Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра вычислительная техника

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

К курсовому проектированию

по курсу «Логика и основы алгоритмизации

в инженерных задачах»

на тему «Реализация алгоритма нахождения изолированных вершин в графе»

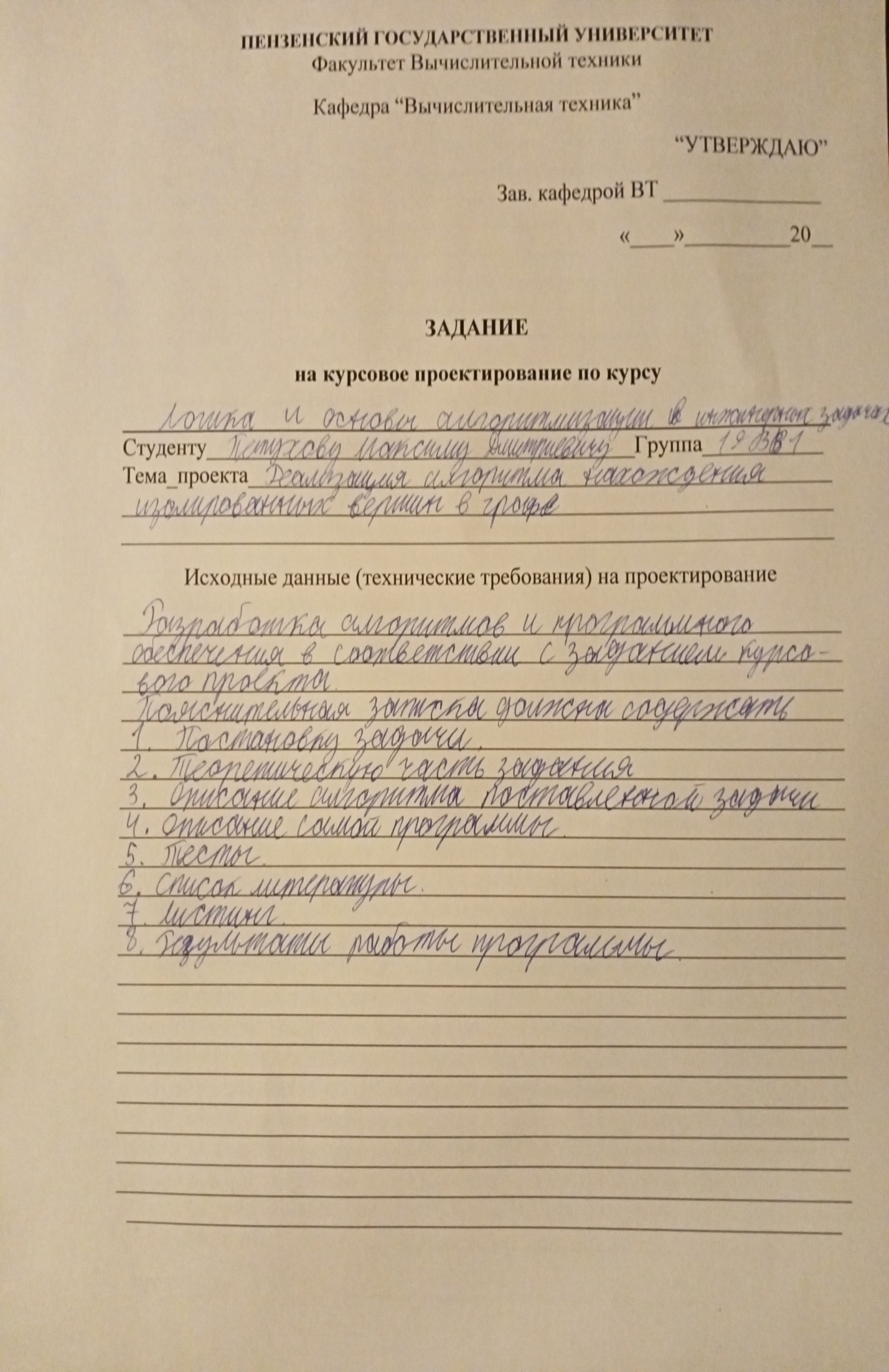
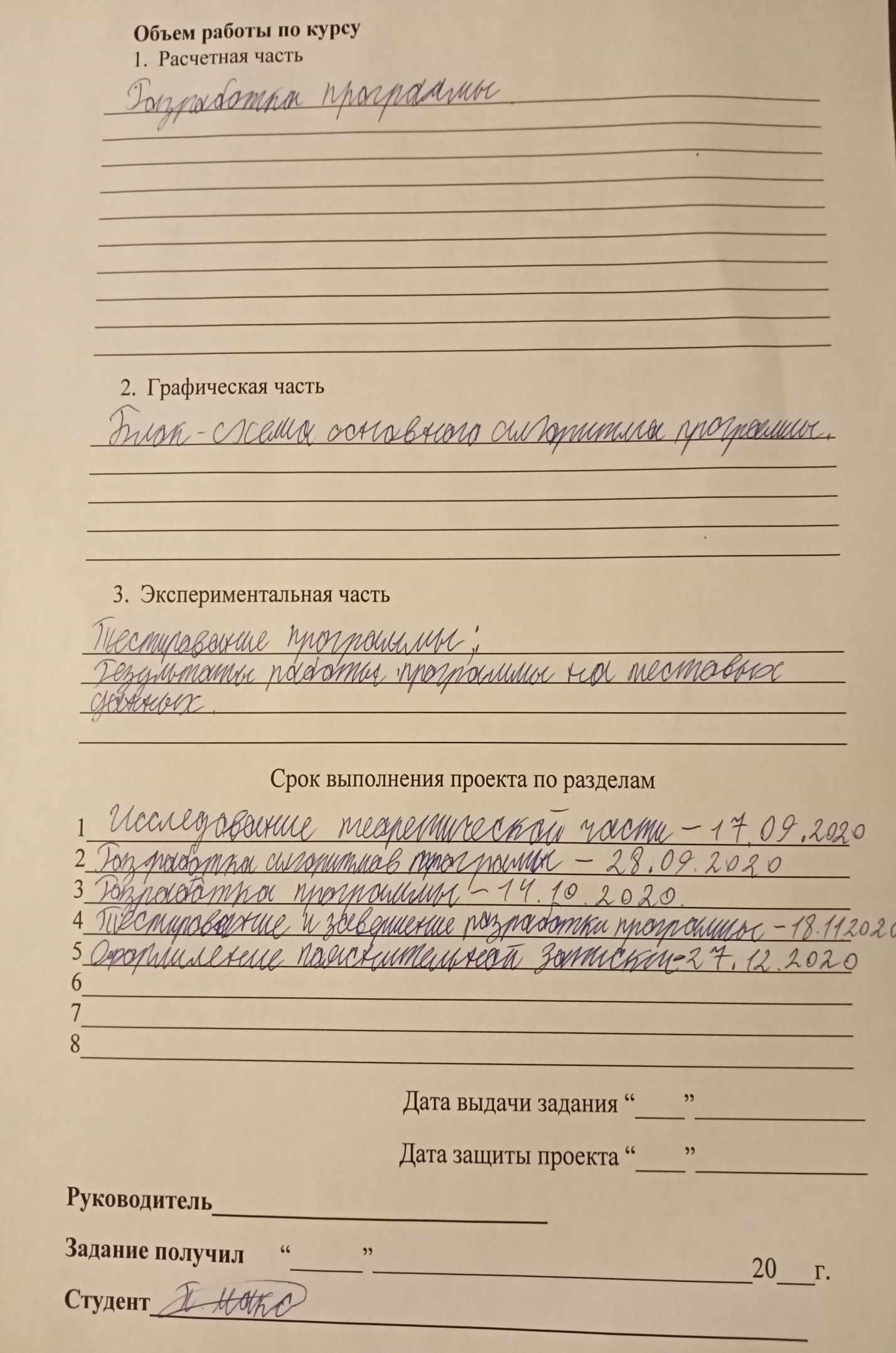
Выполнил студент группы 19ВВ1:

Петухов М. Д.

Принял:

Митрохин М. А.

Пенза 2020

** **

**Содержание**

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

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

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

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

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

4. Описание программы ...................................................................................... 12

5. Тестирование ................................................................................................... 16

Заключение ...................................................................................................... 20

Список литературы ......................................................................................... 21

Приложение A. Листинг программы ............................................................ 22

**Реферат**

Отчет 23 стр, 9 рисунков.

РЕАЛИЗАЦИЯ АЛГОРИТМА НАХОЖДЕНИЯ ИЗОЛИРОВАННЫХ ВЕРШИН В ГРАФЕ.

Цель исследования – разработка программы, находящей изолированные вершины в графе.

В программе используется алгоритм перебора вершин графа на наличие связей с другими его вершинами для поиска изолированных вершин. Установлено, что с помощью данного алгоритма можно обнаружить изолированные вершины или их отсутствие.

**Введение**

Изолированной называется вершина графа, которая инцидентна одной или нескольким петлям. Две вершины a и b называются смежными, если существует по крайней мере одно соединяющее их ребро. То есть изолированная вершина, это та вершина, которая не связана с другими вершинами

В качестве среды разработки мною была выбрана среда MicrosoftVisualStudio2010, 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 поиска изолированной вершины.

**1. Постановка задачи**

Требуется разработать программу, которая найдет изолированные вершины графа.

Исходный граф в программе должен задаваться матрицей смежности, причем при генерации данных должны быть предусмотрены граничные условия. Программа должна работать так, чтобы пользователь вводил количество вершин для генерации матрицы смежности. После обработки этих данных на экран должна выводиться матрица смежности графа. Необходимо предусмотреть различные исходы поиска, чтобы программа не выдавала ошибок и работала правильно.

Устройство ввода – клавиатура.

**2. Теоретическая часть задания**

**Изолированная вершина**– вершина, которая не имеет инцидентных ребер.

Пример: пусть https://pandia.ru/text/77/138/images/image007_13.gif – ориентированный граф, https://pandia.ru/text/77/138/images/image020_5.gif, https://pandia.ru/text/77/138/images/image021_5.gif, тогда https://pandia.ru/text/77/138/images/image013_10.gif*,*https://pandia.ru/text/77/138/images/image022_4.gif – изолированные вершины.

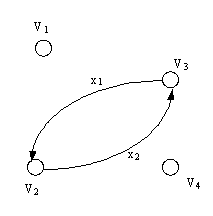


Рисунок 1 – Случайный граф с изолированными вершинами

**3. Описание алгоритма программы**

Для программной реализации алгоритма понадобиться два массива:

F (int) для хранения информации о изолированности вершин (0 или 1), G(int) будет хранить информацию о связанности вершин графа.

Сначала вводится с клавиатуры будущий размер графа в z (int) затем двумерному массиву G(int) присваивается размер z (int) строк и z (int) столбцов. Затем G(int) заполняется вручную или генератором случайных чисел 1- есть связь, 0 – нет. После G(int) выводится на экран.

Параллельно заполняется файл Itog.txt в котором после программы можно будет просмотреть итог работы программы.

После идет обход графа и запись изолированных вершин в массив F (int) 0 – изолированная 1 – нет.

Ниже представлен частичный псевдокод функции main()

**main()**

1. Инициализация G,F,i,j,z,k.

2. Получение z с клавиатуры

3. Присвоить массиву G размер z на z

4. Присвоить массиву F размер z

5. Получить k с клавиатуры

6. Если k = 1, то

7. для i=0 пока i<z делать i+1

8. для j=i пока j<z делать j+1

9. Запросить G[i][j] с клавиатуры

10. G[j][i]=G[i][j]

11. конец цикла

12. конец цикла

13. Иначе

14. для i=0 пока i<z делать i+1

15. для j=i пока j<z делать j+1

16. G[i][j]=rand()%2

17. G[j][i]=G[i][j]

18. конец цикла

19. G[i][i]=0

20. конец цикла

21. конец условия

22. k=0

23. Открыть Itog.txt для записи

24. для i=0 пока i<z делать i+1

25. для j=i пока j<z делать j+1

26. Вывод G на экран

27. Вывод G в файл Itog.txt

28. конец цикла

29. конец цикла

30. для i=0 пока i<z-1 делать i+1

31. для j=i+1 пока j<z делать j+1

32. Если G[i][j] = 1, то

33. F[i]=1

34. F[j]=1

35. конец условия

36. Если F[j]!=1 , то F[j]=0

37. конец условия

38. конец цикла

39. Если F[i]!=1 , то F[i]=0

40. конец условия

41. конец цикла

42. для i=0 пока i<z делать i+1

43. Если F[i]=0

44. Вывести на экран, что вершина i изолирована

45. Записать в Itog.txt, что вершина i изолирована

46. k+1

47. конец условия

48. Если k = 0, то

49. Вывести на экран, что нет изолированных вершин

50. Записать в Itog.txt, что нет изолированных вершин

51. конец условия

52. Закрыть файл Itog.txt

53. Освободить G

54. конец main ()

**4. Описание программы**

Для написания данной программы использован язык программирования Си.Язык программирования Си – универсальный язык программирования, который завоевал особую популярность у программистов, благодаря сочетанию возможностей языков программирования высокого и низкого уровней.

Проект был создан в виде консольного приложения Win32 (Visual C++).

Работа программы начинается с запроса генерации матрицы, на экран выводится запрос на количество вершин в графе. Затем выводится запрос на вид генерации, вручную или генератором. После, если пользователь выбрал сгенерировать вручную, то выводятся запросы на наличие связи между вершинами.

Printf(«Введите размер массива: «);

scanf(«%d»,&z);

printf(“\n”);

G = (int\*)malloc(z\*z \* sizeof(int));

F = (int\*)malloc(z \* sizeof(int));

printf(«Введите 1 если желаете вводить массив вручную, 0 если не желаете: «);

scanf(“%d”,&k);

printf(“\n”);

if (k==1)

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

{

for (j=I;j<z;j++)

{

printf(“Связь между %d и %d вершинами: “,I,j);

scanf(“%d”,(G+i\*z+j));

\*(G+j\*z+i)=\*(G+i\*z+j);

}

}

else

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

{

for (j=I;j<z;j++)

{

\*(G+i\*z+j)=rand()%2;

\*(G+j\*z+i)=\*(G+i\*z+j);

}

\*(G+i\*z+i)=0;

}

Далее матрица выводится на экран и параллельно записывается в файл

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

{

if (i==0)

{

printf(“ “);

fprintf(Itog, “ “);

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

{

printf(“%2d”,j);

fprintf(Itog, “%2d”,j);

}

printf(“\n\n”);

fprintf(Itog, “\n\n”);

}

printf(“%2d “,i);

fprintf(Itog, “%2d “,i);

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

{

printf(“%2d”,\*(G+i\*z+j));

fprintf(Itog, “%2d”, \*(G+i\*z+j));

}

printf(“\n”);

fprintf(Itog, “\n”);

}

printf(“\n”);

fprintf(Itog, “\n”);

Далее нахождение изолированных вершин

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

{

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

{

if (\*(G+i\*z+j)==1)

{

F[i]=1;

F[j]=1;

}

if (F[j]!=1) F[j]=0;

}

if (F[i]!=1) F[i]=0;

}

Ну а под конец вывод на экран изолированных вершин, если таких нет, то сообщение о том, что изолированных вершин нет.

For (i=0;i<z;i++)

if (F[i]==0)

{

printf(«Вершина %d является изолированной\n», i);

fprintf(Itog, «Вершина %d является изолированной\n», i);

k++;

}

if (k==0)

{

printf(«Изолированных вершин нет»);

fprintf(Itog, «Изолированных вершин нет»);

}

Ниже можно увидеть оформление начального запроса и дальнейшие действия с ним.

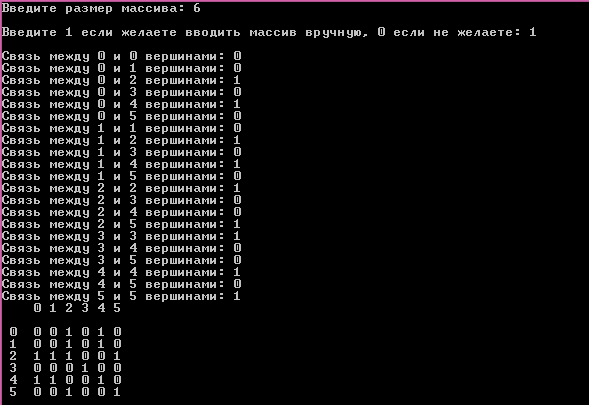


Рисунок 2 – Генерация матрицы вручную

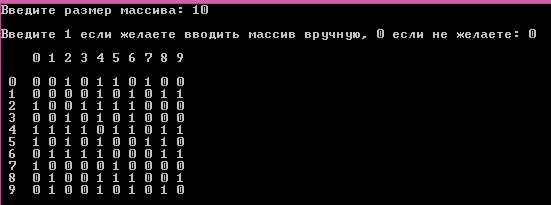


Рисунок 3 – Генерация матрицы случайно

Далее выводится номера всех изолированных вершин (рис. 4).

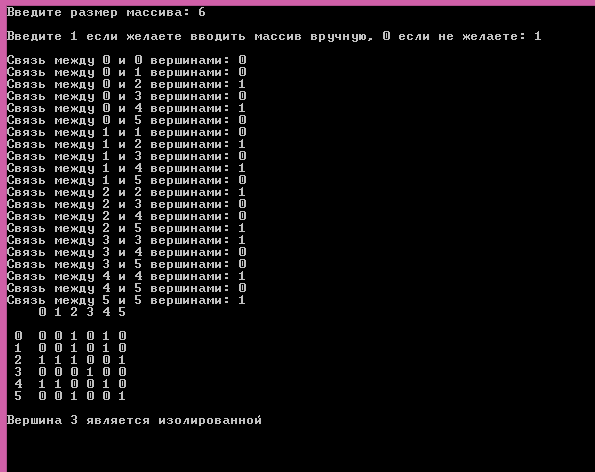


Рисунок 4 – Вывод изолированных вершин

Или если изолированных вершин нет (рис. 5).

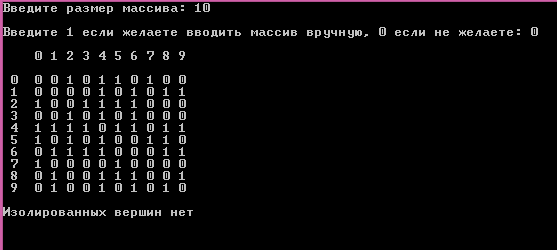


Рисунок 5 – Изолированных вершин нет

**5. Тестирование**

Среда разработки MicrosoftVisualStudio 2010 предоставляет все средства, необходимые при разработке и отладке 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.

Ниже продемонстрирован результат тестирования программы при вводе пользователем различных количеств вершин.

|  |  |  |
| --- | --- | --- |
| Описание теста | Ожидаемый результат | Полученный результат |
| Запуск программы | Вывод сообщения о выборе: сгенерировать матрицу вручную или случайно | Верно |
| Ввод матрицы вручную | Вывод матрицы с введенными значениями | Верно |
| Генерация матрицы случайно | Вывод случайно сгенерированной матрицы | Верно |
| Определение изолированных вершин | Вывод на экран какие вершины являются изолированными  Если их нет, то вывод информации об их отсутствии | Верно |
| Сохранение результата в файл | Запись в файл матицы и информации о изолированных вершинах | Верно |

Таблица 1 – Описание поведения программы при тестировании

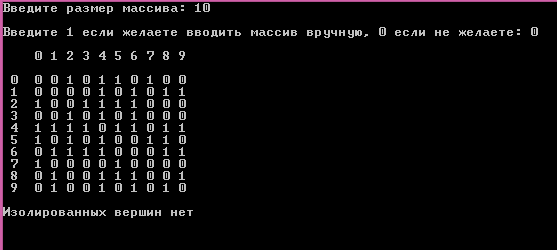


Рисунок 6 – Отсутствие изолированных вершин

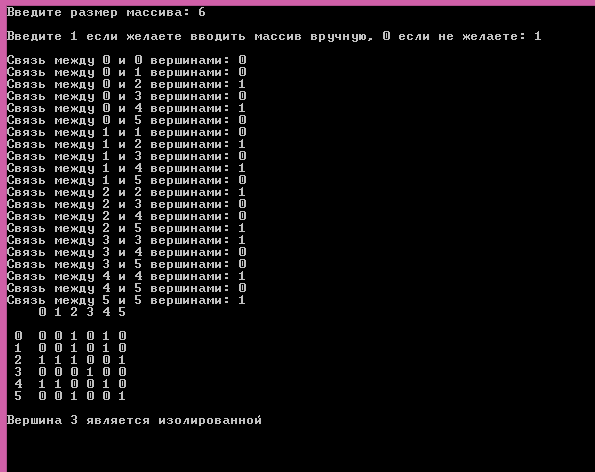


Рисунок 7 – Одна изолированная вершина

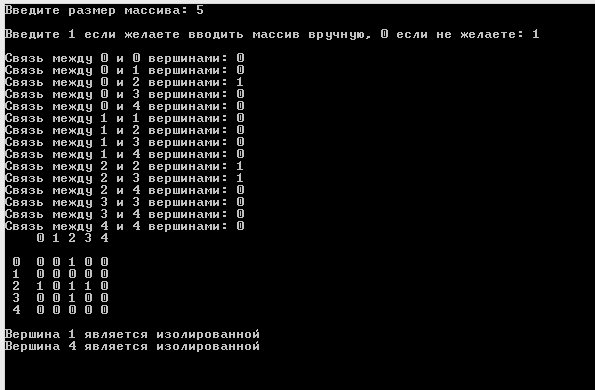


Рисунок 8 – Несколько изолированных вершин

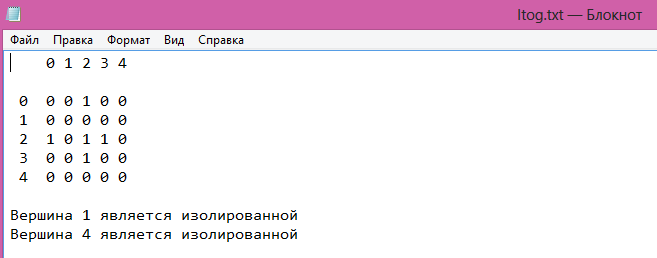


Рисунок 9 – Запись результатов работы программы в файл

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

**Заключение**

Таким образом, в процессе создания данного проекта разработана

программа, реализующая алгоритм поиска изолированных вершин графа в MicrosoftVisualStudio 2010.

При выполнении данной курсовой работы были получены навыки разработки программ и освоены приемы создания матриц смежностей, основанных на теории графов. Приобретены навыки по осуществлению алгоритма поиска изолированных вершин. 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. Кристофидес Н. «Теория графов. Алгоритмический подход» - Мир, 1978

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

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

4. https://pandia.ru/text/77/138/392.php

**Приложение А.**

**Листинг программы.**

#include "stdafx.h"

#include <conio.h>

#include <stdlib.h>

#include <locale.h>

#include <malloc.h>

#include <time.h>

int main()

{

setlocale(LC\_ALL, "RUS");

srand(time(0));

int \*G,\*F;

int i,j,z,k;

printf("Введите размер массива: ");

scanf("%d",&z);

printf("\n");

G = (int\*)malloc(z\*z \* sizeof(int));

F = (int\*)malloc(z \* sizeof(int));

printf("Введите 1 если желаете вводить массив вручную, 0 если не желаете: ");

scanf("%d",&k);

printf("\n");

if (k==1)

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

{

for (j=i;j<z;j++)

{

printf("Связь между %d и %d вершинами: ",i,j);

scanf("%d",(G+i\*z+j));

\*(G+j\*z+i)=\*(G+i\*z+j);

}

}

else

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

{

for (j=i;j<z;j++)

{

\*(G+i\*z+j)=rand()%2;

\*(G+j\*z+i)=\*(G+i\*z+j);

}

\*(G+i\*z+i)=0;

}

k=0;

FILE \*Itog;

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

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

{

if (i==0)

{

printf(" ");

fprintf(Itog, " ");

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

{

printf("%2d",j);

fprintf(Itog, "%2d",j);

}

printf("\n\n");

fprintf(Itog, "\n\n");

}

printf("%2d ",i);

fprintf(Itog, "%2d ",i);

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

{

printf("%2d",\*(G+i\*z+j));

fprintf(Itog, "%2d", \*(G+i\*z+j));

}

printf("\n");

fprintf(Itog, "\n");

}

printf("\n");

fprintf(Itog, "\n");

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

{

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

{

if (\*(G+i\*z+j)==1)

{

F[i]=1;

F[j]=1;

}

if (F[j]!=1) F[j]=0;

}

if (F[i]!=1) F[i]=0;

}

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

if (F[i]==0)

{

printf("Вершина %d является изолированной\n", i);

fprintf(Itog, "Вершина %d является изолированной\n", i);

k++;

}

if (k==0)

{

printf("Изолированных вершин нет");

fprintf(Itog, "Изолированных вершин нет");

}

fclose(Itog);

free(G);

getch();

return 0;

}