МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ЭКОНОМИКИ, СТАТИСТИКИ И ИНФОРМАТИКИ

ИНСТИТУТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

Кафедра Математическое обеспечение и администрирование информационных систем

Дисциплина: «Информатика и программирование»

**ЛАБОРАТОРНАЯ РАБОТА № 2**

|  |  |
| --- | --- |
| Работу выполнил(и):  cтудент(ы) группы ДКО-102 | Оценки по БРС |
|  |  |
| *Вишняков Федор* |  |
|  |  |

Дата: 20.10.2012

Работу принял: Переверзев И.

фамилия преподавателя

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

Москва 2012

Оглавление

[Постановка задачи 3](#_Toc338681380)

[Структура программы на языке C++ 4](#_Toc338681381)

[Операторы и конструкции языка, используемые в программе (описание, синтаксис) 5](#_Toc338681382)

[Текст программы 6](#_Toc338681383)

[Результат работы программы 11](#_Toc338681384)

[Используемая литература и Internet ресурсы 15](#_Toc338681385)

# Постановка задачи

Дана прямоугольная ненулевая матрица A(m,n), состоящая из целых чисел. Матрицей B(q,q) назовем квадратную матрицу, у которой в главной диагонали нет нулевых элементов. Найти в исходной матрице A матрицу B наибольшего размера. Если таких матриц будет несколько, то выбрать ту, у которой элемент B(0,0) будет находится наиболее близко к А(0,0), то есть фактически матрица B будет лежать выше и левее остальных. Матрица B всегда существует, даже если ее размер равен 1.

С матрицей B выполнить следующие действия (выбрать любые **три** действия):

1. Повернуть матрицу B на 90 градусов по часовой стрелке вокруг ее центра, не используя лишней памяти (допускается использовать не более одной временной переменной для хранения целого числа).

2. Транспонировать матрицу B, не используя лишней памяти.

3. Обход матрицы. Создать вектор V(q\*q), в который записать элементы матрицы в указанном ниже порядке. Вектор вывести на экран.

4. Отсортировать методом простых обменов (сортировка пузырьком) [Д. Кнут, Том 3, 5.2.2.Обменная сортировка] элементы матрицы B по цепочке, слева направо и сверху вниз. После выполнения алгоритма сортировки, элемент с минимальным значением окажется в левом верхнем углу матрицы, а с максимальным – в правом нижнем. Сортировку производить непосредственно в матрице B, без использования лишних векторов или дополнительных матриц.

В программе реализовать меню, для того чтобы каждое из указанных выше заданий можно было выполнить неоднократно.

Предусмотреть возможность загрузки не менее трех наборов тестовых данных (исходных матриц A) заранее размещенных в коде программы (или внешнем файле) для облегчения тестирования программы.

# Структура программы на языке C++

Программа составляется из заголовочных файлов (.h) и файлов с исходным кодом (.cpp). Заголовочные файлы подключаются для возможности использования реализованных кем-либо функций, заданных констант, описанных типов и так далее. Заголовочные файлы обычно содержат прототипы функций и задание констант. Файлы с тем же названием, но расширением cpp содержат реализации функций, которые при создании превращаются компилятором в объектные файлы, позволяющие применять описанные в заголовке функции, но не дающие доступ к исходному коду реализаций.

Начинается программа обычно с препроцессорных команд. Программист сообщает компилятору, какие заголовочные файлы нужно подключить к программе, какие константы определить. Далее содержание кода варьируется, но неизменно одно – для того, чтобы программа запускалась с данного cpp-файла, нужно чтобы он содержал функцию main. Она является точкой входа в программу. Из неё вызываются другие функции. По завершении функции main программа также завершается.

При компиляции проверяется верность кода. Заголовочные файлы остаются прежними, а cpp-файлы (включая реализации заголовочных) превращаются в объектные файлы(.obj). Обычно разработчики библиотек предоставляют заголовочные файлы с полным описанием функционала и прилагающиеся к ним объектные файлы, содержащие реализации указынных функций, классов и так далее.

Функции

По мере усложнения ваших программ вам следует делить их на небольшие легко управляемые части, называемые функциями. Каждая функция должна иметь уникальное имя. Присваивайте вашим функциям имена, выражающие смысл задач, которые выполняют функции.

Функции могут возвращать значение вызвавшей функции. При этом вы должны указать тип возвращаемого функцией значения (int, char и т. д.) до имени функции, в противном случае вы должны предварять имя функции словом void.

Программы передают информацию в функции с помощью параметров. Если функция получает параметры, вы должны указать уникальное имя и тип каждого параметра. Если функция не получает параметры, вы должны поместить ключевое слово void внутри круглых скобок, следующих за именем функции.

C++ должен знать тип возвращаемого значения и количество и тип параметров, которые получает функция. Если определение функции следует за использованием функции, вы должны поместить прототип функции в начале исходного файла.

# Операторы и конструкции языка, используемые в программе (описание, синтаксис)

**#include <libname[.h]>**

Директива препроцессора, подключающая библиотеку или файл к программе

**#pragma once**

Избежание подключения заголовочного файла более одного раза

**using namespace**

Использование переменных и операторов пространства имени без указания этого имени

**setlocale**

Определение локализации для определенной категории символов

**int, double**

Типы числовых переменных (целых и вещественных)

**cout**

Оператор вывода библиотеки iostream

**cin**

Оператор считывания данных с клавиатуры

**endl**

Перенос строки для оператора cout

**setw**

Установка количества выводимых символов для переменной. Требуется библиотека iomanip

**while**

Цикл с предусловием

**for**

Итерационный цикл

**system**

Функция, выполняющая процессорную команду

**\***

Сообщает, что переменная является указателем на какой-либо тип. Используется для динамического управления памятью. Для использования переменной как таковой указатель должен ссылаться на какой-либо участок памяти, который программист должен выделить функцией **new**. После работы динамически выделенную память следует освободить, иначе произойдет утечка памяти. Эта память будет «висеть», ничем не используемая, но недоступная.

**delete**

Удаление памяти, выделенной операцией **new**.

# Текст программы

Программа состоит из трех файлов.

1. main.cpp

#include "matrixlib.h"

bool MemoryIsUsed = false;

int main() {

setlocale(LC\_ALL, "rus");

int \*\* MatrixA = NULL;

int Ma, Na;

int MatrixBa, MatrixBb, MatrixBdl;

char e;

while (1) {

cout << "--------------------------------\n0. Выход\n1. Ввести матрицу А\n";

cout << "2. Показать матрицу А\n3. Показать матрицу B\n";

cout << "4. Повернуть матрицу\n5. Транспонировать матрицу\n";

cout << "6. Обход матрицы\n7. Сортировка матрицы\n";

cout << "q. Загрузка тестовых данных 1\nw. Загрузка тестовых данных 2\n";

cout << "e. Загрузка тестовых данных 3\n--------------------------------\n";

e = \_getch();

switch (e) {

case '0':

if (MemoryIsUsed) {

Delete\_Matrix(MatrixA, Ma);

MemoryIsUsed = false;

}

cout << "Выходим...";

return 0;

case '1':

cout << "Ввод матрицы с клавиатуры" << endl;

if (MemoryIsUsed) {

Delete\_Matrix(MatrixA, Ma);

MemoryIsUsed = false;

}

Enter\_Matrix(MatrixA, Ma, Na);

Calculate\_B(MatrixA, Ma, Na, MatrixBa, MatrixBb, MatrixBdl);

MemoryIsUsed = true;

break;

case '2':

cout << "Матрица A\n";

if (!MemoryIsUsed)

cout << "Ошибка. Матрица ещё не загружена\n";

else

Show\_Matrix(MatrixA, 0, 0, Ma, Na);

break;

case '3':

cout << "Матрица B\n";

if (!MemoryIsUsed)

cout << "Ошибка. Матрица ещё не загружена\n";

else

Show\_Matrix(MatrixA, MatrixBa, MatrixBb, MatrixBdl, MatrixBdl);

break;

case '4':

cout << "Поворачиваем матрицу на 90 градусов...\n";

Turn\_Matrix(MatrixA, MatrixBa, MatrixBb, MatrixBdl);

cout << "Успех!\n";

break;

case '5':

cout << "Транспонируем матрицу...\n";

Transponate\_Matrix(MatrixA, MatrixBa, MatrixBb, MatrixBdl);

cout << "Успех!\n";

break;

case '6':

cout << "Вычисляем вектор матрицы...\n";

Go(MatrixA, MatrixBa, MatrixBb, MatrixBdl);

break;

case '7':

cout << "Сортируем матрицу...\n";

Bubble\_Sort(MatrixA, MatrixBa, MatrixBb, MatrixBdl);

cout << "Успех!\n";

break;

case 'q':

case 'w':

case 'e':

if (MemoryIsUsed) {

Delete\_Matrix(MatrixA, Ma);

MemoryIsUsed = false;

}

Load\_Matrix\_From\_File(MatrixA, Ma, Na, e);

Calculate\_B(MatrixA, Ma, Na, MatrixBa, MatrixBb, MatrixBdl);

MemoryIsUsed = true;

break;

default:

cout << "Неверный ввод" << endl;

}

}

}

2. matrixlib.h

#pragma once

#include <iostream>

#include <conio.h>

#include <fstream>

#include <iomanip>

using namespace std;

void Enter\_Matrix(int \*\* &a, int &m, int &n);

void Show\_Matrix(int \*\* m, int a, int b, int c, int d);

void Give\_Memory(int \*\* & a, int m, int n);

void Delete\_Matrix(int \*\* & a, int m);

void Load\_Matrix\_From\_File(int \*\* & a, int &m, int &n, char file);

void Transponate\_Matrix(int \*\*& m, int a, int b, int c);

void Turn\_Matrix(int \*\*& m, int a, int b, int c);

void Calculate\_B(int \*\*& a, int m, int n, int &ba, int &bb, int &bdl);

void Go(int \*\* m, int a, int b, int c);

void Bubble\_Sort(int \*\*& m, int a, int b, int c);

3. matrixlib.cpp

#include "matrixlib.h"

void Give\_Memory(int \*\* & a, int m, int n) {

a = new int\* [m];

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

a[i] = new int[n];

}

void Enter\_Matrix(int \*\* &a, int &m, int &n) {

int i, j;

cout << "Введите размеры матрицы> ";

cin >> m >> n;

Give\_Memory(a, m, n);

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

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

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

cin >> a[i][j];

}

}

void Show\_Matrix(int \*\* m, int a, int b, int c, int d) {

int i, j;

for (i = a; i < a + c; i++) {

for (j = b; j < b + d; j++)

cout << setw(4) << m[i][j];

cout << endl;

}

}

void Delete\_Matrix(int \*\* & a, int m) {

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

delete [] a[i];

delete [] a;

}

void Load\_Matrix\_From\_File(int \*\* & a, int &m, int &n, char file) {

char filename[6] = " .txt";

filename[0] = file;

ifstream input(filename);

if (!input) {

cout << "Файл не найден.\n";

return;

}

input >> m >> n;

Give\_Memory(a, m, n);

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

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

input >> a[i][j];

input.close();

cout << "Данные успешно загружены!\n";

}

void Transponate\_Matrix(int \*\*& m, int a, int b, int c) {

int i, j;

for (i = 0; i < c - 1 ; i++)

for (j = i + 1; j < c; j++)

swap(m[a + i][b + j], m[a + j][b + i]);

}

void Calculate\_B(int \*\*& a, int m, int n, int &ba, int &bb, int &bdl) {

int xStart = m - 1, yStart = 0, x, y, xmax, ymax, xmaxt = 0, ymaxt = 0, dl = 0, dlmax = 0;

while (xStart != 0 || yStart != n) {

for (x = xStart, y = yStart, dl = 0; x != m && y!= n; x++, y++) {

if (a[x][y] == 0)

dl = 0;

else {

if (dl == 0) {

xmaxt = x;

ymaxt = y;

}

dl++;

}

if (dl >= dlmax) {

xmax = xmaxt;

ymax = ymaxt;

dlmax = dl;

}

}

if (xStart > 0)

xStart--;

else

yStart++;

}

ba = xmax;

bb = ymax;

bdl = dlmax;

}

void Turn\_Matrix(int \*\*& m, int a, int b, int c) {

int h, k;

int i, j, i1, j1;

for (h = 0, i = a, j = b, i1 = a + c - 1 , j1 = b + c - 1;

c >= 2;

i++, j++, i1--, j1--, c = c - 2, h = 0) {

while(h != c-1) {

k = m[i + h][j1];

m[i + h][j1]=m[i][j + h];

m[i][j + h]=m[i1 - h][j];

m[i1 - h][j]=m[i1][j1 - h];

m[i1][j1 - h]=k;

h++;

}

}

}

void Go(int \*\* m, int a, int b, int c) {

int steps = 1, j, i, x = a, y = b;

bool go = true;

cout << "Вектор матрицы: ";

for (i = 0; i < c \* 2 - 2; i++) {

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

cout << m[x][y] << " ";

if (steps - 1 != j) {

if (go) {

x++;

y--;

}

else {

y++;

x--;

}

}

}

if (i < c - 1)

steps++;

else

steps--;

if (go) {

go = false;

if (i == c - 1)

y++;

else

x++;

}

else {

go = true;

if (i == c - 1)

x++;

else

y++;

}

}

cout << m[a + c - 1][b + c - 1] << endl;

}

void Bubble\_Sort(int \*\*& m, int a, int b, int c) {

int i, j, k;

for (k = 0; k < c \* c - 1; k++)

for (i = a + c - 1; i >= a; i--)

for (j = b + c - 2; j >= b; j--) {

if (i != a + c - 1 && j + 1 == b + c - 1)

if(m[i][j + 1] > m[i + 1][b])

swap(m[i][j+1], m[i+1][b]);

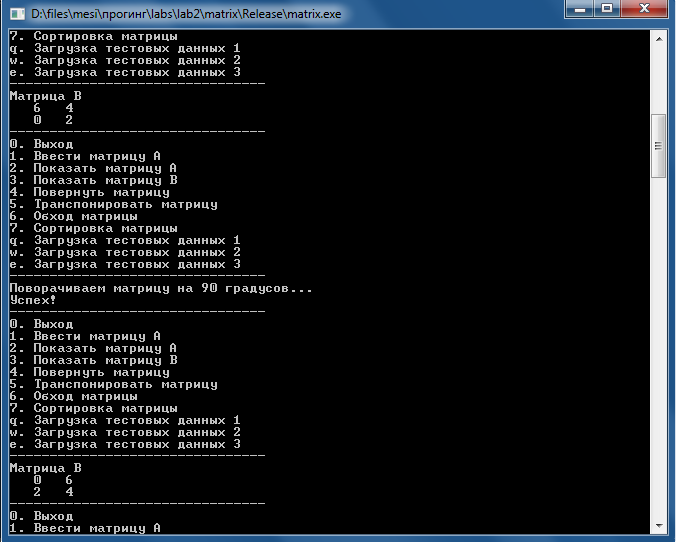
if(m[i][j] > m[i][j + 1])

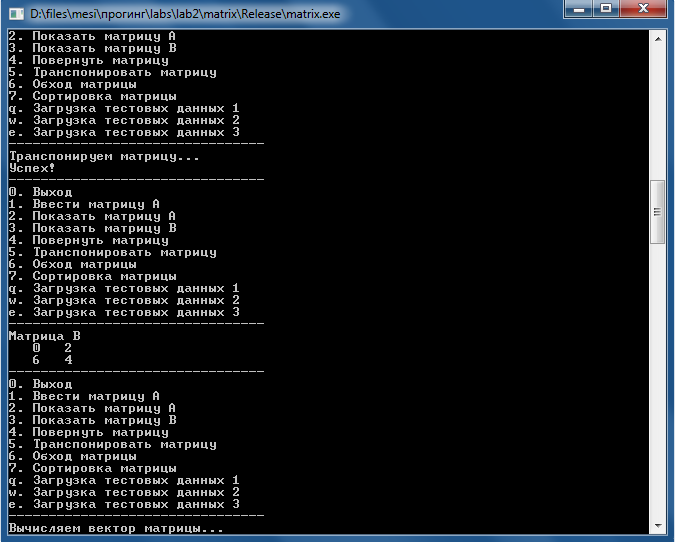
swap(m[i][j], m[i][j+1]);

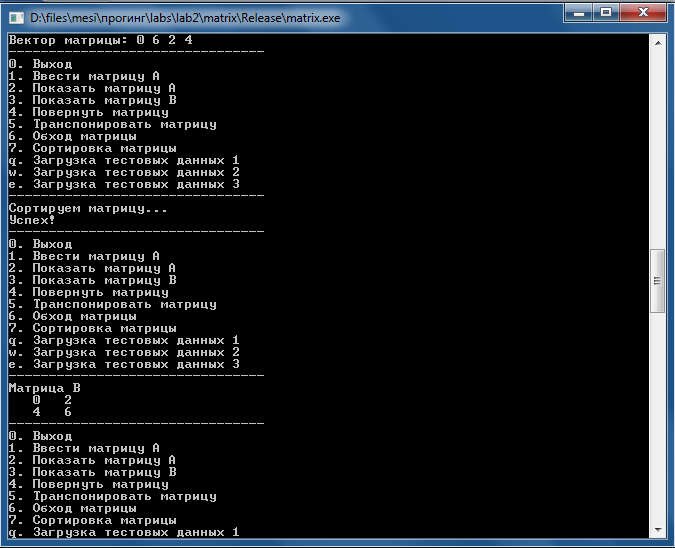
}

}

# Результат работы программы







# Используемая литература и Internet ресурсы

Г. Шилдт «Язык программирования C++»

<http://programmersclub.ru/09/>