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 №7
«ГРАФЫ»

Студент Хамзина Регина Ренатовна
фамилия, имя, отчество

Студент, группа ИУ7-33Б — Хамзина Р.Р.

2020 г.

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.

Задана система двусторонних дорог, где для любой пары городов есть соединяющий их путь. Найти город с минимальной суммой расстояний до остальных городов.

Техническое задание

Входные данные:

1. Целое число — количество городов — вершин графа - *n : [3… INT_MAX* - 1]*
2. Тройки целых чисел — дорога - вершины ребра и его вес — r*ow column value:*
row : [1… n]
column : [1… n]
value : [1… INT_MAX - 1]*
3. Текстовый файл — файл с матрицей смежности графа

**INT_MAX* — недостижимое значение (<*limits.h*>)

Выходные данные:

1. Целое число — номер города — номер вершины графа, сумма расстояний от которой до всех других вершин минимальная *[1… n], n* - число вершин.

2. Матрицы целых чисел — матрица смежности графа и матрица кратчайших расстояний.

3. Изображение графа в формате png — исходный граф и граф с кратчайшими расстояниями и выделенной вершиной.

Функция программы:

Программа принимает матрицу смежности, описывающую граф (систему двусторонних дорог), и находит вершину графа (город), сумма расстояний от которой до других вершин (городов) минимальна. Выводит матрицу смежности,

матрицу кратчайших расстояний и визуализирует исходный граф и граф кратчайших путей с искомой вершиной.

Обращение к программе:

Программа запускается из терминала в директории с проектом при помощи команды «./app.exe», если пользователь хочет осуществить ввод графа с клавиатуры, или при помощи команды «./app.exe name.txt», где name.txt — имя файла с матрицей смежности графа.

Аварийные ситуации

1. Некорректный ввод числа городов — числа вершин графа
Входные данные : не целое число или целое число, меньшее 3.
Выходные данные : сообщение «Неверно введено число городов».

2. Некорректный ввод номера вершины графа или веса ребра
Входные данные : не целое число или целое число, меньшее 1, или целое число, большее n, где n — число вершин.
Выходные данные : сообщение «Неверно задано ребро графа».

3. Ввод отрицательного веса графа
Входные данные : целое отрицательное число.
Выходные данные : сообщение «Вес ребра графа должен быть неотрицательным».

4. Обработка несвязного графа
Входные данные : несвязный граф.
Выходные данные : сообщение «Граф не является связным».

Внутренняя структура данных

Для работы с графом используется следующая структура данных:

```
typedef struct graph_t
{
    int size;
    int **matrix;
    int **path;
} graph_t;
```

Её поля:

int size — число вершин графа;
*int **matrix* — матрица смежности графа;
*int **path* — матрица кратчайших расстояний графа.

Алгоритм

Граф задается при помощи матрицы смежности, которую можно ввести координатным методом или прочитать из файла. Строится матрица кратчайших расстояний при помощи алгоритма Дейкстры, который заключается в следующем:
Для каждой вершины перебираются все другие вершины, которым назначаются метки, которые являются известным минимальным расстоянием от вершины источника до конкретной вершины. До начала обхода у всех вершин метки равны недостижимому значению. Для каждого шага алгоритма: выберем такую вершину W, которая имеет минимальную метку и рассмотрим все вершины в которые из вершины W есть путь, не содержащий вершин посредников. Каждой из рассмотренных вершин назначим метку равную сумме метки W и длинны пути из W в рассматриваемую вершину, но только в том случае, если полученная сумма будет меньше предыдущего значения метки. Если же сумма не будет меньше, то оставляем предыдущую метку без изменений. После рассмотрения всех вершин, в которые есть прямой путь из W, вершину W отмечаем как посещённую, и выбираем из ещё не посещенных такую, которая имеет минимальное значение метки, она и будет следующей вершиной W.

В матрице кратчайших расстояний находится строка с минимальной суммой элементов. Ее номер, увеличенный на единицу является искомой вершиной графа.

Сложность алгоритма: $O(n^2)$

Обоснование выбора алгоритма: по условию задачи граф должен быть связным и с положительными весами ребер. Для поиска кратчайших расстояний для связного графа с положительными весами ребер используется алгоритм Дейкстры. Для решения этой задачи можно использовать и другие алгоритмы, например алгоритм Беллмана-Форда или алгоритм Флойда-Уоршелла, которые работают не только с положительными весами ребер, но и с отрицательными, но преимущества этих алгоритмов не использовались в реализации программы за ненадобностью. Кроме того, сложность алгоритма Флойда-Уоршелла равна $O(n^3)$.

Функции программы

*void init_graph(graph_t *graph)* — инициализация структуры, описывающей граф;

*int create_graph(graph_t *graph)* — создание структуры, описывающей граф;

*void free_graph(graph_t *graph)* — освобождение памяти из-под структуры, описывающей граф;

*int Dijkstra(graph_t *graph, const int start)* - алгоритм Дейкстры;

*int get_median(graph_t *graph, int *median_index)* — нахождение вершины с минимальной суммой расстояний до других городов;

*int input_graph(graph_t *graph)* — ввод графа с клавиатуры;

*int read_graph(graph_t *graph, const char *filename)* — ввод графа из файла;

*void print_graph(int **graph, const int size)* — печать графа на экран;

*void visualizate_graph(int **matrix, const int size, char *name, char *filename, int node)* — визуализация графа.

Тесты

	Тест	Ввод	Вывод
1	Неверный ввод числа городов: меньше 3	0	Неверно введено число городов
2	Неверный ввод числа городов: не целое число	a	Неверно введено число городов
3	Неверный ввод номера вершины или веса ребра: не целое число	a	Неверно задано ребро графа
4	Неверный ввод номера вершины: целое число, меньшее 1	0	Неверно задано ребро графа
5	Неверный ввод номера вершины: целое число, большее числа вершин	6 (при n = 5)	Неверно задано ребро графа

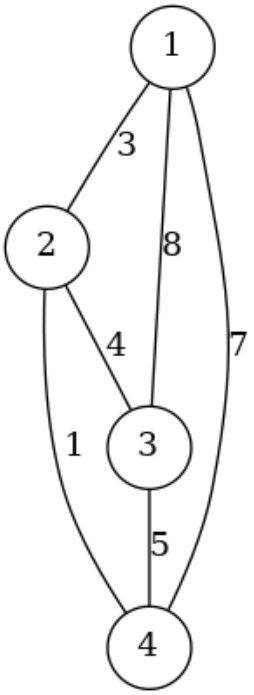
6	Ввод отрицательного веса графа	-3	Вес ребра графа должен быть неотрицательным
7	Ввод несвязного графа	6 1 2 2 1 3 3 2 4 4 3 4 4	Граф не является связным
8	Валидный тест	4 1 2 3 1 3 8 1 4 7 2 3 4 2 4 1 3 4 5	Матрица смежности: 0 3 8 7 3 0 4 1 8 4 0 5 7 1 5 0 Матрица кратчайших расстояний: 0 3 7 4 3 0 4 1 7 4 0 5 4 1 5 0 Город с минимальной суммой расстояний до других городов: 2

Вариант реальной задачи, для которой можно использовать программу

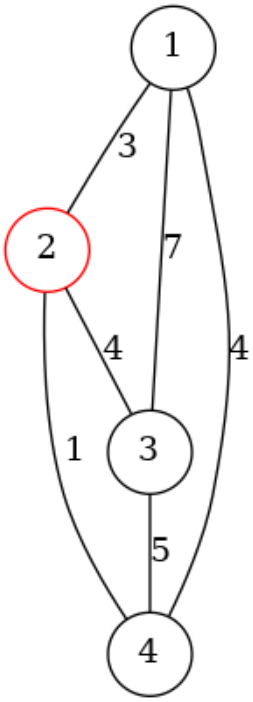
Задана система двусторонних дорог, где для любой пары некрупных населенных пунктов есть соединяющий их путь. В населенных пунктах нет центров скорой медицинской помощи. Необходимо найти населенный пункт, в котором рационально было бы разместить центр скорой помощи, которая бы оказывала медицинскую помощь всем населенным пунктам, входящим в систему.

Примеры визуализации графов

start



result



Ответы на контрольные вопросы

1. Что такое граф?

Граф - это конечное множество вершин и ребер, соединяющих их, т. е.:

$$G = < V,E >,$$

где V – конечное непустое множество вершин; E – множество ребер (пар вершин).

2. Как представляются графы в памяти?

Графы в памяти представляются при помощи матрицы смежности или при помощи списка смежности.

3. Какие операции возможны над графами?

Над графами возможны следующие операции: 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 эйлерова пути (если он есть), поиск гамильтонова пути (если он есть), оюход графа, исключение и включение вершин.

4. Какие способы обхода графов существуют?

Способы обхода графов: в глубину и в ширину.

5. Где используются графовые структуры?

Графовые структуры используются в задачах, где существуют произвольно связанные элементы. Например, в геоинформационных системах (ГИС). Существующие или вновь проектируемые дома, сооружения, кварталы и т. п. рассматриваются как вершины, а соединяющие их дороги, инженерные сети, линии электропередачи и т. п. — как рёбра. Применение различных вычислений, производимых на таком графе, позволяет, например, найти кратчайший объездной путь или ближайший продуктовый магазин, спланировать оптимальный маршрут.

6. Какие пути в графе Вы знаете?

Эйлеров путь - произвольный путь в графе, проходящий через каждое ребро графа точно один раз. При этом, если по некоторым вершинам путь проходит неоднократно, то он является непростым.

Гамильтонов путь - путь в графе, проходящий в точности один раз через каждую вершину графа (а не каждое ребро).

7. Что такое каркасы графа?

Каркас графа - дерево, содержащие все вершины графа.

Выводы

В лабораторной работе граф представлен матрицей смежности, также граф можно представлять списком смежности. Для поиска кратчайших путей использован алгоритм Дейкстры, помимо него поставленную задачу решают алгоритмы Беллмана-Форда и Флойда-Уоршелла. Выбор алгоритма обосновывается работой с графом только с положительным весами ребер и сложностью алгоритма $O(n^2)$ (например, у Флойда-Уоршелла $O(n^3)$).
Программу можно использовать для решение реальной задачи, например выбора населенного пункта для расположения центра скорой помощи, обслуживающей систему населенных пунктов.