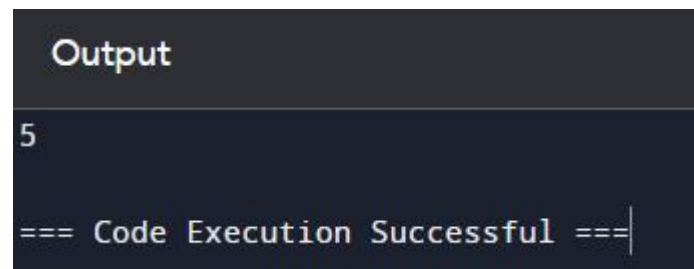


## 102. Longest palindromic\_subsequence

### Program:

```
def longest_palindromic_subsequence(s):  
    n = len(s)  
    dp = [[0] * n for _ in range(n)]  
  
    for i in range(n-1, -1, -1):  
        dp[i][i] = 1  
        for j in range(i+1, n):  
            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])  
  
    return dp[0][n-1]  
s = "character"  
print(longest_palindromic_subsequence(s))
```



The screenshot shows a dark-themed interface with a header labeled "Output". Below the header, the number "5" is displayed, representing the length of the longest palindromic subsequence for the input string "character". At the bottom, a status message reads "=== Code Execution Successful ===" with a vertical cursor at the end.

Time complexity:  $O(n^2)$