3 Technical Interview Questions

Question 1

Given 2 different lists of integers, write a function that will return their intersection.

I love this question because there are several possible solutions, each with different performance characteristics. When the candidate arrives at a solution you can ask them how fast it is and see if they can solve if with additional constraints.

Solution 1
A doubly nested for loop.
Runtime O(n^2)

It's ok if this is the first solution the candidate comes up with, it's pretty easy, but not optimal. This is a good opportunity to ask them if they can make it faster.

Solution 2
Sort both lists, then traverse them with 2 pointers (one per list). Increment the pointer that points at the smaller value. If both pointers point at the same value, insert it into the result and increment both pointers.
Runtime O(nlogn)

This solution has the advantage of using less space than the final solution, while still being pretty fast.

Solution 3
Iterate the first list, put the values into a hashtable, using the value as the key, and the count as the value (if it's already in the list increment count by one, else insert a count of 1). Then iterate list2, if the value is in the hashtable and > 0 insert that value into the result array, decrementing the value in the hash table.
Runtime O(n)

This is the optimal solution. A common mistake will be forgetting to keep a counter and neglecting any duplicates in the result list ([1,2,2,3] and [2,2,3,4] should intersect as [2,2,3] and not just [2,3]).

3 Technical Interview Questions

Question 2

Implement a queue using 2 stacks. Specifically, implement the enqueue and dequeue methods.

This question is good because it shows how well the candidate knows their data structures. It also forces them to write a class with some internal state.

Solution

In the constructor initialize 2 stacks, “entryStack” and “exitStack”. The enqueue method pushes the value onto entryStack. The dequeue method pops from exitStack if it’s not empty, otherwise pop all items from entryStack onto exitStack. If they’re both empty throw an exception.

Question 3

Design the software system for a parking garage. You’ll want to handle the ticketing system and keeping track of when the garage is full.

I like this question because there are a lot of different directions you can go with it. It’s good for a higher level design discussion, but as it progresses you can ask the candidate to start mapping out some of the classes or tables that will be necessary to track the data. You can also get into coding with the ticketing and billing system, depending on how much time you have.

Solution

Lots of possible solutions, but the goal is to have a conversation, not just get to an answer.