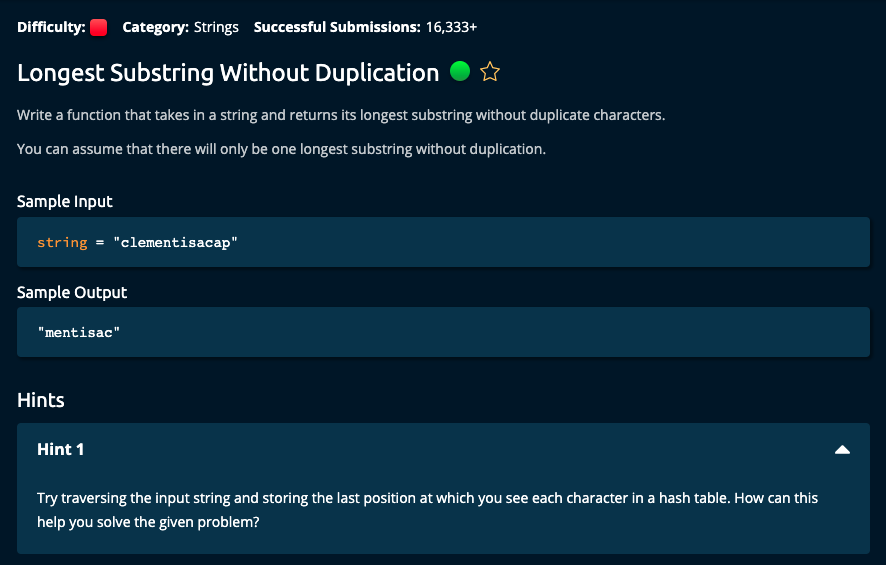
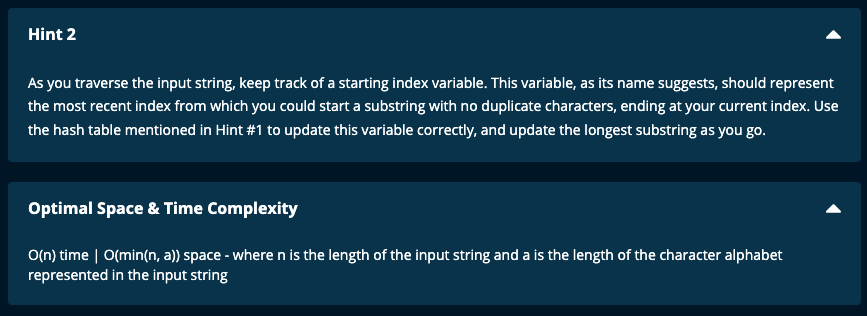
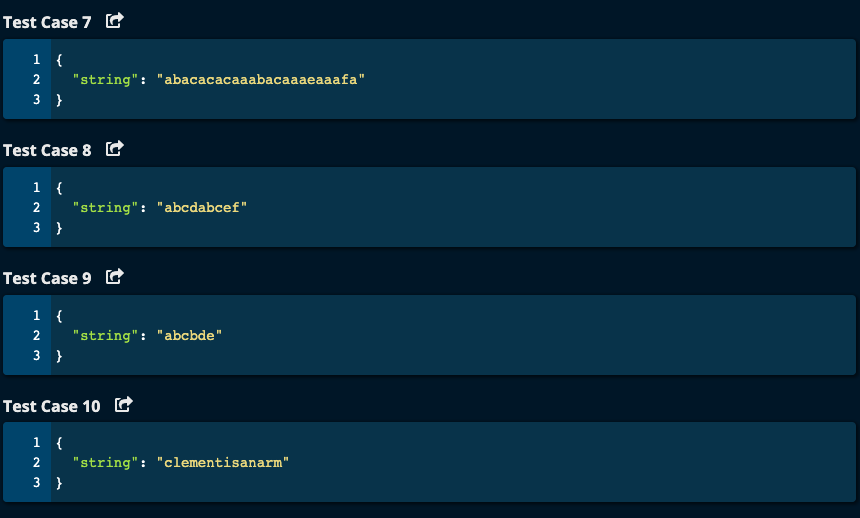
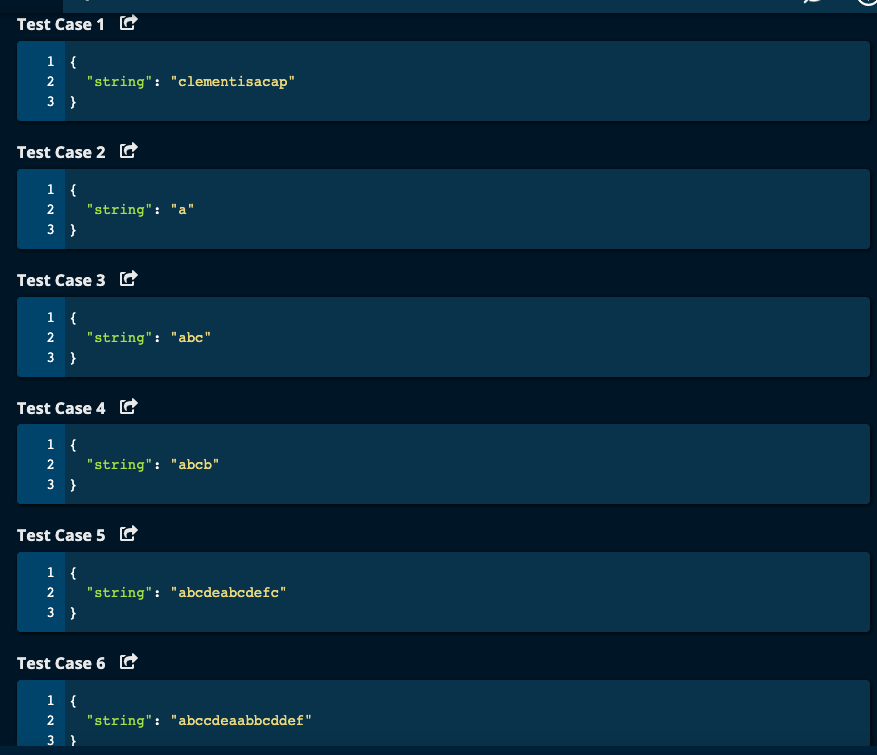
Longest Subarray without Duplication (Hard)







JJ Notes:

1. Initialize a dictionary to hold the characters in the string and the respective index called mydict.
2. Initialize start index to 0.
3. Initialize the longestSubstring as an empty list
4. Initialize the maximum length of the longestSubstring as maxLen to 0.
5. Initialize the currentSubstring as an empty list.
6. Initialize current Substring length as curLen to 0.
7. Iterate through the string array.
8. If the current character is not in mydict, put the character and its index in mydict as key-value pair. Append the character to the currentSubstring and increment curLen by 1.
9. If the current character is in mydict already, then save the index of the character in x.

Then update mydict with the current index as value for the character (key).

1. If the previous index from mydict x is greater than or equal to start, then update start to be the next index i.e. x + 1. Then currentSubstring will start from the new start to the current index + 1, i.e. I + 1. Then update current length to be current index + 1 – start. Note: we have to say one more than the last index value when slicing an array.
2. Of previous index is less than start, the character is not repeated in the current substring. So append the current character to currentSubstring and increment curLen by 1.
3. Finally, if curLen is greater than maxLen, update longestSubstring to be the currentSubstring and maxLen to be the curLen.
4. Finally, Join all the characters in the longestSubstring list to form string and return this string.

# My Solution -- All test cases passed

# LongestSubstringWithoutDuplication.py -- O(n) time | O(min(n, a))

# space where n is the length of the input string

# and a is the length of the character alphabet respresented in the input string

def longestSubstringWithoutDuplication(string):

mydict = {}

start = 0

longestSubstring = []

maxLen = 0

currentSubstring = []

curLen = 0

for i in range(len(string)):

if string[i] not in mydict:

mydict[string[i]] = i

curLen += 1

currentSubstring.append(string[i])

else:

x = mydict[string[i]]

mydict[string[i]] = i # update dictionary with the lastest index

if x >= start:

start = x + 1

currentSubstring = list(string[start: i + 1])

curLen = i + 1 - start

elif x < start:

curLen += 1

currentSubstring.append(string[i])

if curLen > maxLen:

longestSubstring = currentSubstring

maxLen = curLen

return "".join(longestSubstring)

Algoexpert Solution:

# Algoexpert Solution -- O(n) Time | O(min(n, a)) where n = length of the input string

# and a is the length of the character alphabet represented in the input string

def longestSubstringWithoutDuplication(string):

lastSeen = {}

longest = [0, 1]

startIdx = 0

for i, char in enumerate(string):

if char in lastSeen:

startIdx = max(startIdx, lastSeen[char] + 1)

if longest[1] - longest[0] < i + 1 - startIdx:

longest = [startIdx, i + 1]

lastSeen[char] = i

return string[longest[0]: longest[1]]