The Traveling Salesman Problem is solved approximately by MST method.

The program here reads an undirected graph from standard input and it’s represented in adjacency list representation.

Repeated Invocations of Dijkstra on this graph gives the distance matrix in which each entry \(d(i,j)\) is the shortest distance between cities \(i\) and \(j\).

The Dijkstra's Algorithm uses Binary Heap Implementation of the Priority Queue. For this special Datastructure is used which has city index and the value of distance from a source. Initially, we initialize the distance of every city to be infinite which is assumed to be 20,000 in this case. Every time, we extract minimum from the heap, we run heapify and so it runs in \(O(\log|V|)\) time. Here several Datastructures are used to ensure efficiency in runtime. One vector \(Q\) tracks whether a vertex is in Binary Heap or not. Another Heap map Vector keeps track of position of a particular vertex in the Heap so that it can be reached in constant time. The Relax operation uses this Heap map to run it in \(O(\log|V|)\) time.

Then Prim's Algorithm is run on the resultant \(D\) matrix. That uses Array implementation of the queue. So its running time is \(O(E)\) where the graph is assumed to be dense in fact its fully connected graph almost always.

The Prim's Algorithm gives minimum spanning tree. To store this tree we use special data structure again. The spanning tree is store as an adjacency list and then we run Depth first search on the tree with city 0 as the root and this gives us sequence of vertices which is the tour we propose.

For the optimal algorithm, we use an elegant algorithm which produces all the permutations with 0 as the first city in lexicographic order. We produce the permutations only with 0 as the first city because the rest of the permutations are just repetitions of the same.