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 Козлова Ирина Васильевна

Группа ИУ7 – 32Б

2020 г.

Оглавление

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.

Указания к выполнению работы

Интерфейс программы должен быть понятен неподготовленному пользователю. При разработке интерфейса программы следует предусмотреть:

- указание формата и диапазона вводимых данных,
- блокирование ввода данных, неверных по типу,
- указание операции, производимой программой:
  - добавление элемента в стек,
  - удаление элемента из стека,
  - вычисление (обработка данных);
- наличие пояснений при выводе результата.

При тестировании программы необходимо:

- проверить правильность ввода и вывода данных (т.е. их соответствие требуемому типу и формату), обеспечить адекватную реакцию программы на неверный ввод данных;
- обеспечить вывод сообщений при отсутствии входных данных («пустой ввод»);
- проверить правильность выполнения операций;
- предусмотреть вывод сообщения при поиске несуществующих путей в графе .

Описание технического задания

Задана система двусторонних дорог. Найти множество городов, расстояние от которых до выделенного города (столицы) больше, чем T.

Входные данные:

1. **Целое число, представляющее собой номер команды**: целое число в диапазоне от 0 до 7.
2. **Командно-зависимые данные:**
   целочисленные или дробные значения

Выходные данные:

1. Результат выполнения определенной команды.
2. Характеристика сравнения графа на различных размерах.

Функции программы:

1. Загрузить граф из файла.
2. Ввести граф вручную.
3. Вывести на экран матрицу дорог.
4. Вывести граф с помощью GRAPHVIZ.
5. Нахождение путей по условию.
6. Вывести сравнительную характеристику.

0. Выход из программы.

Обращение к программе:

Запускается через терминал. Так же можно собрать программу используя makefile и запустить ее с помощью команды run.

Аварийные ситуации:

1. Некорректный ввод номера команды.
   На входе: число, большее чем 7 или меньшее, чем 0.
   На выходе: сообщение «ERROR!!! Invalid command entered, please re-enter!!!»
2. Некорректный ввод номера команды.
   На входе: пустой ввод.
   На выходе: сообщение «Invalid command entered, please re-enter!!!»
3. Некорректный ввод номера города или рассояние.
   На входе: отрицательное число или число, превышающее максимально допустимое число для номера города, буква.

На выходе: сообщение «ERROR!!! There were problems filling in the table.»

4. Совершение команды с графом, до его загрузки.
   На входе: 3, 4, 5 (номер команды).
   На выходе: сообщение «Graph is emtry, please put 1 at first!.»

Описание структуры данных

Представление графа в программе с помощью матрицы смежности размером count*count, где в ячейки (i*j) хранится длина дороги из (I) в (j).

```
typedef struct
{
    int count;
    double **data;
} matrix_r;
```

count — размер матрицы

data – указатель на данные

Описание алгоритма

1. Выводится меню данной программы.
2. Пользователь вводит номер команды из предложенного меню.
3. Пока пользователь не введет 0 (выход из программы), ему будет предложено вводить номера команд и выполнять действия по выбору.

Набор тестов

№	Название теста	Пользовательский ввод	Результат

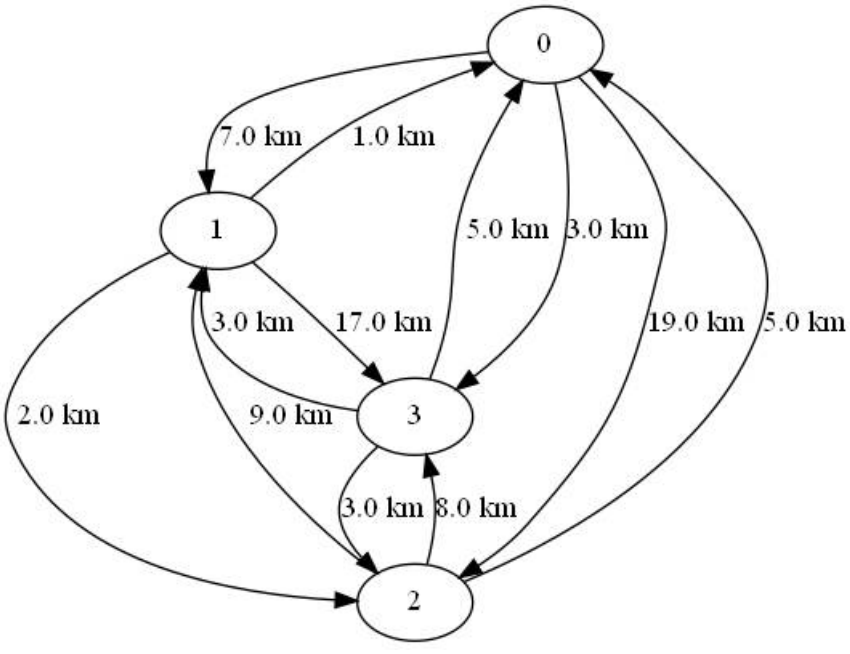
1	Некорректный ввод команды	45	Invalid command entered, please re-enter!!!
2	Пустой ввод	Пустой ввод.	Invalid command entered, please re-enter!!!
3	Не существование графа при вызове команд 3,4,5	Команда 3,4,5	Graph is empty? Please, put 1 at first!
4	Ввод неверного номера города (столицы)	9 (можно от 0 до 5)	Invalid command entered, please re-enter!!!
5	Ввод отрицательного значения длины	-8	Invalid command entered, please re-enter!!!

Пример вывода графа

Входные данные

```
4

0 7 19 3
1 0 2 17
5 9 0 8
5 3 3 0
```

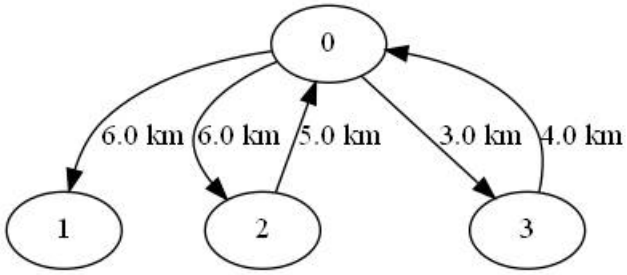



```
6
```

Вывод графа по условию

Столица 0

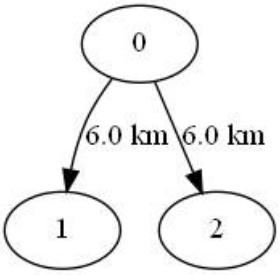
Расстояние 2



Вывод графа по условию

Столица 0

Расстояние 5



Вывод графа по условию

Столица 0

Расстояние 20

Вывод: сообщение, что таких дорог нет.

Оценка эффективности

Измерения эффективности сортировок будут производиться в тактах процессора, с помощью специальной функции будут делаться замеры количества тактов, поэтому погрешность данного измерения минимальна. Частота процессора 1900000000 Гц.

Команда добавления элемента в стек.

Размер	Время в тиках	Память
5	10749	48
10	80830	88
15	196777	128
20	358518	168

Графики оценки эффективности

График зависимости памяти от количества вершин в графе

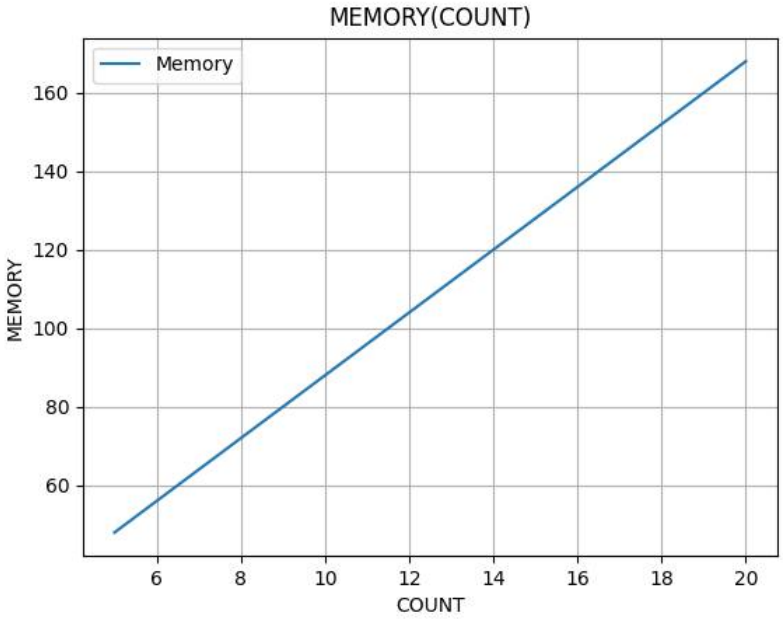
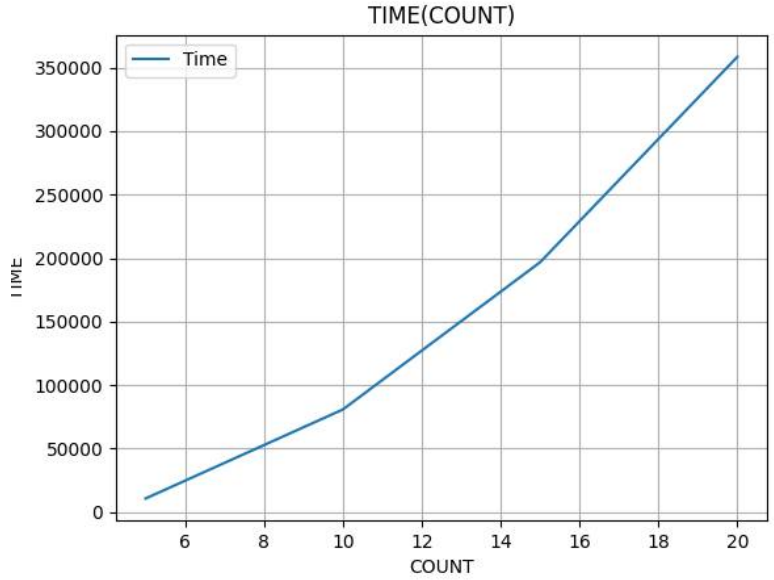


График зависимости времени от количества вершин в графе



Ответы на контрольные вопросы

1. Что такое граф?

Граф – это конечное множество вершин и ребер, соединяющих их, **G = < V,E >,** 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?

В памяти удобно представлять граф в виде матрицы смежности или списка смежности.

Матрица смежности B(n*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?

Обход вершин и поиск различных путей: поиск кратчайшего пути от одной вершины к другой (если он есть), поиск кратчайшего пути, поиск эйлерова пути, поиск гамильтонова пути.

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 последоватеьно заносятся в очередь просмотра.

Обход в глубину (DFS – Depth First Search) - начиная с некоторой вершины v0, 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 = v0. При просмотре используется стек.

5. Где используются графовые структуры?

Графовые стуктуры могут использоваться в задачах, где между элементами могут быть установлены произвольные связи. Наиболее распространеное использование таких структур — при решении различных задачах о путях.

6. Какие пути в графе Вы знаете?

Эйлеровый путь - путь в графе, проходящий через каждое ребро ровно один раз. (путь может проходить по некоторым вершинам несколько раз – в этом случае он является непростым)

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

Такие пути могут не существовать в графах.

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.

Вывод

Для реализации данной задачи был использован алгоритм Дейкстры, который находил все пути, которые больше заданного значения (вводить), от определенной вершины (вводить) до всех остальных вершин.
Алгоритм Дейкстры очень прост, и его удобно использовать в различных сверах жизни, так например, найти кратчайшее расстояние от одного города до другого, или, например, определить, какой маршрут по стоимости перелета из одного города в другой самый дешевый.

Хранение графа в виде матрицы смежности удобно тем, что по матрице можно понять расстояние между вершинами, или узнать, существует ли ребро между вершинами.