Лабораторная работа №7 по дисциплине «Типы и структуры данных»

Графы

Выполнила студентка группы ИУ7-35Б

Мищенко Маргарита

Цель работы

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

Задание

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. Для каждой пары городов найти длину кратчайшего пути между ними.

Исходные данные

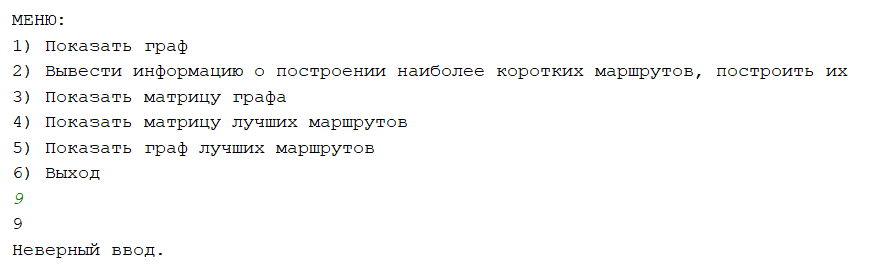
Программа считывает данные о графе из текстового файла. Граф задается количеством вершин и матрицей, в которой указан вес ребер между вершинами (в ячейке (0, 2) вес ребра, связывающего 0 и 2 вершину). Нулевые ребра тоже указываются. Матрица должна быть квадратной.

Тесты

* Если в файле не квадратная матрица, либо другие ошибки

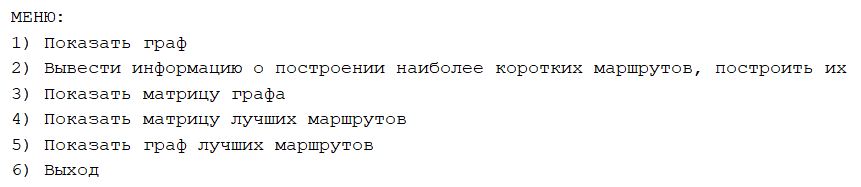


* Попытка ввода несуществующего пункта меню

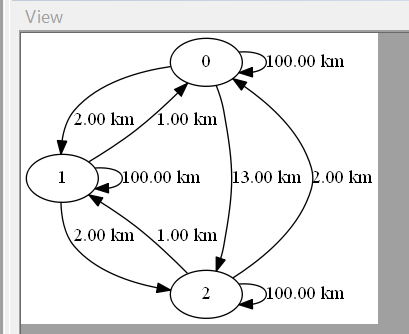


Интерфейс программы

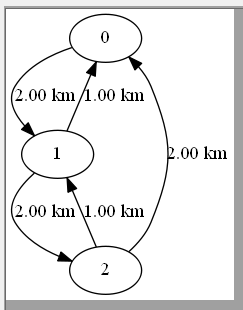
1. Главное меню



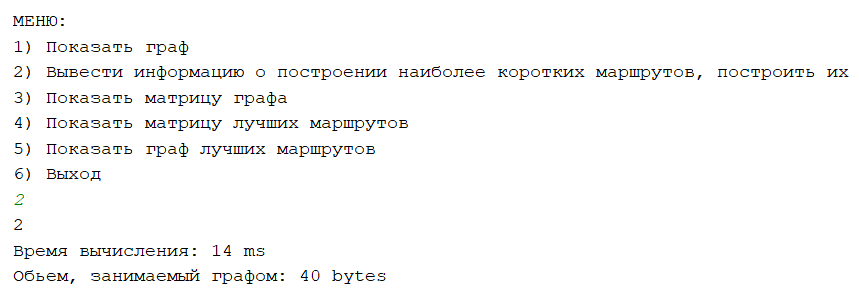
1. Граф со всеми путями



1. Граф только с лучшими путями



1. Вывод информации о процессе поиска идеальных путей



Особенности реализации

Был реализован алгоритм Флоайда-Уоршалла, идея которого описывается следующим образом:

По алгоритму Флойда-Уоршалла сначала ищется кратчайший путь от одной вершины ко всем вершинам, доступным из нее, затем проводятся те же действия, но пытаясь пройти от этой вершины ко всем доступным из нее, проходя каждый раз через новую вершину (сначала через первую, затем – через вторую и т.д.). Таким образом, обрабатываются все вершины.

Внутренняя структура данных

Граф представлен в виде матрицы смежности размера n \* n, где в (i, j) ячейке хранится 0, если ребра между вершинами нет (либо нет петли у вершины, если i = j ), и вес ребра в противном случае. Матрица смежности является более удобным способом хранения данных при обработке и заполнении. Недостатком выбранной реализации является большое количество требуемой памяти – хранятся также нули, которых при другой реализации можно выкинуть.

Вывод

Для реализации задачи был выбран алгоритм Флойда-Уоршалла, так как в данной задаче нет отрицательных значений веса ребер (алгоритм Беллмана-Форда работает с подобными случаями), и нам требуется найти кратчайшие пути между всеми вершинами (а не кратчайшие пути из одной вершины во все другие, как в алгоритме Дейкстры).

В данной программе идеальный и избыточный граф – это два разных графа, которые можно одновременно просматривать и сравнивать.

Контрольные вопросы

1. Что такое граф?

Граф – конечное множество вершин и соединяющих их ребер; G = <V, E>. V – вершины графа, E – ребра. Если пары Е (ребра) имеют направление, то граф называется ориентированным; если ребро имеет вес, то граф называется взвешенным. В моем задании мы имеем дело с ориентированным взвешенным графом.

1. Как представляются графы в памяти?

Существуют различные методы представления графов в программе.

* + - * 1. Матрица смежности B(n \* n) – элемент b[i, j] = вес ребра, если существует ребро, связывающее вершины i и j, и = 0, если ребра не существует.
        2. 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 (в массиве), либо же каждая вершина может хранить свой список смежностей.

1. Какие операции возможны над графами?

Основные операции над графами: обход вершин и поиск различных путей: кратчайшего пути от вершины к вершине; кратчайшего пути от вершины ко всем остальным; кратчайших путей от каждой вершины к каждой; поиск эйлерова пути и гамильтонова пути, если таковые есть в графе.

1. Какие способы обхода графов существуют?

Один из основных методов проектирования графовых алгоритмов – это поиск (или обход графа) в глубину (depth first search, DFS). При поиске в глубину, начиная с произвольной вершины v0, ищется ближайшая смежная вершина v, для которой осуществляется поиск в глубину (т.е. снова ищется ближайшая смежная с ней вершина) 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 = v0. Иными словами, поиск в глубину из вершины v основан на поиске в глубину из всех новых вершин, смежных с вершиной v. Путь, полученный методом поиска в глубину, в общем случае не является кратчайшим путем из вершины v в вершину u. Это общий недостаток поиска в глубину.

Указанного недостатка лишен другой метод обхода графа – поиск в ширину (breadth first search, BFS). Обработка вершины v осуществляется путем просмотра сразу всех новых соседей этой вершины. При этом полученный путь является кратчайшим путем из одной вершины в другую.

1. Где используются графовые структуры?

Графовые структуры могут использоваться в задачах, в которых между элементами могут быть установлены произвольные связи, необязательно иерархические. Наиболее распространенным является использование графов при решении различных задач о путях, будь то построение коммуникационных линий между городами или прокладка маршрута на игровом поле.

1. Какие пути в графе Вы знаете?

Путь в графе, проходящий через каждое ребро ровно один раз, называется эйлеровым путём; путь может проходить по некоторым вершинам несколько раз – в этом случае он является непростым.

Путь, проходящий через каждую вершину ровно один раз, называется гамильтоновым путем. Как эйлеров, так и гамильтонов путь могут не существовать в некоторых графах.

Условие существования эйлерова пути доподлинно известно, гамильтонова – нет.

1. 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.