WEEK-08-CODING-Tuple-Set

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.

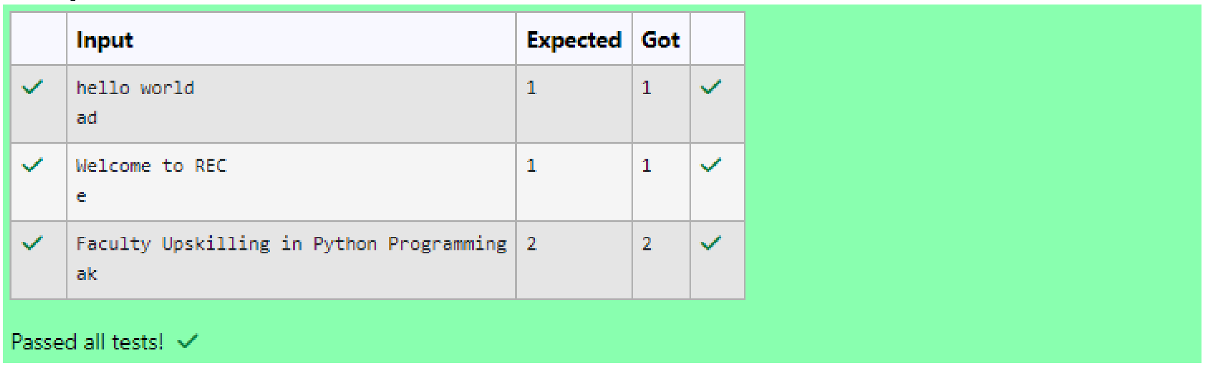
For example:

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

Coding:

```
def main(t,l):
    b = set(l.lower())
    ws = t.split()
    c= 0
    for w in ws:
        if all(letter.lower() not in b for letter in w):
            c +=1
    return c

t = input().strip()
b = input().strip()
res = main(t,b)
print(res)
```

Output:



	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! ✓

2. 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

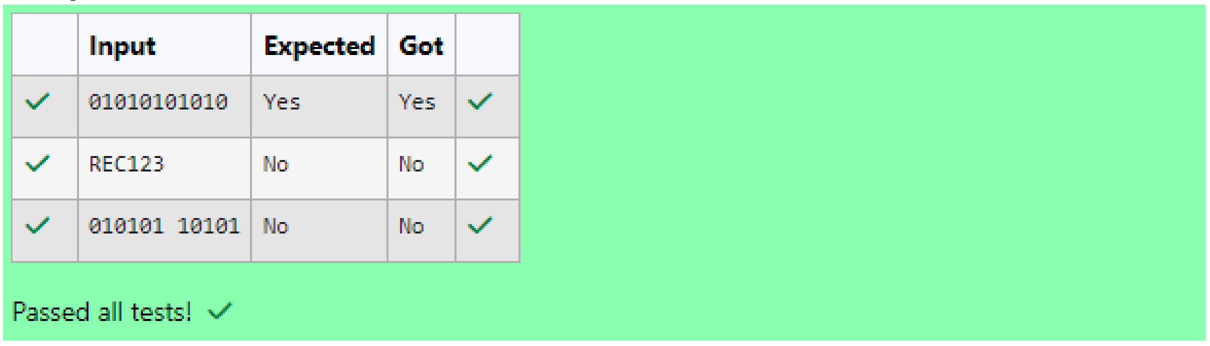
For example:

Input	Result
01010101010	Yes
010101 10101	No

Coding:

```
s=input()
c=set(s)
if c <={'0', '1'}:
    print("Yes")
else:
    print("No")
```

Output:



	Input	Expected	Got	
✓	01010101010	Yes	Yes	✓
✓	REC123	No	No	✓
✓	010101 10101	No	No	✓

Passed all tests! ✓

3. 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:

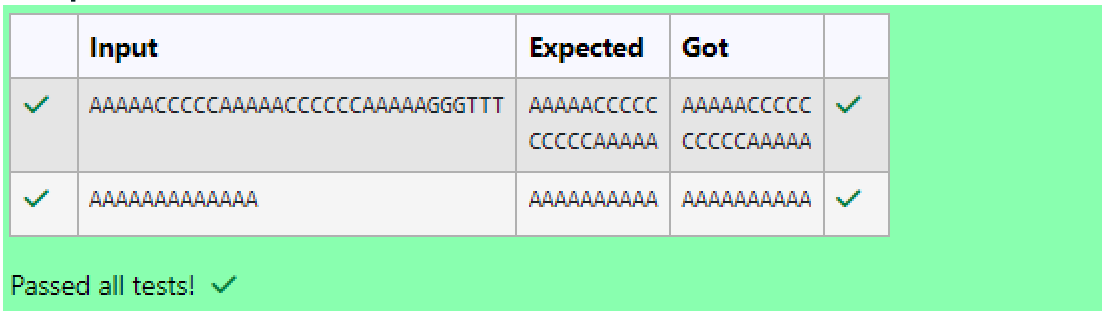
Input: s = "AAAAAAAAAAAAA"
Output: ["AAAAAAAAAA"]
For example:

Input	Result
AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA

Coding:

```
def show(s):
    se = set()
    r = set()
    for i in range(len(s)-9):
        sq = s[i:i+10]
        if sq in se:
            r.add(sq)
        else:
            se.add(sq)
    return list(r)

n = input().strip()
res = show(n)
for resut in res:
    print(resut)
```

Output:



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

Passed all tests! ✓

4. 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:

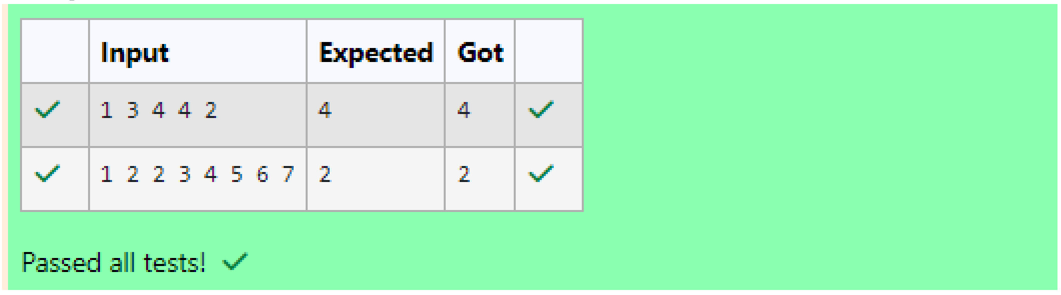
Input: nums = [3,1,3,4,2]
Output: 3
For example:

Input	Result
1 3 4 4 2	4

Coding:

```
def show(num):
    s=set()
    for n in num:
        if n in s:
            return n
        s.add(n)
    return -1

i = input()
num = list(map(int,i.split()))
res = show(num)
print(res)
```

Output:



	Input	Expected	Got	
✓	1 3 4 4 2	4	4	✓
✓	1 2 2 3 4 5 6 7	2	2	✓

Passed all tests! ✓

5. You are given an integer tuple nums containing distinct numbers. Your task is to perform a sequence of operations on this tuple until it becomes empty. The operations are defined as follows:

1. If the first element of the tuple has the smallest value in the entire tuple, remove it.
2. Otherwise, move the first element to the end of the tuple.

You need to return an integer denoting the number of operations required to make the tuple empty.

Constraints

- The input tuple nums contains distinct integers.
- The operations must be performed using tuples and sets to maintain immutability and efficiency.
- Your function should accept the tuple nums as input and return the total number of operations as an integer.

Example:

Input: nums = (3, 4, -1)
Output: 5

Explanation:
Operation 1: [3, 4, -1] -> First element is not the smallest, move to the end -> [4, -1, 3]
Operation 2: [4, -1, 3] -> First element is not the smallest, move to the end -> [-1, 3, 4]
Operation 3: [-1, 3, 4] -> First element is the smallest, remove it -> [3, 4]
Operation 4: [3, 4] -> First element is the smallest, remove it -> [4]
Operation 5: [4] -> First element is the smallest, remove it -> []
Total operations: 5

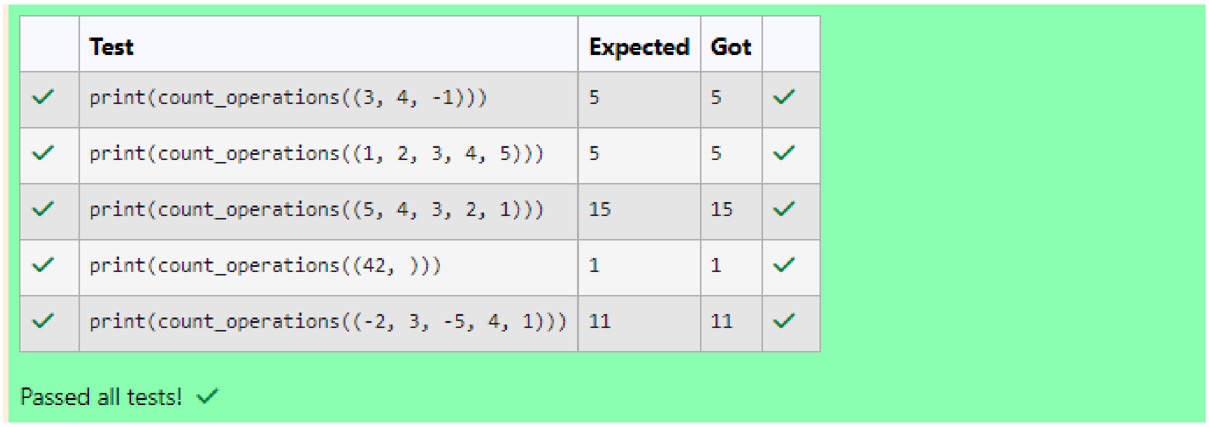
For example:

Test	Result
print(count_operations((3, 4, -1)))	5

Coding:

```
def count_operations(nums: tuple) -> int:
    # Your implementation here
    from collections import deque

    nums = deque(nums)
    c = 0
    while nums:
        m = min(nums)
        if nums[0] == m:
            nums.popleft()
        else:
            nums.append(nums.popleft())

        c +=1

    return c
```

Output:



	Test	Expected	Got	
✓	`print(count_operations((3, 4, -1)))`	5	5	✓
✓	`print(count_operations((1, 2, 3, 4, 5)))`	5	5	✓
✓	`print(count_operations((5, 4, 3, 2, 1)))`	15	15	✓
✓	`print(count_operations((42, )))`	1	1	✓
✓	`print(count_operations((-2, 3, -5, 4, 1)))`	11	11	✓

Passed all tests! ✓

6. 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

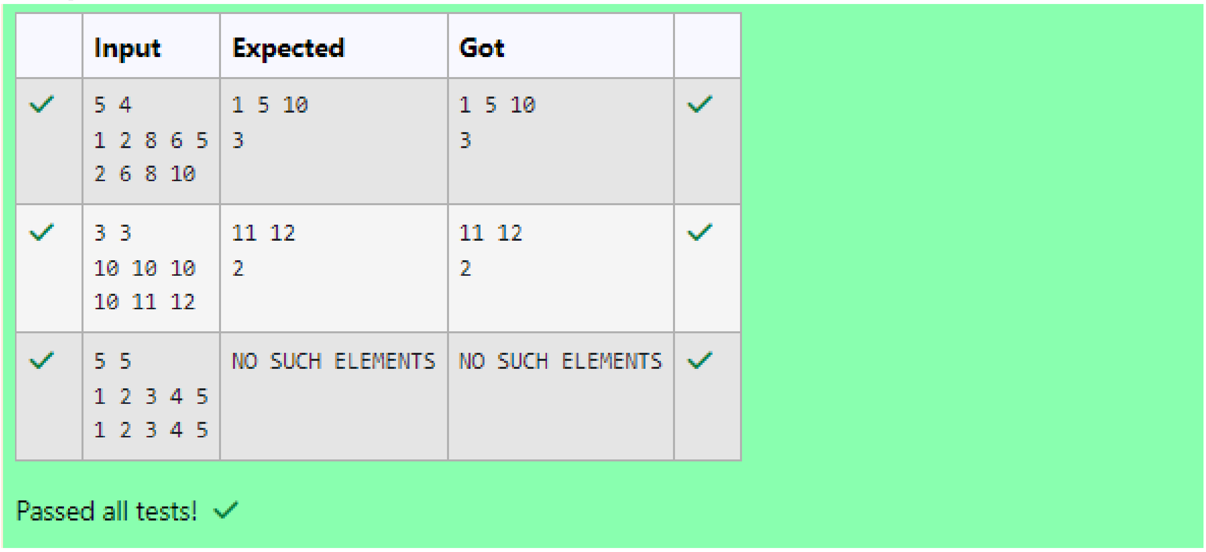
5 4
1 2 8 6 5
2 6 8 10
	1 5 10
	3

5 5
1 2 3 4 5
1 2 3 4 5
	NO SUCH ELEMENTS

Coding:

```
def main(ar1,ar2):
    s1 = set(ar1)
    s2 = set(ar2)
    c = s1.intersection(s2)

    u1 = s1 -c
    u2 = s2 - c

    u = u1.union(u2)
    return list(u)

z = input().strip().split()
z1,z2 = int(z[0]),int(z[1])

a1 = list(map(int,input().strip().split()))
a2 = list(map(int,input().strip().split()))

res = main(a1,a2)

if res:
    print(" ".join(map(str,sorted(res))))
    print(len(res))
else:
    print("NO SUCH ELEMENTS")
```

Output:



	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	✓
✓	5 5 1 2 3 4 5 1 2 3 4 5	NO SUCH ELEMENTS	NO SUCH ELEMENTS	✓

Passed all tests! ✓

7. Program to print all the distinct elements in an array. Distinct elements are nothing but the unique (non-duplicate) elements present in the given array.

Input Format:

First line take an Integer input from stdin which is array length n.

Second line take n Integers which is inputs of array.

Output Format:

Print the Distinct Elements in Array in single line which is space Separated

Example Input:

5

1 2 2 3 4

Output:

1 2 3 4

Example Input:

6

1 1 2 2 3 3

Output:

1 2 3

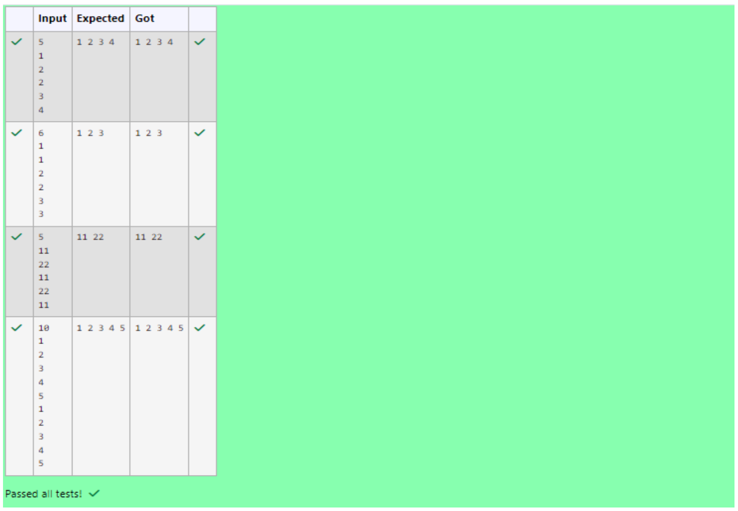
For example:

Input	Result
5 1 2 2 3 4	1 2 3 4

Coding:

```
def show(arr):
    d = set(arr)
    print(" ".join(map(str,sorted(d))))
n = int(input())
arr =[int(input().strip()) for _ in range(n)]
show(arr)
```

Output:



	Input	Expected	Got	
✓	5 1 2 2 3 4	1 2 3 4	1 2 3 4	✓
✓	6 1 1 2 2 3 3	1 2 3	1 2 3	✓
✓	5 11 22 11 22 11	11 22	11 22	✓
✓	10 1 2 3 4 5 1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	✓

Passed all tests! ✓

8. Check if a set is a subset of another set.

Example:

Sample Input1:

mango apple

mango orange

mango

output1:

yes

set3 is subset of set1 and set2

input2:
mango orange
banana orange
grapes
output2:
no

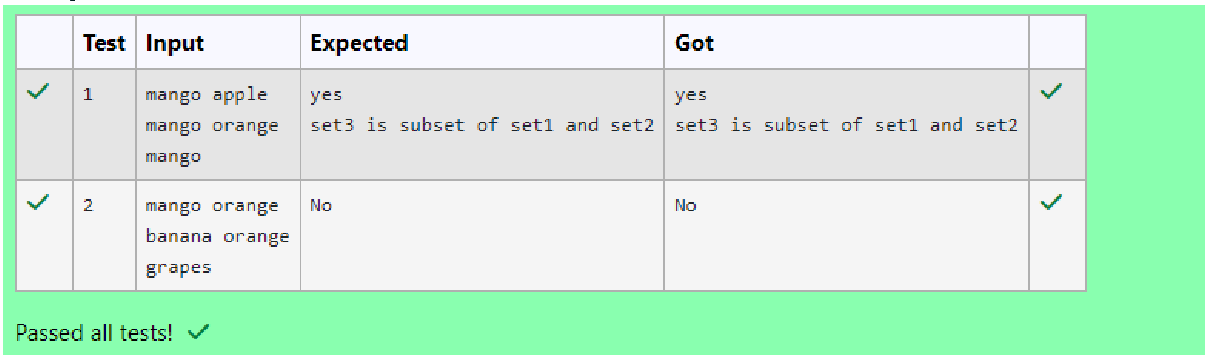
For example:

Test	Input	Result
1	mango apple mango orange mango	yes set3 is subset of set1 and set2
2	mango orange banana orange grapes	No

Coding:

```
def main(s1,s2,ss):
    s1 = set(s1.split())
    s2 = set(s2.split())
    ss = set(ss.split())

    if ss.issubset(s1) and ss.issubset(s2):
        print("yes")
        print("set3 is subset of set1 and set2")
    else:
        print("No")

s1 = input().strip()
s2 = input().strip()
ss = input().strip()

res = main(s1,s2,ss)
```

Output:



	Test	Input	Expected	Got	
✓	1	mango apple mango orange mango	yes set3 is subset of set1 and set2	yes set3 is subset of set1 and set2	✓
✓	2	mango orange banana orange grapes	No	No	✓

Passed all tests! ✓

9. 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.

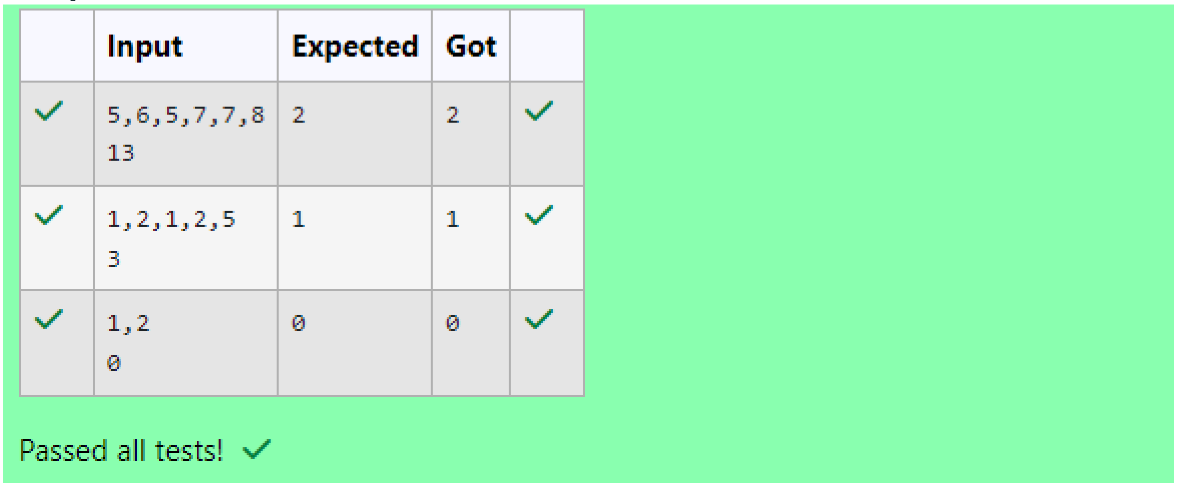
For example:

Input	Result
1,2,1,2,5 3	1
1,2 0	0

Coding:

```
def show(t,K):
    s=set()
    p=set()

    for num in t:
        com = K- num
        if com in s:
            p.add((min(num, com), max(num, com)))
        s.add(num)
    return len(p)

tup = input()
t = tuple(map(int , tup.split(',')))
K = int(input())
res=show(t,K)
print(res)
```

Output:



	Input	Expected	Got	

✓	`5,6,5,7,7,8`
	`13`
		`2`	`2`	✓

✓	`1,2,1,2,5`
	`3`
		`1`	`1`	✓

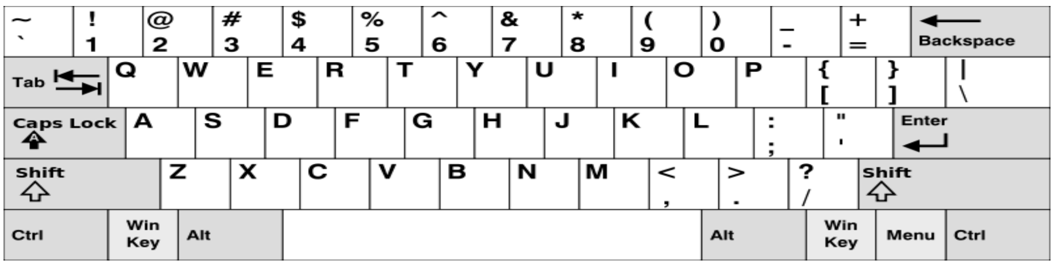
✓	`1,2`
	`0`
		`0`	`0`	✓

Passed all tests! ✓

10. 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".


~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = Backspace
Tab Q W E R T Y U I O P { [}] | \
Caps Lock A S D F G H J K L : ; " ' Enter
Shift Z X C V B N M < , > . ? / Shift
Ctrl Win Key Alt Alt Win Key Menu Ctrl

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:

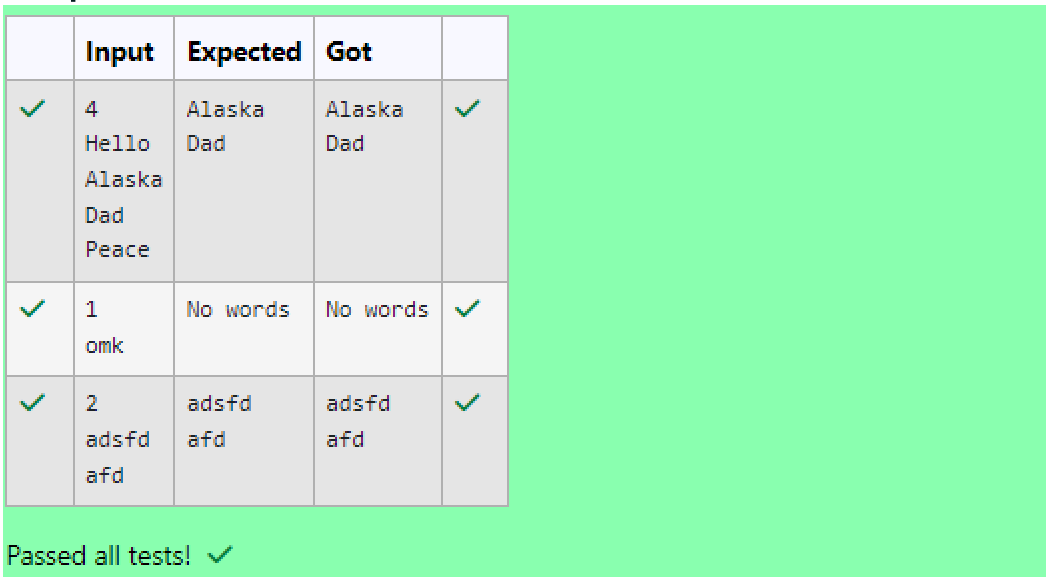
Input	Result
4 Hello Alaska Dad Peace	Alaska Dad

Input	Result
2 adsfd afd	adsfd afd

Coding:

```
def word(w):
    r1=set("qwertyuiop")
    r2=set("asdfghjkl")
    r3=set("zxcvbnm")
    res=[]
    for wo in w:
        l = set(wo.lower())

        if l <= r1 or l <= r2 or l <= r3:
            res.append(wo)
    return res
i = int(input())
w=[]
for _ in range(i):
    wo = input()
    w.append(wo)
res =word(w)
if res:
    for wo in res:
        print(wo)
else:
    print("No words")
```

Output:



	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! ✓