**4.6 LONGEST PALINDROMIC SUBSTRING**

**Question:**

Given a string s, return the longest palindromic substring in S.

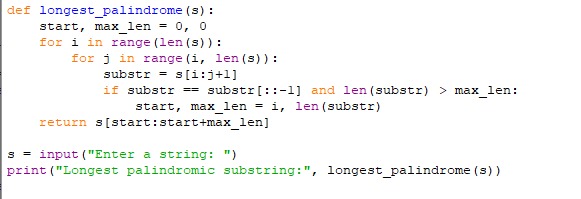
**AIM**

To implement a Python program that finds the longest palindromic substring using dynamic programming.

**ALGORITHM**

1. Let n be the length of the string s.
2. Create a 2D boolean table dp[n][n] where dp[i][j] is True if the substring s[i:j+1] is a palindrome.
3. Initialize all substrings of length 1 as palindromes.
4. Check substrings of length 2 and mark them if both characters are equal.
5. For lengths ≥ 3, use the recurrence:
   * dp[i][j] = True if s[i] == s[j] and dp[i+1][j-1] == True
6. Track the start index and maximum length of the longest palindrome found.
7. Return the substring s[start:start+max\_len].

**PROGRAM**



Input:

Enter a string: reerloooool

Output:

A close up of a sign

AI-generated content may be incorrect.

**RESULT:**

Thus the program is successfully executed, and the output is verified.

**PERFORMANCE ANALYSIS:**

* Time Complexity:
  + O(n²)
* Space Complexity:
  + O(n²)