Micah Kim
08/05/2022

A) The following Python function checks whether a string is a palindrome. Please explain, in 250 words or less, how you'd improve this code and why. We're not looking for a simple one-line rewrite here - submissions will be graded based on the clarity by which you describe what the improvements are, and also WHY they should be made.

```
def is_palindrone(s):
    r=""
    for c in s:
        r = c +r
    for x in range(0, len(s)):
        if s[x] == r[x]:
            x = True
        else:
            return False
    return x
```

* Assuming the current solution works, I will assume the input "s" has no empty spaces, and is lowercase.

Solution Outline

- The current solution stores a reverse of the input in the variable "r" (O(n) time). Then, it runs through the input and compares the equality of the input and the reverse (O(n) time).
- Instead of reverse equality, a palindrome can be thought of as a word that mirrors itself at the center.
- Ex) "anna", "an" is a reflection of "na."

Actions

- Instead of reversing the string, we can set a "l" and "r" pointer to the beginning and end of the input string, respectively. Within a loop, check if the "s[l]" is equal to "s[r]," incrementing "l" and decrementing "r" as long as they don't equal each other. If this condition isn't met, simply return False. If the program exits the loop, this means it is a palindrome and we can return True.
- This solution's time complexity is O(n/2) since we only loop once using two pointers, and a space complexity of O(1) since we only need additional space for the pointers.

Improvements

- By not creating a "r" string, we reduce the space usage of this program. With very large input strings, using more storage than necessary can decrease performance and increase costs for more storage.

- By using pointers, and comparing the two halves of the string, we can minimize the runtime complexity of this program by 4 times. This will save a large amount of time with large inputs since it only has to compare using halves of the string.