516. Longest Palindromic Subsequence

Given a string `s`, find *the longest palindromic **subsequence**'s length in* `s`.

A **subsequence** is a sequence that can be derived from another sequence by deleting some or no elements without changing the order of the remaining elements.

Example 1:

```
Input: s = "bbbab"
Output: 4
Explanation: One possible longest palindromic subsequence is "bbbb".
```

Example 2:

```
Input: s = "cbbd"
Output: 2
Explanation: One possible longest palindromic subsequence is "bb".
```

Constraints:

- `1 <= s.length <= 1000`
- `s` consists only of lowercase English letters.

```
def longestPalindromeSubseq(self, s: str) -> int:
        dp = [[0]*len(s) for _ in range(len(s))]

        for gap in range(len(s)):
            i = 0
            j = gap
            while j<len(s):
                if gap==0:
                    dp[i][j]=1
                elif gap==1:
                    if s[i]==s[j]:
                        dp[i][j] = 2
                    else:
                        dp[i][j] = 1
                else:
                    if s[i]==s[j]:
                        dp[i][j] = 2 + dp[i+1][j-1]
```

```
                else:
                    dp[i][j] = max(dp[i+1][j],dp[i][j-1])
            i = i+1
            j = j+1
        return dp[0][-1]
```