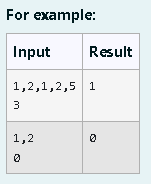
1. Given a tuple and a positive integer k, the task is to find the count of distinct pairs in the tuple whose sum is equal to **K**.

**Examples:**

**Input:** t = (5, 6, 5, 7, 7, 8 ), K = 13   
**Output:** 2   
**Explanation:**   
Pairs with sum K( = 13) are  {(5, 8), (6, 7), (6, 7)}.   
Therefore, distinct pairs with sum K( = 13) are { (5, 8), (6, 7) }.   
Therefore, the required output is 2.



Program:

t = input().split(',')

k = int(input())

s = []

s1 = []

for i in range(len(t)-1):

for j in range(i+1,len(t)):

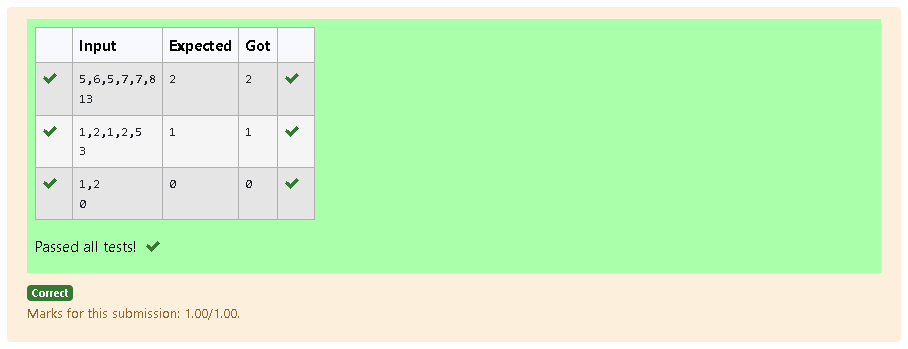
if (int(t[i])+int(t[j])==k) and ((t[i],t[j]) not in s1) and ((t[j],t[i]) not in s1):

s1.append((t[i],t[j]))

s1.append((t[j],t[i]))

s.append((t[i],t[j]))

print(len(set(s)))



2. The **DNA sequence** is composed of a series of nucleotides abbreviated as 'A', 'C', 'G', and 'T'.

* For example, "ACGAATTCCG" is a **DNA sequence**.

When studying **DNA**, it is useful to identify repeated sequences within the DNA.

Given a string s that represents a **DNA sequence**, return all the **10-letter-long** sequences (substrings) that occur more than once in a DNA molecule. You may return the answer in **any order**.

**Example 1:**

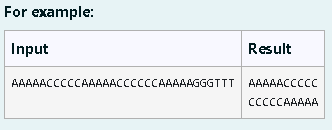
**Input:** s = "AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT"

**Output:** ["AAAAACCCCC","CCCCCAAAAA"]

**Example 2:**

**Input:** s = "AAAAAAAAAAAAA"

**Output:** ["AAAAAAAAAA"]



Program:

def findRepeatedDnaSequences(s):

if len(s) < 10:

return []

seq = {}

repeated = set()

for i in range(len(s) - 9):

sequence = s[i:i + 10]

if sequence in seq:

repeated.add(sequence)

else:

seq[sequence] = 1

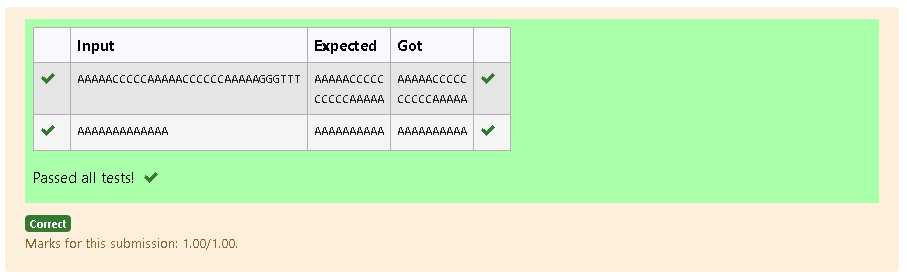
return list(repeated)

dna = input()

repeated = findRepeatedDnaSequences(dna)

for i in repeated:

print(i)



3. Coders here is a simple task for you, Given string str. Your task is to check whether it is a binary string or not by using python set.

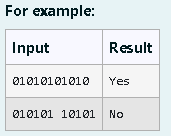
Examples:

Input: str = "01010101010"

Output: Yes

Input: str = "REC101"

Output: No



Program:

def isbinary(s):

binary={'0','1'}

for char in s:

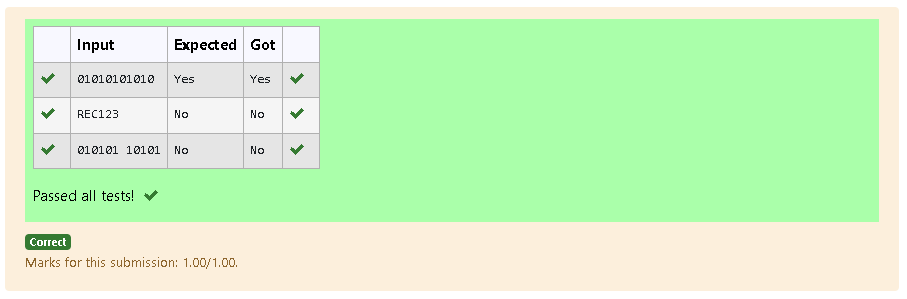
if char not in binary:

return "No"

return "Yes"

s=input()

print(isbinary(s))



4. There is a malfunctioning keyboard where some letter keys do not work. All other keys on the keyboard work properly.

Given a string text of words separated by a single space (no leading or trailing spaces) and a string brokenLetters of all distinct letter keys that are broken, return the number of words in text you can fully type using this keyboard.

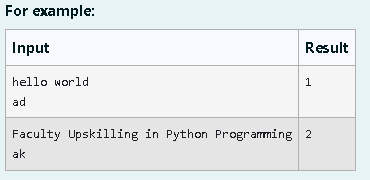
Example 1:

Input: text = "hello world", brokenLetters = "ad"

Output:

1

Explanation: We cannot type "world" because the 'd' key is broken.



Program:

s = input()

sub = list(input())

c= 0

for i in sub:

if i in s:

c+=1

print(c)



5. Write a program to eliminate the common elements in the given 2 arrays and print only the non-repeating

elements and the total number of such non-repeating elements.

Input Format:

The first line contains space-separated values, denoting the size of the two arrays in integer format respectively.

The next two lines contain the space-separated integer arrays to be compared.

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Input:

5 4

1 2 8 6 5

2 6 8 10

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Output:

1 5 10

3

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127)  Input:

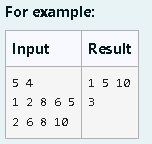
5 5

1 2 3 4 5

1 2 3 4 5

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Output:

NO SUCH ELEMENTS



Program:

def ece(arr1, arr2):

nre = []

for num in arr1:

if num not in arr2:

nre.append(num)

for num in arr2:

if num not in arr1 and num not in nre:

nre.append(num)

return nre

size1, size2 = map(int, input().split())

arr1 = list(map(int, input().split()))

arr2 = list(map(int, input().split()))

nre = ece(arr1, arr2)

if nre:

print(\*nre)

print(len(nre))

else:

print("NO SUCH ELEMENTS")

