



**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 **«Графы»**

Студент Городский Юрий Николаевич

Группа ИУ7 – 32Б

2023г.

ОПИСАНИЕ УСЛОВИЯ ЗАДАЧИ

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

ОПИСАНИЕ ТЕХНИЧЕСКОГО ЗАДАНИЯ

Найти все вершины графа, к которым от заданной вершины можно добраться по пути не длиннее A .

Входные данные:

1. **Номер команды:** целое число в диапазоне от 0 до 3.
2. **Файл с данными:** Число элементов, id связанных узлов, веса связей

Выходные данные:

1. Сообщение об ошибке
2. Граф в виде изображения
3. Список доступных узлов

Обращение к программе:

Запуск через терминал

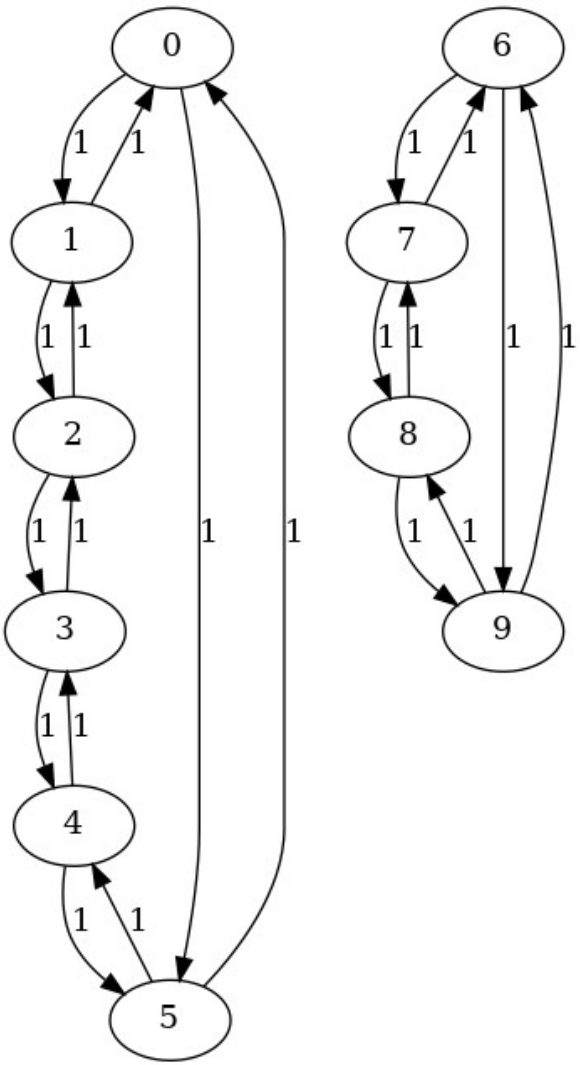
Аварийные ситуации:

1. Неверная команда
2. Ошибка при работе с файлом
3. Ошибка при вводе из файла
4. Ошибка при вводе данных при поиске доступных узлов

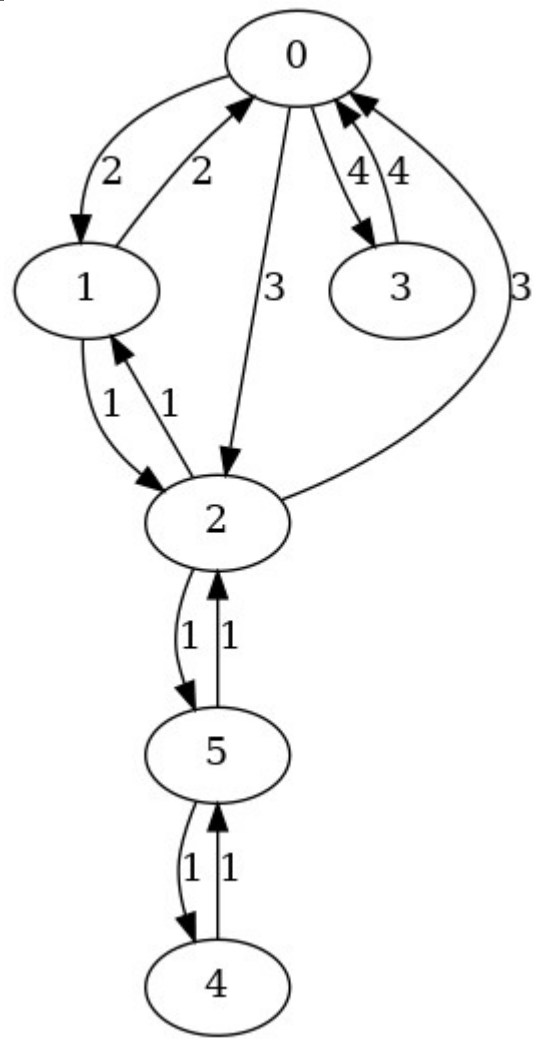
ОПИСАНИЕ СТРУКТУРЫ ДАННЫХ

```
// Узел графа
typedef struct graph_type
{
    size_t id; // Идентификатор
    struct graph_type **connections; // Связанные узлы
    int *weights; // Стоимости связей
    size_t n; // Количество связей
    bool visited; // Посещен
} graph_t;
```

ПРИМЕР РАБОТЫ ПРОГРАММЫ

Входные данные	Выходные данные
10 0 1 1 1 2 1 2 3 1 3 4 1 4 5 1 5 0 1 6 7 1 7 8 1 8 9 1 9 6 1	

6
0 1 2
0 2 3
0 3 4
1 2 1
2 5 1
5 4 1



Поиск доступных узлов

10
0 1 1
1 2 1
2 3 1
3 4 1
4 5 1
5 0 1
6 7 1
7 8 1
8 9 1
9 6 1

Идентификатор начального узла: 0
Максимальная стоимость пути: 2
1 2 5 4
...
Идентификатор начального узла: 6
Максимальная стоимость пути: 2
7 8 9

НАБОР ТЕСТОВ

Файлы в тестах		
1.txt	2.txt	a.txt
10 0 1 1 1 2 1 2 3 1 3 4 1 4 5 1 5 0 1 6 7 1 7 8 1 8 9 1 9 6 1	a	Не существует
Описание	Пользовательский ввод	Результат
Некорректный ввод команды	-1 4 a	Неверная команда
Файл не существует	1 a.txt	Ошибка работы с файлом
Файл с ошибкой	1 2.txt	Ошибка ввода
Ошибка при вводе id	1 1.txt 3 a 1	Ошибка ввода
Id за пределами	1 1.txt	Ошибка

	3 11 1	диапазона данных
Ошибка при вводе максимальной длины пути	1 1.txt 3 1 a	Ошибка ввода

РАЗРАБОТАННАЯ ЗАДАЧА

Логистическая фирма производит перевозки между несколькими городами, между которыми существуют платные и бесплатные дороги, необходимо составить оптимальные маршруты при перевозке товаров между городами, в одном пути можно посетить несколько городов.

Вывод

Для решения задачи был выбран способ хранения графа в виде списка смежностей. Плюсы данного метода заключаются в экономии памяти, в случае если количество связей меньше количества вершин и возможности легко построить граф в формате DOT.

Использовался алгоритм поиска в глубину, т. к. он легко реализуется, а для поиска всех доступных вершин необходимо обойти все ближайшие вершины, до которых путь меньше заданного.

Графы полезны при решении задач о путях или об иных связях и зависимостях. Они помогают при отображении и визуализации связей, что иногда является ключевым для быстрого решения задач.

Контрольные вопросы

1. Что такое граф?

Граф – это конечное множество вершин и ребер, соединяющих их, $G = \langle V, E \rangle$, где V – конечное непустое множество вершин; E – множество ребер (пар вершин).

Если пары E (ребра) имеют направление, то граф называется ориентированным (*орграф*), если иначе - неориентированный (*неорграф*). Если в пары E входят только различные вершины, то в графе нет петель. Если ребро графа имеет вес, то граф называется *взвешенным*.

Неорграф называется *связным*, если существует путь из каждой вершины в любую другую.

2. Как представляются графы в памяти?

Граф в памяти представляется в виде матрицы смежности или списка смежности.

Матрица смежности $B(n \times n)$ – элемент $b[i, j] = 1$, если существует ребро, связывающее вершины i и j , и $= 0$, если ребра не существует.

Список смежностей содержит для каждой вершины из множества вершин V список тех вершин, которые непосредственно связаны с ней. Входы в списки смежностей могут храниться в отдельной таблице, либо же каждая вершина может хранить свой список смежностей.

3. Какие операции возможны над графами?

- поиск кратчайшего пути от одной вершины к другой (если он есть);
- поиск кратчайшего пути от одной вершины ко всем другим;
- поиск кратчайших путей между всеми вершинами;
- поиск эйлера пути (если он есть);
- поиск гамильтонова пути (если он есть).

4. Какие способы обхода графов существуют?

Обход в ширину (BFS – Breadth First Search) - обработка вершины V осуществляется путём просмотра сразу всех «новых» соседей этой вершины, которые последовательно заносятся в очередь просмотра.

Обход в глубину (DFS – Depth First Search) - начиная с некоторой вершины v_0 , ищется ближайшая смежная ей вершина v , для которой в свою очередь осуществляется поиск в глубину до тех пор, пока не встретится ранее просмотренная вершина, или не закончится список смежности вершины v (то есть вершина полностью обработана). Если нет новых вершин, смежных с v , то вершина v считается использованной, идет возврат в вершину, из которой попали в вершину v , и процесс продолжается до тех пор, пока не получим $v = v_0$. При просмотре используется стек.

5. Где используются графовые структуры?

Графовые структуры могут использоваться в задачах, где между элементами могут быть установлены произвольные связи. Распространенное применение — решение задач о путях.

6. Какие пути в графе Вы знаете?

Эйлеровый путь - путь в графе, проходящий через каждое ребро ровно один раз. Если путь проходит по некоторым вершинам несколько раз – он называется *непростым*, иначе – *простым*.

Гамильтонов путь - путь, проходящий через каждую вершину ровно один раз.

7. Что такое каркасы графа?

Каркас графа – дерево, в которое входят все вершины графа, и некоторые (не обязательно все) его рёбра. Для построения каркасов графа используются алгоритмы Крускала и Прима.