420-SVJ-AS Internship 31 january 2020
Felix Soumpholphakdy

Chapter 1 - Arrays and Strings

Here are questions from the book (Cracking the Coding Interview) that have been adapted by me. The changes are minor and are aimed to increase clarity and simplify the questions.

1.1 Is Unique: Implement an algorithm to determine if a string has all unique characters. What if you cannot use additional data structures?

1.2 Check Permutation: Given two strings, write a method to decide if one is a permutation of the other.

1.3 URLify: Write a method to replace all spaces in a string with '%20'. You may assume that the string has sufficient space at the end to hold the additional characters, and that you are given the "true" length of the string. (Note: If implementing in Java, please use a character array so that you can perform this operation in place.)

1.4 Palindrome Permutation: Given a string, write a function to check if it is a permutation of a palindrome. A palindrome is a word or phrase that is the same forwards and backwards. A permutation is a rearrangement of letters. The palindrome does not need to be limited to just dictionary words.

1.5 One Away: There are three types of edits that can be performed on strings: insert a character, remove a character, or replace a character. Given two strings, write a function to check if they are one edit (or zero edits) away.

1.6 String Compression: Implement a method to perform basic string compression using the counts of repeated characters. For example, the string aabcccccaaa would become a2b1c5a3. If the "compressed" string would not become smaller than the original string, your method should return the original string. You can assume the string has only uppercase and lowercase letters (a - z).

1.7 Rotate Matrix: Given an image represented by an NxN matrix, where each pixel in the image is 4 bytes, write a method to rotate the image by 90 degrees. Can you do this in place?

1.8 Zero Matrix: Write an algorithm such that if an element in an MxN matrix is 0, its entire row and column are set to 0.

1.9 String Rotation: Assume you have a method isSubstring which checks if one word is a substring of another. Given two strings, s1 and s2, write code to check if s2 is a rotation of s1 using only one call to isSubstring (e.g., “waterbottle” is a rotation of “erbottlewat”). Hint: If s1 is a rotation of s2, then it follows that the string resulting from s2 concatenated with itself (s2 + s2) must contain s1 as a substring.