



**Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ

Информатика и системы управления

КАФЕДРА

Программное обеспечение ЭВМ и информационные технологии

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8
«ОБРАБОТКА ГРАФОВ»
Вариант 6

Студент

Кладницкий А. Б.

Преподаватель

Группа

ИУ7 – 32Б

2022 г.

1. Описание условия задачи

Обработать графовую структуру в соответствии с указанным вариантом задания. Обосновать выбор необходимого алгоритма и выбор структуры для представления графов. Ввод данных – на усмотрение программиста. Результат выдать в графической форме.

2. ТЗ

Найти минимальное (по количеству ребер) подмножество ребер, удаление которых превращает заданный связный граф в несвязный.

1. Исходные данные:

- Имя файла
- Файл, содержащий в себе представление графовой структуры (в первой строке – кол-во вершин, в последующих связи вершин – номера 2-х вершин через пробел)

2. Результирующие данные:

- Визуализированный граф
- Найденные ребра (красный цвет на графе и печать в консоль)

3. Задача программы:

- Поиск минимального подмножества ребер, удаление которых превращает связный граф в несвязный

4. Способ обращения к программе:

- Запуск через терминал, имя файла передается как параметр (./app.exe filename)

5. Возможные ошибки:

- Ошибка выделения динамической памяти
- Ошибка открытия файла
- Ошибка неверной передачи аргументов
- Ошибка чтения из файла

3. Описание внутренних структур данных

Тип данных graph_t:

```
typedef struct graph
{
    int size;
    int **mtr;
} graph_t;
```

Тип содержит в себе кол-во вершин в графе и матрицу смежности для них.

Тип данных edge_t:

```
typedef struct edge
{
    int beg;
    int end;
} edge_t;
```

Тип содержит в себе номера двух вершин ребра графа.

Тип данных node_t:

```
typedef struct node node_t;
struct node
{
    edge_t *data;
    node_t *next;
};
```

Узел линейного односвязного списка, хранящего в себе элемент типа edge_t (ребро графа).

4. Алгоритм

1. Считывается граф из файла
2. Производится полный перебор ребер и ищется минимальное подмножество, удовлетворяющее условию
3. Проверка связности осуществляется модифицированным поиском в глубину.

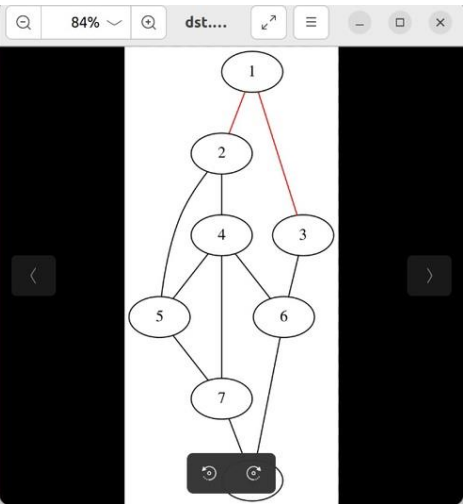
5. Набор тестов

	Описание	Входные данные	Результат
1.	Запуск без аргумента	./app.exe	ERROR_ARGS
2.	Ошибка выделения динамической памяти	???	ERROR_ALLOCATE
3.	Неверное имя файла	Имя несуществующего файла	ERROR_BAD_FILE
4.	Пустой файл	-	ERROR_READ
5.	Некорректный файл	q qw wqe erb poi	ERROR_READ
6.	Несвязный граф	файл	«Граф несвязный», граф
7.	Граф, в котором 1 вершина имеет 1 ребро	файл	номер ребра, граф
8.	«Обычный» граф	файл	номера ребер, граф
9.	Граф с полной связностью	файл	номера ребер, граф

6. Пример работы

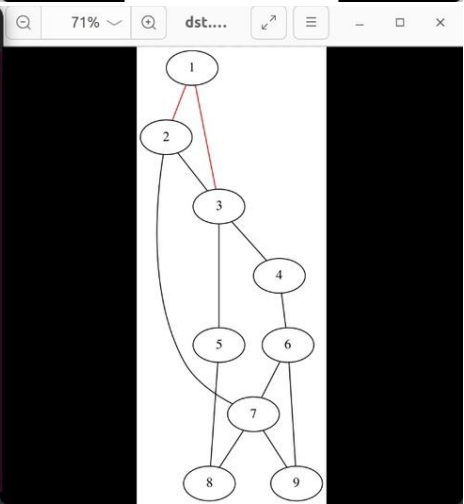
packled@packled-honor: ~/Рабочий стол/ТаDS/Lab8

```
packled@packled-honor:~/Рабочий стол/ТаDS/Lab8$ ./app.exe ./test/test1.txt
1-2
1-3
packled@packled-honor:~/Рабочий стол/ТаDS/Lab8$
```



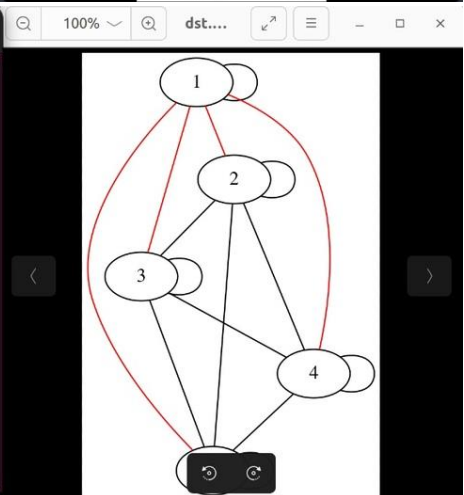
packled@packled-honor: ~/Рабочий стол/ТаDS/Lab8

```
packled@packled-honor:~/Рабочий стол/ТаDS/Lab8$ ./app.exe ./test/test3.txt
1-2
1-3
packled@packled-honor:~/Рабочий стол/ТаDS/Lab8$
```



packled@packled-honor: ~/Рабочий стол/ТаDS/Lab8

```
packled@packled-honor:~/Рабочий стол/ТаDS/Lab8$ ./app.exe ./test/test5.txt
1-2
1-3
1-4
1-5
packled@packled-honor:~/Рабочий стол/ТаDS/Lab8$
```



7. Вывод

Изучил способы обработки и хранения графовых структур. Ознакомился с алгоритмом поиска в глубину и другими алгоритмами.

Разработанный алгоритм может быть применен, например, для оценки возможности перекрытия дорог или участков линий метро.

8. Контрольные вопросы

1. Что такое граф?

Граф – это конечное множество вершин и ребер, соединяющих их.

2. Как представляются графы в памяти?

2 основных способа:

- Матрица смежности (в элементе i, j 1, если есть ребро, 0 иначе)
- Список смежностей (для каждой вершины в виде списка хранятся вершины, смежные с данной)

3. Какие операции возможны над графами?

Основные операции:

- Поиск кратчайшего пути до одной вершины
- Поиск кратчайшего пути ко всем другим вершинам
- Поиск кратчайших путей между всеми вершинами
- Поиск эйлера пути
- Поиск гамильтонов пути

4. Какие способы обхода графов существуют?

В ширину и в глубину

В глубину – просмотр ближайших смежных вершин, пока есть новые вершины

В ширину – одновременный просмотр всех «соседей»

5. Где используются графовые структуры?

В основном в математическом моделировании

6. Какие пути в графе Вы знаете?

Эйлеров и гамильтонов

Эйлеров путь – путь, проходящий через каждое ребро графа только один раз

Гамильтонов путь – путь, проходящий через каждую вершину графа только один раз

7. Что такое каркасы графа?

Каркас графа (остов графа, остовое дерево) – дерево, содержащее в себе все вершины графа