Problem 3503: Longest Palindrome After Substring Concatenation I

Problem Information

Difficulty: Medium
Acceptance Rate: 43.44%
Paid Only: No
Tags: Two Pointers, String, Dynamic Programming, Enumeration

Problem Description

You are given two strings, `s` and `t`.

You can create a new string by selecting a substring from `s` (possibly empty) and a substring from `t` (possibly empty), then concatenating them **in order**.

Return the length of the **longest** palindrome that can be formed this way.

Example 1:

Input: s = "a", t = "a"

Output: 2

Explanation:

Concatenating `"a"` from `s` and `"a"` from `t` results in `"aa"`, which is a palindrome of length 2.

Example 2:

Input: s = "abc", t = "def"

Output: 1

Explanation:

Since all characters are different, the longest palindrome is any single character, so the answer is 1.

Example 3:

Input: s = "b", t = "aaaa"

Output: 4

Explanation:

Selecting "`aaaa`" from `t` is the longest palindrome, so the answer is 4.

Example 4:

Input: s = "abcde", t = "ecdba"

Output: 5

Explanation:

Concatenating `"abc"` from `s` and `"ba"` from `t` results in `"abcba"`, which is a palindrome of length 5.

Constraints:

* `1 <= s.length, t.length <= 30` * `s` and `t` consist of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
int longestPalindrome(string s, string t) {

}
```

```
};
```

Java:

```
class Solution {
public int longestPalindrome(String s, String t) {

}
}
```

Python3:

```
class Solution:
def longestPalindrome(self, s: str, t: str) -> int:
```