


Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

	«Московский государственный технический университет имени Н.Э. Баумана» (МГТУ им. Н.Э. Баумана)
---	--

ФАКУЛЬТЕТ _____ ИУ _____

КАФЕДРА _____ ИУ7 _____

Отчет

по лабораторной работе № 6

Дисциплина: Типы и Структуры Данных

Название лабораторной работы: Графы

Студент гр. **ИУ7-34Б Малышев Илья Николаевич**

(Подпись, дата) (И.О. Фамилия)

Преподаватель **Силантьева Александра Васильевна**

(Подпись, дата) (И.О. Фамилия)

Москва, 2021

Цель работы

Реализовать алгоритмы обработки графовых структур: поиск различных путей, проверку связности, построение остовых деревьев минимальной стоимости.

Условия задачи

Задана система двусторонних дорог. Определить, можно ли, построив еще три новые дороги, из заданного города добраться до каждого из остальных городов, проезжая расстояние не более T единиц.

Техническое задание

Входные данные:

- Файл с записанным заранее графом;
- Запрашиваемые целые числа, необходимые для выполнения программы.

Выходные данные:

- Текстовые файлы с представлениями графов на языке dot;
- png-файлы с графическими представлениями графов.

Задача программы:

Программа определяет, возможно ли попасть из заданной вершины графа во все остальные за заданное количество ходов, при условии, что граф может быть добавлено не более трех ребер.

Способ обращения к программе:

Программа является консольной. После запуска исполняемого файла из консоли, программа выведет справку и приглашение на ввод.

Аварийные ситуации:

- Неверный ввод целого числа;
- Отсутствие требуемых для работы программы текстовых файлов;
- «Испорченность» переданных текстовых файлов.

Описание алгоритма:

1. Программа считывает граф из файла;
2. Программа генерирует svg исходного графа;
3. Программа проверяет выполнение условия для заданного графа;
4. Программа генерирует svg графа-решения, если такой есть.

Структуры данных:

graph_t – структура, фактически представляющая целочисленную квадратную матрицу, однако в программе используется исключительно для хранения матриц смежности для графов. Так как никакой иной информации о графах хранить не потребовалось, структура была названа по полностью определяемому ей объекту.

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

Тесты

#	Ошибка	Пример
1	Неправильный ввод целого числа	q
2	Выход за пределы множества вершин графа	-1
3	Ввод неверного числа ходов	0, -1
4	Отсутствие переданного файла	nofile.txt
5	Испорченность переданного текстового файла	broken.txt

Оценка эффективности

Измерения эффективности способов сложения будут производиться в единицах измерения – тактах процессора.

Таблица эффективности программы по памяти и по скорости выполнения поставленной задачи.

Количество узлов графа	Память (в байтах), необходимая для хранения графа	Время (в тактах процессора), затраченное на выполнение поставленной задачи
5	156	362712
10	496	28364541
15	1036	2147483647

Вывод

Оценочное время работы используемого алгоритма $O(n^8)$. Такая высокая сложность обусловлена необходимостью нахождения трех дополнительных ребер (что выполняется не всегда), только на этапе генерации этих значений алгоритм уже достигает сложности $O(n^6)$. Для реализации функции проверки графа по условию, использовался алгоритм Дейкстры со сложностью $O(n^2)$. Алгоритм Беллмана-Форда не был использован, так как его функционал избыточен, в поставленной задаче не рассматриваются отрицательные переходы, поэтому нет смысла перенагружать функционалом вспомогательную функцию. Относительно алгоритма Флойда – Уоршелла можно сказать то же: из-за перенагруженности повышается сложность, что, в нашей ситуации, критично.

Контрольные вопросы

1. Что такое граф?

Граф – конечное множество вершин и соединяющих их ребер; $G = (V, E)$. Если пары E (ребра) имеют направление, то граф называется ориентированным; если ребро имеет вес, то граф называется взвешенным.

2. Как представляются графы в памяти?

С помощью матрицы смежности или списков смежности.

3. Какие операции возможны над графами?

Обход вершин, поиск различных путей, исключение и включение вершин.

4. Какие способы обхода графов существуют?

Обход в ширину (BFS – Breadth First Search), обход в глубину (DFS – Depth First Search).

5. Где используются графовые структуры?

Графовые структуры могут использоваться в задачах, в которых между элементами могут быть установлены произвольные связи, необязательно иерархические.

6. Какие пути в графе Вы знаете?

Эйлеров путь, простой путь, сложный путь, гамильтонов путь.

7. Что такое каркасы графа?

Каркас графа – дерево, в которое входят все вершины графа, и некоторые (необязательно все) его рёбра.