Our Solution(s) Run Code

Solution 1 Solution 2

```
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

// O(n^2) time | O(1) space
function longestPalindromicSubstring(string) {
  let currentLongest = [0, 1];
  for (let i = 1; i < string.length; i++) {
    const odd = getLongestPalindromeFrom(string, i - 1, i + 1);
    const even = getLongestPalindromeFrom(string, i - 1, i);
    const longest = odd[1] - odd[0] > even[1] - even[0] ? odd : even;
    currentLongest = currentLongest[1] - currentLongest[0] > longest[1] - longest[
  }
  return string.slice(currentLongest[0], currentLongest[1]);
}

function getLongestPalindromeFrom(string, leftIdx, rightIdx) {
  while (leftIdx >= 0 && rightIdx < string.length) {
    if (string[leftIdx] !== string[rightIdx]) break;
    leftIdx--;
    rightIdx++;
  }
  return [leftIdx + 1, rightIdx];
}

exports.longestPalindromicSubstring = longestPalindromicSubstring;

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
function longestPalindromicSubstring(string) {
  // Write your code here.
}

// Do not edit the line below.
exports.longestPalindromicSubstring = longestPalindromicSubstring;

```

Custom Output Raw Output Submit Code

Run or submit code when you're ready.