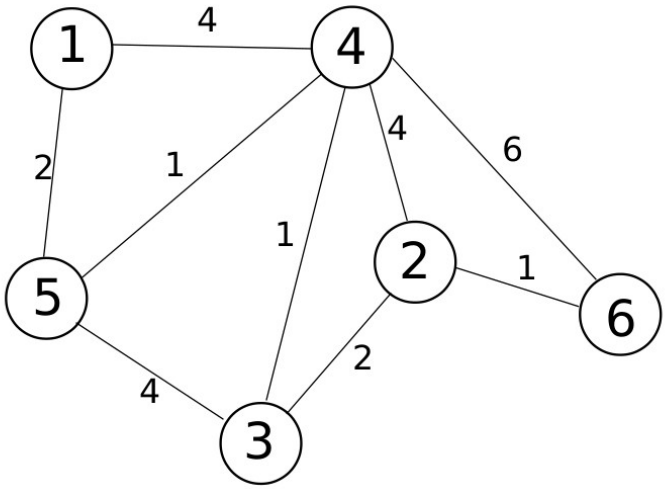
Graph algorithms

Given a transport network of *n* cities (1 <= n <=1000000) and roads between them write algorithms that solve problems:
1) Finding path between given two cities that passes through least number of other cities.
2) Finding path between given two cities that has the shortest total distance (standard algorithm)
3) Finding path between given two cities that has the shortest total distance (use geographical heuristic)
Compare algorithms execution time for given data.

Input:
In the first line there are two integers *n* and *m* (1<=*n*<=1000, 1<=*m*<=100000) meaning number of cities and number of roads between them. In the next n lines there are coords of cities (numbered from 1 to *n*): latitude and longitude. In the next *m* lines road descriptions are given: in each line three numbers: *v1*, *v2* and *w*. They mean there is a road between cities *v1* and *v2* (undirected graph, both directions) and its distance is *w*. In the last line starting and ending cities (s and e) are given.

Output:
In the first line a path length for algorithm variant 1) is given. In the second line vertices on the path are given (separated by spaces). In the third line a path length for variant 2) is given and in the fourth line its vertices (separated by spaces)



Example: (graph illustrated below)
Input:
6 9 // 6 vertices, 9 edges
55 16 // exemplary coords of 6 cities
54.30 18.20
53.70 17
54.5 18
53 15.50
53.50 19.3
1 4 4 // road between city 1 and city 4 has distance of 4
1 5 2 // road between city 1 and 5 has distance of 2
2 3 2 //etc
2 4 4
2 6 1
3 4 1
3 5 4
4 5 1
4 6 6
1 6 // path from 1 to 6

Output:
1 // the path from 1 to 6 that passes through least other cities has only one city on path.
1 4 6 // the path
7 // the shortest path from 1 to 6 (in terms of distances sum) has length of 7
1 5 4 3 2 6 // the path