Started on	Friday, 10 May 2024, 7:56 AM
State	Finished
Completed on	Saturday, 18 May 2024, 2:58 PM
Time taken	8 days 7 hours
Overdue	5 days 7 hours
Marks	5.00/5.00
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 1.00 out of 1.00

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

Answer: (penalty regime: 0 %)

```
a=list(map(int,input().split()))
for i in a:
    if a[i] in a[:i]:
        print(a[i])
        break
```

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

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **2**

Correct

Mark 1.00 out of 1.00

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

Answer: (penalty regime: 0 %)

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

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

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **3**

Correct

Mark 1.00 out of 1.00

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())
c=list(map(int, input().split()))
d=list(map(int, input().split()))
k=[x for x in c if x not in d]+[x for x in d if x not in c]
if k==[]:
    print("NO SUCH ELEMENTS")
for i in k:
    print(i,end=" ")
print("\n",len(k),sep="")
```

	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 **4**

Correct

Mark 1.00 out of 1.00

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 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)
```

	Input	Expected	Got	
✔	AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA	AAAAACCCCC CCCCCAAAAA	✔

	Input	Expected	Got	
✔	AAAAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **5**

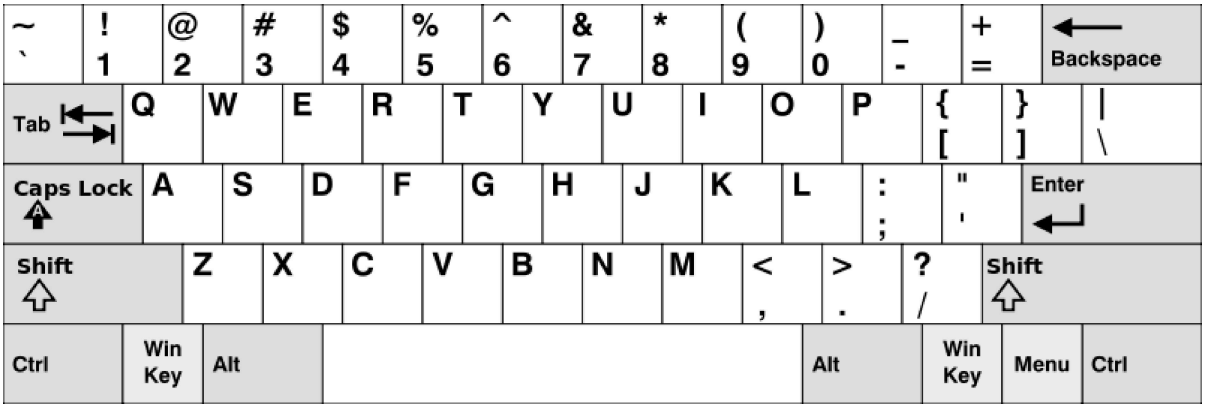
Correct

Mark 1.00 out of 1.00

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 %)

```
n=int(input())
l=[]
x=1
y=1
z=1
```

```
a=0
r1="qwertyuiop"
R1="QWERTYUIOP"
r2="asdfghjkl"
R2="ASDFGHJKL"
r3="zxcvbnm"
R3="ZXCVBNM"
for i in range(n):
    l.append(input())
for i in range(n):
    s=l[i]
    x=1
    y=1
    z=1
    for j in range(len(s)):
        if(s[j] in r1):
            x*=1
        elif(s[j] in R1):
            x*=1
        else:
            x*=0
        if(s[j] in r2):
            y*=1
        elif(s[j] in R2):
            y*=1
        else:
            y*=0
        if(s[j] in r3):
            z*=1
        elif(s[j] in R3):
            z*=1
        else:
            z*=0
    if(x==1 or y==1 or z==1):
        print(s)
        a+=1
    s=""
if(a==0):
    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.

◄ Week7_MCQ

Jump to...

Week8_MCQ ►