```
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
```

Unclarified things about this problem:

- Whether there are white spaces in the input s
- Does the case-sensitivity matters for this problem
    - If there are upper case chars, whether a and A are the same characters?
- Does s only contain alphanumerical characters, or special characters could exist in s?
    - If s does contain special characters, should we ignore them or take them into the consideration
        - abc => True
        - How about a!b&c? Is this still true?

Based on the given code implementation, I will assume the string only consists of lower case alphanumerical characters without spaces and special characters.

Improvements**:**

- The first three lines of code are creating **O(n)** extra spaces to store the reversed string, which is not necessarily.
- x = True and return x will produce an **error** for certain test cases.
    - If s == “”, we will get error UnboundLocalError: local variable 'x' referenced before assignment
- For the for loop, we can just return False directly when s[x] != r[x], no need to assign True to x
- My approach to this problem is using pointers to traverse the string from the beginning and from the end simultaneously without creating any extra spaces.
    - **Time O(n/2) = O(n)**
    - **Space = O(1)**

```
def is_palindrone(s):
    if not s:
        return True
```

```
    l, r = 0, len(s) - 1

    while l < r:
        if s[l] != s[r]:
            return False
    return True
```