Problem 3734: Lexicographically Smallest Palindromic Permutation Greater Than Target

Problem Information

Difficulty: Hard
Acceptance Rate: 23.10%
Paid Only: No
Tags: Two Pointers, String, Enumeration

Problem Description

You are given two strings `s` and `target`, each of length `n`, consisting of lowercase English letters.

Return the **lexicographically smallest string** that is **both** a **palindromic permutation** of `s` and **strictly** greater than `target`. If no such permutation exists, return an empty string.

Example 1:

Input: s = "baba", target = "abba"

Output: "baab"

Explanation:

* The palindromic permutations of `s` (in lexicographical order) are `"abba"` and `"baab"`. * The lexicographically smallest permutation that is strictly greater than `target` is `"baab"`.

Example 2:

Input: s = "baba", target = "bbaa"

Output: ""

Explanation:

* The palindromic permutations of `s` (in lexicographical order) are `"abba"` and `"baab"`. * None of them is lexicographically strictly greater than `target`. Therefore, the answer is `""`.

Example 3:

Input: s = "abc", target = "abb"

Output: ""

Explanation:

`s` has no palindromic permutations. Therefore, the answer is `""`.

Example 4:

Input: s = "aac", target = "abb"

Output: "aca"

Explanation:

* The only palindromic permutation of `s` is `"aca"`. * `"aca"` is strictly greater than `target`. Therefore, the answer is `"aca"`.

Constraints:

* `1 <= n == s.length == target.length <= 300` * `s` and `target` consist of only lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
string lexPalindromicPermutation(string s, string target) {


}
};
```

Java:

```
class Solution {
public String lexPalindromicPermutation(String s, String target) {

}
}
```

Python3:

```
class Solution:
def lexPalindromicPermutation(self, s: str, target: str) -> str:
```