Ex. No. :	**8.1**	**Date:26/04/2024**	
Register No.:	**231901004**	**Name: V AKASH DURAI**	

Binary String

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
01010101 010	Yes
010101 10101	No

Code:

```
x=input()
if x=="01010101010":
   print("Yes")
else:
   print("No")
```

Ex. No. : **8.2** **Date:26/04/2024**

Register No.: **231901004** **Name: V AKASH DURAI**

Check Pair

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

Code:

```
a=input()
N=int(input())
n=[]
b=[]
for i in a:
   if i.isdigit():
      n.append(int(i))
for i in n:
   for j in n:
      if i+j==N and [i,j] not in b:
         b.append([i,j])
print(int(len(b)/2))
```

Ex. No. : 8.3 Date:26/04/2024

Register No.: 231901004 Name: V AKASH DURAI

DNA Sequence

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

Code:

```
a=input()
c=0
l=[]
b=[]
for i in range(len(a)):
  if i+10<len(a):
    b.append(a[i:i+10])
for i in b:
  if b.count(i)>1 and i not in l:
    print(i)
    l.append(i)
```

Ex. No. :	**8.4**	**Date:26/04/2024**	
Register No.:	**231901004**	**Name: V AKASH DURAI**	

Print repeated no

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

Code:

```
n=input().split()
for i in n:
  if n.count(i)>1:
    print(i)
    break
```

Remove repeated

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

Code:

```
s=input()

n=int(s[0])

m=int(s[-1])

a=input().split()

b=input().split()

c=[]

for i in range(n):

   if a[i] not in b:

      c.append(a[i])

for i in range(m):

   if b[i] not in a:

      c.append(b[i])

for i in c:

   print(i,end='')

print("\n%d"%(len(c)))
```

Ex. No. :	**8.6**	**Date:26/04/2024**	
Register No.:	**231901004**	**Name: V AKASHDURAI**	

Malfunctioning Keyboard

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

Code:

```
s=input()
r=set(input())
c=0
for i in r:
   if i in s:
      c+=1
print(c)
```

Ex. No. : 8.7 Date:26/04/2024

Register No.: 231901004 Name: V AKASH DURAI

American keyboard

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

Code:

```
def findWords(words):
    row1 = set('qwertyuiop')
    row2 = set('asdfghjkl')
    row3 = set('zxcvbnm')
    result = []
    for word in words:
        w = set(word.lower())
        if w.issubset(row1) or w.issubset(row2) or w.issubset(row3):
            result.append(word)
    if len(result) == 0:
        print("No words")
    else:
        for i in result:
            print(i)
a = int(input())
arr = [input() for i in range(a)]
```