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

Группа ИУ7-35Б

Тема работы **Графы**

Студент Баранов Николай Алексеевич

Преподаватель Силантьева Александра Васильевна

2022 г.

Цель работы: реализовать алгоритмы работы графовых структур: поиск различных путей, проверку связности, построение остовных деревьев минимальной стоимости.

1. Описание условия задачи

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

2. Описание ТЗ

2.1. Описание исходных данных и результатов.

При чтении имени файла программа принимает на вход произвольную непустую строку.

При чтении числа программа считывает число в указанном диапазоне.

Выбор действия происходит в меню, где вводится число – номер действия.

При добавлении ребра ребро просто записывается в граф. Если такое ребро уже было, оно заменяется. Нельзя создать ребро из вершины в себя.

Граф при необходимости сохраняется в файле в виде графа на языке описания графов DOT. Далее с помощью утилиты graphviz граф можно визуализировать.

2.2. Описание задачи, реализуемой программой.

Программа позволяет добавлять и удалять ребра из графа (длины ребер неотрицательные), найти все вершины, от которых можно дойти до заданной по пути не длиннее A, а также сохранить граф в файл.

2.3. Способ обращения к программе.

Запуск программы:

./app.exe

В этом случае программа начнёт выполнять свою задачу. Ввод осуществляется вручную с клавиатуры.

2.4. Описание возможных аварийных ситуаций и ошибок пользователя.

2.4.1. Переполнение буфера. Возникает, если строка при чтении оказалась длиннее чем максимальное количество цифр в числе (вместо этого также может случится ошибка выхода числа из диапазона, если число пометилось в буфер).

2.4.2. Неожиданный символ. Возникает, если в строке присутствуют символы, не предусмотренные программой, или из-за неправильного формата входных данных (например, лишний пробел, лишняя точка в числе, пробел посреди числа и т. д.).

2.4.3. Число не входит в диапазон. Возникает, если пользователь ввёл число, которое не входит в диапазон допустимых значений.

2.4.4. Ошибка при выделении памяти. Практически не зависит от действий пользователя. Возникает только при неудачной попытке выделить память на куче, поэтому вероятность возникновения крайне мала.

2.4.5. Пустая строка. Возникает, если пользователь ничего не ввёл.

3. Описание внутренних структур данных.

Для графа используется следующая структура:

```
struct my_graph
{
    size_t roads_cnt;
    uint32_t **buf;
};
```

Её поля:
size – целое число. Хранит количество ребер графа.
buf – указатель на указатель на беззнаковый целый тип размера 4 байта. Хранит граф в виде матрицы. Матрица создаётся один раз, и её размер больше не меняется в течение всего времени работы программы.

Данный способ хранения выбран из-за своей простоты реализации и возможности быстрого доступа к ребру графа, однако из-за этого приходится расплачиваться большими затратами памяти.

4. Описание алгоритма.

Для поиска кратчайших расстояний используется алгоритм Дейкстры. Его сложность O(v^2+e), где v – количество вершин, e – количество ребер в графе. Данный алгоритм был выбран, так как в графе не используются отрицательные ребра, из-за чего отпадает необходимость проверки на отрицательные циклы. Также время работы алгоритма Дейкстры меньше зависит от количества ребер в графе, чем время работы алгоритма Беллмана-Форда, у которого сложность O(v*e).

Сначала создаётся массив кратчайших расстояний, который заполняем максимально возможным расстоянием, и массив состояний, который сначала заполняем нулями. В массиве состояний 0 означает, что вершина не просматривалась, 1 – вершина в очереди на просмотр, 2 – вершина уже просмотрена. Для рассматриваемой вершины ставим кратчайшее расстояние 0, а состояние 1.

Далее заходим в цикл, который завершается, когда не осталось вершин в состоянии 1. В нём сначала выбираем вершину в состоянии 1 с наименьшим расстоянием. Далее переводим её в состояние 2 и просматриваем все смежные вершины в состояниях 0 и 1. При просмотре смежных вершин мы переводим их в состояние 1 и уменьшаем расстояние, если расстояние через ранее рассмотренную вершину и ребро между ними меньше.

Данный алгоритм можно применить, например, при расчете расстояния на карте. Если за значения ребер принять количество требуемого ресурса (например, топлива или денег), то данный алгоритм можно использовать для расчёта требуемых ресурсов для выполнения какого-либо действия.

5. Набор тестов.

Тесты описаны только для ввода значений. Отсутствие выходных данных означает в данном случае что тест позитивный.

В меню проверяется, что после каждого действия запускается нужная функция. При добавлении дополнительно проверяется, что стартовая и конечная вершины ребра не совпадают. При удалении проверяется только существование ребра. При поиске вершин за расстояние проверятся их наличие. Если они есть, то их список выводится на экран вместе с расстояниями. При сохранении графа проверяется, что граф непустой, и что имя файла непустое.

Входные данные	**Выходные данные**	**Что и где проверялось**
1		Ввод минимального целого числа
3		Ввод целого числа не на границе
1000		Ввод целого числа максимального размера
0000003		Ввод целого числа с ведущими нулями (без переполнения буфера)
filename		Ввод имени файла
	Ошибка. Пустой ввод.	Ввод пустой строки при вводе числа или имени файла
what	Ошибка. Встречен неожиданный символ.	Ввод не числа, если нужно целое число
1234567890	Ошибка. Переполнение буфера при вводе.	Переполнение буфера (при чтении целого числа)
2024	Ошибка. Число выходит за пределы допустимого диапазона.	Ввод числа, не входящего в допустимый диапазон чисел.

6. Выводы по проделанной работе.

В ходе выполнения работы я изучил способы хранения графовых структур и алгоритмы поиска кратчайших путей из одной вершины во все остальные: алгоритм Дейкстры и алгоритм Беллмана-Форда. Также был приведён пример задачи, в которой можно применять выбранный алгоритм (в данном случае алгоритм Дейкстры).

7. Ответы на вопросы.

7.1. Что такое граф?

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

7.2. Как представляются графы в памяти?

Граф можно представить в виде матрицы смежности или списка смежности.

7.3. Какие операции возможны над графами?

Поиск кратчайших путей к одной вершине/всем вершинам/между всеми вершинами, поиск эйлеровых или гамильтоновых путей (последние 2 - при наличии).

7.4. Какие способы обхода графа существуют?

Обходы графа существуют в глубину или в ширину.

7.5. Где используются графовые структуры?

Графовые структуры могут использоваться в задачах, где произвольные элементы можно представить в виде верин графа, а связи между ними – в виде ребер.

7.6. Какие пути в графе вы знаете?

В графе может быть простой, замкнутый, эйлеров или гамильтонов путь.

7.7. Что такое каркасы графа?

Каркасы графа – это подграфы, содержащие все вершины и не содержащие циклов.