Problem

You are given a string S, and a list of words L i.e array/vector of strings (Words in list L are all of the same length). Find the starting indices of the substrings in string S, which contains all the words present in list L.

```
Input : S: "barfoothefoobarman"
        L: ["foo", "bar"]
Output : 0 9
Explanation :
// at index 0 : barfoo
// at index 9 : foobar

Input : S: "catbatatecatatebat"
        L: ["cat", "ate", "bat"]
Output : 0 3 9
Explanation :
// at index 0 : catbatate
// at index 3 : batatecat
// at index 9 : catatebat

Input : S : "abcdababcd"
        L : ["ab", "ab", "cd"]
Output : 0 2 4
Explanation :
// at index 0 : abcdab
// at index 2 : cdabab
// at index 4 : ababcd

Input : S : "abcdababcd"
        L : ["ab", "ab"]
Output : 4
```

Approach

Let's see the steps :

1. Declare a map (**hash_map**) which stores all words of List L corresponding to their occurrences inside list L.

2. Traverse through all possible substrings in string S which are equal to size_L(total number of characters produced if all the words in list L are concatenated).

3. Create a temporary map (**temp_hash_map**) and initialize it with original map(**hash_map**) for every possible substring.

4. Extract the words from the substring and if the word is present in temp_hash_map we decrease it's corresponding count, if it's not present in temp_hash_map we simply break.

5. After traversing the substring we traverse temp_hash_map and look for any key which has it's count > 0. If we found no such key it means that all the words in list L were found in substring and store the given starting index of the substring, if we find a key which has it's

count > 0 it means we did not traversed whole substring because we came across a word which was not in temp_hash_map.