DAY-45

Hackerrank:-

#!/bin/python3

import math

import os

import random

import re

import sys

#

# Complete the 'weightedUniformStrings' function below.

#

# The function is expected to return a STRING\_ARRAY.

# The function accepts following parameters:

#  1. STRING s

#  2. INTEGER\_ARRAY queries

#

def weightedUniformStrings(s, queries):

    # Write your code here

    weights = set()

    prev\_char = ''

    count = 0

    for c in s:

        if c == prev\_char:

            count += 1

        else:

            count = 1

            prev\_char = c

        weight = (ord(c) - ord('a') + 1) \* count

        weights.add(weight)

    # For each query, return "Yes" if it exists in weights set

    return ["Yes" if q in weights else "No" for q in queries]

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    s = input()

    queries\_count = int(input().strip())

    queries = []

    for \_ in range(queries\_count):

        queries\_item = int(input().strip())

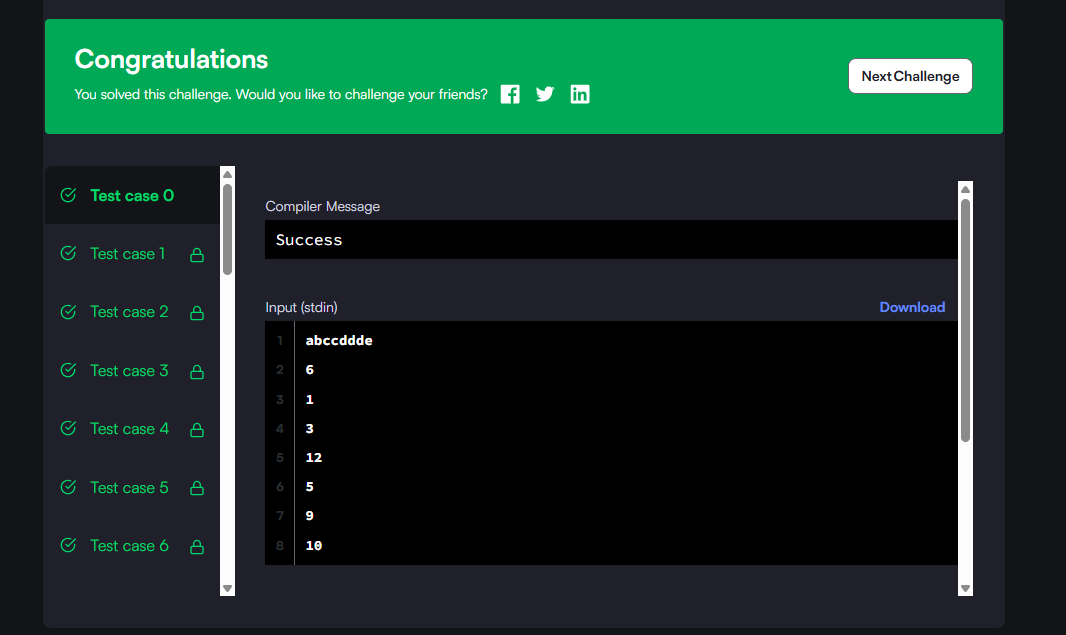
        queries.append(queries\_item)

    result = weightedUniformStrings(s, queries)

    fptr.write('\n'.join(result))

    fptr.write('\n')

    fptr.close()



class Solution(object):

def compress(self, chars):

"""

:type chars: List[str]

:rtype: int

"""

write = 0 # position to write compressed characters

read = 0 # position to read characters

while read < len(chars):

char = chars[read]

count = 0

# Count consecutive characters

while read < len(chars) and chars[read] == char:

read += 1

count += 1

# Write the character

chars[write] = char

write += 1

# If count > 1, write the count as digits

if count > 1:

for digit in str(count):

chars[write] = digit

write += 1

return write # new length



def min\_operations\_to\_reverse(s):

t = s[::-1]

n = len(s)

i = j = 0

while i < n and j < n:

if s[i] == t[j]:

j += 1

i += 1

return n - j # Minimum operations needed

# Read input

T = int(input())

for \_ in range(T):

S = input().strip()

print(min\_operations\_to\_reverse(S))

