647. Palindromic Substrings

Medium

Given a string s, return the number of palindromic substrings in it.

A string is a palindrome when it reads the same backward as forward.

A substring is a contiguous sequence of characters within the string.

Example 1:

Input: s = "abc"
Output: 3
Explanation: Three palindromic strings: "a", "b", "c".

Example 2:

Input: s = "aaa"
Output: 6
Explanation: Six palindromic strings: "a", "a", "a", "aa", "aa", "aaa".

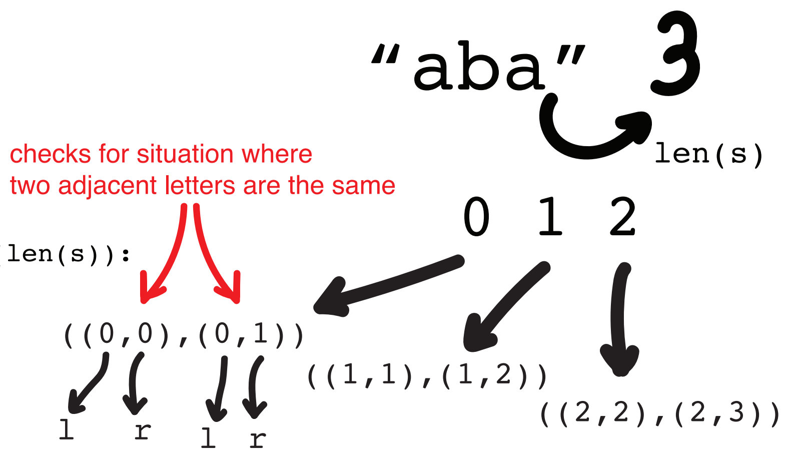
Constraints:
1 <= s.length <= 1000
s consists of lowercase English letters.

```python
class Solution:
    def countSubstrings(self, s: str) -> int:
        answer = []

        for i in range(len(s)):
            for left, right in ((i,i), (i,i+1)):
                while left >= 0 and right < len(s) and s[left] == s[right]:
                    answer.append(s[left:right+1])
                    left -= 1
                    right += 1

        return len(answer)
```

```
for i in range(len(s)):
```



```
    for l, r in ((i,i), (i,i+1)):
```

while l and r pointers are in bound, and s[l] and s[r] are the same

```
        while l >= 0 and r < len(s) and s[l] == s[r]:
```

append the string, s[left:right+1] to the answer list

```
            answer.append(s[left:right+1])

            l -= 1
            r += 1
```

1. Loop through the string, s.
2. There are two parts:
    a. set left (l) and right (r) pointers the same as i, expand left (l-=1) and right (r+=1), and check to see if the s[l] and s[r] are the same.
    b. set left (l) as i, and right (r) as i + 1, so that it accounts for situations where there are two adjacent characters that are the same, and expand left(l-=1) and right (r+=1), and check to see if the s[l] and s[r] are the same.