МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г. ШУХОВА»

(БГТУ им. В.Г. Шухова)

**Кафедра информационных технологий БГТУ им.Шухова**

* + - 1. **Отчет о прохождении учебной практики**
      2. **База практики: кафедра информационных технологий**

Выполнил:

студент группы ИТ -192

Кузнецова В.Е.

Принял:

Имайкина Л.Е.

Белгород

2020

**Оглавление**

1 Общая характеристика преприятия 3

2 Задачи практики 4

2.1 Индивидуальное задание 4

3 Выполнение индивидуального задания 4

3.1 Знакомство с Git 4

3.1.1 Что такое git, и зачем это нужно? 4

3.1.2 Плюсы и минусы Git 4

3.1.3 Работа с GIT 4

3.2 Обработка двумерных массивов, файловый ввод-вывод, применение итеративных и рекурсивных функций. 8

3.3 Обработка динамических массивов и связанных списков данных. 17

3.4 Заключение 24

1. Общая характеристика преприятия

Кафедра Информационных Технологий была создана 29 июня 1999 года.

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

С 2003 года кафедра становится выпускающей и ведет подготовку  
по следующим направлениям:

* [**09.03.02 - Информационные системы и технологии**](http://kit.bstu.ru/specialties#z1)

С 2014 года открыта подготовка по направлению

* [**09.03.03 - Прикладная информатика (в бизнесе)**](http://kit.bstu.ru/specialties#z2).

C 2016 года началась подтоговка магистров по направлению

* [**09.04.02 - Информационные системы и технологии**](http://kit.bstu.ru/specialties#z3).
  + - 1. **Заведующие кафедрой**:
* кандидат физико-математических наук, доцент Ломазов В.А.  
  (с 1999 по 2002)
* кандидат технических наук, доцент Иванов И.В.  
  (с 2002 по настоящее время).
  + - 1. Проводится повышение квалификации по программам:
* подготовка пользователей ЭВМ с международной сертификацией ECDL (European Computer Driving License);
* подготовка пользователей справочно-правовой системы «Консультант Плюс» с российской сертификацией;
* подготовка пользователей с сертификацией 1C: Профессионал по направлениям "Платформа 1C: Предприятие 8", "1C: Бухгалтерия 8".
  + - 1. Кафедра расположена:
      2. кабинет 418, главный корпус, тел.: 30-99-01, внутр. 14-91.
      3. Сайт кафедры:
      4. <http://it.bstu.ru/>
      5. Во время практики работа на ноутбуке lenovo ideapad 320-17iKB
      6. Тип системы: 64 – разрядная
      7. Model Name 80XM0010RK
      8. Программное обеспечение: MS Office Word 2016, MS Office Visio 2010, Visual Studio 2019, Git CMD.

1. Задачи практики
   1. Индивидуальное задание

* Ознакомиться с Git;
* Научиться добавлять в Git репозиторий новые версии программы;
* Получить навыки работы с двумерными массивами в языке С/С++;
* Получить навыки работы с передачей параметров в функции по ссылке;
* Получить навыки описания рекурсивных функций;
* Получить навыки работы с многомерными динамическими массивами в языке С/С++;
* Закрепить навыки работы, выполнив задание по программированию на языке С++

1. Выполнение индивидуального задания
   1. Знакомство с Git
      1. Что такое git, и зачем это нужно?
         1. Если в общем, то это одна из множества разных систем контроля версий (VCS), но что это значит?

**Система контроля версий** — это система, записывающая изменения одного или множества файлов в течение времени и позволяющая вернуться позже к определённой версии.

* + - 1. Если просто, то git — это прокаченная кнопка «Сохранить». Сохранение в git зовётся коммит (commit). Когда мы делаем коммит, мы создаём копию состояния всех файлов в рабочей директории — git репозитории.
      2. Git можно использовать для сохранения состояния файлов любого типа, не обязательно только для разработки ПО. Но именно в этом случае, в процессе работы необходимо постоянно вносить изменения в большое количество разных файлов, и git в данном случае мощный инструмент для просмотра изменений, дающий возможность откатывать изменения в случае ошибок, или совмещать изменения нескольких разработчиков.
    1. Плюсы и минусы Git
       1. Плюсы:
* Это крутая кнопка сохранения которая делает копию состояния вашего проекта на определенный момент времени
* Это инструмент который помогает работать над разными версиями вашего проекта одновременно.
* Это потрясающая система для взаимодействия нескольких людей в рамках проекта

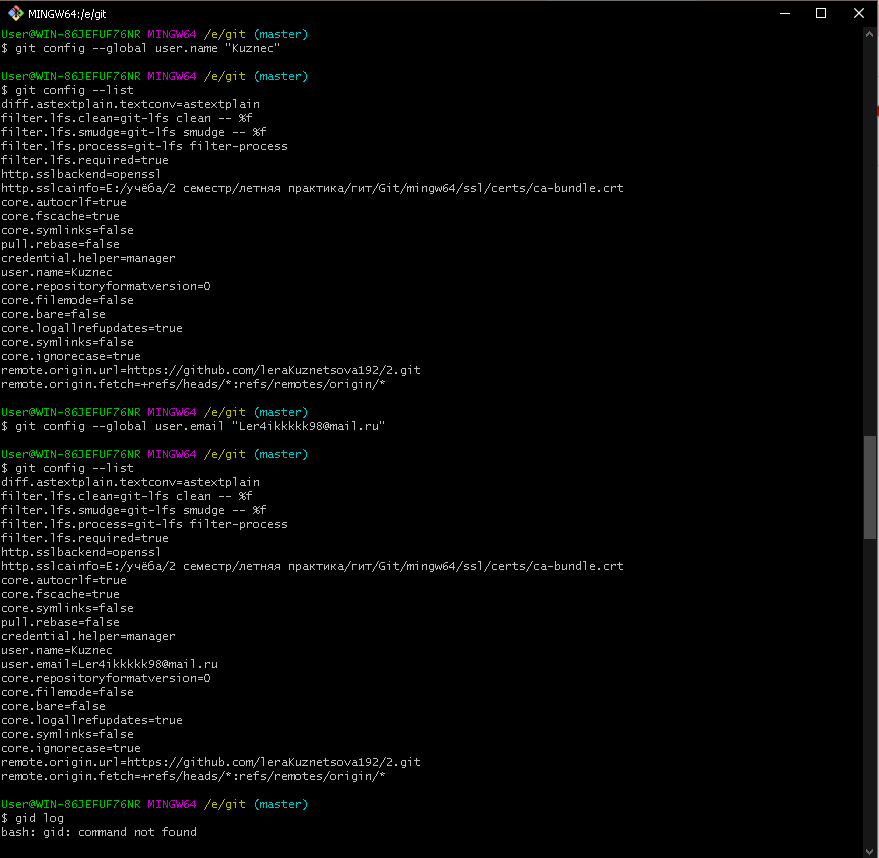
Минусы:

* + - * Поддержка Windows отстает.
      * Отсутствует возможность резервного копирования в нескольких местах;
      * Весь интерфейс на английском.
    1. Работа с GIT
       1. Для того чтобы начать работать с GIT нужно скачать само приложение. После открытия терминала Git Bash в Windows нужно перейти в нужную нам папку с помощью команды cd <адрес пути папки>. После надо настроить имя и адрес электронной почты в git. Каждый коммит имеет своего автора, с записанными именем и адресом электронной почты. Для настройки можно воспользоваться следующими двумя командами:

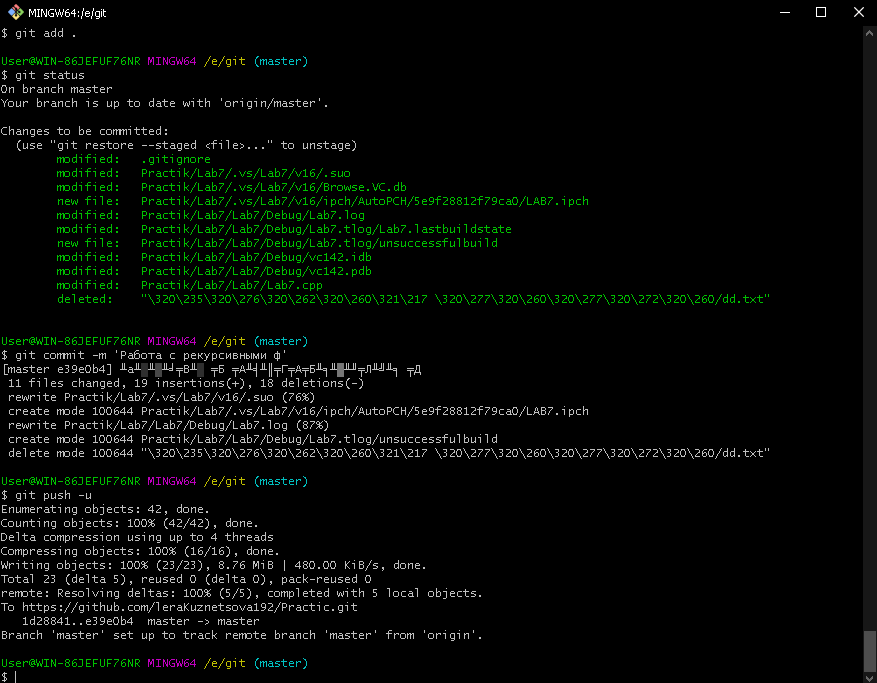
git config --global user.name “Ваше имя”

git config --global user.email “Ваш email”

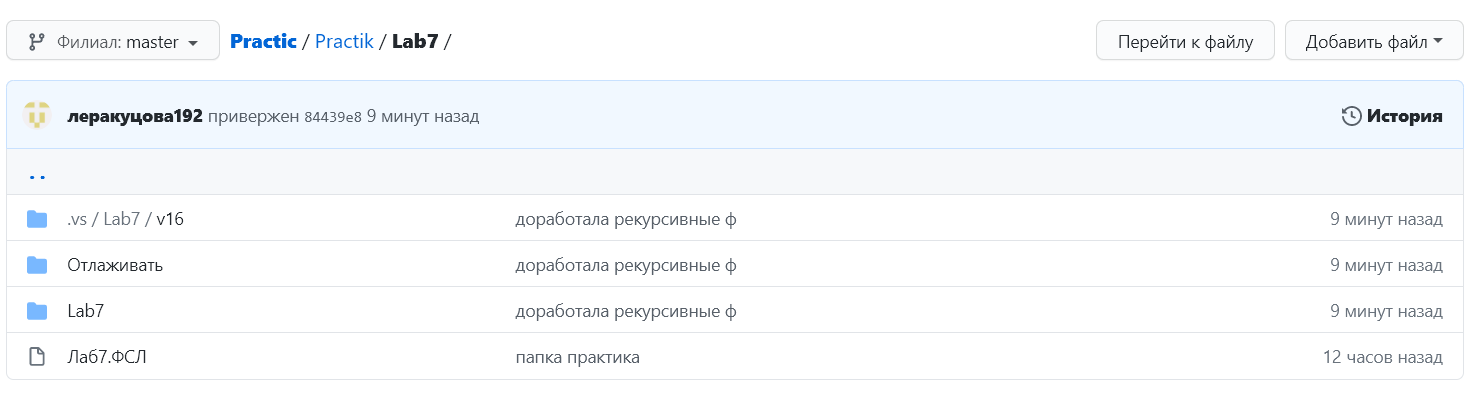
* + - 1. Авторов проекта можно посмотреть используя команду git log.
      2. С помощью команды git --help мы можем посмотреть все функции и их значение
      3. Далее расмотрим часто используемые функции:
      4. git init – создает папку ркпозиторий
      5. git status – показывает статус проэкта
      6. git log – позволяет просмотреть все комментарии
      7. git add - добавляет в коммит все файлы директории
      8. git commit –m ‘’ добавляет ком.
      9. git push –u -загружает проэкт и ком. в github
      10. git clone – копирует проект
      11. Ниже приложила скрины работы с GitBash (см. Рис.1-Рис.2) и с GitHub (см. Рис.3-Рис.5)



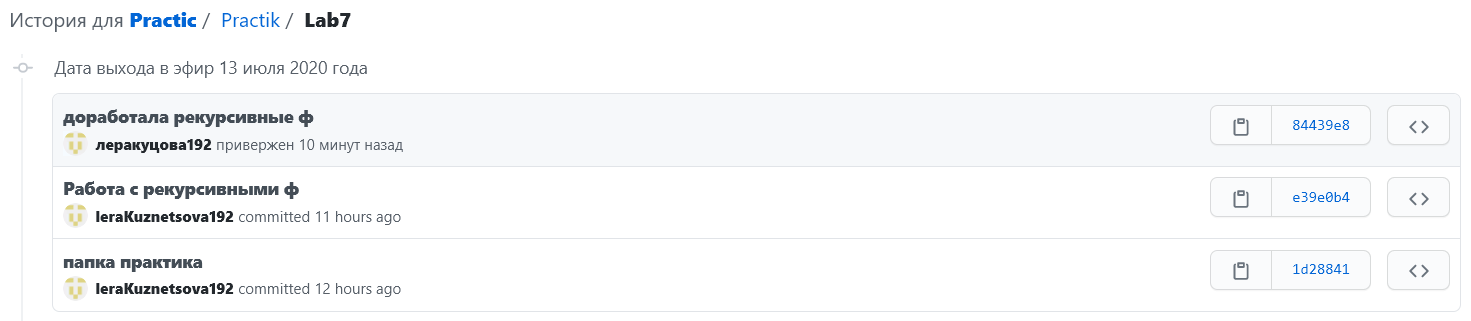
* + - * 1. Найстройка имени и почты



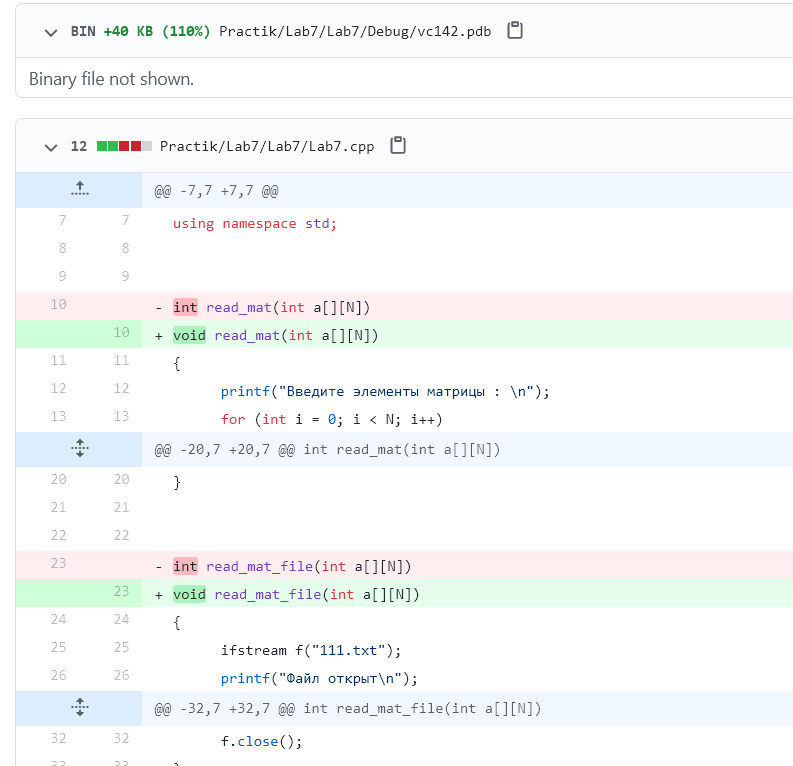
* + - * 1. Загрузка в сервер изменений
      1. Для просмотра изменений сначала зашла на GitHub в свой репозиторий и зашла в папку, в который были изменения, после нажала на кнопку «History»(История) (см. ниже), открылось окно со всеми зафиксированными изменениями в этой папке (см. ниже), выбираю самое верхнее изменение и просматриваю изменения (см. Рис.5 )



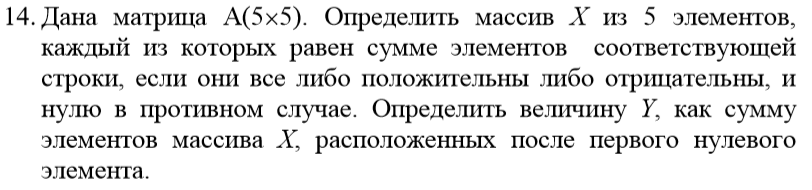
* + - * 1. Просмотр изменений



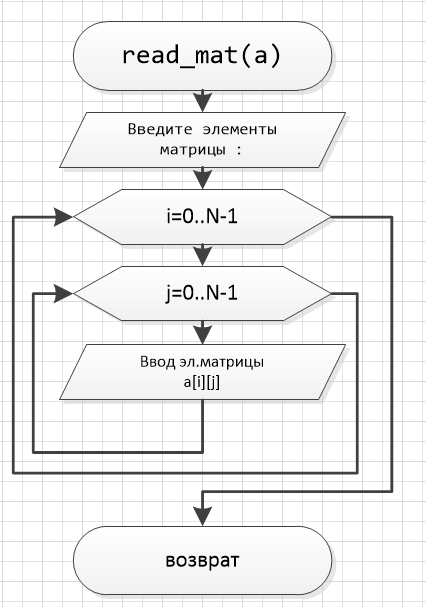
* + - * 1. Просмотр изменений2



* + - * 1. Просмотрела изменения в файле
      1. Ссылка на GIT проект <https://github.com/leraKuznetsova192/Practic>
  1. Обработка двумерных массивов, файловый ввод-вывод, применение итеративных и рекурсивных функций.
     + 1. **СОДЕРЖАНИЕ РАБОТЫ**
       2. Выбрать алгоритм, составить его блок-схему и программу для решения выбранного варианта задания (см. Рис.6). 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.



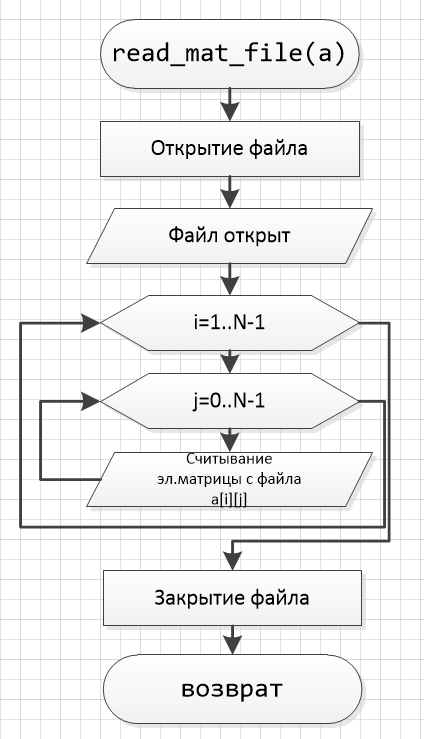
* + - * 1. Мой вариант
      1. **Ход работы**
      2. 1. Составила блок-схемы .
      3. Назначение: ввод элементов массива с клавиатуры (см. Рис.7).
      4. Входные параметры: матрица a[n][n].
      5. Выходные параметры: нет.



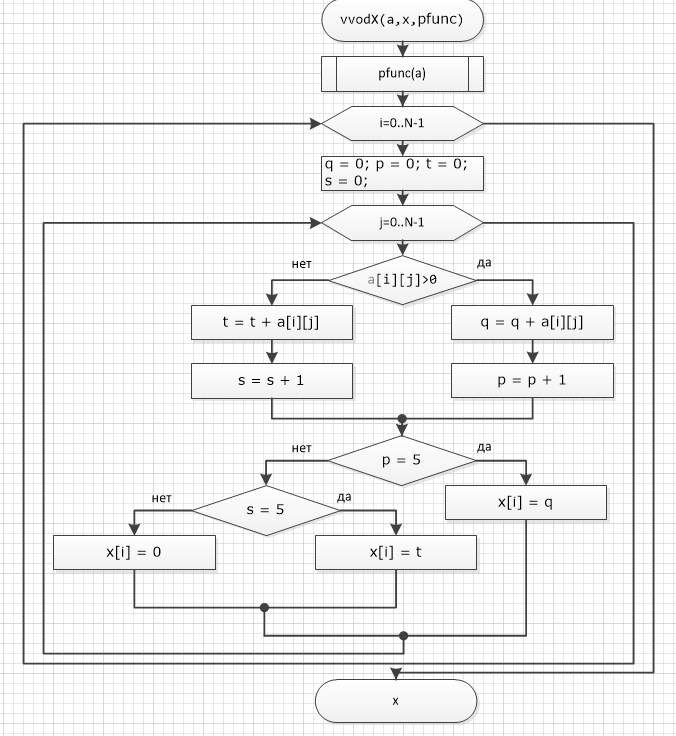
* + - * 1. Заполнение матрицы с клавиатуры

Назначение: заполнение матрицы данными из файла. (см. Рис.8)

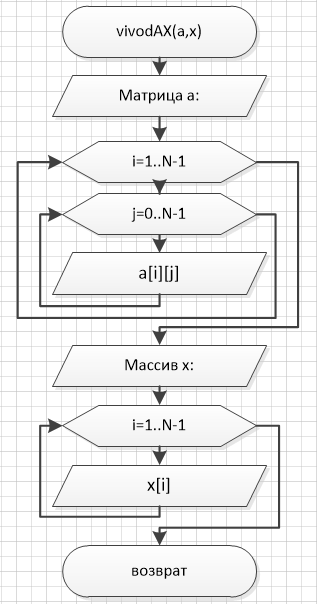
* + - 1. Входные параметры: матрица a[n][n].
      2. Выходные параметры: нет.



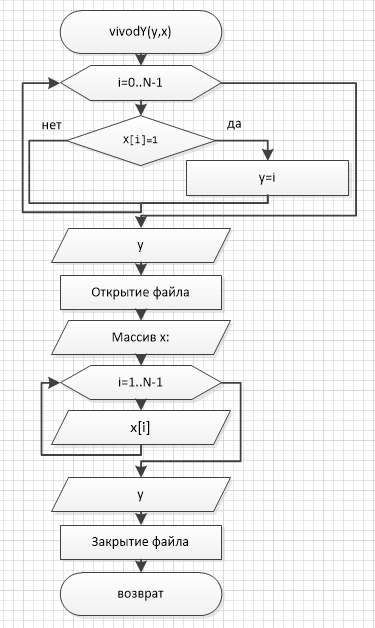
* + - * 1. Заполнение матрицы с файла
      1. Назначение: заполнение массива x[n] (см. Рис.9)
      2. Входные параметры: массив x[n], матрица a[n][n], указатель pfunc
      3. Выходные параметры: указатель на x



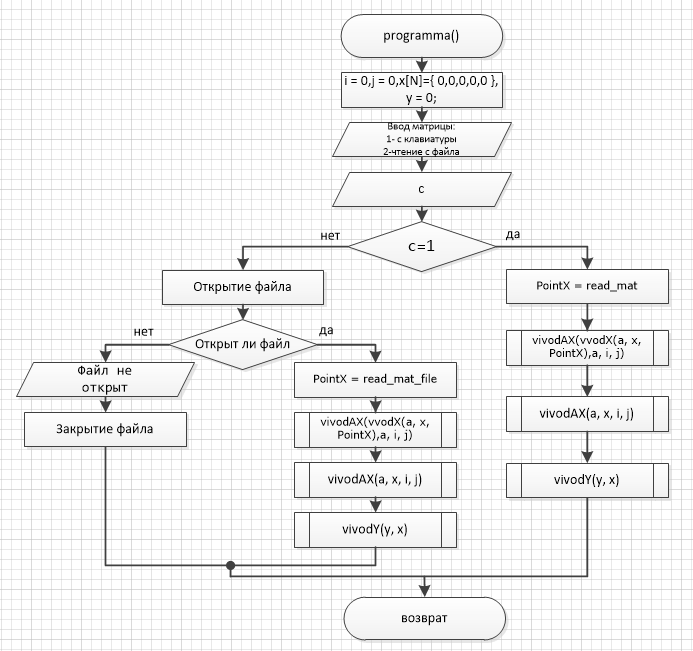
* + - * 1. заполнение массива x
      1. Назначение: вывод элементов массива и матрицы (см. Рис.10).
      2. Входные параметры: массив x[n], матрица a[n][n].
      3. Выходные параметры: нет.



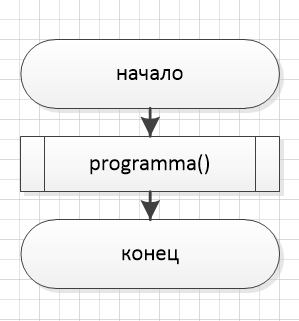
* + - * 1. Вывод на экран массива и матрицы
      1. Назначение: подсчёт y, вывод его на экран и если заполнение с клавиатуры, то создаётся файл в который выводится массива х и у(см. Рис.11).
      2. Входные параметры: массив x[n], матрица a[n][n].
      3. Выходные параметры: нет.



* + - * 1. Работа с у
      1. Назначение:выполнение поставленной задачи (см. задание-Рис.6, Рис.12).
      2. Входные параметры: нет.
      3. Выходные параметры: нет.



* + - * 1. Решение моего варианта
      1. Основная программа. (см. Рис.13) Вызов функции programm().



* + - * 1. Основная программа

2. Программа:

#include <stdio.h>

#include <locale.h>

#include <iostream>

#include <fstream>

#include <iomanip>

#define N 5

using namespace std;

void read\_mat(int a[][N])

{

printf("Введите элементы матрицы : \n");

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

{

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

{

cin >> a[i][j];

}

}

}

void read\_mat\_file(int a[][N])

{

ifstream f("111.txt");

printf("Файл открыт\n");

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

{

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

f >> a[i][j];

}

f.close();

}

int \*vvodX(int a[][N], int x[N], void (\*pfunc) (int mas[][N]))

{

int q, p, t, s;

pfunc(a);

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

{

q = 0; p = 0; t = 0; s = 0;

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

{

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

{

q = q + a[i][j];

p = p + 1;

}

else

{

t = t + a[i][j];

s = s + 1;

}

}

if (p == 5)

x[i] = q;

else

{

if (s == 5)

x[i] = t;

else

x[i] = 0;

}

}

return x;

}

void vivodAX(int x[N], int a[N][N], int i, int j)

{

printf("Матрица a:\n");

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

{

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

printf("%d ", a[i][j]);

printf("\n");

}

printf("Массив x:\n");

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

printf("%d ", x[i]);

}

void vivodY(int y, int x[N],int i)

{

int t; i = 0;

while (x[i]!=0)

{

t = i+1;

i = i + 1;

}

if (t > 0)

{

for (i = t; i < N; i++)

y = y + x[i];

}

else printf("\n Невозможно посчитать Y");

printf("\n Y=%d", y);

ofstream f("111.txt");

f << "Массив x:\n";

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

{

f << x[i] << ' ';

}

f << "\n Y:" << y;

f.close();

}

void programma()

{

int c, i = 0, j = 0, a[N][N], x[N] = { 0,0,0,0,0 }, y = 0;

setlocale(LC\_CTYPE, "");

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

printf("Ввод матрицы:\n 1- с клавиатуры\n 2-чтение с файла\n");

cin >> c;

if (c == 1)

{

PointX = &read\_mat;

vivodAX(vvodX(a, x, PointX), a, i, j);

vivodY(y, x,i);

}

else

{

ifstream f("111.txt");

if (f.is\_open())

{

PointX = &read\_mat\_file;

vivodAX(vvodX(a, x, PointX),a, i, j);

vivodY(y, x,i);

}

else

{

printf("Файл не открыт\n");

f.close();

}

}

}

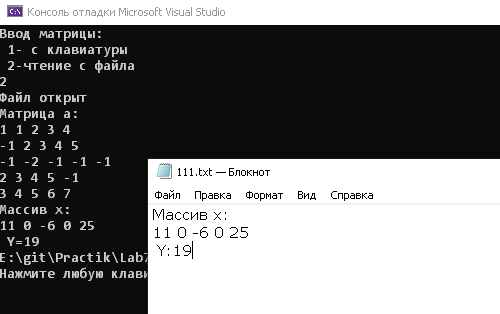
int main(void)

{

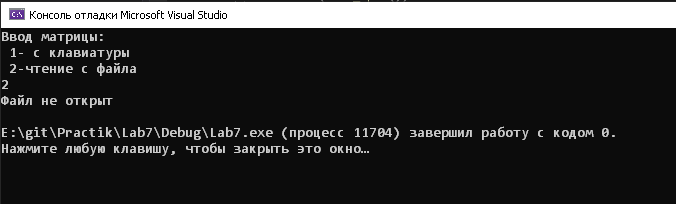
programma();

}

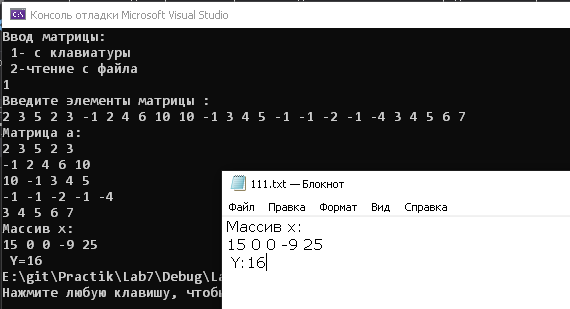
* + - 1. 3. Результаты работы программы (см. Рис.14-Рис.16):
      2. Заполнение с файла (файл сущ.)



* + - * 1. Работа программы 1
      1. Заполнение с файла (файл не сущ.)



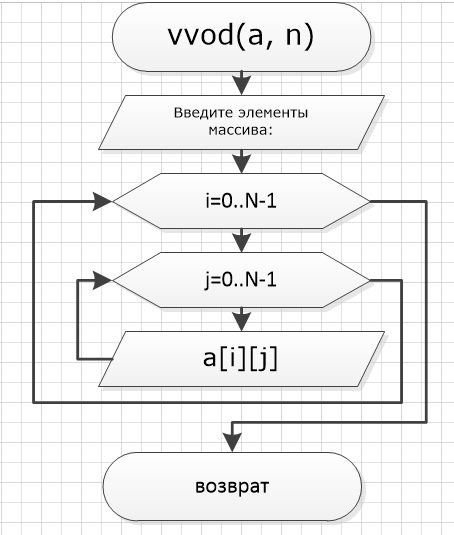
* + - * 1. Работа программы 2
      1. Заполнение с клавиатуры



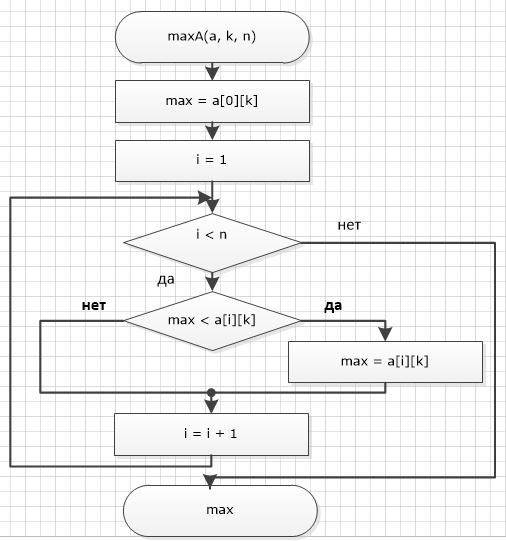
* + - * 1. Работа программы 3
  1. Обработка динамических массивов и связанных списков данных.
     + 1. **СОДЕРЖАНИЕ РАБОТЫ**
       2. Задание А
       3. Выбрать алгоритм, составить его блок-схему и программу, выполняющую создание и обработку двумерного динамического массива, в соответствии со своим вариантом задания (см. Рис.17). 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.



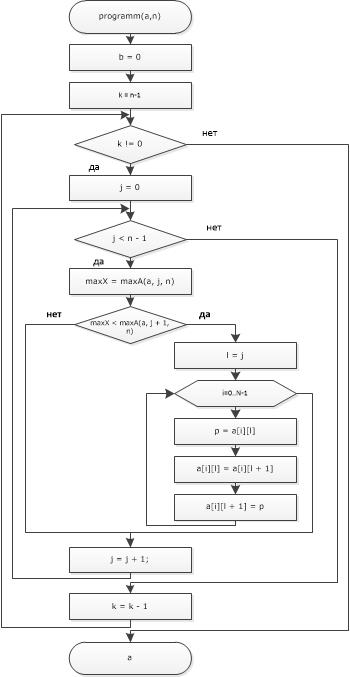
* + - * 1. Мой вариант
      1. **Ход работы**
      2. 1. Составила блок-схему.
      3. Назначение: ввод элементов массива с (см. Рис.18).
      4. Входные параметры: динамический массив a[n][n].
      5. Выходные параметры: нет.



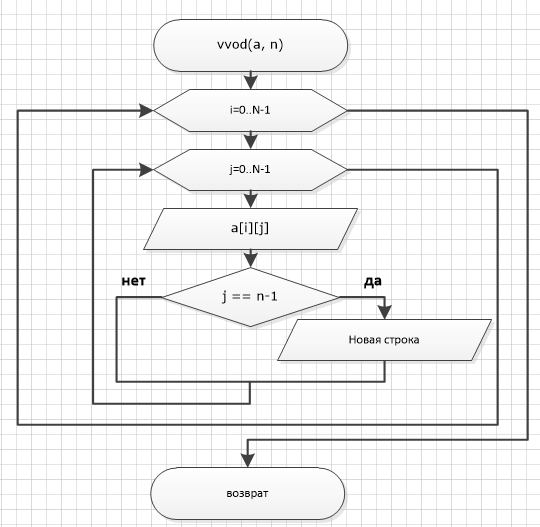
* + - * 1. Ввод массива
      1. Назначение: нахождение максимального эл. в столбце (см. Рис.19).
      2. Входные параметры: динамический массив a[n][n], k,n
      3. Выходные параметры: max.



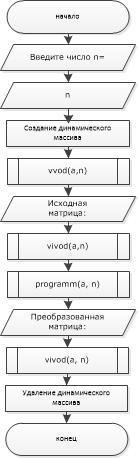
* + - * 1. Нахождение максимального эл. в столбце
      1. Назначение: выполнение поставленной задачи (см. Рис.17) (см. Рис.20).
      2. Входные параметры: динамический массив a[n][n],n
      3. Выходные параметры: a.



* + - * 1. Решение поставленной задачи
      1. Назначение: вывод массива на экрам (см. Рис.21).
      2. Входные параметры: динамический массив a[n][n],n
      3. Выходные параметры: нет.



* + - * 1. Вывод массива на экран
      1. Основная программа. (см. Рис.22) Ввод пользователем значения n. Создание динамического массива.Вызов программ ввода, вывода и решения задачи. После удаление динамического массива.



* + - * 1. Основная программа
      1. 2. Программа:

#include <stdio.h>

#include <locale.h>

#include <iostream>

#include <fstream>

#include <stdlib.h>

using namespace std;

void vvod(int\*\* a, int n)

{

int i, j;

cout << "Введите элементы массива: ";

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

{

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

cin >> a[i][j];

}

}

int maxA(int\*\* a, int k, int n)

{

int max = a[0][k]; int i = 1;

while (i < n )

{

if (max < a[i][k])

max = a[i][k];

i = i + 1;

}

return max;

}

int\*\* programm(int\*\* a, int n)

{

int i, j,l,k,maxX; int p;

k = n-1;

while (k != 0)

{

j = 0;

while (j < n - 1)

{

maxX = maxA(a, j, n);

if (maxX < maxA(a, j + 1, n))

{

l = j;

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

{

p = a[i][l];

a[i][l] = a[i][l + 1];

a[i][l + 1] = p;

}

}

j = j + 1;

}

k = k - 1;

}

return a;

}

void vivod(int\*\* a, int n)

{

int i, j;

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

{

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

{

cout << a[i][j] << ' ';

if (j == n-1)

{

cout << "\n";

}

}

}

}

int main()

{

setlocale(LC\_CTYPE, "rus");

int n, i;

cout << "Введите число n= ";

cin >> n;

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

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

a[i] = new int[n];

vvod(a,n);

cout << "Исходная матрица: \n";

vivod(a,n);

programm(a, n);

cout << "Преобразованная матрица: \n";

vivod(a, n);

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

{

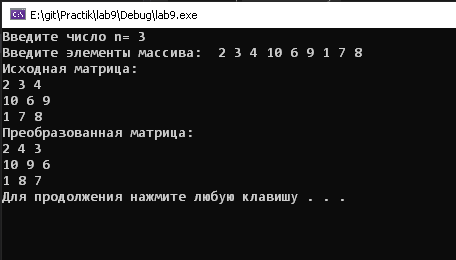
delete[] a[i];

}

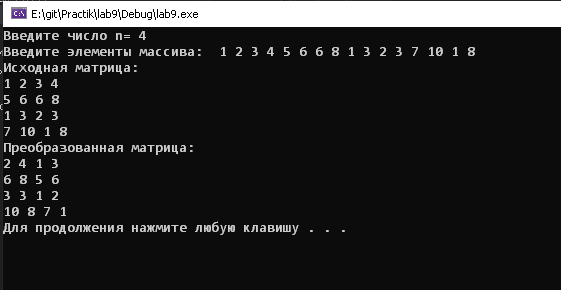
delete[] a;

system("pause");

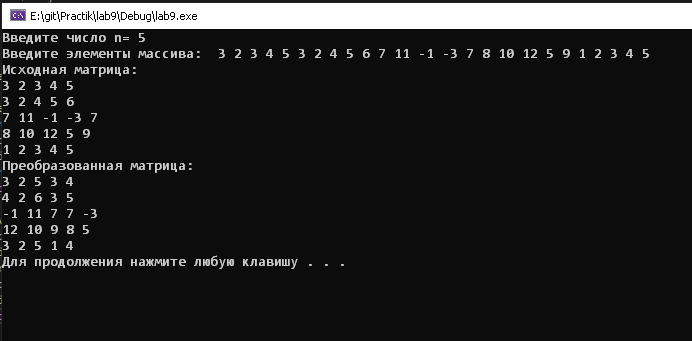
* + - 1. }
      2. 3. Результаты работы программы (см. Рис.23-Рис.25):



* + - * 1. А(3\*3)



* + - * 1. A(4\*4)



* + - * 1. А(5\*5)
  1. Заключение
     + 1. На практике я научилася основам работы с GIT и GitHub и закрепила знания работы с языком C/C++ по темам : Обработка двумерных массивов, файловый ввод-вывод, применение итеративных и рекурсивных функций на языке С/С++.. Обработка динамических массивов на языке С/С++.