.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

135

# 08 – Tuple/Set

**Register No.: 231501046 Name: Ganesh S**

## 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** |
| 01010101010 | Yes |
| 010101 10101 | No |

**Program:** str=input() bset={'0','1'}

flag=0

if set(str).issubset(bset): print("Yes")

else:

print("No")

.

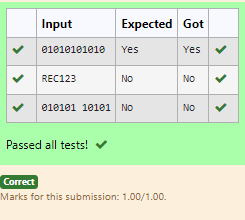
**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

136

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

137



**Register No.: 231501046 Name: Ganesh S**

# 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 |

**Program:**

t = eval(input()) k = int(input()) seen = set() pairs = set() count = 0

for num in t: complement = k - num

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

138

if complement in seen:

.

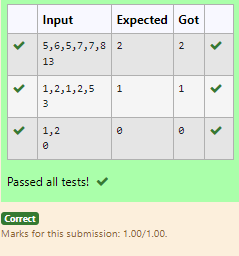
**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

139

pair = (min(num, complement), max(num, complement)) if pair not in pairs:

pairs.add(pair) count += 1

seen.add(num) print(count) **Output:**



## 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 |

**Program:**

s = input() sequences = {} result = []

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

sequences[substring] = sequences.get(substring, 0) + 1 for substring, frequency in sequences.items():

if frequency > 1:

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

140

result.append(substring) for i in result:

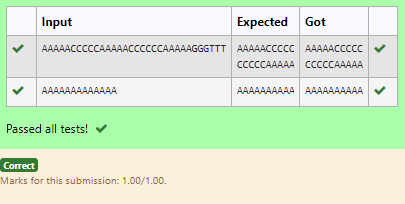
.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

141

print(i)

### Output:



**Register No.: 231501046 Name: Ganesh S**

## 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:** nums = input() l=nums.split() seen = set()

for num in l:

if num in seen: r=num

seen.add(num) print(r)

.

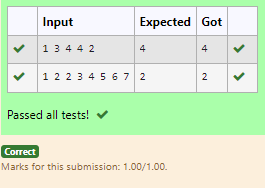
**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

142

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

143



.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

144

**Register No.: 231501046 Name: Ganesh S**

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

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

145

**Register No.: 231501046 Name: Ganesh S**

## 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:**

text = input().upper() brokenLetters = input().upper() broken\_set = set(brokenLetters) count = 0

for word in text.split():

if all(letter not in broken\_set for letter in word): count += 1

print(count)

.

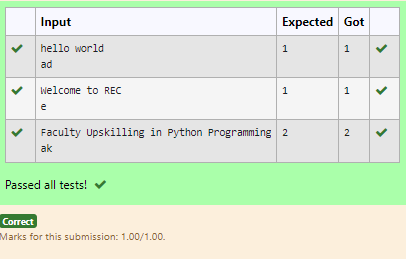
**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

146

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

147



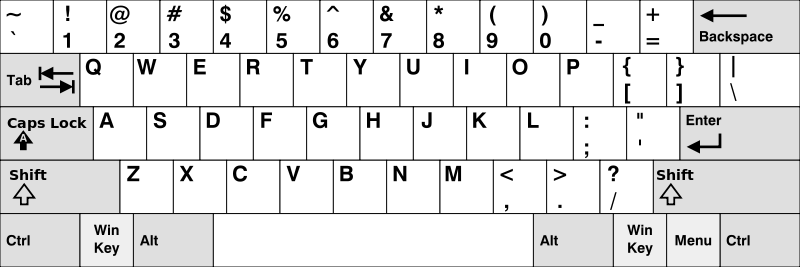
**Register No.: 231501046 Name: Ganesh S**

## 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"]

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

148

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

149

|  |  |
| --- | --- |
| **Input** | **Result** |
| 4 | Alaska |
| Hello | Dad |
| Alaska |  |
| Dad |  |
| Peace |  |

**Program:**

n=int(input()) l=[]

for j in range(n): l.append(str(input()))

row1 = set("qwertyuiop") row2 = set("asdfghjkl") row3 = set("zxcvbnm") result = []

for word in l:

lower\_word = set(word.lower())

if lower\_word <= row1 or lower\_word <= row2 or lower\_word <= row3: result.append(word)

if(len(result)!=0):

for i in range(len(result)): print(result[i])

else:

print("No words")

.

**Department of Computer Science and Engineering** | **Rajalakshmi Engineering College**

150

