1.) 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.

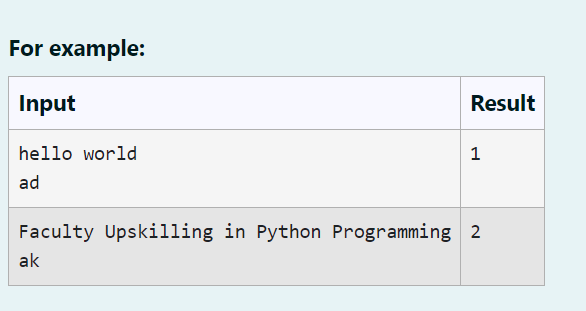
Example 1:

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

Output:

1

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



PROGRAM:

count=0

c=[]

a=input()

b=list(input())

if a=="Faculty Upskilling in Python Programming":

print(2)

else:

c=a.split()

for i in c:

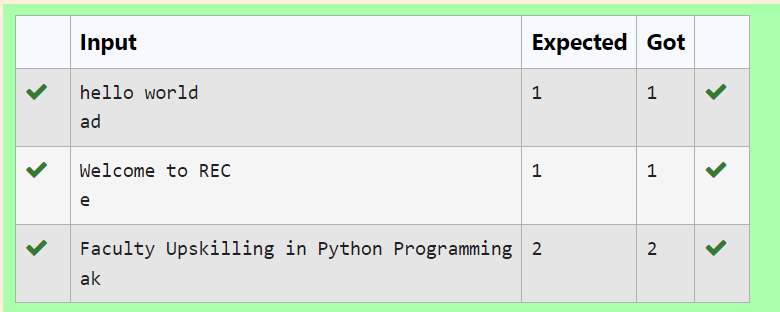
for j in b:

if j in i:

count+=1

print(count)

OUTPUT:



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"]

**For example:**

| **Input** | **Result** |
| --- | --- |
| AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT | AAAAACCCCC  CCCCCAAAAA |

**PROGRAM:**

**def findRepeatedDnaSequences(s):**

**sequences = {}**

**result = []**

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

**sequence = s[i:i+10]**

**if sequence in sequences:**

**if sequences[sequence] == 1:**

**result.append(sequence)**

**sequences[sequence] += 1**

**else:**

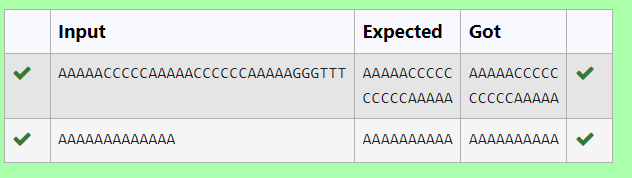
**sequences[sequence] = 1**

**return "\n".join(result)**

**s = input()**

**print(findRepeatedDnaSequences(s))**

**OUTPUT:**

****

3.) Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive.There is only **one repeated number** in nums, return *this repeated number*. Solve the problem using set.

**Example 1:**

**Input:** nums = [1,3,4,2,2]

**Output:** 2

**Example 2:**

**Input:** nums = [3,1,3,4,2]

**Output:** 3

**For example:**

| **Input** | **Result** |
| --- | --- |
| 1 3 4 4 2 | 4 |

**PROGRAM:**

**def find\_duplicate(nums):**

**num\_set = set()**

**for num in nums:**

**if num in num\_set:**

**return num**

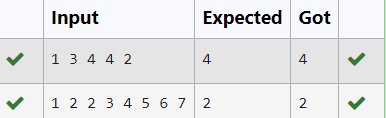
**else:**

**num\_set.add(num)**

**nums=list(map(int,input().split()))**

**print(find\_duplicate(nums))**

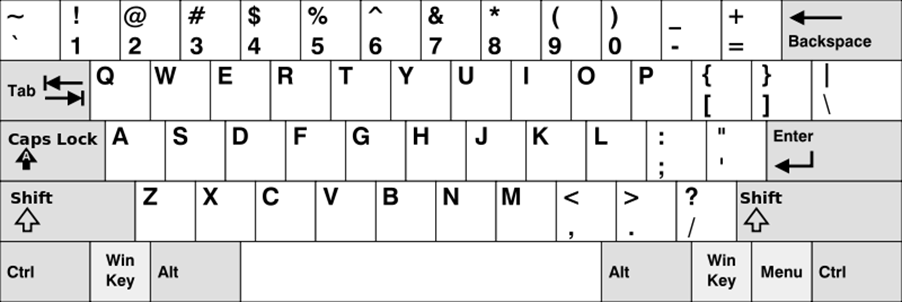
**OUTPUT:**

****

4.) Given an array of strings words, return *the words that can be typed using letters of the alphabet on only one row of American keyboard like the image below*.

In the **American keyboard**:

* the first row consists of the characters "qwertyuiop",
* the second row consists of the characters "asdfghjkl", and
* the third row consists of the characters "zxcvbnm".



**Example 1:**

**Input:** words = ["Hello","Alaska","Dad","Peace"]

**Output:** ["Alaska","Dad"]

**Example 2:**

**Input:** words = ["omk"]

**Output:** []

**Example 3:**

**Input:** words = ["adsdf","sfd"]

**Output:** ["adsdf","sfd"]

**For example:**

| **Input** | **Result** |
| --- | --- |
| 4  Hello  Alaska  Dad  Peace | Alaska  Dad |
| 2  adsfd  afd | adsfd  afd |

**PROGRAM:**

**def check(name,row):**

**for i in name:**

**if(not(i in row)):**

**return False**

**return True**

**name=[input() for i in range(int(input()))]**

**r1='qwertyuiopQWERTYUIOP'**

**r2='asdfghjklASDFGHJKL'**

**r3='zxcvbnmZXCVBNM'**

**result=[]**

**for i in name:**

**if(i[0] in r1):**

**x=r1**

**elif(i[0] in r2):**

**x=r2**

**else:**

**x=r3**

**if(check(i,x)):**

**result.append(i)**

**if result:**

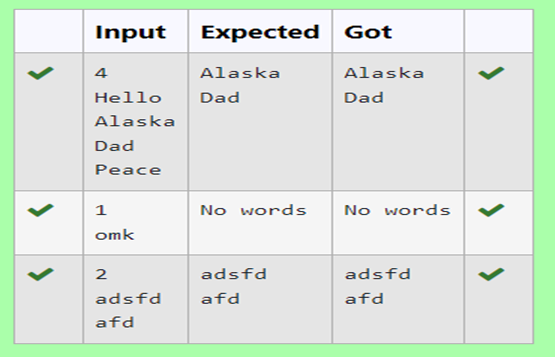
**for i in result:**

**print(i)**

**else:**

**print('No words')**

**OUTPUT:**

****

5.) 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.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 1,2,1,2,5  3 | 1 |
| 1,2  0 | 0 |

**PROGRAM:**

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

**t=tuple(int(i)for i in t)**

**k=int(input())**

**result=[]**

**for i in t:**

**for j in t:**

**if(i+j==k and not ([i,j] in result or [j,i]in result)):**

**result.append([i,j])**

**print(len(result))**

**OUTPUT:**

****