Jacob Archer

Prof. Shen CS 210

6/6/24

Program 1 Complexity Analysis

We're essentially looking at how the runtime of our function grows as the size of the input array grows. In our case, we're iterating over the array a couple of times. Each iteration runs through the entire array once, which makes it linear, or O(N), where N is the size of the array. This is because the time it takes to run the function scales linearly with the size of the input array. The other stuff we do inside the loop, like comparing strings and swapping elements, doesn't really change the overall picture because they're constant time operations, meaning they don't take longer as the array gets bigger. So, the bottom line is that our function's time complexity is O(N), which means it runs in linear time.

Now, onto auxiliary space complexity, which is about how much extra space our function uses besides the input array. Here, we're using a few extra variables to keep track of things and a couple of strings to store lowercase versions of our promotion items. But here's the key: no matter how big the input array is, the amount of extra space we use stays the same. It doesn't grow with the size of the input array. That's why we call it constant space complexity, or O(1). So, our function uses only a constant amount of extra space, no matter how big the input array is.