Problem 2842: Count K-Subsequences of a String With Maximum Beauty

Problem Information

Difficulty: Hard
Acceptance Rate: 30.04%
Paid Only: No
Tags: Hash Table, Math, String, Greedy, Combinatorics

Problem Description

You are given a string `s` and an integer `k`.

A **k-subsequence** is a **subsequence** of `s`, having length `k`, and all its characters are **unique** , **i.e**., every character occurs once.

Let `f(c)` denote the number of times the character `c` occurs in `s`.

The **beauty** of a **k-subsequence** is the **sum** of `f(c)` for every character `c` in the k-subsequence.

For example, consider `s = "abbbdd"` and `k = 2`:

* `f('a') = 1`, `f('b') = 3`, `f('d') = 2` * Some k-subsequences of `s` are: * `"_**ab**_ bbdd"` -> `"ab"` having a beauty of `f('a') + f('b') = 4` * `"_**a**_ bbb** _d_** d"` -> `"ad"` having a beauty of `f('a') + f('d') = 3` * `"a** _b_** bb _**d**_ d"` -> `"bd"` having a beauty of `f('b') + f('d') = 5`

Return _an integer denoting the number of k-subsequences_ _whose**beauty** is the **maximum** among all **k-subsequences**_. Since the answer may be too large, return it modulo `109 + 7`.

A subsequence of a string is a new string formed from the original string by deleting some (possibly none) of the characters without disturbing the relative positions of the remaining characters.

Notes

* `f(c)` is the number of times a character `c` occurs in `s`, not a k-subsequence. * Two k-subsequences are considered different if one is formed by an index that is not present in the other. So, two k-subsequences may form the same string.

Example 1:

Input: s = "bcca", k = 2 **Output:** 4 **Explanation:** From s we have f('a') = 1, f('b') = 1, and f('c') = 2. The k-subsequences of s are: **_bc_** ca having a beauty of f('b') + f('c') = 3 **_b_** c _**c**_ a having a beauty of f('b') + f('c') = 3 **_b_** cc** _a_** having a beauty of f('b') + f('a') = 2 b** _c_** c _**a**_**** having a beauty of f('c') + f('a') = 3 bc** _ca_** having a beauty of f('c') + f('a') = 3 There are 4 k-subsequences that have the maximum beauty, 3. Hence, the answer is 4.

Example 2:

Input: s = "abbcd", k = 4 **Output:** 2 **Explanation:** From s we have f('a') = 1, f('b') = 2, f('c') = 1, and f('d') = 1. The k-subsequences of s are: _**ab**_ b** _cd_** having a beauty of f('a') + f('b') + f('c') + f('d') = 5 _**a**_ b _**bcd**_ having a beauty of f('a') + f('b') + f('c') + f('d') = 5 There are 2 k-subsequences that have the maximum beauty, 5. Hence, the answer is 2.

Constraints:

* `1 <= s.length <= 2 * 105` * `1 <= k <= s.length` * `s` consists only of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
int countKSubsequencesWithMaxBeauty(string s, int k) {

}
};
```

Java:

```
class Solution {
public int countKSubsequencesWithMaxBeauty(String s, int k) {
```

```
}
}
```

Python3:

```
class Solution:
def countKSubsequencesWithMaxBeauty(self, s: str, k: int) -> int:
```