GE19211 / GE23233 / GE23231 - PSPP/PUP

Quiz navigation
1 2 3 4 5
Show one page at a time
Finish review

Started on	Thursday, 23 May 2024, 8:02 PM
State	Finished
Completed on	Friday, 24 May 2024, 8:48 AM
Time taken	12 hours 46 mins
Marks	5.00/5.00
Grade	**100.00** out of 100.00

Question **1**
Correct
Mark 1.00 out of 1.00
Flag question

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 %)

```
n,m = map(int,input().split())
array1 = list(map(int,input().split()))
array2 = list(map(int,input().split()))
set1 = set(array1)
set2 = set(array2)
symmentric_diff = set1.symmetric_difference(set2)
non_repeating_elements=[x for x in symmentric_diff if x not in set1 or x not in set2]
if non_repeating_elements:
    print(*non_repeating_elements)
    print(len(non_repeating_elements))
else:
    print("NO SUCH ELEMENTS")
```

	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 **2**
Correct
Mark 1.00 out of 1.00
Flag question

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 %)

```
s = input()
A = set()
B = set()
for i in range(len(s) - 9):
    C = s[i:i + 10]
    if C in A:
        B.add(C)
    else:
        A.add(C)
for seq in B:
    print(seq)
```

	Input	Expected	Got	
✓	AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA	AAAAACCCCC CCCCCAAAAA	✓
✓	AAAAAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	✓

Passed all tests! ✓

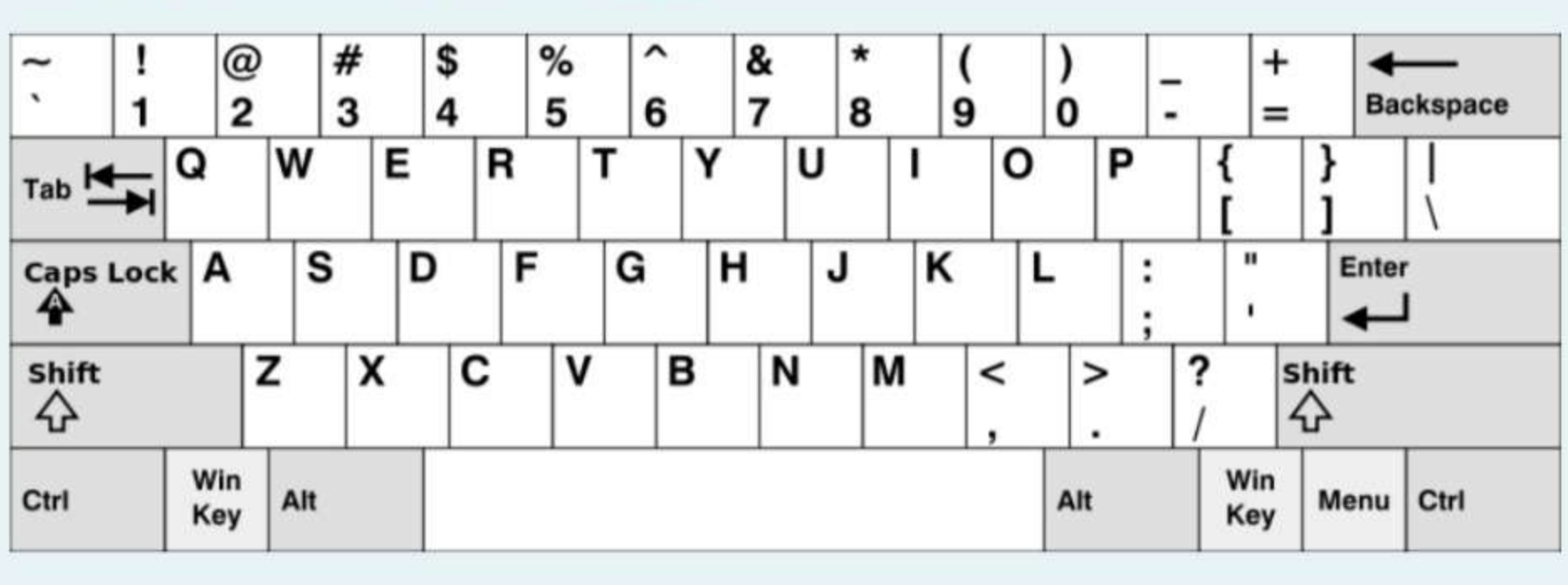
Correct
Marks for this submission: 1.00/1.00.

Question **3**
Correct
Mark 1.00 out of 1.00
Flag question

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 %)

```
A = int(input())
words = [input() for _ in range(A)]
rows = [set("qwetyuiop"), set("asdfghjkl"),set("zxcvbnm")]
result=[word for word in words if any(set(word.lower()).issubset(row) for row in rows)]
if result:
    print("\n".join(result))
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 **4**
Correct
Mark 1.00 out of 1.00
Flag question

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 %)

```
def countWords(text, brokenLetters):
    brokenSet = set(brokenLetters)
    words = text.split(' ')
    count = 0
    for word in words:
        if not set(word) & brokenSet:
            if not set(word) & brokenSet:
                count +=1
    return count
text = input().lower()
brokenLetters=input()
print(countWords(text, brokenLetters))
```

	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 **5**
Correct
Mark 1.00 out of 1.00
Flag question

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(map(int,input().split(',')))
k = int(input())
seen = {}
distinct_pairs=set()
for num in t:
    complement = k-num
    if complement in seen and seen[complement]>0:
        distinct_pairs.add((min(num , complement), max(num, complement)))
    else:
        seen[num]= seen.get(num, 0) + 1
print(len(distinct_pairs))
```

	Input	Expected	Got	
✓	5,6,5,7,7,8 13	2	2	✓
✓	1,2,1,2,5 3	1	1	✓
✓	1,2 0	0	0	✓

Passed all tests! ✓

Correct
Marks for this submission: 1.00/1.00.

Finish review

◄ Week7_MCQ
Jump to...
Dictionary ►