g. No. : 8.1

Register No.: 230401023 Date: 20/5/2024

Name: CAROLINE

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.

```
text = input (" ")
a = '01'
count = 0
for char in text:
    if char not in a:
        count = 1
        break
if (count != 0):
    Print ("No")
else:
    Print ("Yes")
```

Ex. No. : 8.2

Register No.: 230401023 Date: 20/5/2024

Name: CAROLINE

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.

```
t = tuple(map(int, input().split(',')))
K = int(input())
s = set(t)
count = 0
for x in s:
    if K-x in s:
        count += 1

result = count//2
print(result).
```

Ex. No. : 8.3 Date: 20/5/2024

Register No.: 230401023 Name: CAROLINE

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**.

```
s = input()
substring_counts = {}
for i in range(len(s) - 9):
    substring = s[i:i+10]
    substring_counts[substring] = substring_counts.
        get(substring, 0) + 1
repeated_substrings = [substring for substring,
    count in substring_counts.items() if count > 1]
for substring in repeated_substrings:
    print(substring)
```

8.4

No. :

Register No.: 230401023

Date: 20/5/2024

Name: CAROLINE

Print repeated no

ven an array of integers nums containing n + 1 integers where each integer is in the nge [1, n] inclusive.There is only one repeated number in nums, eturn this repeated number. Solve the problem using set.

```
nums = list (map(int, criput().split()))
num_set = set()
for num in nums:
    if num in num_set:
        print(num)
        break
    else:
        num_set.add(num)
```

Ex. No. : 8.5 Date: 20/5/2024

Register No.: 230401023 Name: CAROLINE

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.

```
size 1, size 2 = map(int, input().split())
arr 1 = set(map(int, input().split()))
arr 2 = set(map(int, input().split()))
non-repeating = sorted(list((arr1 - arr2) |
                            (arr 2 - arr 1)))

if non-repeating:
    Print(non-repeating)
    Print(len(non-repeating))
else:
    print("NO SUCH ELEMENTS").
```

Ex. No. : 8.6

Register No.: 230401023

Date: 20/5/2024

Name: CAROLINE

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.

```
text = input()
brokenLetters = input()
words = text.split()
valid_words = 0
for word in words:
    if any(letter in brokenLetters for letter in word):
        continue
    else:
        valid_words += 1
print(valid_words)
```

Ex. No. : 8.7

Register No.: 230401023

Date: 20/5/2024

Name: CAROLINE

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".

```
row1 = set ('qwertyuiop')
row2 = set ('asdfghjkl')
row3 = set ('zxcvbnm')

num_words = int (input ())
found = False
for _ in range (num_words):
    word = input ()
    word_lower = work.lower()
  if all (char in row1 for char in work_lower) or \
    all (char in row2 for char in work_lower) or \
    all (char in row3 for char in work_lower):
     print (word)
     found = True
if not found:
  print ("no words")
```

A.Jerin