Task 1: Voting City (`voting city`)

Authored and prepared by: Aloysius Lim

Subtask 1

Limits $Q = 1, K = 1$, No tickets sold

This is just a standard shortest path problem.

Time Complexity: $O(E \log V)$

Subtask 2

Limits: $K = 1$, No tickets sold.

There are 2 ways to do it. You can run the shortest path problem Q times which will work for this subtask. However, in preparation for the later tasks, the intended way of reverse the problem. Instead of finding the shortest path to the voting city, find the shortest path from the only voting city to each city. You have to reverse the edges however, and do shortest path from the only voting city. Then, you can answer the query in O(1) time.
Time Complexity: $O(E \log V + Q)$ or $O(QE \log V)$

Subtask 3

Limits: $K = 1$, $Q = 1$. No tickets sold.

Similar to the above, running the shortest path Q times would work, and you check each voting city. However, you can continue with the latter approach as well. In this case, there are multiple voting cities, so a common technique is to use a super source city connecting the voting city with road edge of 0.

Time Complexity: $O(E \log V + Q)$ or $O(QE \log V)$

Subtask 4

Limits: $K = 1, Q = 1$

From this sub task onwards, we have to consider the effects of the tickets. We know we can use a ticket at most once, and once per road. Furthermore, notice that the number of possible tickets is small at 5. This motivates keeping a state of the tickets that have been used. Thus, we can make 32 copies of the cities and compute the shortest path to (city, tickets used). We reconnect the edges and modify the weights between the new cities whether or not we use a ticket between them. Then, at the destination, we compare the 32 different possible ticket state and choose the best cost of them. Note that, while it is possible to incorporate the ticket cost when constructing the new edges for this subtask, to facilitate further subtasks, it is better to just leave the cost at the end and then add the costs while computing the best state.
To analysise time complexity, let the number of tickets be T. We thus have $T2^T E$ edges and $2^T V$ vertices.
Time Complexity: $O(T^2 2^T E \log V)$

Subtask 5

Limits: $K = 1$

Notice that in this case, we can no longer just blindly run Q queries. Thus, we have to apply the trick we did in subtask 2.
Time Complexity: $O(T^2 2^T E \log V + Q2^T)$

Subtask 6

Limits: There is only at most 1 ticket.

This is a simplified version of the 5 tickets. However, if the contestant is unable to do the above, he can just code out a simpler state (city, taken) where taken is just a T/F boolean. Note that here, since the number of edges and vertices is small, you can indeed run the algorithm Q times.

Time Complexity: $O(E \log V + Q)$ with a higher constant.

Subtask 7

Limits: $N \leq 100, E \leq 1000$

I added this subtask to allow bad implementation to pass or any not so optimal solutions. I personally do not know of a weaker solution that can pass this solution without passing the final one. However, because the constraints are small, if you pass subtask 4 while running Q times, you should be able to pass this

Time Complexity: $O(T^2 2^T E \log V + Q 2^T)$

Subtask 8

Limits: None

Finally in this subtask, you add the final trick of the supersource node and finally this concludes the question.
Time Complexity: $O(T^2 2^T E \log V + Q 2^T)$