1684. Count the Number of Consistent Strings

You are given a string `allowed` consisting of **distinct** characters and an array of strings `words`. A string is **consistent** if all characters in the string appear in the string `allowed`.

Return *the number of **consistent** strings in the array* `words`.

Example 1:

```
Input: allowed = "ab", words = ["ad","bd","aaab","baa","badab"]
Output: 2
Explanation: Strings "aaab" and "baa" are consistent since they only contain characters 'a' and 'b'.
```

Example 2:

```
Input: allowed = "abc", words = ["a","b","c","ab","ac","bc","abc"]
Output: 7
Explanation: All strings are consistent.
```

Example 3:

```
Input: allowed = "cad", words = ["cc","acd","b","ba","bac","bad","ac","d"]
Output: 4
Explanation: Strings "cc", "acd", "ac", and "d" are consistent.
```

Constraints:

- `1 <= words.length <= 104`
- `1 <= allowed.length <= 26`
- `1 <= words[i].length <= 10`
- The characters in `allowed` are **distinct**.
- `words[i]` and `allowed` contain only lowercase English letters.

1684. Count the Number of Consistent Strings

```
/*
    Time complexity: O(n+mk)
    Space complexity: O(26)=O(1)
*/
class Solution {
    public:
        int countConsistentStrings(std::string allowed, std::vector<std::string>& words) {
            bool exist[26]={false};
            for(auto& letter: allowed) exist[letter-'a']=true;

            int ans=0;
            for(auto& word: words){
                int i=0,k=word.size();
                while(i<k&&exist[word[i]-'a']) i++;
                ans+=(i==k);
            }

            return ans;
        }
};
```