Our Solution(s) Run Code

Solution 1 Solution 2

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

using namespace std;

bool isPalindrome(string str);

// O(n^3) time | O(1) space
string longestPalindromicSubstring(string str) {
  string longest = "";
  for (int i = 0; i < str.length(); i++) {
    for (int j = i; j < str.length(); j++) {
      string substring = str.substr(i, j + 1 - i);
      if (substring.length() > longest.length() && isPalindrome
        longest = substring;
      }
    }
  }
  return longest;
}

bool isPalindrome(string str) {
  int leftIdx = 0;
  int rightIdx = str.length() - 1;
  while (leftIdx < rightIdx) {
    if (str[leftIdx] != str[rightIdx]) {
      return false;
    }
    leftIdx++;
    rightIdx--;
  }
  return true;
}

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```cpp
using namespace std;

string longestPalindromicSubstring(string str) {
  // Write your code here.
  return "";
}

```

Our Tests

Custom Output Submit Code

```
package
  void Run() {

    Run("Test Case 1",
            () => { assert(LongestPalindrome.Solution("a") == "a
    Run("Test Case 2", () => {
      assert(LongestPalindrome.Solution("ac") == "a
    });

    Run("Test Case 3", () => {
      assert(LongestPalindrome.Solution("aaaa high is is") == "
    });

    Run("Test Case 4", () => {
      assert(LongestPalindrome.Solution("abcdba") == "abcdba");
    });

    Run("Test Case 5", () => {
      assert(LongestPalindrome.Solution("abaaxyzzyxyf") == "xyzzy
    });
```

Run or submit code when you're ready.