прямоугольной матрицы, после чего присвоить этим элементам такие значения, что бы сумма элементов каждой строки и сумма элементов каждого столбца, за исключением l-го, была равна нулю. Значения k и l ввести с клавиатуры.</p> <p>Для прямоугольной матрицы найти минимальный из положительных элементов и максимальный из отрицательных элементов, после чего обменять их местами. Нулевые элементы не учитывать.</p> <p>найти максимальный по модулю элемент и заменить его средним арифметическим значением остальных элементов данной строки.</p>
16.	<p>В прямоугольной матрице определить количество столбцов, полностью состоящих из положительных элементов.</p> <p><i>Указание. Последовательный просмотр элементов столбца организовать таким образом, чтобы при обнаружении первого неположительного элемента остальные элементы столбца не проверялись.</i></p>

№	Задание
	<p>В прямоугольной матрице часть элементов имеют нулевое значение. Заменить каждый такой элемент полусуммой смежных ему элементов.</p> <p>Примечание. Угловые элементы матрицы имеют два смежных элемента; элементы, расположенные на периметре матрицы, но не являющиеся угловыми, имеют три смежных элемента; остальные элементы матрицы имеют четыре смежных элемента.</p> <p>найти минимальный из положительных элементов и максимальный из отрицательных элементов.</p>
17.	<p>В каждой строке прямоугольной матрицы, элементами которой являются целые положительные числа, определить сумму элементов, являющихся удвоенными нечетными числами,</p> <p>после чего сгруппировать строки в порядке убывания этих сумм.</p> <p>Сгруппировать элементы каждой строки, расположенные выше главной диагонали, в порядке убывания их абсолютных значений.</p>
18.	<p>В каждой строке прямоугольной матрицы определить разность <math>d</math> между средним арифметическим значением <math>S1</math> элементов, стоящих на четных местах, и средним арифметическим значением <math>S2</math> элементов, стоящих на нечетных местах. При этом учесть, что количество столбцов может быть как четное, так и нечетное.</p> <p>Сгруппировать столбцы в порядке убывания параметра <math>d</math>.</p> <p>найти минимальный из положительных элементов и максимальный из отрицательных элементов.</p>
19.	<p>Для каждой строки прямоугольной матрицы определить количество нарушений <math>k</math> условия упорядоченности ее элементов по возрастанию, т.е. условия <math>a[i,j] \leq a[i,j+1]</math>.</p> <p>Сгруппировать строки в порядке возрастания параметра <math>k</math>.</p> <p>В каждой строке прямоугольной матрицы удалить максимальный элемент, сдвинув на одну позицию влево расположенные после него элементы данной строки. Последнему элементу строки присвоить нулевое значение.</p>
20.	<p>В каждой строке прямоугольной целочисленной матрицы определить процент четных положительных чисел по отношению к общему количеству положительных чисел в данной строке,</p> <p>после чего сгруппировать строки в порядке уменьшения указанного процента.</p> <p>В каждом столбце прямоугольной матрицы поменять местами минимальный элемент с элементом главной диагонали, если элемент главной диагонали, относящийся к данному столбцу, имеет отрицательное значение.</p>
21.	<p>Рассматривая в квадратной матрице диагональ, соединяющую левый нижний элемент с правым верхним элементом, определить минимальное по модулю число среди элементов, расположенных выше данной диагонали, и максимальное по модулю число среди элементов, расположенных ниже этой же диагонали,</p> <p>после чего обменять местами соответствующие этим числам элементы матрицы</p> <p>В каждой строке квадратной матрицы расположить элементы в порядке убывания их абсолютных значений, не затрагивая при этом положения элементов главной диагонали.</p>
22.	<p>Для прямоугольной матрицы <math>A (m \times n)</math> сформировать одномерный массив <math>B</math>, <math>i</math>-му элементу которого присвоить значение 1, если в <math>i</math>-ой строке имеются по крайней мере три элемента, упорядоченных по убыванию, и значение 0 в противном случае. Переместить в начальную часть матрицы строки, для которых определено значение <math>b[i] = 1</math>, сохранив при этом относительное расположение этих строк.</p> <p>Рассматривая каждый столбец прямоугольной вещественной матрицы как вектор, определить, имеются ли в матрице ортогональные вектор-столбцы. Подсчитать количество таких пар векторов. Примечание. Для ортогональных векторов их скалярное произведение <math>S</math> равно нулю. Значение <math>S</math> считать равным нулю, если <math>\text{abs}(S) &lt; \text{eps}</math>, где <math>\text{eps}</math> - малое число (например, 0.001) .</p> <p>Для каждой строки прямоугольной матрицы, кроме последней, определить количество <math>k</math> элементов, совпадающих по знаку с соответствующими элементами последней строки.</p>
23.	<p>В прямоугольной матрице определить значение и местоположение максимального по модулю и минимального по модулю четных элементов, после чего обменять их местами.</p> <p>В прямоугольной матрице определить количество столбцов, содержащих только числа одного знака (положительные или отрицательные) и не содержащих нулевых элементов.</p>



№	Задание
	Для каждой строки прямоугольной матрицы определить количество $k$ элементов, значения которых расположены в заданном интервале $[a, b]$ . Значения $a$ и $b$ ввести с клавиатуры.
24.	<p>Если в <math>j</math>-ом столбце квадратной матрицы (<math>j=1..n</math>) максимальный элемент находится на главной диагонали, то разделить все элементы данного столбца на значение максимального элемента; в противном случае столбец матрицы оставить без изменений. Подсчитать общее количество преобразованных столбцов матрицы.</p> <p>Для прямоугольной матрицы определить значение и местоположение элемента, являющегося седловой точкой матрицы (если такая имеется). Указание. Седловой точкой матрицы считать не расположенный на ее периметре элемент с индексом <math>(i, j)</math>, который является минимальным в <math>i</math>-ой строке и одновременно максимальным в <math>j</math>-ом столбце.</p>
	найти максимальный по модулю элемент и заменить его средним арифметическим значением остальных элементов данной строки.
25.	<p>Для каждого столбца прямоугольной матрицы, элементами которой являются целые положительные числа, определить сумму входящих в него элементов и, если она нечетная, добавить единицу к значению последнего элемента данного столбца,</p> <p>после чего сгруппировать элементы столбца в порядке убывания.</p> <p>найти номера векторов-строк, для которых модуль их скалярного произведения имеет максимальное значение.</p>

## 5. Требования к отчету

1. Обоснование выбора структуры памяти класса и его интерфейса.
2. Блок-схемы алгоритмов функций обработки.
3. Описание класса + схемы.
4. Результаты тестирования класса.
5. Заключение и рекомендации по совершенствованию класса и/или тестирующей программы.