10.01.2025

Word subsets

we are storing the frequency of characters from the word2 array of strings in a vector and making sure that we are storing the maximum required character only since if we store the maximum all the other would automatically be covered (as we are not subtracting or anything), therefore it would work.
(nested for loop)

then we iterate on the first array of strings we will check if the required maximum characters are present in the current string, if it does it would be the superset or second array strings are subset of current string, therefore we add the current string to the resultant array.
(nested for loop)

at last we return the array.

```
class Solution {
public:
    bool isSubSet(vector<int>&charRequired, vector<int>& temp){
        for(int i = 0; i < 26; i++){
            if(temp[i] < charRequired[i]) return false;
        }
        return true;
    }
    vector<string> wordSubsets(vector<string>& words1, vector<string>& words2) {
        vector<string>res;
        vector<int>charRequired(26);
        for(string& it: words2){
            vector<int>temp(26, 0);
            for(char& ch: it){
                // index pe increment krdo
                temp[ch - 'a']++;
                charRequired[ch  - 'a'] = max(charRequired[ch-'a'], temp[ch -
'a']);
            }
        }
        for(string& it: words1){
            vector<int>temp(26, 0);
            for(char& ch: it){ // l2 is the number of average characters in a
word.
                temp[ch - 'a']++;
            }
            if(isSubSet(charRequired, temp) == true){
            res.push_back(it);
        }
        }
        return res;
    }
};
// time complexity: O(n * l) + O(n * l2)where n is the number of words, l is
```

```
number of characters
// in a word (average).
```