WEEK 7

Experiments based on Tuples,Sets and its Operations

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.

For example:

Input	Result
hello world ad	1
Faculty Upskilling in Python Programming Ak	2

Answer:(penalty regime: 0 %)

```
s=input().split()

b=input()

b1=b.upper()

broken=b+b1

a=[]

for i in s:

    for j in i:

        if j in broken:

            a.append(i)

            break

print(len(s)-len(a))
```

	Input	Expected	Got	
	hello world ad	1	1	
	Welcome to REC e	1	1	
	Faculty Upskilling in Python Programming ak	2	2	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **2**
Correct

Mark 1.00 out of 1.00

Flag question

Question text

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 Input:

5 4

1 2 8 6 5

2 6 8 10

Sample Output:

1 5 10

3

Sample Input:

5 5

1 2 3 4 5

1 2 3 4 5

Sample Output:

NO SUCH ELEMENTS

For example:

Input	Result
5 4 1 2 8 6 5 2 6 8 10	1 5 10 3

Answer:(penalty regime: 0 %)

```
a,b = map(int, input().split())

l1 = list(map(int, input().split()))

l2 = list(map(int, input().split()))

x = [i for i in l1 if i not in l2]

y = [i for i in l2 if i not in l1]

z = x+y

if z:

    print(' '.join(map(str, z)))

    print(len(z))
```

	Input	Expected	Got	
	5 4 1 2 8 6 5 2 6 8 10	1 5 10 3	1 5 10 3	
	3 3 10 10 10 10 11 12	11 12 2	11 12 2	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **3**
Correct

Mark 1.00 out of 1.00

Flag question

Question text
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

Answer:(penalty regime: 0 %)

```
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))
```

	Input	Expected	Got	
	5,6,5,7,7,8 13	2	2	
	1,2,1,2,5 3	1	1	

	Input	Expected	Got	
	1,2 0	0	0	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **4**
Correct

Mark 1.00 out of 1.00

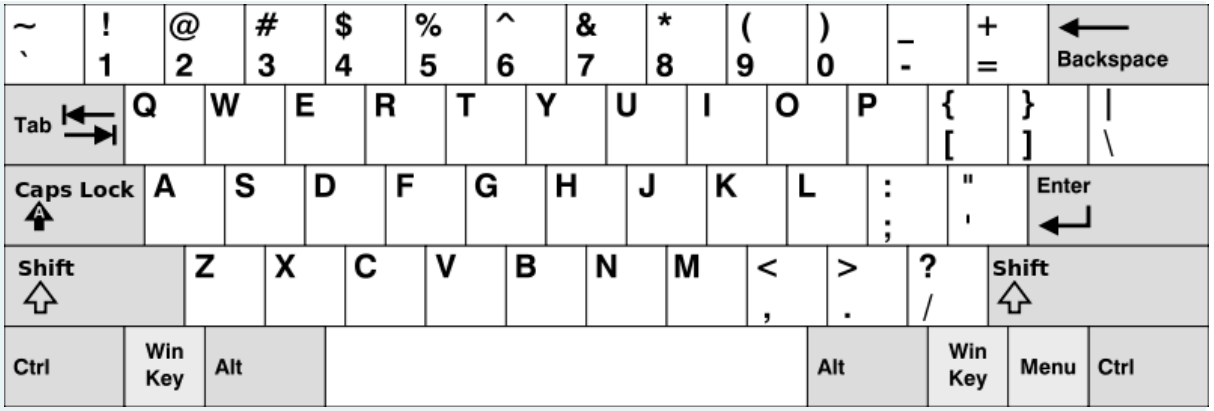
Flag question

Question text

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

Answer:(penalty regime: 0 %)

```
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')
```

	Input	Expected	Got	
	4 Hello Alaska Dad Peace	Alaska Dad	Alaska Dad	
	1 omk	No words	No words	
	2 adsfd afd	adsfd afd	adsfd afd	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **5**
Correct

Mark 1.00 out of 1.00

Flag question

Question text
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

Answer:(penalty regime: 0 %)

```
def findRepeatedSequences(s):
    sequence_count = {}
    result = set()
    for i in range (len(s) - 9):
        sequence = s[i:i+10]
        if sequence in sequence_count:
            result.add(sequence)
        else:
             sequence_count[sequence] = 1
    return list(result)
s = input()
repeated_sequences = findRepeatedSequences(s)
for sequence in repeated_sequences:
    print(sequence)
```

	Input	Expected	Got	
	AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA	AAAAACCCCC CCCCCAAAAA	
	AAAAAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.