**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ**

**УНИВЕРСИТЕТ им. В.Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Институт ЭИТУС

Кафедра информационных технологий

**Отчет о прохождении учебной практики**

База практики: кафедра информационных технологий

Выполнил:

студент группы ИТ-192

Жулега И.Д.

Проверил:

старший преподаватель

Имайкина Л.Е.

Белгород 2020

Оглавление

[1. Общая характеристика организации 2](#_Toc45645173)

[2. Задачи практики 3](#_Toc45645174)

[Работа с системой контроля версий GIT 3](#_Toc45645175)

[Работа с двумерными массивами, работа с файлами, применение рекурсивных функций на языке С/С++ . 8](#_Toc45645176)

[Обработка динамических массивов на языке С/С++. 17](#_Toc45645177)

[3. Заключение 26](#_Toc45645178)

# Общая характеристика организации

**Информация о кафедре Информационных технологий:**

Кафедра информационных технологий входит в состав [института информационных технологий и управляющих систем](http://www.bstu.ru/ru/faculties/itus/), организована приказом ректора от 29.06.1999 г.

Первоначальной задачей кафедры являлось проведение занятий по информатике и смежным дисциплинам у неспециалистов.

С 2003 года кафедра становится выпускающей и ведет подготовку по направлению:

* 09.03.02 - Информационные системы и технологии.

С 2014 года открыта подготовка по направлению:

* 09.03.03 - Прикладная информатика (в бизнесе).

C 2016 года началась подготовка магистров по направлению:

* 09.04.02 - Информационные системы и технологии.

Будущие специалисты по информационным технологиям обучаются с применением современных информационных технологий. Компьютерные залы оборудованы новейшими компьютерами, видеопроекционными средствами. Широко применяются автоматизированное тестирование, электронные учебники, выполнение виртуальных лабораторных работ. Посредством сети Интернет студенты могут из своего дома или общежития получать удаленный доступ к учебным материалам, расписанию занятий. При кафедре функционирует учебно-методический центр информационно-технического обеспечения образования, который проводит работу по созданию и обеспечению применения в университете прогрессивных информационно-технических средств обучения (ИТСО), компьютерных информационных технологий, мультимедийных и других технических средств, способствующих совершенствованию учебного процесса.

**Информация о ПК:**

ОС – Windows 10

Процессор: Intel Core i5-3470 3.20 GHz

ОЗУ: 8 Гб

Тип системы: 64 – разрядная

Монитор: ASUS (старый) формат 5:4, 1280х1024 точек

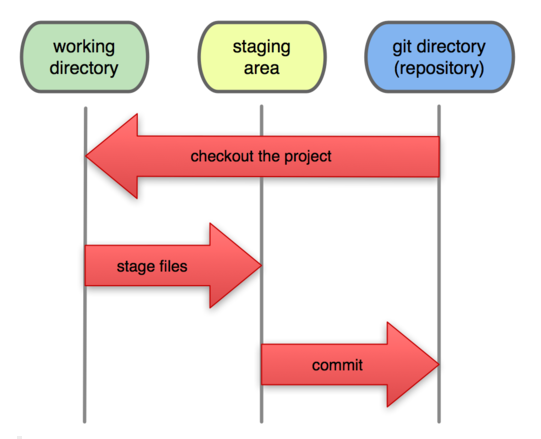
ПО: MS Office Word 2016, MS Office Visio 2016, Visual Studio 2019

# Задачи практики

## Работа с системой контроля версий GIT

* + 1. Что такое GIT?

GIT – система контроля версий (СКВ). Эта система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов.



1. Б-с того как работает GIT

GIT directory - каталог GIT - это место, где GIT хранит метаданные и базу данных объектов вашего проекта. Это наиболее важная часть GIT, и именно она копируется, когда вы клонируете репозиторий с другого компьютера.

Working directory - рабочий каталог - это извлечённая из базы копия определённой версии проекта. Эти файлы достаются из сжатой базы данных в каталоге GIT и помещаются на диск для того, чтобы вы их просматривали и редактировали.

Staging area - область подготовленных файлов — это обычный файл, обычно хранящийся в каталоге GIT, который содержит информацию о том, что должно войти в следующий коммит. Иногда его называют индексом (index), но в последнее время становится стандартом называть его областью подготовленных файлов.

* + 1. Что нам даёт СКВ?

СКВ даёт множество полезных функций. В первую очередь эта система позволит нам сделать «откат», привести отдельные файлы к предыдущему виду или даже весь проект. В следующую же очередь, в ней есть возможность просматривать происходящие изменения файлов – проекта, и так же, кто и когда внёс ошибку в код.

* + 1. Начало работы с GIT

Для того чтобы начать работать с GIT я скачал само приложение с официального сайта, открыл удобную рабочую область GIT (ПКМ –> GIT Bush Here) либо в самой рабочей папке либо в другом месте компьютера открыть рабочую область и при помощи $ cd <Диск>:\<Путь до папки>\<Папка>. Пробовал оба варианта, первый конечно же проще.

После того как оказался в исходной папке я создал сам проект, для этого написал команду $ git intit (после этого в исходной папке создаётся новая папка (.git) в которой и будет хранится ваш коммит и все данные что вы делали).

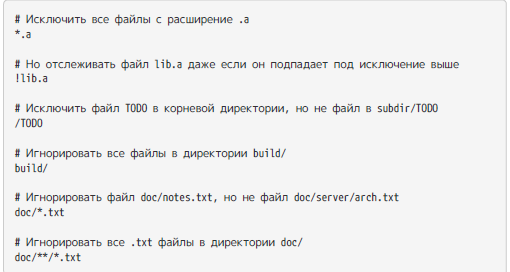
Для того что бы не забывать операнды GIT, есть удобный способ увидеть их все, для этого написал $ git help после чего появится текст с операндом и что он делает (правда всё на английском к сожалению).

После того как я инициализировал рабочую область - репозиторий я смог работать с GIT полноценно.

* + 1. Работа с GIT

После того как я добавил файлы в репозиторий, отредактировал или же, даже просто удалил файлы нужно добавить изменения в гит-проект, для этого написал команду $ git add <название папки или файла> или < . > (при написании (.) добавляются все новые или изменённые файлы). Мне удобней пользоваться $ git add . т.к. добавление отдельных файлов не только занимает время, но и если вы хотите не добавлять некоторые файлы в репозиторий то просто «за игнорируйте» его, об этом я написал дальше. Проверил работоспособность этой функции. Его (файл с игнорированием) можете посмотреть в моём проекте ссылка в конце темы про GIT.

Для того, чтобы удалить файл нужно написать git rm –cached <название файла>. Но я использовал для игнорирования некоторых фалов текстовый документ, для этого зашёл в него записываем что игнорируем (см. пример Рис.2) и перевёл в тип .gitignore. (Знак # игнорирует что находится в строке после него, т.е. можно сделать пояснения почему вы хотите игнорировать данный файл).



1. Пример

Написал git status – для просмотра статуса репозитория.

Зарегистрировал себя в репозитории при помощи команды git config - -global user.<name>или<@mail> для привязки акк. с сервер GIT (без этого вы не сможите коммитеть )

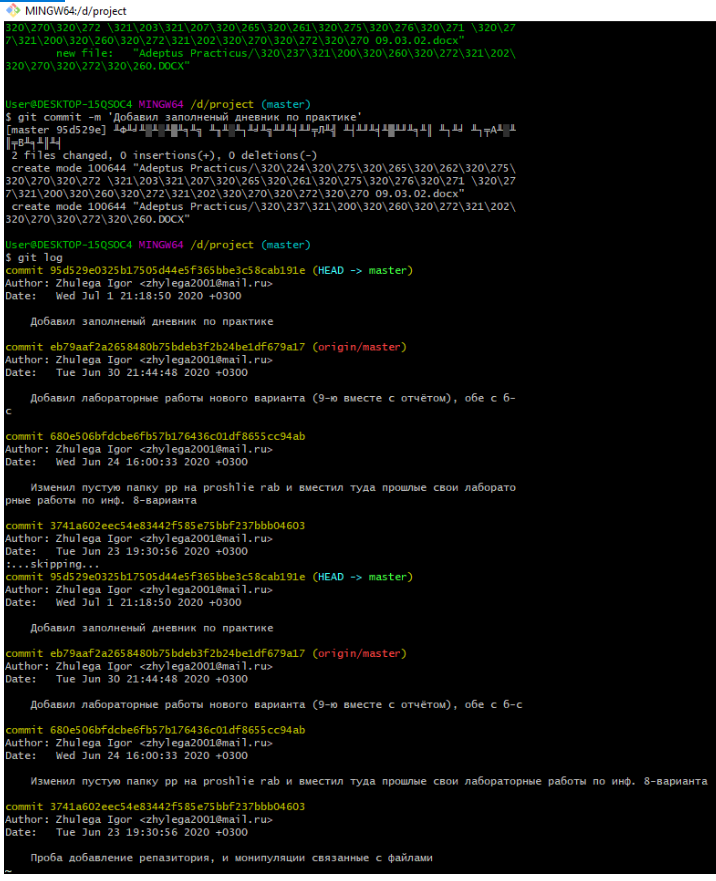


Связал аккаунт и репозиторий github при помощи команды $ git remote add origin <ссылка на проект в github>.

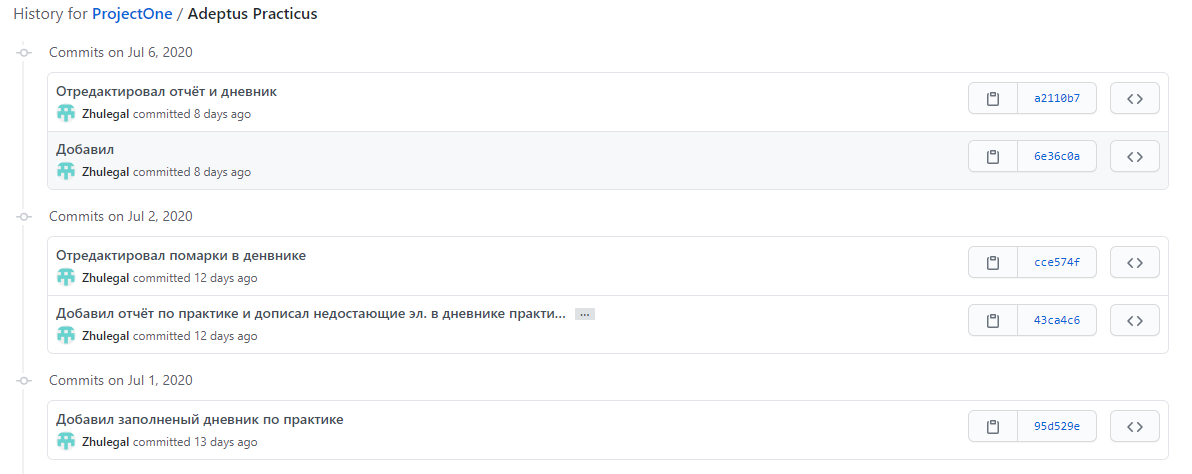
Загрузил проект на github при помощи команды $ git push –u origin master; <origin master> можно будет в дальнейшем писать только $ git push –u это будет загружать ваш коммит в сам сервер GitHub.

Зашёл в другую папку и попробовал клонировать (копировать) свой проект с сайта на компьютер при помощи команды $ git clone <ссылка на репозиторий с сайта> - скачивает.

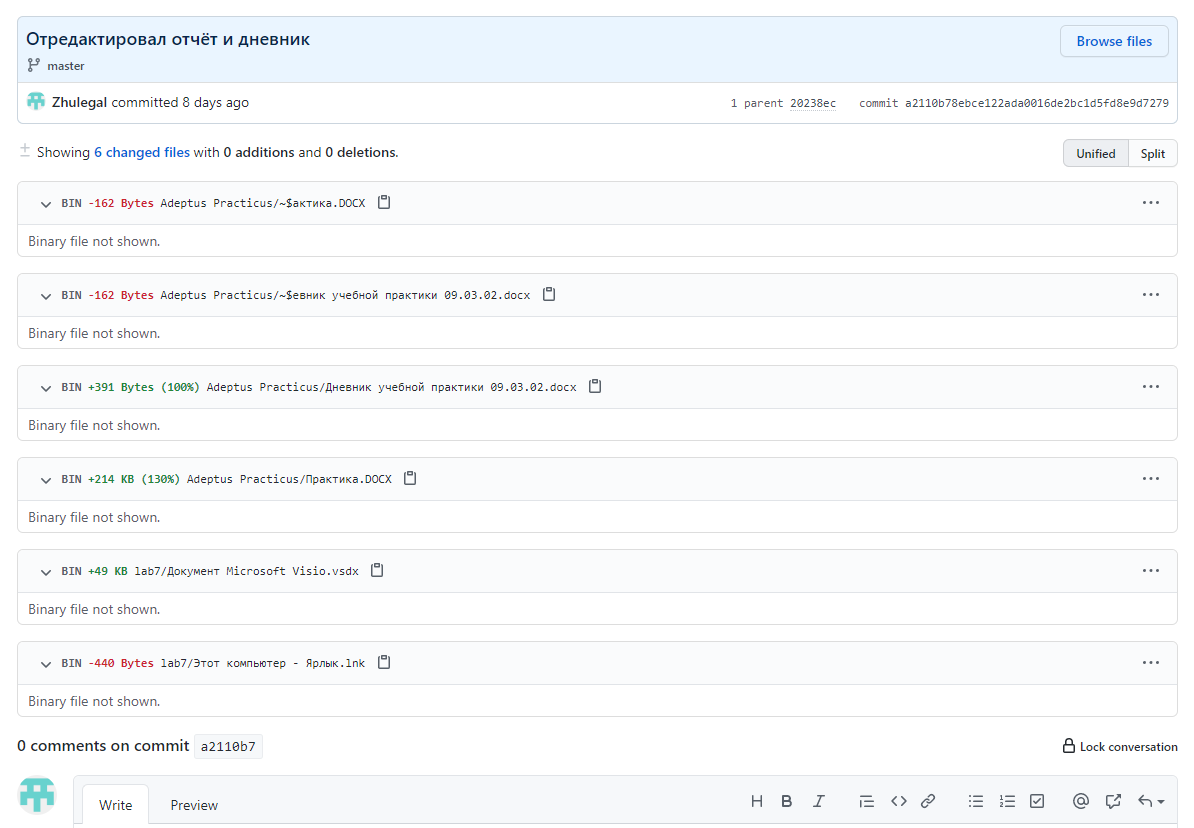
Доказательство работы с GIT’ом (см. Рис.4, Рис.5, Рис.6)







1. История одной из папок проекта на сайте (комиты).



1. последние обновление

Ссылка на GIT проект (https://github.com/ZhulegaI/ProjectOne.git)

## Работа с двумерными массивами, работа с файлами, применение рекурсивных функций на языке С/С++ .

**Постановка задания:**

Выбрать алгоритм, составить его блок-схему и программу для решения выбранного варианта задания. Программа должна по выбору пользователя осуществлять ввод исходной матрицы с клавиатуры или из файла. Для этого программа должна содержать две соответствующие функции, указатель на одну из которых необходимо передавать в функцию для вычисления элементов массива Х. Данная функция должна вызывать через указатель одну из функций ввода элементов матрицы, производить вычисление элементов массива X в соответствии с заданием и возвращать указатель на этот массив. Кроме того, программа должна содержать функцию для вывода на экран и в файл исходной матрицы и результирующего массива, а также рекурсивную функцию определения в соответствии с заданием величины Y. В программе не должно быть глобальных переменных.

Вариант 9: Дана матрица А(5х5). Определить массив *Х* из 5 элементов, каждый из которых равен 1, если элементы упорядочены по возрастанию или по убыванию и –1 в противном случае. Определить величину *Y*, как среднее арифметическое наибольшего и наименьшего элемента главной диагонали матрицы A.

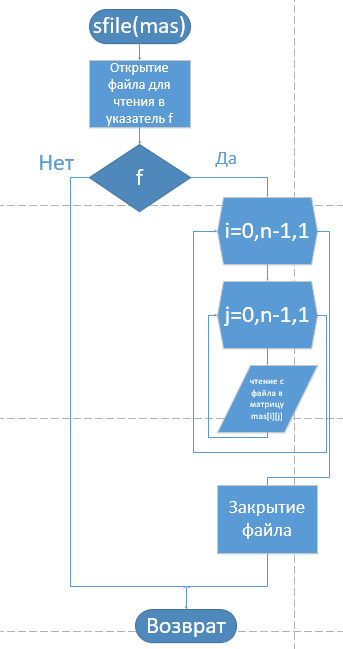
**Блок-схемы функций и основного алгоритма:**

Функция sfile:

Назначение: заполнение матрицы данными из файла. (см. Рис.7)

Входные параметры: матрица mas[n][n].

Выходные параметры: нет.



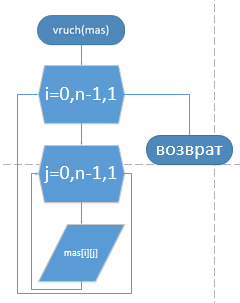
1. функция заполнения массива с файла

Функция vruch:

Назначение: ввод элементов массива с клавиатуры (см. Рис.8).

Входные параметры: матрица mas[n][n].

Выходные параметры: нет.



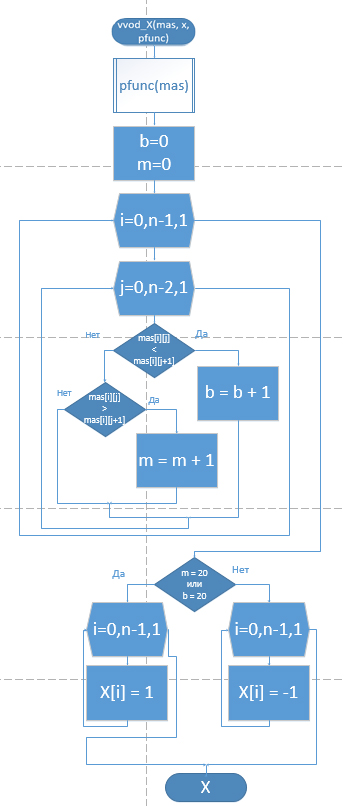
1. заполнение матрицы с клавиатуры

Функция vvod\_X:

Назначение: заполнение массива Х[n] (массив *Х* из 5 элементов, каждый из которых равен 1, если элементы упорядочены по возрастанию или по убыванию и –1 в противном случае) и матрицы mas[n][n], матрица заполняется посредством передачи в функцию указателя на функцию, которую выбирает пользователь в основной программе. (см. Рис.9)

Входные параметры: массив X[n], матрица mas[n][n], указатель pfunc

Выходные параметры: указатель на Х



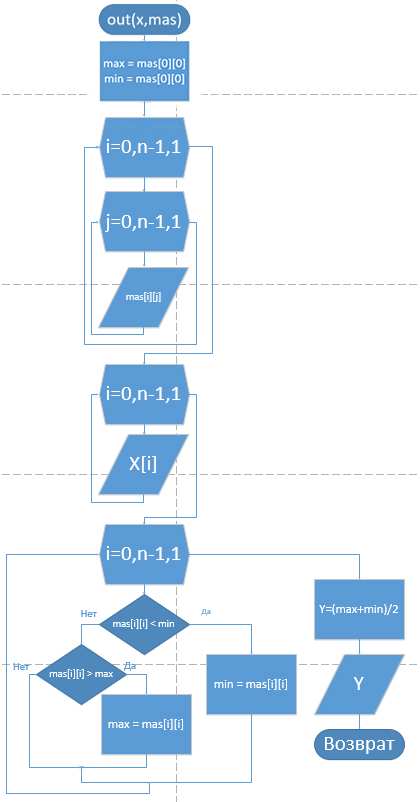
1. заполнение массива Х

Функция out:

Назначение: вывод элементов массива и матрицы, а так же подсчёт Y и вывод его на экран (см. Рис.10).

Входные параметры: массив X[n], матрица mas[n][n].

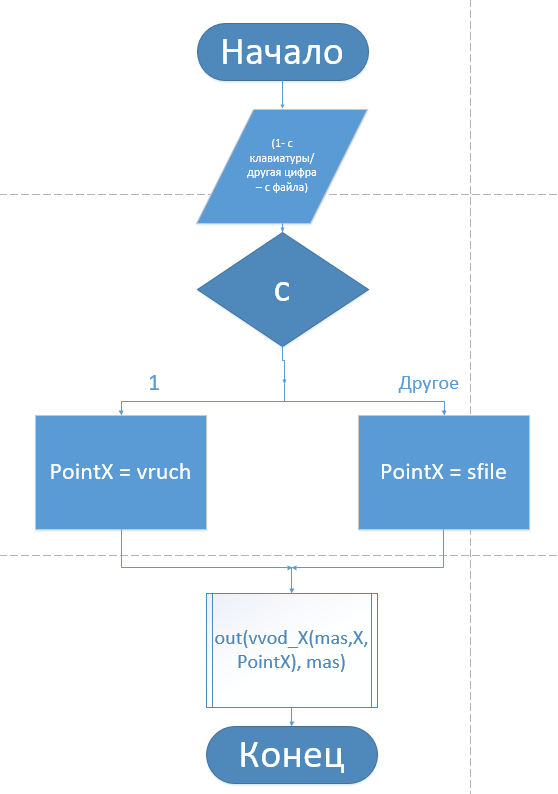
Выходные параметры: нет.



1. вывод на экран

Основная программа. (см. Рис.11)   
Алгоритм: даётся выбор пользователю как заполнить массив. После того как он сделал выбор в указатель на функцию передаётся адресс функции и далее в функции out за место параметра X[n] предаётся функция возрощающая указатель на этот массив (в функции vvod\_X же, в свою очередь идёт заполнение как массива, так и матрицы. В неё передаётся как раз таки указатель на функцию Point\_X) и матрица mas.

Всё выводится на экран массив, матрица и Y.



1. Основная программа

**Текст программы:**

#include <iostream>

#include <locale.h>

#include <stdio.h>

# define N 5

using namespace std;

// Вариант с считыванием из файла

void sfile(int mas[][N]) {

FILE\* f;

fopen\_s(&f, "1.txt", "r");

if (f) {

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

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

fscanf\_s(f, "%d", &mas[i][j]);

}

}

fclose(f);

}

}

void vruch(int mas[][N]) {

cout << "Введите эл. матрицы A(5 x 5):" << endl;

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

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

cout << "A[" << i << "][" << j << "]:\t";

cin >> mas[i][j];

}

}

}

int \*vvod\_X(int mas[][N], int X[N],void (\*pfunc) (int masp[][N])) {

pfunc(mas);

int b=0, m=0;

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

for (int j = 0; j < N - 1; j++) {

if (mas[i][j] < mas[i][j + 1]) b++; // по возростанию

else if (mas[i][j] > mas[i][j + 1]) m++; //по убыванию

}

}

if ((m == 20) or (b == 20))

for (int i = 0; i < N; i++) X[i] = 1;

else for (int i = 0; i < N; i++) X[i] = -1;

return X;

}

void out(int X[], int mas[][N]) {

int i, j, max = -100000, min = 1000000 ;

cout << "Массив A(5 x 5): " << endl;

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

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

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

}

cout << endl;

}

cout << "Массив X[5]:" << endl;

for (i = 0; i < N; i++) cout << X[i] << "\t";

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

if (mas[i][i] < min) min = mas[i][i];

else if (mas[i][i] > max) max = mas[i][i];

}

cout << endl << "Y являющийся средним арифм. главной диагонали равен: " << (max + min) / 2 << endl;

}

void main() {

setlocale(LC\_CTYPE, "rus");

int c;

int mas[N][N];

int X[N];

void (\*PointX)(int mas[][N]);

cout << "Вы хотите ввести матрицу в ручную (1-да\\другая цифра - нет)" << endl;

cin >> c;

switch(c) {

case 1: PointX = &vruch; break;

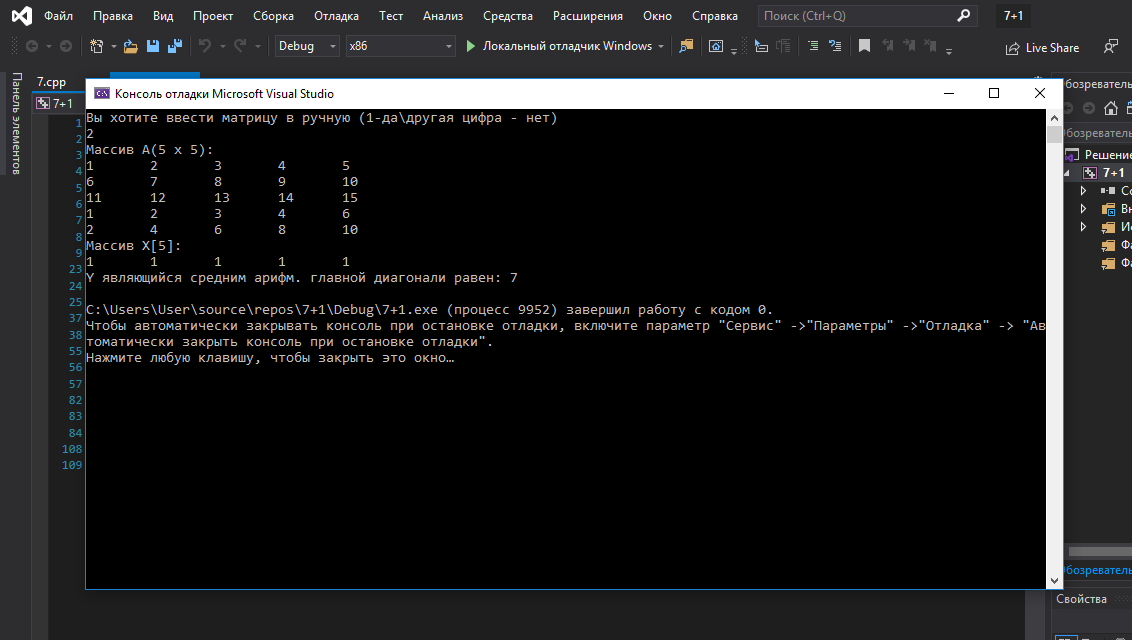
default: PointX = &sfile;

}

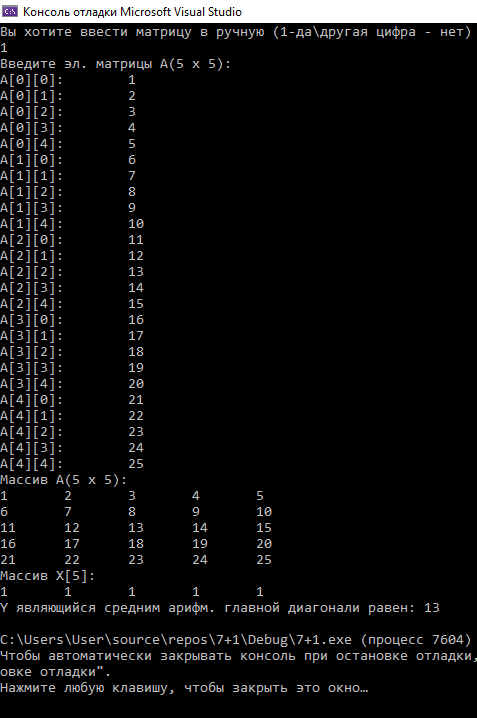
out(vvod\_X(mas, X, PointX), mas);

}

**Тестирование программы:**



1. с файла



1. с клавиатуры

## Обработка динамических массивов на языке С/С++.

Выбрать алгоритм, составить его блок-схему и программу, выполняющую создание и обработку двумерного динамического массива, в соответствии со своим вариантом задания. Во всех вариантах предполагается, что размерность массива задается на этапе выполнения пользователем. Элементы матрицы вводятся с клавиатуры. На экран выводится исходная матрица и результаты работы программы.

Вариант 9: Дана матрица А(*n* х *n*), состоящая из ненулевых элементов. Упорядочить по возрастанию элементы каждого из столбцов. Если имеются несколько столбцов, содержащих одинаковый набор элементов, то оставить без изменений только столбец с младшим индексом, заполнив остальные нулями.

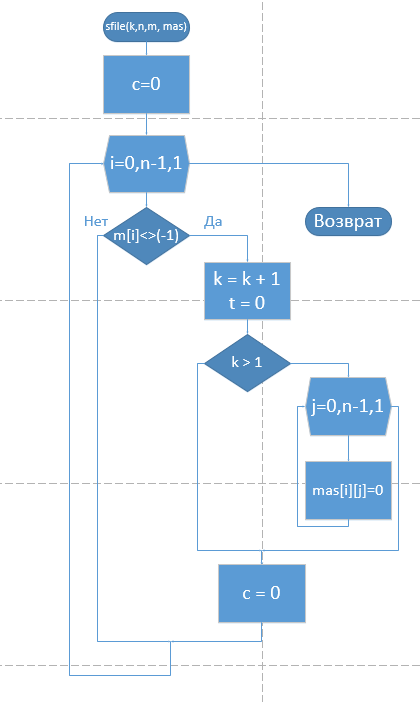
**Блок-схемы используемых функций и основного алгоритма:**

Функция sfile:

Назначение: обнуление столбцов матрицы mas. (см. Рис.14)

Входные данные: k – счётчик столбцов с повт. эл., n - константа, m – динамический массив, mas - матрица

Выходные данные: нет



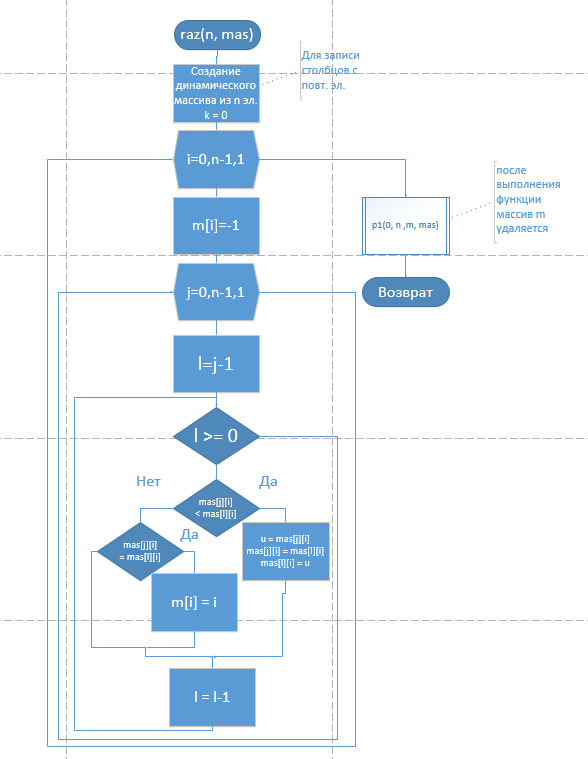
1. Функция обнуления столбцов

Функция raz:

Назначение: упорядочивание элементов (в столбцах) матрицы mas по возрастанию. (см. **Ошибка! Источник ссылки не найден.**)

Входные данные: n - константа, mas - матрица

Выходные данные: нет

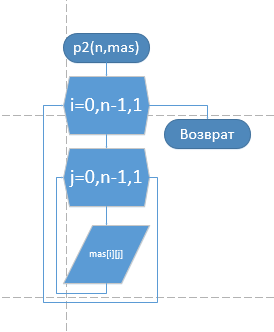


Функция p2:

Назначение: вывод на экран матрицы mas (см. Рис.16)

Входные данные: n - константа, mas - матрица

Выходные данные: нет

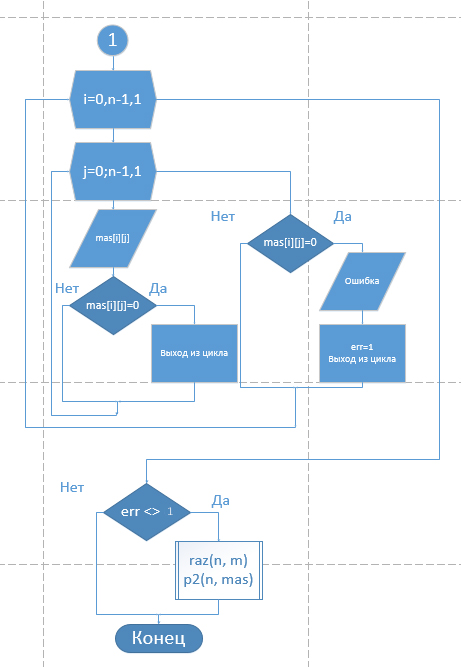


1. Функция вывода на экран эл.

Смысл программы: пользователь вводит кол-во эл. квадратной матрицы с клавиатуры, после чего создаётся динамический массив из указателей на массивы. После чего идёт заполнение матрицы и проверка на ненулевые элементы (если найдёт, то пользователю выйдет сообщение об ошибке, не соответствии эл. и заданного условия и запись в err). (см. Рис.17, Рис.18)



1. Начало блок-схемы основного алгоритма



1. Продолжение блок-схемы основного алгоритма

**Текст программы:**

#include <iostream>

#include <stdio.h>

#include <clocale>

using namespace std;

void polozhit(const int n, int\* mas[]) {

int t = 0;

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

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

if (mas[i][j] > 0) t++;

}

}

if (t == 0) cout << "Нет положительных эл." << endl;

}

void p1(int k, const int n, int m[], int\* mas[]) {

int l, t = 0, c = 0;

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

if (m[i] != (-1)) {

k++;

t = 0;

if (k > 1) {

if (c == 0) {

cout << "Меняем столбец " << (i + 1) << " обнуляя" << endl;

c++;

}

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

mas[j][i] = 0;

}

}

}//Обнуление повторяющихся (см. на условие варианта)

c = 0;

}

}

void kol\_vo(int &i, const int n, int m[], int\* mas[]) {

int l = n - 1, kol = 0;

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

while ((l >= 0) && (j!=l)) {

if (mas[j][i] == mas[l][i]) {

m[i] = i;//запомнил столбец с повт. эл.

kol++;

}

l--;

}

}

cout << "Одинаковое кол-во эл. в столбце " << (i + 1) << " равняется: " << kol << endl;

kol = 0;

}

void raz(const int n, int\* mas[]) {

int l, u;

int\* m = new int[n];

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

polozhit(n, mas);

m[i] = -1;

int c = 0;

kol\_vo(i, n, m, mas);

for (int j = n - 1; j >= 0; j--) {

l = j - 1;

while (l >= 0) {

if (mas[j][i] < mas[l][i]) {

if (c == 0) {

cout << "Меняем столбец " << (i+1) << " по возростанию" << endl;

c++;

}

u = mas[j][i];

mas[j][i] = mas[l][i];

mas[l][i] = u;

}

l--;

}

}

c = 0;

}// Расспределение по возростанию эл. и запоминание столбцов с повт.

p1(0, n, m, mas);

delete[]m;

}

void p2(const int n, int\* mas[]) {

cout << "После выполнения:" << endl;

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

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

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

}

cout << endl;

}//Вывод массива

}

void main() {

setlocale(LC\_CTYPE, "rus");

cout << "Введите кол-во элементов n для A(n x n): " << endl;

int n, i, j, err = 0;

cin >> n;

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

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

mas[i] = new int[n];

}

//////////////////////////////////////////////////////////////////////

cout << "Введите элементы не равные 0:,\n";

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

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

cin >> mas[i][j];

if (mas[i][j] == 0) break;

}

if (mas[i][j] == 0) {

cout << "Ошибка, найден элемент не удовлетворяющий условию!" << endl;

err = 1;

break; }

}// заполнение двумерного массива

if (err != 1) {

raz(n, mas);

p2(n, mas);

}

///////////////////////////////////////////////////////////////////////

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

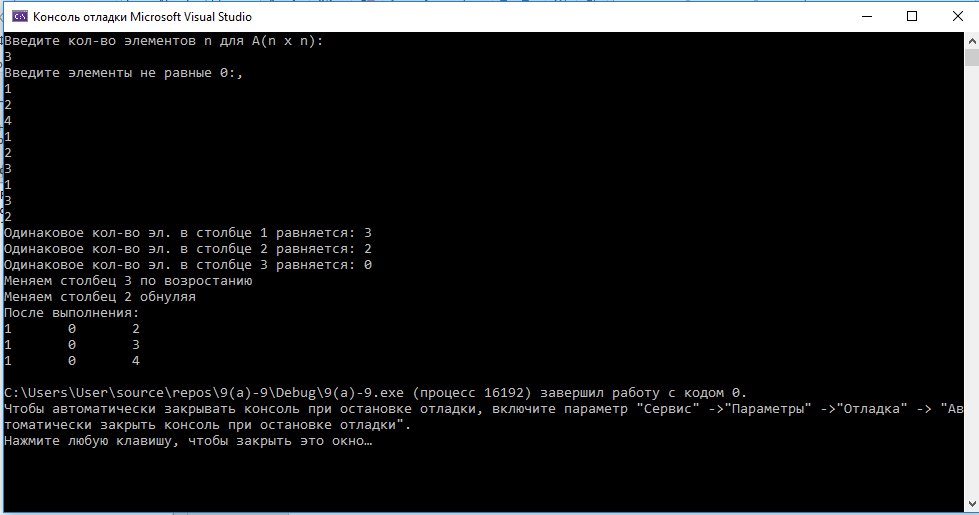
delete[] mas[i];

}// Удалил одномерные массивы

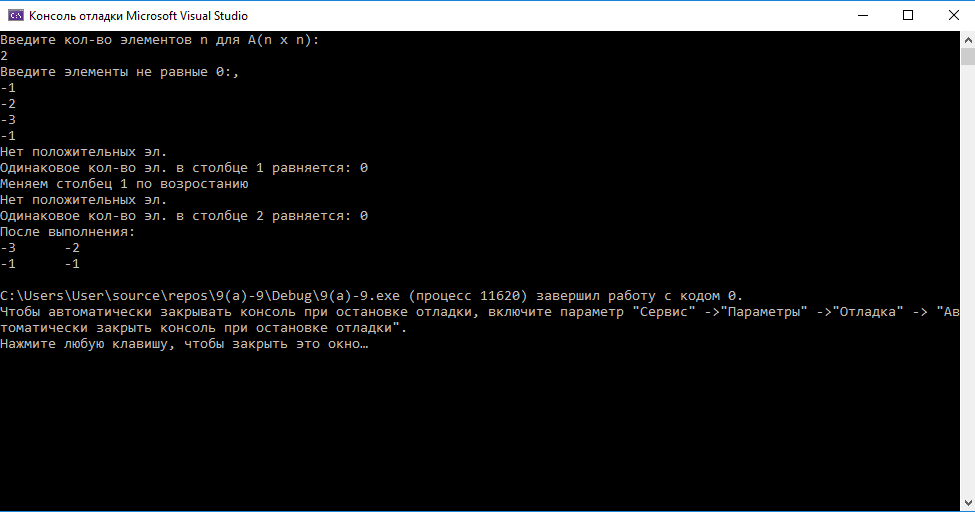
delete[]mas;// Удалил динамический массив

}

**Тестирование программы:**



1. Подсчёт столбцов и сообщение о том что меняется



1. Сообщение об отсутствии положительных элементов

# Заключение

Во время прохождения учебной практики я научился основам работы с GIT, понял его основную задачу и смог закрепить знания и навыки в написании программ на языке C/C++.