Problem # 763: Partition Labels

<https://leetcode.com/problems/partition-labels/>

My Solution:

<https://leetcode.com/problems/partition-labels/discuss/829164/Simple-Python-3-Solution-using-dictionary-and-lists-runtime-beats-97.13>

1. Create a dictionary with key = letter and value = a list of indexes from the string where the letters appear.

2. Define a partition as a two element array, where the first element represents the beginning of the partition and the last element the end of the partition. Initialize an array res to -1 where res holds the max values of the partition.

3. If a character has min index less than the max index of the partition, this character belongs to the partition. If the max index of the character is more than of the partition

4. However, if the min index of the character is more than the max index of the partition, then the character does not belong to this partition. Append the max value of the partition to res array and then form the new partition with the new min index and max index.

5. Iterate through the res array and calculate the difference with the preceding element to get the length. That is why we have res array with first element being -1.

Store these lengths in an array called result and return it.

--------------------------------------------

class Solution:

def partitionLabels(self, S: str) -> List[int]:

if len(S) == 1:

return(1)

my\_dict = {}

for idx, char in enumerate(S):

if char not in my\_dict:

my\_dict[char] = [idx]

else:

my\_dict[char].append(idx)

res = [-1]

partition = [0, 0]

for key in my\_dict.keys():

# the partition has 2 numbers -- min\_ and max\_

min\_ = my\_dict[key][0]

max\_ = my\_dict[key][-1]

if min\_ <= partition[-1]: # min index of the character < max value in the partition

if max\_ > partition[1]: # partition is updated with the max value.

partition[-1] = max\_

elif min\_ > partition[-1]:

res.append(partition[-1])

partition = [min\_, max\_]

res.append(partition[-1]) # add the last partition

result = []

for i in range(1, len(res)):

result.append(res[i] - res[i-1])

return(result)