Homework 6: Data Structures

Due: October 31, 2024

Problem 1. You're an announcer at the hottest rock-paper-scissors competition in North America. The biggest match of the evening is coming up, Rocky Rick vs. Paper Pete, and you need make sure the audience stays engaged during the event. For past announcers, the biggest obstacle is keeping the audience engaged during the halftime show (the players need time to rest their hands).

In order to step things up, you're planning to do something that has never been done before and want to be able to say that this is the nth time that the score between Rocky Rick and and Paper Pete has been i to j at halftime. The main challenge that has prevented previous announcers from doing this is that you won't know what i and j are, as the game has not yet happened. You also cannot afford to scan through the entire list of previous halftime scores between Rick and Pete and count the number of i vs j appearances as the audience would surely revolt.

1. Devise an efficient method of processing the list of previous Rick-Pete halftime scores before their match begins, so that you can quickly say, right at the start of half-time, how many times the pair (i, j) has occurred at similar moments in the past. Your pre-match processing should take time proportional to the number of previous games and the querying task should take constant time.

2. Justify the runtime and correctness of your scheme.

Problem 2. There are n marbles rolling along a straight one-lane track. They each are at some distance (in inches) away from the start of the track (inch 0) and are all traveling to the end of the track which is at inch *end*. You are given two arrays: *position* and *speed*. Both are of length n so *position*$[i]$ and *speed*$[i]$ denote the starting position and speed of the ith marble, respectively. The positions are given in inches and speeds are given in inches per second.

Since the track is narrow, marbles cannot roll pass other marbles: this means that when a faster marble catches up to a slower marble, they will travel together at the speed of the slower marble. We can then define a *group* of marbles to be a set of marbles (of size at least 1) that reach the end of the track at the same time. You are tasked with figuring out the number of groups of marbles that arrive at the end of the track.

1. Devise an efficient method to find the number of groups of marbles we expect to see at the end of track given their starting positions, speeds, and *end*.
2. Justify the runtime and correctness of your scheme.