Spiral Matrix

Median of 2 Sorted Arrays

Need to get more familiar with binary search to not index out of range on the solution to this problem.

Word Search

This is a backtracking problem (not exactly DFS) and not really BFS either

Best Time to buy/sell stock:

Overthought yielded an O(nlogn) solution, when I could have done a less advanced sliding window solution

I did learn a bit about how to handle heaps in python though…

Next Permutation:

Comments were universally opposed to this as way too hard for a medium question.

Needed to clarify if all numbers in a given contiguous range are present. This may have helped me find the location where I needed to swap. However, I would not have gotten the reverse the last part of the array without help.

I had no idea what was going on once my original solution failed…

Very difficult problem and entirely unreasonable to ask in an interview without heavy hints, but not completely impossible to solve on your own. First of all, the requirements of **in-place replacement and constant space** should immediately indicate **swapping** (this goes for other questions too). Secondly, it should be obvious that **if the elements are increasing from the right, they are currently at their largest possible permutation**, so nothing can be done. I think the tricky part is simply knowing where to swap and reversing the last digits. This problem is certainly at the harder-end of medium, or even hard itself.

String:

Basic Calculator 1,2 (Legitimately hard divide/conquer problem)

Can also use a stack to do calculations, the idea I missed is to reverse the input to have commutativity work

RegEx matching

The answer is DP, however knowing more about RegEx from courses could help. RegEX is implemented with an NFA (nondeterministic – finite -automaton) and the ideal solution replicates that. However, DP is what’s expected during an interview.

Minimum Window Substring

Had the approach exactly right, just need to become more familiar with dictionary defaults

Also, should use single-variable counters to keep track of the window state, rather than doing an O(t) operation every time to check if the window is good.

Maximal Square/Rectangle -> This one is a DP problem that I should really consider getting good at.

Longest Increasing Subsequence -> The dynamic programming solution is O(n^2), however I should also focus on the O(nlogn) solution using binary search as well.

Frog Jump, want to do a DP solution instead of a DFS one

Insert, Delete, GetRandom O(1)

Apparently, random.choice() is O(n) so need to use list + dict() implementation

Time-based Key/value store

Real solution uses binary search instead of linear search

Should have seen this, as the dict entries are lists sorted in ascending order by timestamp

Top-K-Frequent-Elements (Uses Quick-Select) or a Heap