

222. 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"]

PROGRAM:-

```
def is_subset(small_count, big_count):
    for i in range(26): # 26 letters in the English alphabet
        if small_count[i] > big_count[i]:
            return False
    return True
```

```
def count_characters(word):
    char_count = [0] * 26
    for char in word:
        char_count[ord(char) - ord('a')] += 1
    return char_count
```

```
def universal_strings(words1, words2):
    result = []

    for word1 in words1:
        word1_count = count_characters(word1)
        all_subset = True
        for word2 in words2:
            word2_count = count_characters(word2)
            if not is_subset(word2_count, word1_count):
                all_subset = False
                break
        if all_subset:
            result.append(word1)

    return result
```

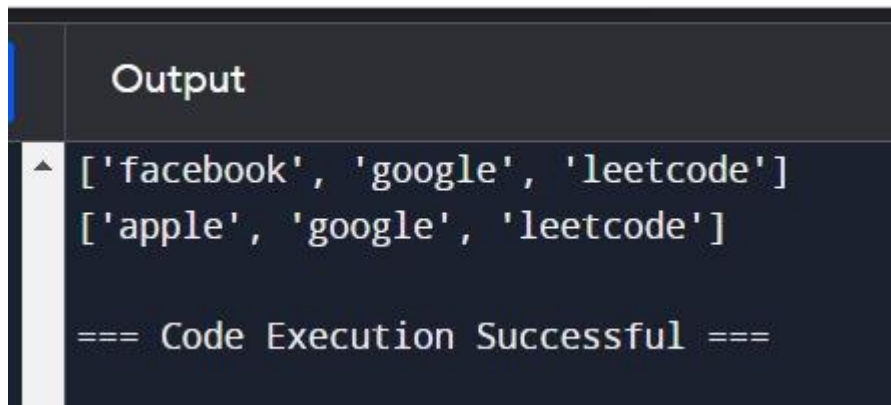
Example usage:

```
words1 = ["amazon","apple","facebook","google","leetcode"]
words2 = ["e","o"]
print(universal_strings(words1, words2)) # Output: ["facebook","google","leetcode"]
```

```
words2 = ["l","e"]
```

```
print(universal_strings(words1, words2)) # Output: ["apple","google","leetcode"]
```

OUTPUT:-

A screenshot of a code execution environment's output window. The window has a dark background with a light-colored title bar that says "Output". Below the title bar, there is a scrollable area containing two lines of text: `['facebook', 'google', 'leetcode']` and `['apple', 'google', 'leetcode']`. At the bottom of the scrollable area, it says `=== Code Execution Successful ===`. A small upward-pointing arrow is visible on the left side of the scrollable area, indicating it can be scrolled up.

```
Output
['facebook', 'google', 'leetcode']
['apple', 'google', 'leetcode']

=== Code Execution Successful ===
```

TIME COMPLEXITY:- $O(n * k + m * k)$