Problem Set 5 - Shortest Paths

Problem B

This question wasn't too bad either, I think when first reading it, I was wrongly under the impression that there could be trains from ANY city to any other city. And thus, I started with the wrong - and a more overly complicated approach. But I had it all largely there from the beginning.

I understood that this was a Djikstra question as there were weighted edges, and we were checking all shortest paths from a single source with no negative edge weights. And that when we were running Djikstra, we just wanted to mark how many trains were used. Initially though, since I thought there could be a train from any city to any city, I thought the way to track which trains were used was to store the trains used in a bitset and pass a bit set through Djikstra states. Every time a shortest path was found, we would union the trains used to get to that state with the total trains used, and then by the end, we would just get the bitcount. Honestly, that should work with the actual question (?), but it definitely wasn't passing all the test cases.

I then read the question again, and realised that there were only trains from the capital to every other city. So this could pretty easily be done by just running Djikstra, and for every path, checking if it used a train to get there, if it did mark the prev array as having used a train (with -1). Finally a count through all the -1 occurrences in the prev array would give us the number of trains.

A few changes to the edge struct needed to be done in order to get this working, I created an extra field in the edge struct called 'is_train', and then a custom operator to prioritise edges that weren't trains in the priority queue.