Ex. No. : 9.1 Date: 6/6/2024

Register No.: 230401023 Name: CAROLINE

Uncommon words

A sentence is a string of single-space separated words where each word consists only of lowercase letters.A word is uncommon if it appears exactly once in one of the sentences, and does not appear in the other sentence.

Given two sentences s1 and s2, return a list of all the uncommon words. You may return the answer in any order.

```
s1 = input().strip()
s2 = input().strip()
words_s1 = s1.split()
words_s2 = s2.split()
word_count_s1 = {}
word_count_s2 = {}
for word in words_s1:
    word_count_s1[word] = word_count_s1.get(word,0)+1
for word in words_s2:
    word_count_s2[word] = word_count_s2.get(word,0)+1
uncommon_words = set()
for word, count in word_count_s1.items():
    if count == 1 and word not in word_count_s2:
        uncommon_words.add(word)
```


Example 1:
Input: s1 = "this apple is sweet", s2 = "this apple is sour"
Output: ["sweet","sour"]
Example 2:

Input: s1 = "apple apple", s2 = "banana"
Output: ["banana"]
Constraints:
1 <= s1.length, s2.length <= 200
s1 and s2 consist of lowercase English letters and spaces.
s1 and s2 do not have leading or trailing spaces.
All the words in s1 and s2 are separated by a single space.
Note:
Use dictionary to solve the problem
For example:

Input	Result
this apple is sweet this apple is sour	sweet sour

```
for word, count in word_count_s2.items():
    if count == 1 and word not in word_count_s1:
        uncommon_words.add(word)

if len(uncommon_words) == 0:
    print("No uncommon words")
else:
    print("uncommon_words")
```

s. No. : 9.2 Date: 6/6/2024

egister No.: 230401023 Name: CAROLIN G

Sort Dictionary by Values Summation

ive a dictionary with value lists, sort the keys by summation of values in value list.

```
T= int(input())
result_dict = {}
for _ in range(T):
    key, *values = input().split()
    values = list(map(int, values))
    sum_values = sum(values)
    result_dict[key] = sum_values
sorted_result = dict(sorted(result_dict.items(),
 key = lambda item: item[1]))
for key, value in sorted_result.items():
    print(key, value)
print("No input provided.")
```

. No. : 9.3 Date: 6/6/2024

gister No.: 230401023 Name: CAROLINE

Winner of Election

ven an array of names of candidates in an election. A candidate name in the array presents a vote cast to the candidate. Print the name of candidates received Max vote. here is tie, print a lexicographically smaller name.

```
n= int(input())
votes_dict = {}
for _ in range(n):
    candidate = input()
    if candidate in votes_dict:
        votes_dict[candidate] += 1
    else:
        votes_dict[candidate] = 1
max_votes = max(votes_dict.values())
winners = [candidate for candidate, votes in votes_dict.
        items() if votes == max_votes]
winner = min(winners)
print(winner)
print("No input provided.")
```

Ex. No. : 9.5 Date: 6/6/2024

Register No.: 230401023 Name: CAROLINE

Scramble Score

In the game of Scrabble™, each letter has points associated with it. The total score of a word is the sum of the scores of its letters. More common letters are worth fewer points while less common letters are worth more points.

Write a program that computes and displays the Scrabble™ score for a word. Create a dictionary that maps from letters to point values. Then use the dictionary to compute the score.

A Scrabble™ board includes some squares that multiply the value of a letter or the value of an entire word. We will ignore these squares in this exercise.

```
letter_value = {
    'A': 1, 'E': 1, 'I': 1, 'L': 1, 'N': 1, 'O': 1, 'R': 1, 'S': 1, 'T': 1, 'U': 1
    'D': 2, 'G': 2
    'B': 3, 'C': 3, 'M': 3, 'P': 3
    'F': 4, 'H': 4, 'V': 4, 'W': 4, 'Y': 4,
    'K': 5,
    'J': 8, 'X': 8
    'Q': 10, 'Z': 10
word = input()
score = sum(letter_values.get(letter.upper(), 0) for letter in word)
print(f"{word}'s worth {score} points.")
```

A. Arun