## 08 – Tuple/Set

**Ex. No. : 8.1 Date:30/5/24 Register No.: 231401036 Name:Hemasree.B**

### 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:**

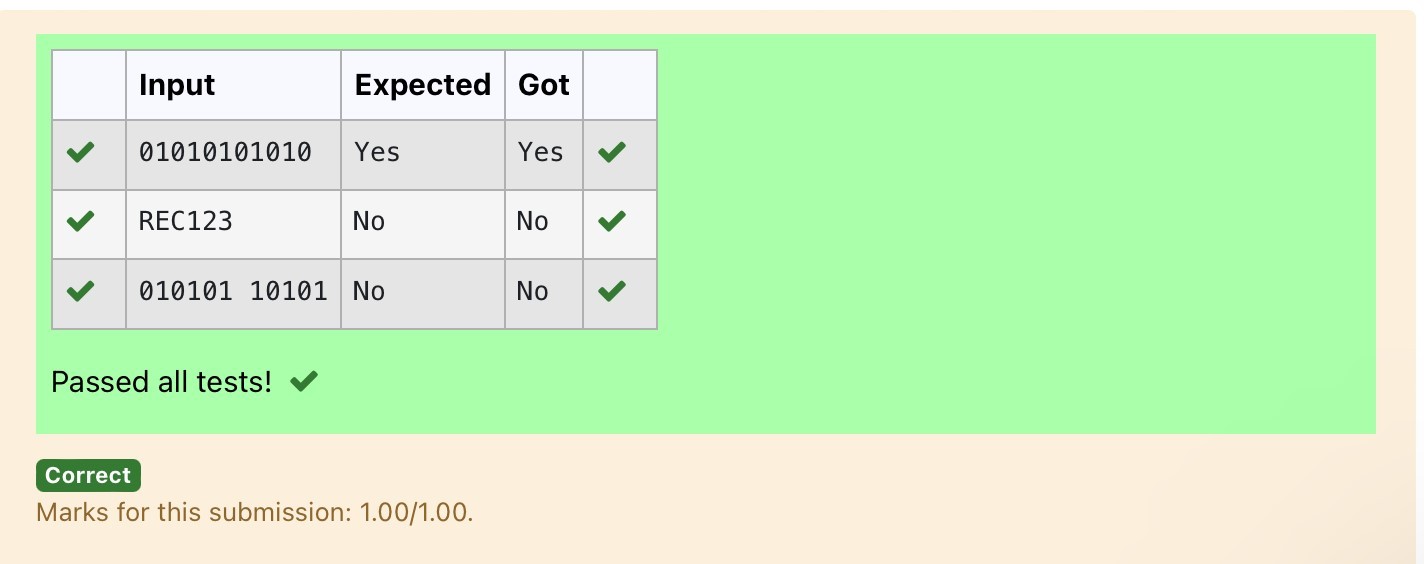
|  |  |
| --- | --- |
| **Input** | **Result**  Yes |
| 01010101010 |
| 010101 10101 | No |

# Program:

a = input() try: c = int(a)

print("Yes") except: print("No")

# Output:



**Ex. No: 8.2 Date:30/5/24**

**Register No.:231401036 Name:Hemasree**

## 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 |
| 1,2,1,2,5  3 |
| 1,2  0 | 0 |

# Program:

t = input() k = int(input()) a = t.split(",") l = [int(x) for x in a] count = 0 x = set()

for i in range(len(l)):

for j in range(i + 1, len(l)): if l[i] + l[j] == k: s = (l[i], l[j]) if s not in x and (l[j], l[i]) not in x:

count += 1 x.add(s)

print(count)

# Output:

**Ex. No: 8.3 Date:30/5/24**

Regi**ster No:231401036 Name: Hemasree**

### 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-letterlong**

sequences (substrings) that occur more than once in a DNA molecule. You may return the answer in **any order**.

**Ex. No. : Date:**

**Register No.: Name:**

**Example 1:**

**Input:** s = "AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT"

**Output:** ["AAAAACCCCC","CCCCCAAAAA"] **Example 2:**

**Input:** s = "AAAAAAAAAAAAA"

**Output:** ["AAAAAAAAAA"]

**For example:**

|  |  |
| --- | --- |
| **Input** | **Result** |
| AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT | AAAAACCCCC CCCCCAAAAA |

# Program:

s = input() j

= []

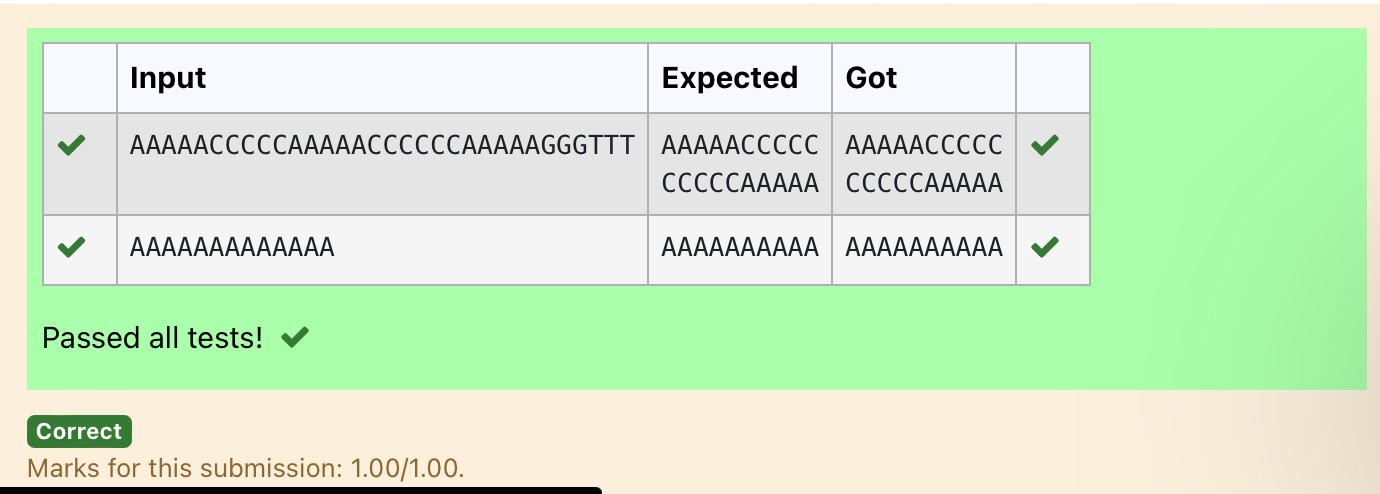
repeated = set() for i in range(len(s) - 9): sequence = s[i:i+10]

if sequence in j:

repeated.add(sequence) else:

j.append(sequence) l=list(repeated) l=list(reversed(l)) for i in l: print(i)

# Output:



**Ex.No:8.4 Date:30/5/24 Register No:231401039. Name: A.isaipriya**

### 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 |

# Program:

**Ex. No. : Date:**

**Register No.: Name:**

n =input().split(" ") n

= list(n) for i in range(len(n)):

for j in range(i+1,len(n)): if n[i] == n[j]:

print(n[i]) exit(0)

# Output:

**Ex. No:8.5. Date:30/5/24 Register No:231401036 Name: Hemasree**

### 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 | 1 5 10  3 |
| 2 6 8 10 |  |

# Program:

**Ex. No. : Date:**

**Register No.: Name:**

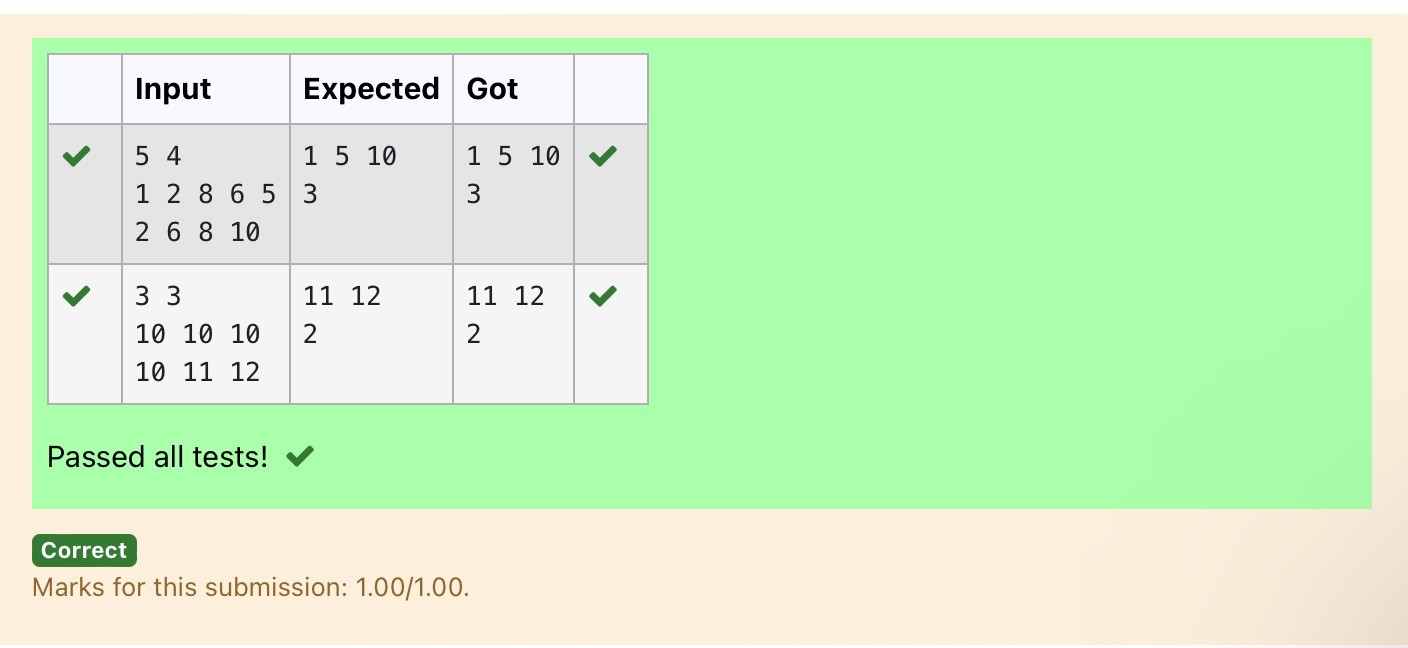
a=input() d=[] b=input() c=input() b=tuple(b.split(" "))

c=tuple(c.split(" ")) for i in b: if i not in c:

d.append(i) for i in c: if i not in b:

d.append(i) for i in range(len(d)): print(int(d[i]),end=' ') print() print(len(d))

# Output:



**Ex. No. : 8.6 Date:30/5/24 Register No.:231401036 Name:Hemasree**

### 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:**

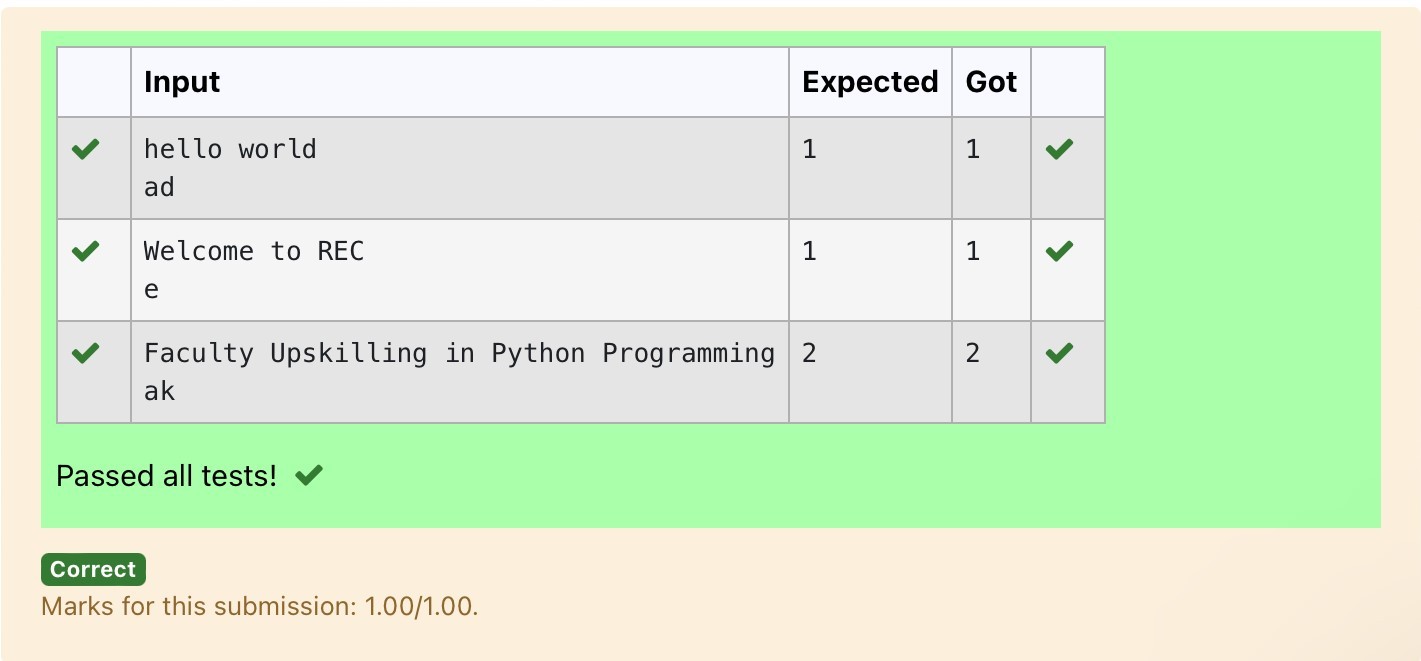
|  |  |
| --- | --- |
| **Input** | **Result** |
| hello world ad | 1 |

# Program:

### a=input() b=input() c=set() for i in a: for j

**in b: if j in i:**

**c.add(i) print(len(c))** Output:



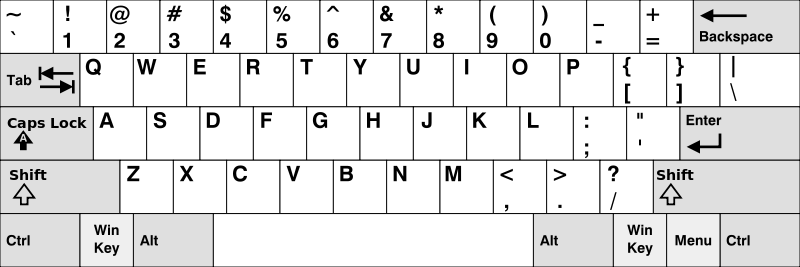
**Ex. No. : 8.7 Date: 30/5/24 Register No.: 231401036 Name:Hemasree**

### 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:**

**Result**

**Input**

|  |  |
| --- | --- |
| 4 | Alaska |
| Hello | Dad |
| Alaska |  |
| Dad |  |
| Peace |  |

# Program:

**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)] findWords(arr)**

# Output:

