**МГТУ им. Н.Э.Баумана**

**Дисциплина “Типы и структуры данных”**

**Отчет по Лабораторной работе №7 по теме “** Графы **”.**

Работу выполнил:

Студент: Зейналов Зейнал

ИУ7-31Б

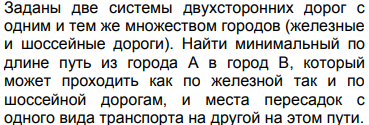
Варинат 14

Москва, 2018.

Работа № 7

***Цель работы*:** реализовать алгоритмы обработки графовых структур: поиск различных путей, проверка связности, построение остовых деревьев минимальной стоимости. 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.

Описание условия задачи



Описание ТЗ

Общая концепция системы

Программа предоставляет пользователю возможность увидеть граф, описанный в графовой структуре, определить минимальный вес пути из введенной вершины, в вершину, в которую ищем тот самый путь (так же вводится пользователем). В качестве результата программа предоставляет пользователю информацию о минимальной цене пути, если таковой существует, а так же информацию о вершинах, участвующих в этом пути, с информацией о том, можно ли на конкретной вершине совершить пересадку с шоссейной дороги на железную.

Входные данные

## Графовая структура.

Файл, содержащий графовую структуры выглядит так:

M M (количество вершин в графе)

First (шоссейные дороги)

1 2 5 ( откуда, куда, вес)

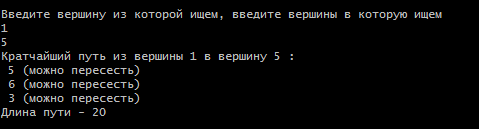
...

Second (железные дороги)

1 2 5

…

Ввод вершин поиска и результат.



### Требования к функциональным характеристикам

### 

## Способ обращения к программе

Программа представляет собой файл app.exe. Запускается в консоли. Для запуска достаточно команды ./app.exe in.txt ( файл, содержащий графовую структуру, пример этой структуры будет приведен ниже). Если файл app.exe отсутствует можно собрать его с помощью утилиты make .

Аварийные ситуации:

* Введена несуществующая вершина поиска  
  Программа выведет соответствующую информацию
* Введена не вершина  
  Программа выведет соответствующую информацию

Описание структур данных

В ходе выполнения работы в качестве структуры представления графовой структуры, была выбрана матрица. Это обусловлено тем, что выполнять проверки на возможность пересадки с одной дороги на другую становится проще, а так же занимаемая память в данном виде и в виде списков – практически одинакова.



Память под матрицы выделяется динамически.



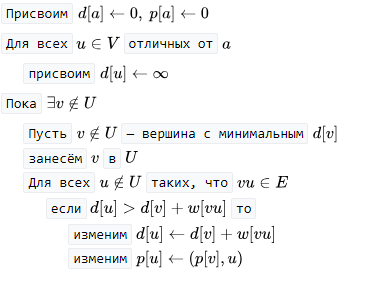
Массив – содержащий вершины пересечения

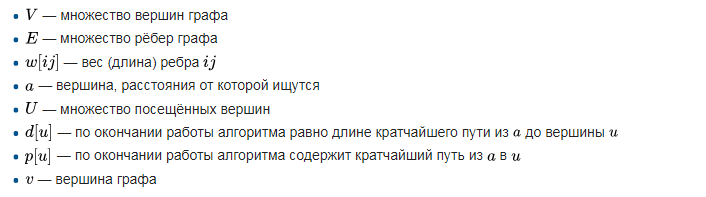
Описание алгоритма

На вход поступает файл с графовой структурой, на основе которой формируются 2 матрицы, благодаря которым формируется изображения графа.

На основе двух матриц формируется новая матрица, с минимальными метками, для точного определения минимальной длины.

Данная матрица , с введенными вершинами поиска отправляется в алгоритм Флойда, проделанным p-ое количество раз, который превращается в алгоритм Дейкстры (p – количество вершин в графе. Проделывается для нахождения до каждой из вершин).





Сложность Алгоритма:

В простейшем случае, когда для поиска вершины с минимальным *d*[*v*] просматривается всё множество вершин, а для хранения величин *d* используется массив, время работы алгоритма есть {\displaystyle O(n^{2})} .

Основной цикл выполняется порядка *n* раз, в каждом из них на нахождение минимума тратится порядка *n* операций. На циклы по соседям каждой посещаемой вершины тратится количество операций, пропорциональное количеству рёбер *m* (поскольку каждое ребро встречается в этих циклах ровно дважды и требует константное число операций). Таким образом, общее время работы алгоритма {\displaystyle O(n^{2}+m)} , но, так как {\displaystyle m\leq n(n-1)} , оно составляет {\displaystyle O(n^{2})} .

Реальное применение данной задачи.

В повседневной жизни каждому человеку ежедневно нужно добраться до какой то точки назначения с минимальным затратами времени. Если предположить что каждая метка, заданного графа – количество времени, которое человек потратит для того, чтобы добраться в ту или иную точку, то можно применить реализованную задачу. Здесь же подразумевается возможность добираться на железных дорогах и на шоссейных с пересадками.

Выводы по проделанной работе

В результате выполнения данной работы, программист ознакомился с реализацией графовых структур, а так же осуществлять поиск минимальной цены между необходимыми вершинами, выводить графовую структуру на экран, для убеждения в верности решения данной задачи. Реализация данной задачи матрицами является более наглядной и простой, нежели реализация списками. Так как реализация со списками кроме информативной части обладает указателем на следующий элемент, что в свою очередь добавляет к каждой вершине 8 байт памяти, подтверждает проделанный выше вывод.

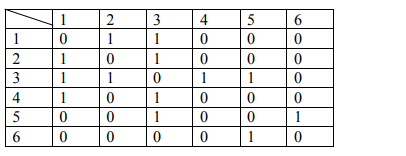
# Ответы на вопросы

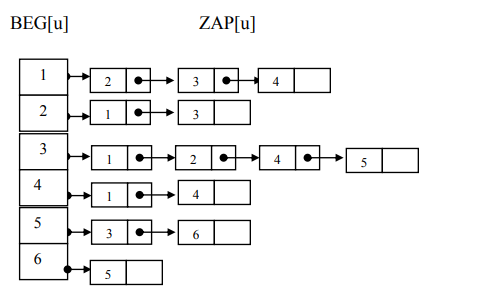
1. Что такое граф?

**Граф** — абстрактный математический объект, представляющий собой множество вершин **графа** и набор рёбер, то есть соединений между парами вершин.

1. Как представляются графы в памяти?

В виде матриц достижимости и в виде списков





1. Какие операции возможны над графами?

Допустимые операции:

\* Нахождение кратчайшего пути из заданных вершин.

\* Определение, является ли граф связным или несвязным.

\* Построение остовых деревьев минимальной стоимости

1. Какие способы обхода графов существуют?

Существует 2 вида обхода графов:

\* В ширину

\* В глубину

5. Где используются графовые структуры?

Графовые структуры используются для решения задач различного рода

6. Какие пути в графе Вы знаете?

\* Гамильтонов путь

\* Эйлеров путь

7. Что такое каркасы графа?

Остовное дерево или **каркас** (скелет) **графа** – это подграф, который :

1) содержит все вершины **графа**

2) является деревом