Problem Pattern

We use Dijkstra's algorithm to find the shortest path from our source to all targets. This is a textbook algorithm, refer to this link for more details.

Problem Approach

1. Put all delay into a delay map -> Map<Integer, Map<Integer, Integer>>

// source node : Map<destination node, delay>

2. Keep a visited array.
3. init a min priority queue PQ -> each object in PQ should be a pair with

top.first = current delay
top.second = current source city
PQ compares each object by total delay so far

4. add original source nodes to PQ with delay = 0
5. while exists nodes to explore

→ get min object then remove it from PQ
→ get the current total delay, current source city
→ mark the current node as visited
→ reduce the count of nodes by one
→ update the max delay to the current total delay
→ for all connected nodes for the current node: if the node is not visited then add the {currentTotaldelay + delay} to PQ

6. If N == 0 return max delay else return -1.