6.17 UNIVERSAL STRINGS FROM TWO WORD LISTS

Question:

You are given two string arrays words1 and words2. A string b is a subset of string a if every letter in b occurs in a including multiplicity. For example, "wrr" is a subset of "warrior" but is not a subset of "world". A string a from words1 is universal if for every string b in words2, b is a subset of a. Return an array of all the universal strings in words1. You may return the answer in any order.

Example 1:
Input: words1 = ["amazon","apple","facebook","google","leetcode"], words2 = ["e","o"]
Output: ["facebook","google","leetcode"]

Example 2:
Input: words1 = ["amazon","apple","facebook","google","leetcode"], words2 = ["l","e"]
Output: ["apple","google","leetcode"]

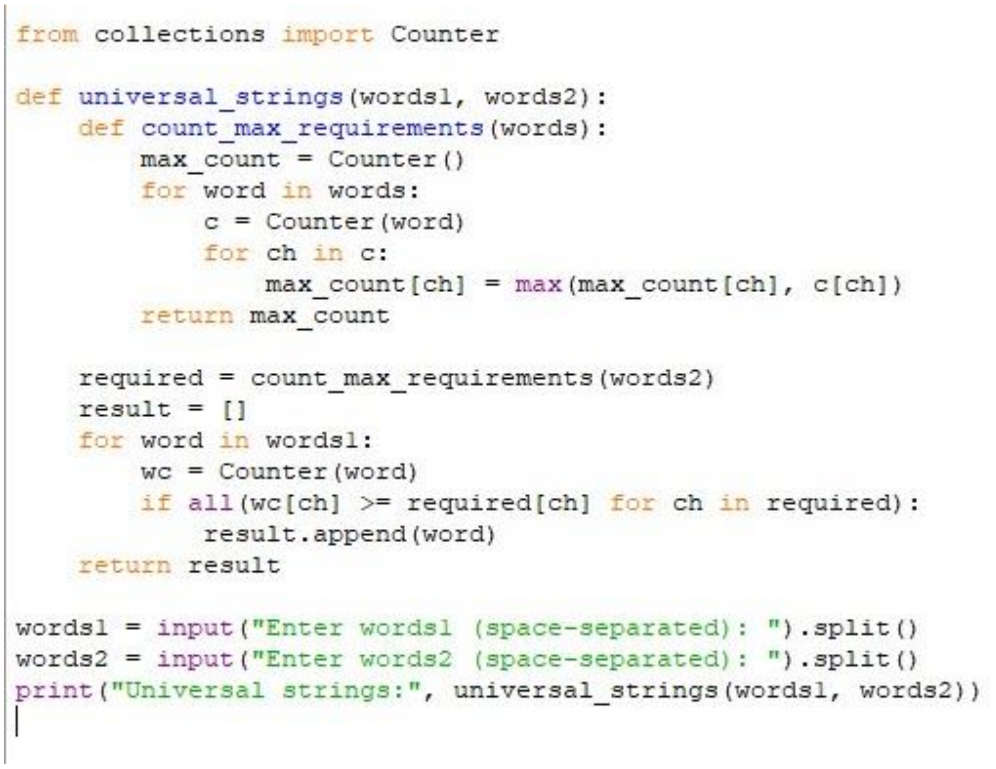
AIM

To implement a Python program that finds all universal strings from words1 that contain all subsets of words2.

ALGORITHM

1. Initialize a frequency dictionary max_freq to store the maximum frequency of each character across all words in words2.
2. For each word in words2:
   - Count character frequencies.
   - Update max_freq with the maximum frequency for each character.
3. For each word in words1:
   - Count character frequencies.
   - Check if the word contains at least the required frequency for every character in max_freq.
   - If yes, include it in the result list as a universal string.
4. Return the result list.

PROGRAM

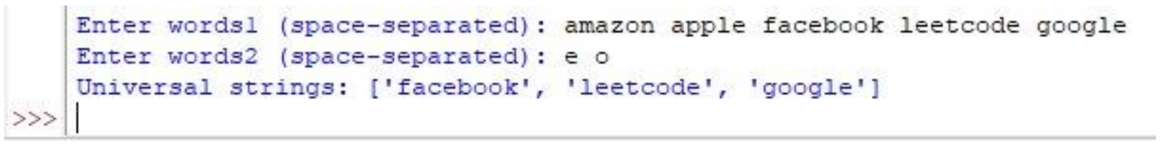


```python
from collections import Counter

def universal_strings(words1, words2):
    def count_max_requirements(words):
        max_count = Counter()
        for word in words:
            c = Counter(word)
            for ch in c:
                max_count[ch] = max(max_count[ch], c[ch])
        return max_count

    required = count_max_requirements(words2)
    result = []
    for word in words1:
        wc = Counter(word)
        if all(wc[ch] >= required[ch] for ch in required):
            result.append(word)
    return result

words1 = input("Enter words1 (space-separated): ").split()
words2 = input("Enter words2 (space-separated): ").split()
print("Universal strings:", universal_strings(words1, words2))
```

Input:

words1 = ["amazon","apple","facebook","google","leetcode"], words2 = ["e","o"]

Output:



```
Enter words1 (space-separated): amazon apple facebook leetcode google
Enter words2 (space-separated): e o
Universal strings: ['facebook', 'leetcode', 'google']
>>>
```

RESULT:

Thus, the program is successfully executed and verified to return all universal strings from words1 that contain all subsets from words2.

PERFORMANCE ANALYSIS:

Time Complexity: O(n * m + k), where n is the length of words1, m is the length of words2, and k is the average word length.

Space Complexity: O(1), since only 26 lowercase English letters are used for frequency counting.