

**Содержание**

Реферат .............................................................................................................5

Введение............................................................................................................6

1.Постановка задачи.........................................................................................7

2.Теоретическая часть задания .......................................................................8

3.Описание алгоритма программы..................................................................9

4.Описание программы.....................................................................................10

5.Руководство по использованию....................................................................13

6.Тестирование...................................................................................................15

7.Ручной расчёт задачи......................................................................................20

Заключение ........................................................................................................21

Список литературы ...........................................................................................22

Приложение A. Листинг программы ..............................................................23

**Реферат**

Отчет 25 стр,15 рисунков.

ГРАФ, ТЕОРИЯ ГРАФОВ, ИЗОЛИРОВАННЫЕ ВЕРШИНЫ.

Цель исследования – разработка программы, способной выявлять изолированные вершины в графе.

В работе рассмотрен алгоритм поиска изолированных вершин. Установлено, что с помощью данного алгоритма можно найти изолированные вершины в графе.

**Введение**

Алгоритм нахождения изолированных вершин в графе основан на просмотре строк в матрице смежности графа.

В качестве среды разработки мною была выбрана среда MicrosoftVisualStudio2017, язык программирования – Си.

Целью данной курсовой работы является разработка программы на языке Си, который является широко используемым. Именно с его помощью в данном курсовом проекте реализуется алгоритм нахождения изолированных вершин графа.

**1.Постановка задачи**

Требуется разработать программу, котораянаходит изолированные вершины в графе. Исходный граф в программе должен задаваться матрицей смежности.Программа должна работать так, чтобы пользователь вводил количество вершин для генерации матрицы смежности. После обработки этих данных на экран должна выводиться матрица смежности и изолированные вершины. Необходимо предусмотреть различные исходы поиска, чтобы программа не выдавала ошибок и работала правильно. Устройство ввода – клавиатура и мышь. Задания выполняются в соответствии с вариантом №8.

**2.Теоретическая часть задания**

Граф G (рисунок 1) задается множеством вершин X1, X2, ..., Xn. и множеством ребер, соединяющих между собой определенные вершины. Ребра из множества А ориентированы, что показывается стрелкой, которая указывает достижимость данной вершины, граф с такими ребрами называется ориентированным графом.

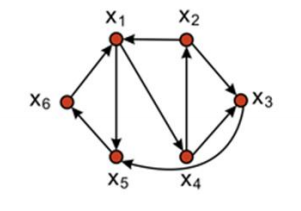


Рисунок 1 – Граф

При представлении графа матрицей смежности информация о ребрах графа хранится в квадратной матрице, где присутствие пути из одной вершины в другую обозначается единицей, иначе нулем.

Вершина называется **изолированной**, если она не является концом ни для одного ребра.

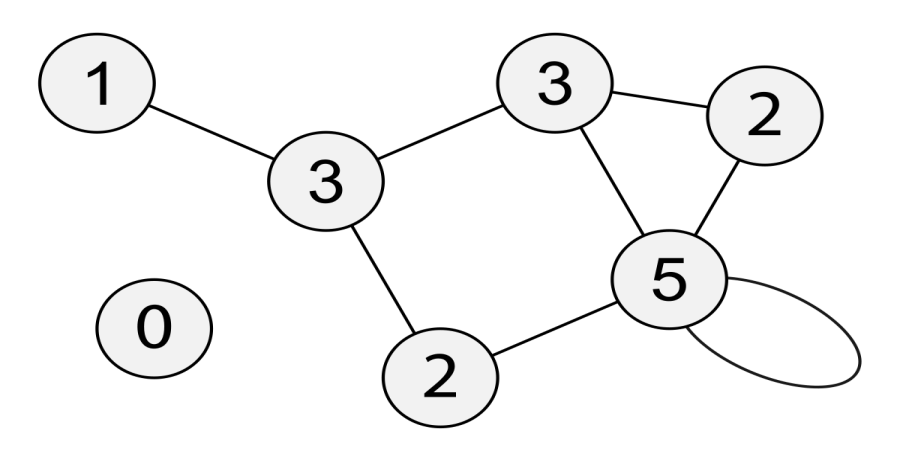


Рисунок 2 – Граф с изолированной вершиной

Вершина 0 является **изолированной**.

**3.Описание алгоритма программы**

Алгоритм нахождения изолированных вершин в матрице смежности заключается в поиске степеней вершины. Идёт построчный пробег по матрице, и все найденные степени суммируются. Затем следует проверка: если степень равняется 0, то вершина называется изолированной, и наоборот. Ниже представлен псевдокод функцииvivod():

**vivod**

1)дляi = 0 пока i<m делать i++

2)iz = 0

3)для j = 0 пока j<m делать j++

4)если(mat[i][j] != 0){

5)iz += 1}

6) mass[i] = iz

7) flag = 0

8) дляi = 0 покаi<mделатьi++

9) если (mass[i] == 0){

10)вывод – изолированная вершина, i +1

11) flag = 1}

12) если(flag == 0 ){

13)вывод – нет изолированных вершин }

**4. Описание программы**

Для написания данной программы использован язык программирования Си.Язык программирования Си - универсальный язык программирования,который завоевал особую популярность у программистов, благодаря сочетаниювозможностей языков программирования высокого и низкого уровней. Проект был создан в виде консольного приложения Win32 (Visual C++).Данная программа является многомодульной, поскольку состоит из нескольких функций: vvod1, vvod2, vivod.

Для реализации программы понадобится двумерный динамический массив(матрица \*\*mat), куда в дальнейшем будут записывать значения матрицы смежности,переменная m, отвечающая за размер матрицы, динамический массив(\*mass), куда в будущем будут занесены степени вершин.

Следующими командами выделяется память под двумерный динамический и динамический массивы:

mat = (int\*\*)malloc(m \* sizeof(int\*));

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

mat[i] = (int\*)malloc(m \* sizeof(int));

}

mass = (int\*)malloc(m \* sizeof(int));

Далее выполняется заполнение матрицы в зависимости от выбранной функции:

**vvod1** –двойной цикл, главная диагональ заполняется ‘0’, все что выше – случайными числами от 0 до 30, нижняя часть дублируется.

void vvod1(intm, int \*\*mat) {

srand(time(NULL));

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

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

if (i == j) {

mat[i][j] = 0;

}

if (i> j) {

mat[i][j] = rand() % 2;

if (mat[i][j] == 0)

mat[i][j] = mat[i][j];

else

mat[i][j] = 1 + rand() % 30;

}

mat[j][i] = mat[i][j];

}

}

}

**vvod2** -двойной цикл, главная диагональ заполняется ‘0’, все что выше –задаётся пользователем с клавиатуры, нижняя часть дублируется.

void vvod2(intm, int \*\*mat) {

srand(time(NULL));

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

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

if (i == j) {

mat[i][j] = 0;

}

if (i> j) {

scanf\_s(" %d", &mat[i][j]);

}

mat[j][i] = mat[i][j];

}

}

}

Далее в функции **vivod**() идёт пробег по матрице, и все степени записываются в массив, затем идёт пробег по массиву, если условие совпадает, то выводится конкретная изолированная вершина, и наоборот.

voidvivod(FILE \*f, intm, int \*\*mat, int \*mass) {

for (inti = 0; i<m; i++) { //VIVOD

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

printf(" %4d", mat[i][j]);

fprintf(f, " %4d", mat[i][j]);

}

printf("\n");

fprintf(f, "\n");

}

;

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

intiz = 0;

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

if (mat[i][j] != 0) {

iz += 1;

}

}

mass[i] = iz;

}

int flag = 0;

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

if (mass[i] == 0) {

printf("\nВершина %d является изолированной", i + 1);

fprintf(f, "\nВершина %d является изолированной", i + 1);

flag = 1;

}

}

if (flag == 0) {

printf("Матрица не имеет изолированных вершин!");

fprintf(f, "Матрица не имеет изолированных вершин!");

}

Также помимо функции, программа включает в себя меню выбора. Реализацию которого представлены в **Приложение A. Листинг программы.**

**5. Руководство по использованию**

После запуска программы у пользователя появляется выбор между “Начать программу” при вводе (1) и “Завершить программу” при вводе (2). Так как этот выбор находится в бесконечном цикле, при нажатии другой цифры, будет происходить тот же самый запрос.

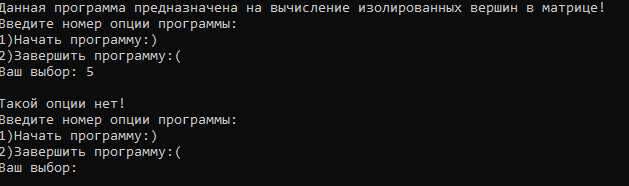


Рисунок 3– Меню

При выборе (2) “Завершить программу” – программа завершает свою работу.

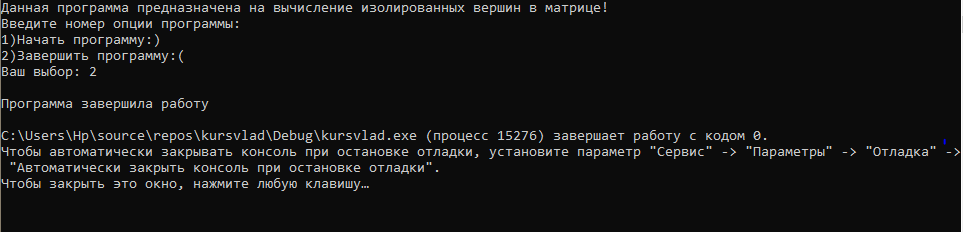
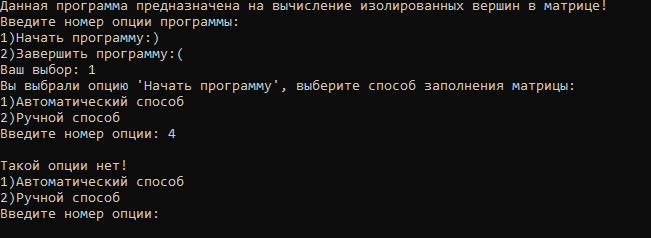


Рисунок 4– Завершение программы

При выборе (1)“Начать программу” пользователю дают выбор между форматом заполнения матрицы: (1)”Автоматический способ” и (2)”Ручной способ”. Так как этот выбор находится в бесконечном цикле, при нажатии другой цифры, будет происходить тот же самый запрос.Рисунок5 –Выбор в меню

При выборе (1)”Автоматический способ”: пользователю дают выбрать размер матрицы, вызывается функция(vvod1(m, mat)), в которую передаются значения размера(m)и выделенную память под саму матрицу. Заполнение происходит следующим образом: идёт пробег по двум циклам, т.е по строкам и столбикам матрицы, главная диагональ заполняется “0”, часть, которая находится выше главной диагонали заполняется случайным образом(от 0 до 30), затем она дублируется с нижней частей матрицы.

После данной функции, вызывается следующая - vivod(f, m, mat, mass), в которую передают размер; заполненную матрицу; файл, в котором ,будет храниться информация о изолированных вершинах, а также о матрице смежности; массив, в котором будут находиться степени изолированных вершин. Функция сначала выводит сгенерированную матрицу на экран, затем идет двойной цикл, с условием на нахождение степени, все эти степени записываются в массив(mass), которые в дальнейшем выведут пользователю есть изолированные вершины в графе или их нет.

При выборе (2)”Ручной способ”: пользователю дают выбрать размер матрицы, вызывается функция(vvod2(m, mat)), в которую передаются значения размера(m)и выделенную память под саму матрицу. Заполнение происходит следующим образом: идёт пробег по двум циклам, т.е по строкам и столбикам матрицы, главная диагональ заполняется “0”, часть, которая находится выше главной диагонали заполняется самим пользователем, затем она дублируется с нижней частей матрицы. После данной функции, вызывается следующая - vivod(f, m, mat, mass).

**6.Тестирование**

Среда разработки MicrosoftVisualStudio 2010 предоставляет все средства, необходимые при разработке и отладке многомодульной программы. Тестирование проводилось в рабочем порядке, в процессе разработки, после завершения написания программы. В ходе тестирования было выявлено и исправлено множество проблем, связанных с вводом данных, изменением дизайна выводимых данных, алгоритмом программы, взаимодействием функций.

Таблица 1 - Описание поведения программы при тестировании

|  |  |  |
| --- | --- | --- |
| Описание теста | Ожидаемый результат | Полученный результат |
| Запуск | Запуск программы | Верно |
| Проверка на не правильный выбор в меню | Программа должна оповестить пользователя в его неправильном выборе и предложить снова варианты | Верно |
| Выход | Выход из программы | Верно |
| Выбор “Начать программу” | Перейти на следующий этап программы(Формат заполнения) | Верно |
| Проверка на не правильный выбор в меню “формат заполнения” | Программа должна оповестить пользователя в его неправильном выборе и предложить снова варианты | Верно |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| Проверка автоматического способа и ввод данных | Программа должна перейти на автоматическое заполнение матрицы, при этом должна выйти надпись | Верно |
| Сохранение данных в файл | Данные матрицы и информация о нахождении изолированных вершин должны записаться в файл | Верно |
| Проверка ручного формата и его результаты на наличие изолированных вершин + сохранение в файл | Изолированные вершины определены заранее, при заполнении должно вывести на экран результат, который будет сохранен в файл | Верно |
| Автоматический режим, уменьшенный размер матрицы | Возможность появления изолированной вершины, вывод | Верно |

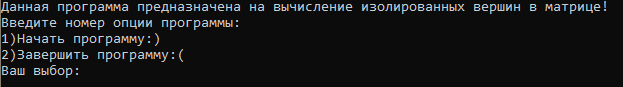
Тест 1. Запуск

Рисунок 6 – Тест 1

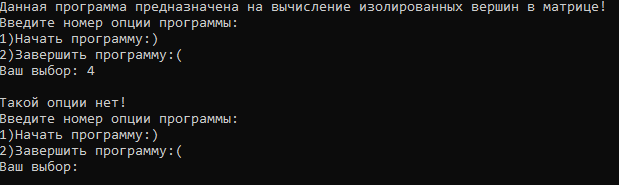
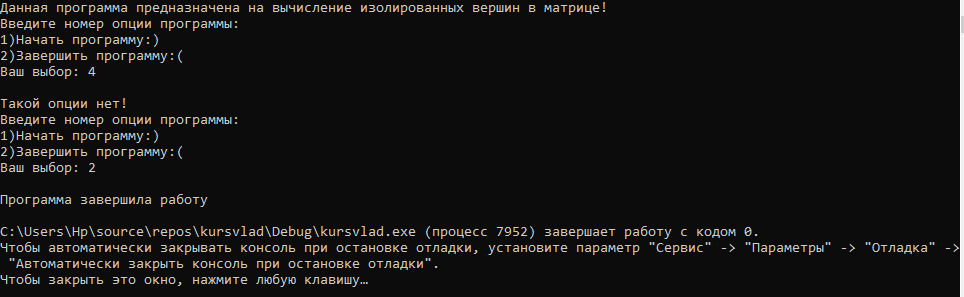
Тест 2. Проверка на не правильный выбор в меню

Рисунок 7 – Тест 2

Тест 3. Выход Рисунок 8 – Тест 3

Тест 4. Выбор “Начать программу”

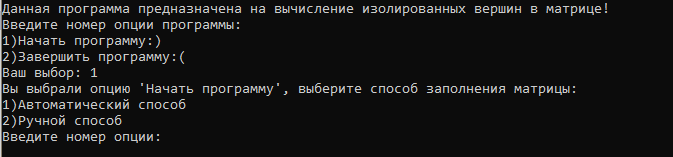


Рисунок 9 – Тест 4

Тест 5. Проверка на не правильный выбор в меню “формат заполнения”

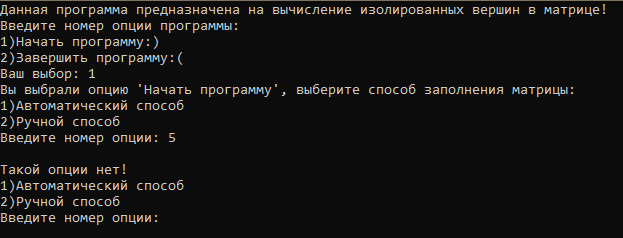


Рисунок 10 – Тест 5

Тест 6. Проверка автоматического способа и ввод данных

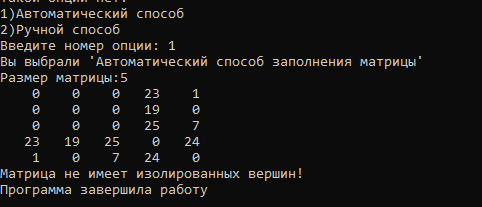


Рисунок 11 – Тест 6

Тест 7. Сохранение данных в файл

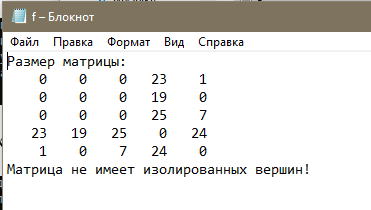


Рисунок 12 – Тест 7

Тест 8. Проверка ручного формата и его результаты на наличие изолированных вершин + сохранение в файл

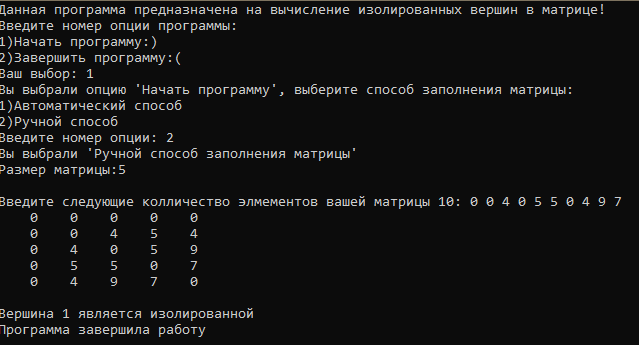


Рисунок 13 – Тест 8

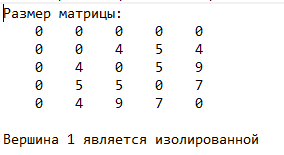


Рисунок 14 – Тест 8

Тест 9. Автоматический режим на матрице размером 3 на 3

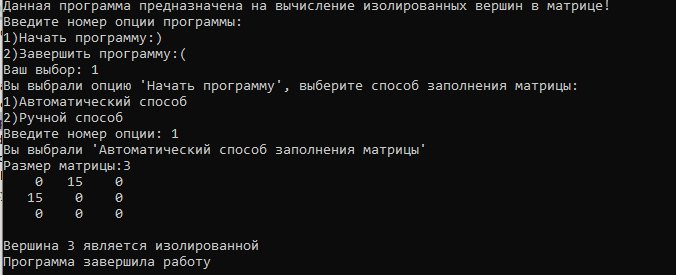


Рисунок 15 – Тест 9

В результате тестирования было выявлено, что программа успешно проверяет данные на соответствие необходимым требованиям.

**7.Ручной расчёт задачи**

Проведем проверку программы посредством ручных вычислений на примере Теста №9

Пробегаемся по первой строке нашей таблице смежности. Замечаем, что на второй позиции имеется значение “15”, следовательно, можно сделать вывод, что вершина под номером 1 не является изолированной. Точно так же пробегаемся по второй строчке. При пробеге 3-ей строки, вы видим, что она полностью заполнена “0”. Делаем вывод, что 3-ья вершина является изолированной.

**Заключение**

Таким образом, в процессе создания данного проекта разработана программа, реализующая алгоритм поиска изолированных вершин в графе в MicrosoftVisualStudio 2017.

При выполнении данной курсовой работы были получены навыки разработки программ и освоены приемы создания матриц смежности, основанных на теории орграфов. Приобретены навыки по осуществлению алгоритма поиска изолированных вершин в графа. Углублены знания языка программирования Cи.

Недостатком разработанной программы является примитивный пользовательский интерфейс. Потому что программа работает в консольном режиме, не добавляющем к сложности языка сложность программного оконного интерфейса.

Программа имеет небольшой, но достаточный для использования функционал возможностей.

**Список литературы**

1. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: Построение и анализ - М.: МЦНМО, 2001. - 960 с.

2. Кристофидес Н. «Теория графов. Алгоритмический подход» - Мир, 1978

3. Герберт Шилдт «Полный справочник по C++» - Вильямс, 2006

4. Уилсон Р. Введение в теорию графов. Пер. с анг. 1977. 208 с.

5. Харви Дейтел, Пол Дейтел. Как программировать на C/C++. 2009 г.

6. 3. Оре О. Графы и их применение: Пер. с англ. 1965. 176 с.

**Приложение A. Листинг программы**

**main**

#include"Header.h"

int main()

{

FILE \*f;

int \*\*mat;

int \*mass;

int m;

int number1, number2;

f = fopen("f.txt", "w");

setlocale(LC\_ALL, "Rus");

int r = 0;

int t = 0;

printf("Данная программа предназначена на вычисление изолированных вершин в матрице!");

while (r < 1) {

printf("\nВведите номер опции программы: ");

printf("\n1)Начать программу:)");

printf("\n2)Завершить программу:(");

printf("\nВашвыбор: ");

scanf\_s("%d", &number1);

switch (number1)

{

case 1:

printf("Вывыбралиопцию'Начатьпрограмму', выберитеспособзаполненияматрицы:");

r++;

while (t < 1) {

printf("\n1)Автоматический способ");

printf("\n2)Ручной способ");

printf("\nВведите номер опции: ");

scanf\_s("%d", &number2);

switch (number2) {

case 1:

t++;

printf("Вы выбрали 'Автоматический способ заполнения матрицы'");

fprintf(f, "Размер матрицы:");

fprintf(f, "\n");

printf("\nРазмерматрицы:");

scanf\_s(" %d", &m);

mat = (int\*\*)malloc(m \* sizeof(int\*));

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

mat[i] = (int\*)malloc(m \* sizeof(int));

}

mass = (int\*)malloc(m \* sizeof(int));

vvod1(m, mat);

vivod(f, m, mat, mass);

break;

case 2:

t++;

printf("Вы выбрали 'Ручной способ заполнения матрицы'");

fprintf(f, "Размер матрицы:");

fprintf(f, "\n");

printf("\nРазмерматрицы:");

scanf\_s(" %d", &m);

mat = (int\*\*)malloc(m \* sizeof(int\*));

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

mat[i] = (int\*)malloc(m \* sizeof(int));

}

mass = (int\*)malloc(m \* sizeof(int));

printf("\nВведите следующие колличествоэлмементов вашей матрицы %d: ", ((m\*m) - m) / 2);

vvod2(m, mat);

vivod(f, m, mat, mass);

break;

default:

printf("\nТакойопциинет!");

break;

}

}

case 2:

r++;

printf("\nПрограмма завершила работу");

break;

default:

printf("\nТакойопциинет!");

break;

}

}

getchar();

getchar();

fclose(f);

return 0;

}

**Header**

#pragmaonce

#pragmaonce

#define\_CRT\_SECURE\_NO\_WARNINGS

#define\_CTR\_NONSTDC\_NO\_WARNINGS

#include<stdio.h>

#include<conio.h>

#include<time.h>

#include<random>

#include<malloc.h>

#include<locale>

void vvod1(intm, int \*\*mat);

void vvod2(intm, int \*\*mat);

voidvivod(FILE \*f, intm, int \*\*mat, int \*mass);

**func**

#include"Header.h"

void vvod1(intm, int \*\*mat) {

srand(time(NULL));

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

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

if (i == j) {

mat[i][j] = 0;

}

if (i> j) {

mat[i][j] = rand() % 2;

if (mat[i][j] == 0)

mat[i][j] = mat[i][j];

else

mat[i][j] = 1 + rand() % 30;

}

mat[j][i] = mat[i][j];

}

}

}

void vvod2(intm, int \*\*mat) {

srand(time(NULL));

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

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

if (i == j) {

mat[i][j] = 0;

}

if (i> j) {

scanf\_s(" %d", &mat[i][j]);

}

mat[j][i] = mat[i][j];

}

}

}

voidvivod(FILE \*f, intm, int \*\*mat, int \*mass) {

for (inti = 0; i<m; i++) { //VIVOD

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

printf(" %4d", mat[i][j]);

fprintf(f, " %4d", mat[i][j]);

}

printf("\n");

fprintf(f, "\n");

}

;

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

intiz = 0;

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

if (mat[i][j] != 0) {

iz += 1;

}

}

mass[i] = iz;

}

int flag = 0;

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

if (mass[i] == 0) {

printf("\nВершина %d является изолированной", i + 1);

fprintf(f, "\nВершина %d является изолированной", i + 1);

flag = 1;

}

}

if (flag == 0) {

printf("Матрица не имеет изолированных вершин!");

fprintf(f, "Матрица не имеет изолированных вершин!");

}

free(mass);

free(mat);

}