Arrays and Strings Quickstart Guide

In this mini-workbook, we'll get you up and running with arrays and strings. These problems come up incredibly often and most people overlook them because they seem like they're going to be easy. However, there is some tricky stuff in here, so make sure to take your time going through this.

In this guide, we will quickly highlight the key concepts that you need to know for your interview. If you're unfamiliar with any of these, I highly recommend taking some time to review them.

Then, I've provided you with a series of daily exercises to help you get up and running quickly. I recommend using these similarly to how you use your daily workbook. Do one set of exercises each day and be sure to log them in your tracker. This will help you ensure that you've reviewed all the most important points.

You can find additional resources here: https://www.byte-by-byte.com/strings/.

Key String & Array Terminology

- **Mutable vs. Immutable**
  This is a key concept that you need to understand based on the programming language you're using. Know whether strings are mutable or immutable and what the implications are
- **ASCII and Unicode**
  Make sure that you understand the difference and are aware of how character encoding works
- **Object or Primitive**
  Make sure to know whether your language treats strings as an object or primitive type

Key String & Array Patterns

- **Using a length-256 array**
  Rather than using a set or map to count characters in a string, we have a small fixed character set so we can often just use an array instead
- **Using 2 pointers**
  This comes up all the time, whether you're using a sliding window or just trying to find all the substrings
- **String math**
  It's critical that you be able to convert between strings/characters and integer values
- **Sliding windows**
  This concept is fairly straightforward, but as soon as you give something a fancy name it sounds scary. We'll go over some examples because this is super important
- **Comparison, Alignment, and Matching**
  Here, our strings & arrays start to overlap with recursion. For example, you might want to find the way to align 2 strings to minimize the necessary edits. This is fairly easy to do recursively
- **Regular Expressions**
  Implementing your own matching really falls into the previous pattern, but understanding how to use basic regular expressions in your language is a valuable skill
- **String Algorithms**
  These aren't that important, but there are algorithms like KMP, Boyer Moore, and Rabin-Karp that may be useful after you've mastered everything else

Daily Exercises

Day 1

Today, we're going to focus on the first pattern, using a length-256 array to store character counts. With all these problems, I encourage you to follow the entire problem solving framework we discussed in the course.

1. Anagrams (Leetcode)
2. Sorting the characters in a string (Geeks for Geeks)
3. Longest substring without a repeating character (Leetcode)

Day 2

Day 2 we're going to look at our second string pattern, using 2 pointers. By doing this focused practice, you'll be able to really reinforce the core patterns.

1. Remove Duplicates (Leetcode)
2. Is String a Palindrome (Leetcode)
3. Reverse Words in a String (Interview Cake)

Day 3

Today, we're doing string math! It's really important that we know how to properly process strings and arrays and convert back and forth between integer values and strings.

1. Convert Strings to Binary (Geeks for Geeks)
2. String to Integer (Leetcode)
3. Compare Version Numbers (Leetcode)

Day 4

Woohoo, now we're moving on to sliding windows! If you're feeling intimidated, don't. While they might seem a little complicated initially, sliding windows are quite straightforward and a super useful strategy to be able to use confidently.

1. Find All Anagrams in a String (Leetcode)
2. Minimum Window Substring (Leetcode)
3. Substring with Concatenation of All Words (Leetcode)

Day 5

Today, we're going to do string alignment and matching. This is a super important topic, but if your recursion skills are rusty, I'd highly recommend that you refresh those first. Here, we are really combining strings with recursion and it will get challenging if you're not up on your recursion game.

Start here to refresh your recursion, and if you want to go deeper, we do have a full course on recursion as well.

1. Longest Common Substring (Byte by Byte)
2. Edit Distance (Leetcode)
3. Regex Matching (Leetcode)

Day 6

For this final day, we're going to take a look at some regular expression problems.

1. Regex Matching (Leetcode)
2. Wildcard Matching (Leetcode)

Exercises Solutions

Coming soon!