5. Longest Palindromic Substring

Medium 19456 1128 Add to List Share

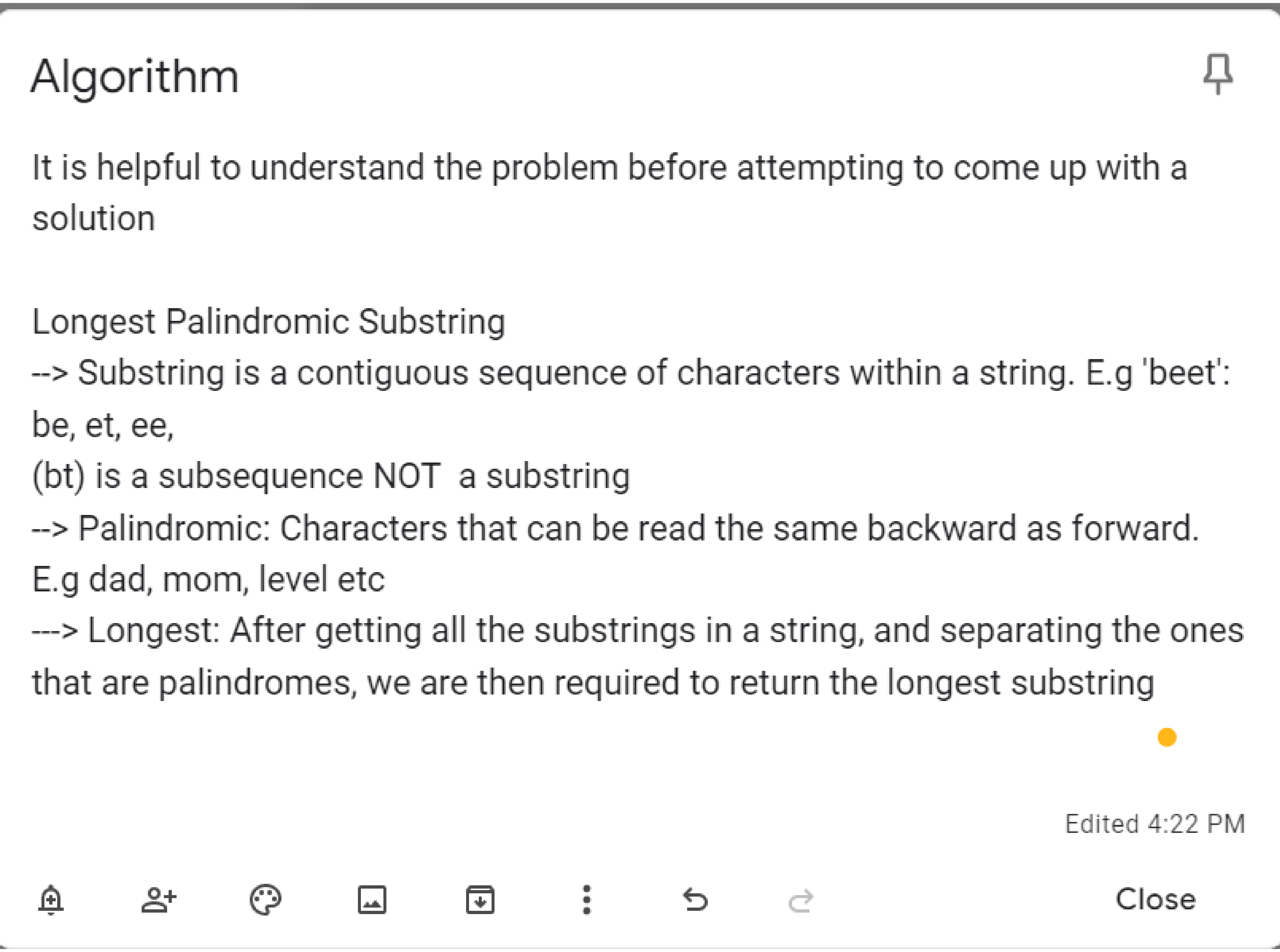
Given a string `s`, return *the longest palindromic substring* in `s`.

Example 1:

```
Input: s = "babad"
Output: "bab"
Explanation: "aba" is also a valid answer.
```

Example 2:

```
Input: s = "cbbd"
Output: "bb"
```

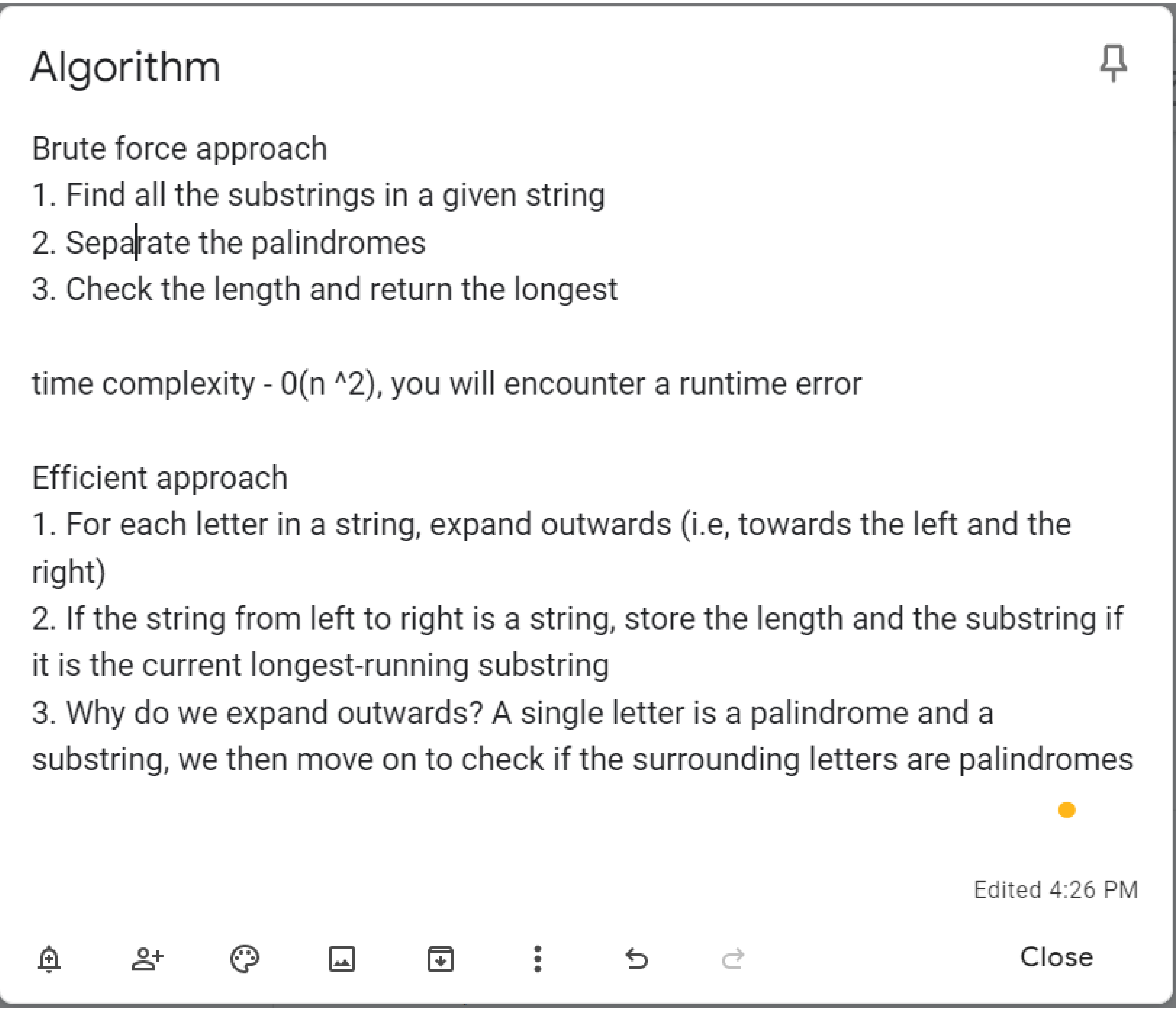


Algorithm

It is helpful to understand the problem before attempting to come up with a solution

Longest Palindromic Substring
--> Substring is a contiguous sequence of characters within a string. E.g 'beet': be, et, ee,
(bt) is a subsequence NOT a substring
--> Palindromic: Characters that can be read the same backward as forward. E.g dad, mom, level etc
---> Longest: After getting all the substrings in a string, and separating the ones that are palindromes, we are then required to return the longest substring

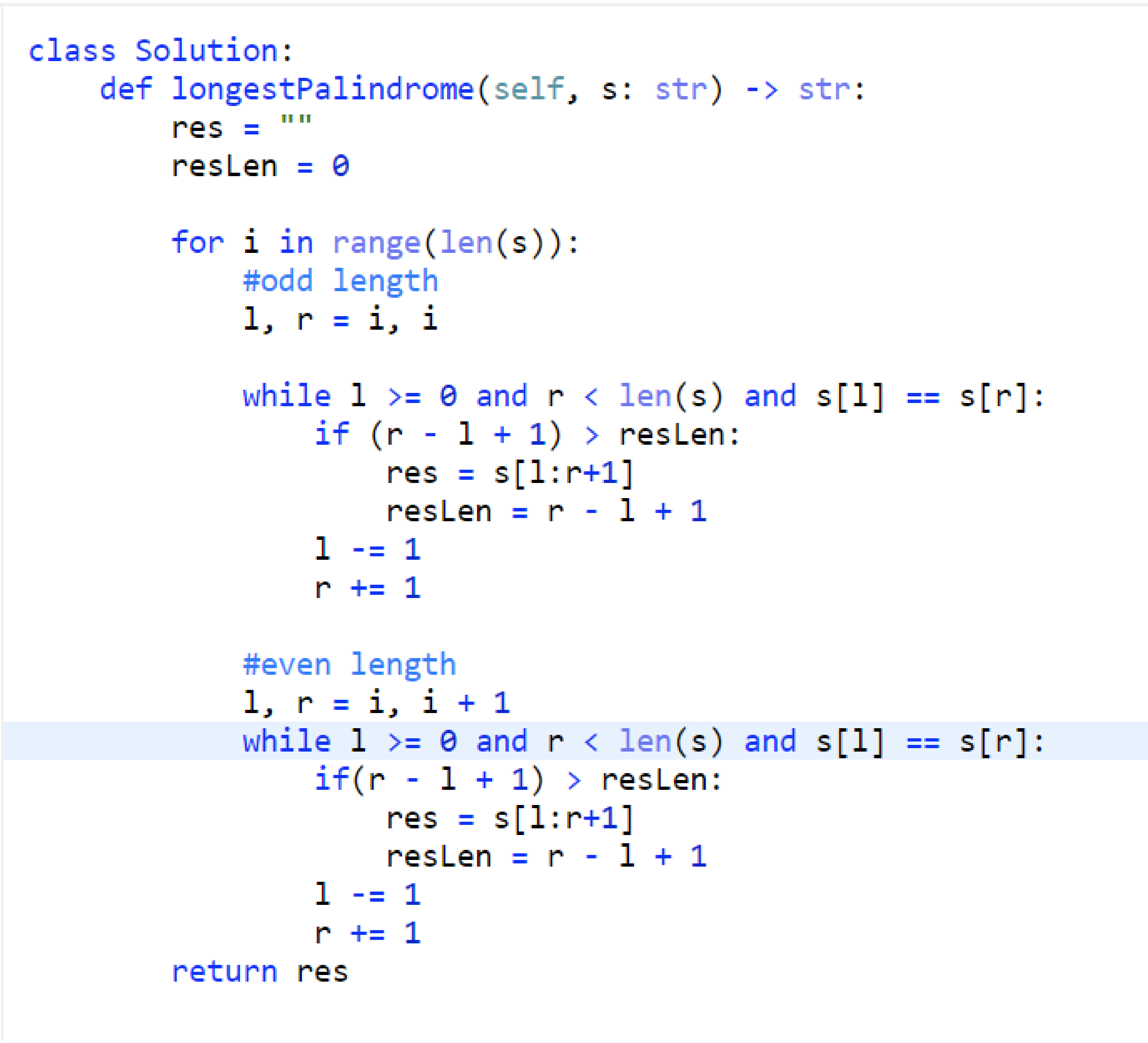
Edited 4:22 PM


Algorithm

Brute force approach
1. Find all the substrings in a given string
2. Separate the palindromes
3. Check the length and return the longest

time complexity - 0(n ^2), you will encounter a runtime error

Efficient approach
1. For each letter in a string, expand outwards (i.e, towards the left and the right)
2. If the string from left to right is a string, store the length and the substring if it is the current longest-running substring
3. Why do we expand outwards? A single letter is a palindrome and a substring, we then move on to check if the surrounding letters are palindromes



```python
class Solution:
    def longestPalindrome(self, s: str) -> str:
        res = ""
        resLen = 0

        for i in range(len(s)):
            #odd length
            l, r = i, i

            while l >= 0 and r < len(s) and s[l] == s[r]:
                if (r - l + 1) > resLen:
                    res = s[l:r+1]
                    resLen = r - l + 1
                l -= 1
                r += 1

            #even length
            l, r = i, i + 1
            while l >= 0 and r < len(s) and s[l] == s[r]:
                if(r - l + 1) > resLen:
                    res = s[l:r+1]
                    resLen = r - l + 1
                l -= 1
                r += 1
        return res
```