Solution 1   Solution 2

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

public class Program {
  // O(n^2) time | O(1) space
  public static string LongestPalindromicSubstring(string str) {
    int[] currentLongest = {0, 1};
    for (int i = 1; i < str.Length; i++) {
      int[] odd = getLongestPalindromeFrom(str, i - 1, i + 1);
      int[] even = getLongestPalindromeFrom(str, i - 1, i);
      int[] longest = odd[1] - odd[0] > even[1] - even[0] ? odd : even;
      currentLongest = currentLongest[1] - currentLongest[0] >
        longest[1] - longest[0] ? currentLongest : longest;
    }
    return str.Substring(currentLongest[0], currentLongest[1] - currentLongest[0]
  }

  public static int[] getLongestPalindromeFrom(string str, int leftIdx, int right
    while (leftIdx >= 0 && rightIdx < str.Length) {
      if (str[leftIdx] != str[rightIdx]) {
        break;
      }
      leftIdx--;
      rightIdx++;
    }
    return new int[] {leftIdx + 1, rightIdx};
  }
}
```

Solution 1   Solution 2   Solution 3

```csharp
public class Program {
  public static string LongestPalindromicSubstring(string str) {
    // Write your code here.
    return null;
  }
}
```

Run or submit code when you're ready.