**6.16 Super Set of Every String**

**Aim:** The aim is to find all strings in a primary list (words1) that are a "superset" of every string in a secondary list (words2), based on character frequency.

**Algorithm:**

1. Let the inputs be two lists of lowercase words:  
  - words1 = [w₁, w₂, …, wm] (primary list)  
  - words2 = [u₁, u₂, …, uk] (secondary list)

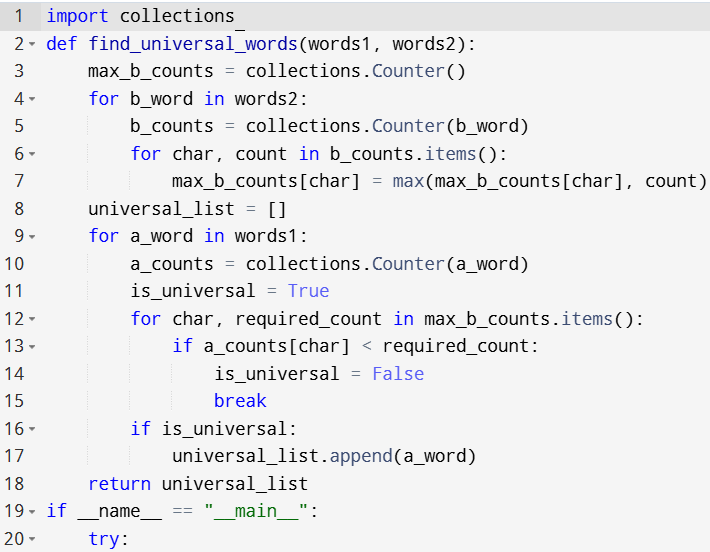
2. Definition: A word x is a superset of word y if for every character c, the frequency of c in x is greater than or equal to the frequency of c in y.

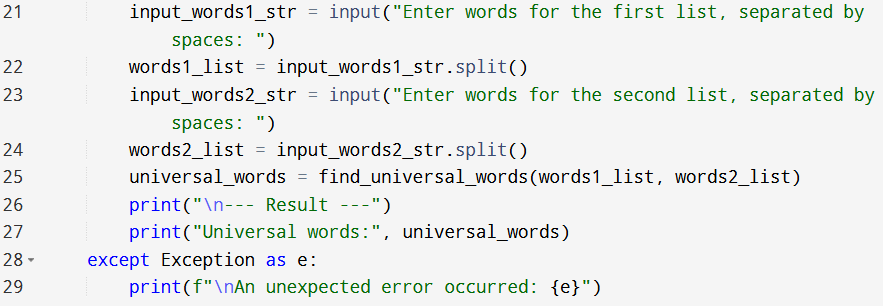
3. Preprocessing step (combine requirements from words2):  
  - For each word u ∈ words2, compute its frequency count freq(u).  
  - Maintain a global maximum frequency array required[26], where: required[c] = max(freq(u)[c] for all u ∈ words2)  
  - This ensures that any valid superset must satisfy all words in words2.

4. Checking step: For each word w ∈ words1:  
  - Compute its frequency count freq(w).  
  - For every character c ∈ [a…z]: If freq(w)[c] < required[c], then w is not a superset.  
  - Otherwise, include w in the result list.

5. Return the list of all words from words1 that satisfy the superset condition.

**Program:**

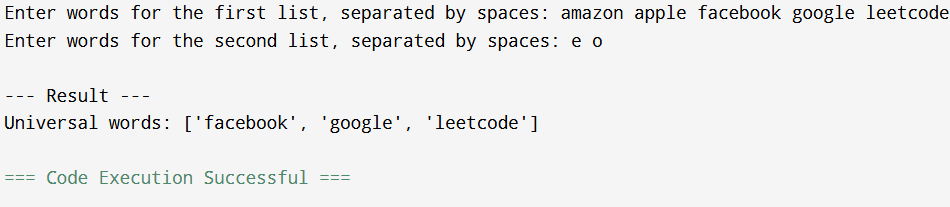
****



**Input:**

* Enter words for the first list, separated by spaces:
* Enter words for the second list, separated by spaces:

**Output:**

****

**Result:** Thus, the program is executed successfully and output is verified.

**Performance analysis:**

* Time Complexity: O(N cdotS + M cdotT)
* Space Complexity: O(S).