

Examples:

Input: str = "01010101010"

Output: Yes

Input: str = "REC101"

Output: No

Input	Result
01010101010	Yes
010101 10101	No

Ex. No. : 8.1 Date:

Register No.: Name:

# **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.

```
a=input()
b=list(a)
flag=1
for i in range(0,len(b)):
    if(b[i]=='1' or b[i]=='0'):
        continue
    else:
        flag=0
        break
if(flag==1):
        print("Yes")
else:
    print("No")
```

## **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.

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

Ex. No. : 8.2 Date:

Register No.: Name:

# **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.

```
a=input()
 aaa=int(input())
 bbb=[]
 aaaaaaaaaaa=0
 l=[]
 for j in range(0, len(a)):
    if(a[j]!=','):
      l.append(int(a[j]))
aa=tuple(l)
     for k in range(0,len(aa)):
       for bb in range(k,len(aa)):
           if(bb!=k):
               if((aa[k] not in bbb) and (aa[bb] not in bbb)):
                  if(aa[k]+aa[bb]==aaa):
                        aaaaaaaaaaaa=aaaaaaaaaaaa+1
                       bbb.append(aa[k])
                       bbb.append(aa[bb])
print(aaaaaaaaaaa)
```

Input: s = "AAAAACCCCCCAAAAACCCCCCAAAAAGGGTTT"

Output: ["AAAAACCCCC","CCCCCAAAAA"]

Example 2:

Input: s = "AAAAAAAAAAA" Output: ["AAAAAAAAAA"]

Input	Result
AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCC CCCCAAAAA

Ex. No. : 8.3 Date:

Register No.: Name:

## **DNA Sequence**

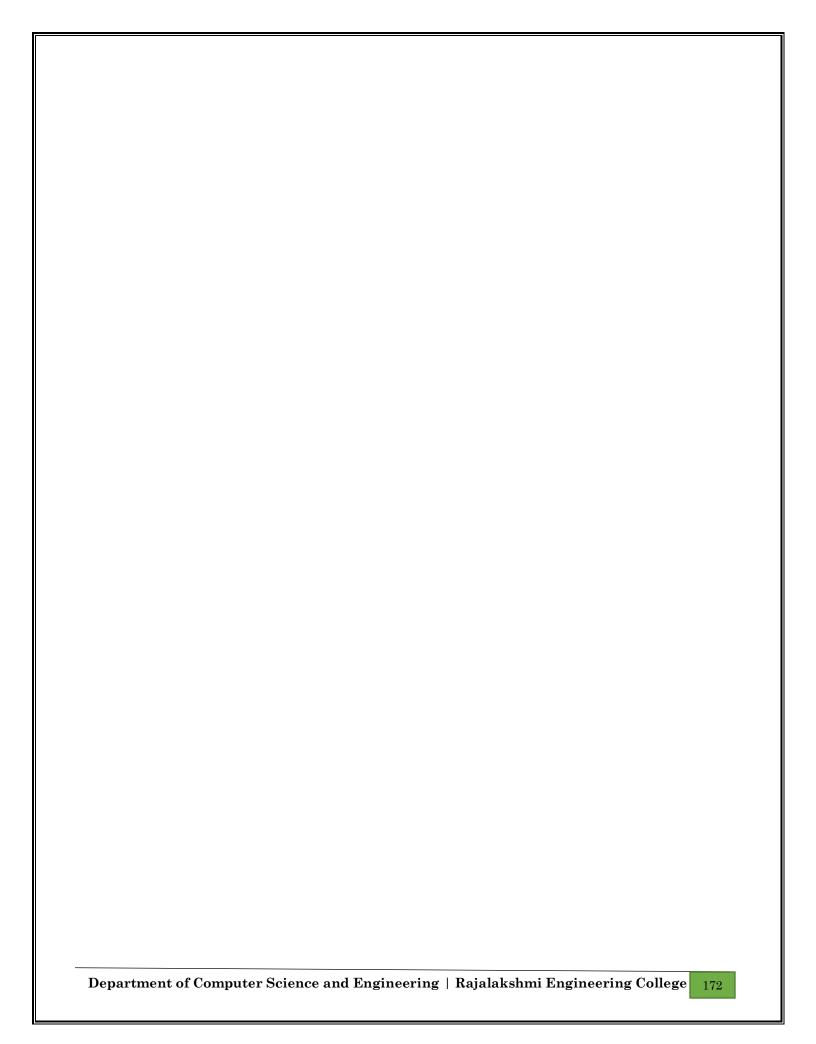
The **DNA sequence** is composed of a series of nucleotides abbreviated as 'A', 'C', 'G', and 'T'.

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**.

```
a=input()
aaa=[]
bb=[]
bbbb=[]
for m in range(0,len(a)):
  b=a[m:m+10]
  if(len(b)==10):
      bb.append(b)
for p in range(0,len(bb)):
  if(bb[p] not in bbbb):
  if((bb.count(bb[p]))>1):
      print(bb[p])
  bbbb.append(bb[p])
```



**Input:** nums = [1,3,4,2,2]

Output: 2

## Example 2:

**Input:** nums = [3,1,3,4,2]

Output: 3

Input	Result
1 3 4 4 2	4

Ex. No.	:	8.4	Date:
Register No.	. <b>:</b>		Name:

# 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  $\underline{set}$ .

```
n=input().split()
for i in n:
  if n.count(i)>=2:
    print(i)
    break
```

Sample Input:

5 4

 $1\,2\,8\,6\,5$ 

26810

Sample Output:

 $15\ 10$ 

3

Sample Input:

5 5

 $1\ 2\ 3\ 4\ 5$ 

 $1\ 2\ 3\ 4\ 5$ 

Sample Output:

NO SUCH ELEMENTS

Input	Result
54 $12865$ $26810$	1 5 10 3

Ex. No. : 8.5 Date:

Register No.: Name:

## 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.

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

bb=[]

cc=[]

aaa=[]

aaaaaa=input()

bb=aaaaaa.split(" ")

bbbbbb=input()

cc=bbbbbb.split(" ")

for m in range(0, len(bb)):

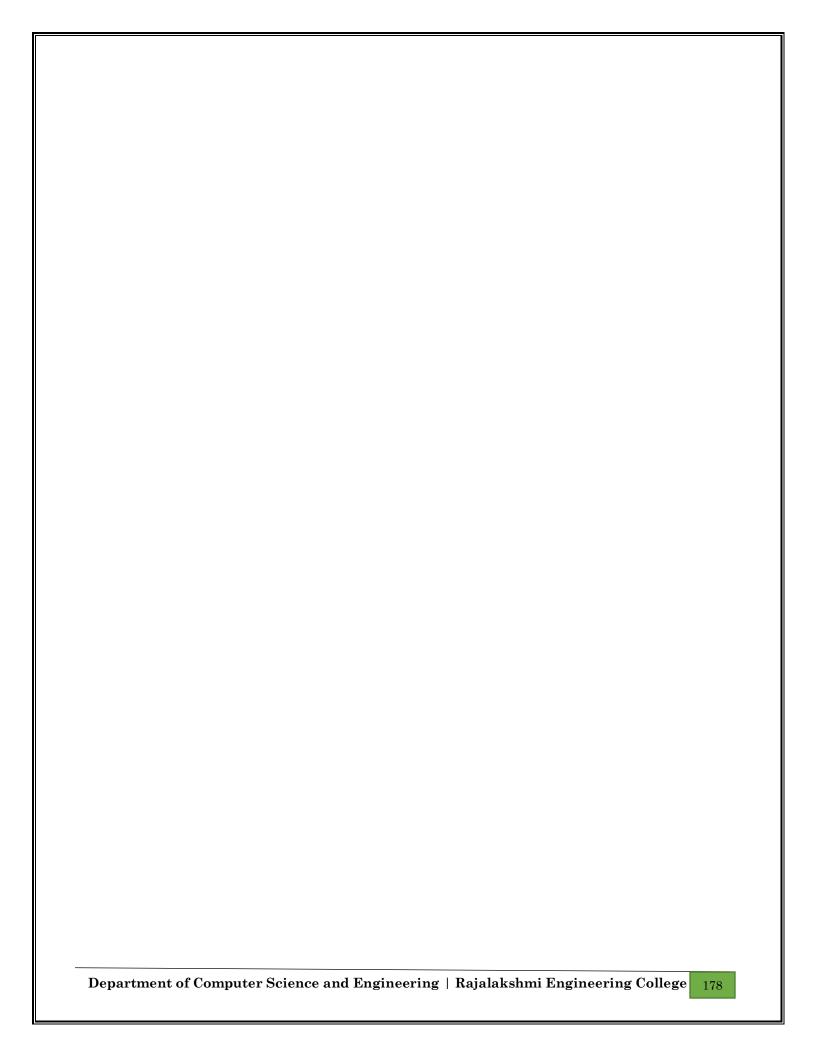
    if(bb[m] not in aaa):

    if(bb[m] not in cc):

        aaa.append(bb[m])

for p in range(0,len(cc)):

    if(cc[p] not in aaa):
```



Input: text = "hello world", brokenLetters = "ad"

Output:

1

Explanation: We cannot type "world" because the 'd' key is broken.

Input	Result
hello world ad	1

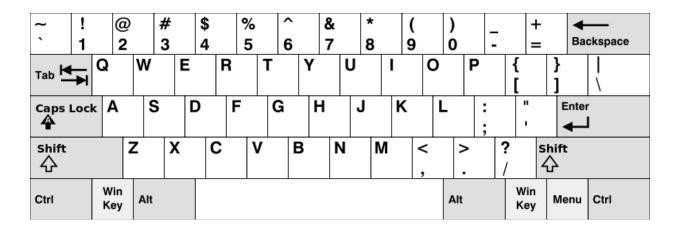
Ex. No.	:	8.6	Date:
Register No.	:		Name:

## **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.

```
a=input()
aa=a.lower()
b=input()
c=list(b)
d=aa.split()
b=len(d)
cccc=[]
for j in range(0,b):
    for aaaa in range(0,len(c)):
        if(c[aaaa] in d[j]):
        if(d[j] not in cccc):
        cccc.append(d[j])
print(len(d)-len(cccc))
```



Input: words = ["Hello","Alaska","Dad","Peace"]

Output: ["Alaska","Dad"]

Example 2:

Input: words = ["omk"]

Output: []
Example 3:

Input: words = ["adsdf","sfd"]
Output: ["adsdf","sfd"]

Input	Result
4 Hello Alaska Dad Peace	Alaska Dad

Ex. No.	:	8.7	Date:
Register No	<b>.:</b>		Name:

# 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".

