Лабораторная работа №7

**«Двумерные массивы»**

**1) Индивидуальное задание на разработку проекта**

Разработать программный код решения следующей задачи:

Дан двумерный массив положительных и отрицательных чисел. Разработайте функцию, которая выполняет нахождение среднего арифметического всех четных элементов, лежащих ниже побочной диагонали. Разработайте функцию, которая формирует новый массив, из тех элементов исходной матрицы, значения которых больше найденного числа.

**2) Техническое задание**

1. Добавить в существующую DLL-библиотеку функции, решающие предписанные задачи. Обмен данными между подпрограммами должен осуществляться через параметры, без использования глобальных переменных. Событийная процедура должна содержать только операторы вызова пользовательских подпрограмм.

2. Изобразить написанные алгоритмы на соответствующих схемах: алгоритм нахождения среднего арифметического чётных чисел и лежащих ниже побочной диагонали, алгоритм создания нового массива из элементов матрицы, которые больше заданной границы, и алгоритм событийной процедуры.

3. В ходе решения задачи своего варианта предусмотреть написание функций для генерации матрицы, вывода матрицы, очистки памяти, выделенной под матрицу, нахождения среднего арифметического чётных чисел и лежащих ниже побочной диагонали, и создания нового массива из элементов матрицы, которые больше заданной границы. Для решения задачи своего варианта использовать многочисленные методматериалы, которые перечислены в пояснениях (ЭОИС).

4. Подготовить тесты для контрольного решения задачи. Показать их использование, отобразив в отчете по заданию

5. Выполнить проект и получить результаты, которые отобразить в виде рисунков при написании отчета по заданию.

6. Выполнить отчет в соответствии с ГОСТ на схемы алгоритмов и согласно ГОСТ оформления отчетов по НИР.

**3) Схемы алгоритмов**

Схема алгоритма нахождения количества элементов, заканчивающихся на цифру 3 и количество элементов, среднего арифметического чётных чисел и лежащих ниже побочной диагонали, приведена на рисунке 1.

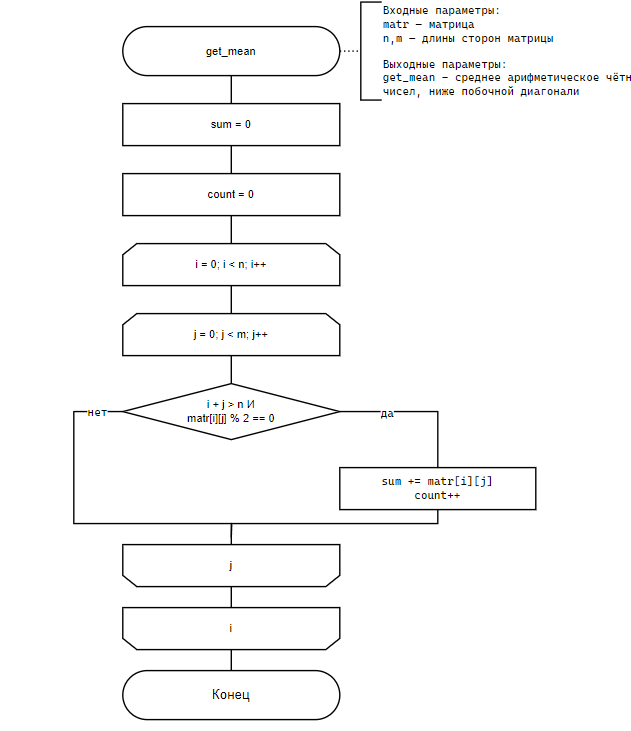


Рисунок 1 – алгоритм поиска среднего арифметического

Схема алгоритма создания нового массива из элементов матрицы, которые больше заданной границы, приведена на рисунке 2.

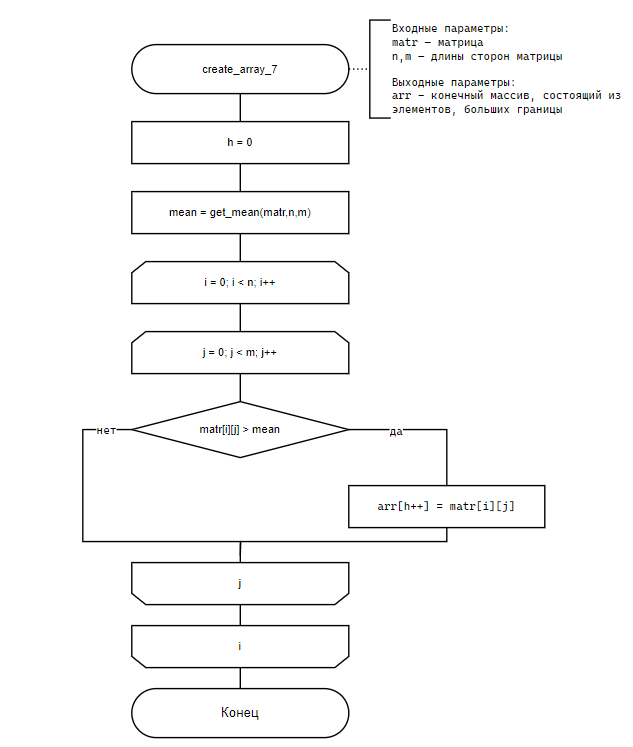


Рисунок 2 – алгоритм создания нового массива из элементов больших границы

Схема событийной процедуры приведена на рисунке 3

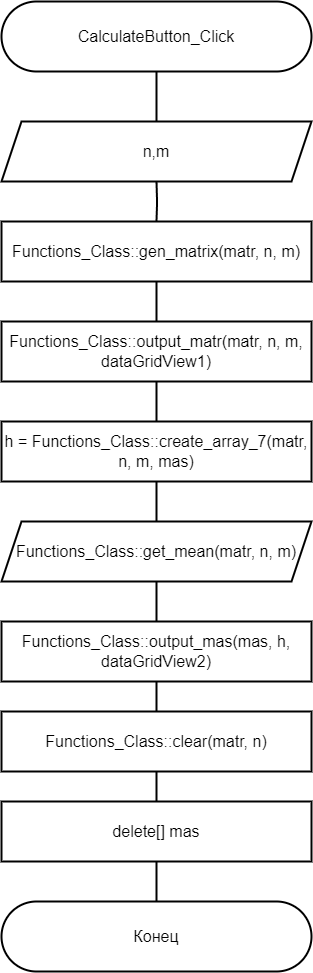


Рисунок 3 – схема событийной процедуры

**4 Программный код библиотеки**

В ходе создания DLL-библиотеки необходимо было написать код в двух файлах: заголовочном (.h) и исходном (.cpp).

Программный код заголовочного файла представлен ниже.

#pragma once

using namespace System;

using namespace System::Windows::Forms;

namespace dll\_kuleshov\_techprog\_sem3\_bei2202 {

public ref class Functions\_Class

{

public:

// ЛР #7

static void gen\_matrix(int\*\* matr, int n, int m);

static void clear(int\*\* mas, int n);

static void output\_matr(int\*\* matr, int n, int m, DataGridView^ grid);

static double get\_mean(int\*\* matr, int n, int m);

static int create\_array\_7(int\*\* matr, int n, int m, int\* arr);

static void output\_mas(int\* mas, int len, DataGridView^ grid);

};

}

Программный код исходного файла со всеми функциями представлен ниже.

#include "pch.h"

#include "dll\_kuleshov\_techprog\_sem3\_bei2202.h"

#include "Math.h"

#include <cstdlib>

#include <time.h>

using namespace System::Windows::Forms;

using namespace System;

using namespace System::IO;

using namespace System::Data::OleDb;

using namespace System::Runtime::InteropServices;

using namespace Microsoft::Office::Interop;

using namespace System::Globalization;

namespace dll\_kuleshov\_techprog\_sem3\_bei2202 {

double Functions\_Class::get\_mean(int\*\* matr, int n, int m) {

double sum = 0;

int count = 0;

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

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

if ((i + j) > n && (matr[i][j] % 2 == 0)) {

sum += matr[i][j];

count += 1;

}

}

}

return sum / count;

}

int Functions\_Class::create\_array\_7(int\*\* matr, int n, int m, int\* arr) {

int h = 0;

double mean = get\_mean(matr, n, m);

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

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

if (matr[i][j] > mean) {

arr[h++] = matr[i][j];

}

}

}

return h;

}

void Functions\_Class::output\_mas(int\* mas, int len, DataGridView^ grid) {

grid->RowCount = 2;

grid->ColumnCount = len;

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

{

grid->Rows[0]->Cells[i]->Value = String::Format("[{0}]", i);

grid->Rows[1]->Cells[i]->Value = mas[i].ToString();

}

}

}

**5 Программный код событийной процедуры**

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

String^ g = Microsoft::VisualBasic::Interaction::InputBox("Введите количество строк в матрице", "Ввод", "", -1, -1);

String^ p = Microsoft::VisualBasic::Interaction::InputBox("Введите количество столбцов в матрице", "Ввод", "", -1, -1);

int n = Convert::ToInt32(g);

int m = Convert::ToInt32(p);

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

int\* mas = new int[n \* m];

Functions\_Class::gen\_matrix(matr, n, m);

Functions\_Class::output\_matr(matr, n, m, dataGridView1);

labelh->Text = "Среднее арифметическое чётных элеметнов: " + Functions\_Class::get\_mean(matr, n, m).ToString("F3");

int h = Functions\_Class::create\_array\_7(matr, n, m, mas);

Functions\_Class::output\_mas(mas, h, dataGridView2);

Functions\_Class::clear(matr, n);

delete[] mas;

}

**6) Результаты работы программы**

На рисунке 3 представлен результат работы программы при выдуманных размерах матрицы. Как видно, программа работает корректно.

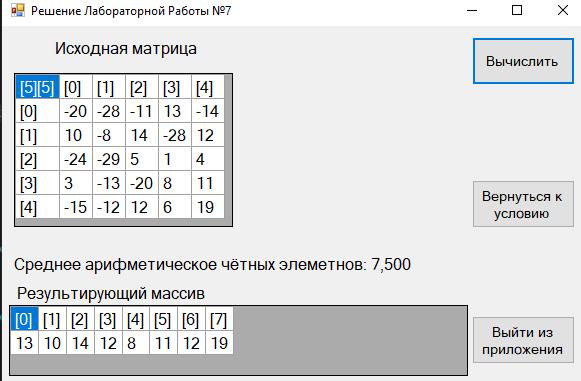


Рисунок 4 – Результаты работы при тестовых данных

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Гуриков С. Р. Основы алгоритмизации и программирования на Visual C++ : учебное пособие / С.Р. Гуриков. — Москва: ИНФРА-М, 2022. — 515 с. — (Высшее образование: Бакалавриат). — DOI 10.12737/1039154. – ISBN 978-5-16-015500-5. – Текст: электронный. – URL: https://znanium.com/catalog/product/1039154 – Режим доступа: по подписке.