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

PROGRAM:

```
class Solution:

   def longestPalindrome(self, s: str) -> str:

      if len(s) < 1:

         return ""


      start = 0

      end = 0


      for i in range(len(s)):

         len1 = self.expandAroundCenter(s, i, i)

         len2 = self.expandAroundCenter(s, i, i + 1)

         max_len = max(len1, len2)


         if max_len > end - start:

            start = i - (max_len - 1) // 2

            end = i + max_len // 2


      return s[start:end+1]


   def expandAroundCenter(self, s: str, left: int, right: int) -> int:

      while left >= 0 and right < len(s) and s[left] == s[right]:

         left -= 1

         right += 1


      return right - left - 1
```

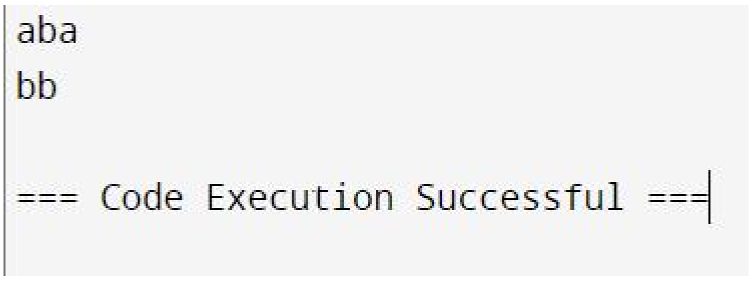
```
# Test cases

sol = Solution()

print(sol.longestPalindrome("babad"))  # Output: "aba" or "bab"

print(sol.longestPalindrome("cbbd"))   # Output: "bb"
```

OUTPUT:



```
aba
bb

=== Code Execution Successful ===
```

TIME COMPLEXITY:O(N^2)