1170. Compare Strings by Frequency of the Smallest Character

Let the function `f(s)` be the **frequency of the lexicographically smallest character** in a non-empty string `s`. For example, if `s = "dcce"` then `f(s) = 2` because the lexicographically smallest character is `'c'`, which has a frequency of 2.

You are given an array of strings `words` and another array of query strings `queries`. For each query `queries[i]`, count the **number of words** in `words` such that `f(queries[i])` < `f(W)` for each `W` in `words`.

Return *an integer array* `answer`*, where each* `answer[i]` *is the answer to the* `ith` *query*.

Example 1:

Input: queries = ["cbd"], words = ["zaaaz"]
Output: [1]
Explanation: On the first query we have f("cbd") = 1, f("zaaaz") = 3 so f("cbd") < f("zaaaz").

Example 2:

Input: queries = ["bbb","cc"], words = ["a","aa","aaa","aaaa"]
Output: [1,2]
Explanation: On the first query only f("bbb") < f("aaaa").
On the second query both f("aaa") and f("aaaa")
are both > f("cc").

```
def numSmallerByFrequency(self, queries: List[str], words: List[str]) ->
List[int]:
        for i in range(len(queries)):
            queries[i] = queries[i].count(min(queries[i]))
        for i in range(len(words)):
            words[i] = words[i].count(min(words[i]))
        words.sort()
        res = []
        N = len(words)
        for ele in queries:
            idx= self.binarySearch(words,ele)
            res.append(N-idx)
        return res
```

```
    def binarySearch(self,arr,target):
        lo = 0
        hi = len(arr)-1

        while lo<=hi:
            mid = (lo+hi)//2
            if arr[mid]==target:
                lo = mid+1
            elif arr[mid]<target:
                lo = mid+1
            else:
                hi = mid-1
        return lo
```