Evan Zhang | z5383657

Question 2

2.1

You want to be consuming the rations from the town with the smallest c_i in the entire sequence of towns (excluding the last town, since you won't need to buy anything once you are there). Since you are going to the last town, you would have passed the town with the cheapest rations (excluding the last town). To spend the least money, you should have bought enough rations from that town to reach the last town, as there are no towns with a lower price.

2.2

The last time you should change using a town's rations is when you reach the town with the cheapest rations (excluding the last town). You should have switched from the previous cheapest town's rations at that point.

2.3

We are going to plan our trip out and assign each town how many rations are needed to be bought in a 1 indexed array of n size.

Put town 1 as the cheapest town and add 1 to array[1] (as you need rations to travel to the next town). Go through the given ration price array until a new town that is cheaper is found. At each town that is more expensive or equal to the price, add 1 to array[current_cheapest_town]. If/When a cheaper town is found, go to array[current_town] and repeat the above until the current town being looked at is the last town.

Since, at any point in the journey, we will have bought enough rations to make it to a cheaper town/the last town, we have now planned the trip with the minimum cost to travel from town 1 to town n, with the results in an array of n size, with each index corresponding with how many rations to buy in that town.

This is done in O(n) time since at each town in the planning stage, at most O(1) order operations are carried out n times. Then in actually travelling (not sure if this is needed), n towns are visited, with total O(n) time, with rations already pre-determined to be bought or not.

O(2n) = O(n), therefore, the algorithm runs in O(n) time.