

07 – Tuple/Set



Ex. No. : 7.1

Date: 16.05.24

Register No.: 231401081

Name: Ramya R

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:

```
a = input() try:
```

```
    c = int(a)
```

```
print("Yes") except:
```

```
    print("No")
```



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



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**.

Example 1:

Input: s = "AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT"

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

Input: s = "AAAAAAAAAAAAA"

Output: ["AAAAAAAAAAAA"]

For example:

Input	Result
AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA



Program: def

findRepeatedSequences(s):

 sequences = {}

result = []

for i in range(len(s) - 9):

 seq = s[i:i+10]

 sequences[seq] = sequences.get(seq, 0) + 1

if sequences[seq] == 2:

 result.append(seq)

return result s1 = input()

for i in findRepeatedSequences(s1):

 print(i)

	Input	Expected	Got	
✓	AAAAACCCCCAAAAACCCCCAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA	AAAAACCCCC CCCCCAAAAA	✓
✓	AAAAAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	✓



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".

~ `	!	@	#	\$	%	^	&	*	()	-	+	← Backspace
Tab ↩↪	Q	W	E	R	T	Y	U	I	O	P	{	}	 _
Caps Lock ⬆	A	S	D	F	G	H	J	K	L	:	"	'	↵ Enter
Shift ⬆	Z	X	C	V	B	N	M	<	>	?	!	Shift ⬆	
Ctrl	Win Key	Alt							Alt	Win Key	Menu	Ctrl	

-
-
- **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	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)
```



	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	✓



Ex. No. : 7.4

Date: 16.05.24

Register No.: 231401081

Name: Ramya R

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:

```
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)
```



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



Ex. No. : 7.5

Date: 16.05.24

Register No.: 231401081

Name: Ramya R

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: def

```
count_distinct_pairs(t, K):
```

```
    distinct_pairs = set()
```

```
    for i in range(len(t)):
```

```
        for j in range(i + 1, len(t)):
```

```
            if t[i] + t[j] == K:
```



```

distinct_pairs.add((min(t[i], t[j]),
max(t[i], t[j])))
return len(distinct_pairs)

t_input = input()
t=tuple(map(int,t_input.split(',')))
K = int(input())
print(count_distinct_pairs(t, K))

```

	Input	Expected	Got	
✓	5,6,5,7,7,8 13	2	2	✓
✓	1,2,1,2,5 3	1	1	✓
✓	1,2 0	0	0	✓

