Leetcode Problem 1. (Easy)

Valid Palindrome

A phrase is a **palindrome** if, after converting all uppercase letters into lowercase letters and removing all non-alphanumeric characters, it reads the same forward and backward. Alphanumeric characters include letters and numbers.

Given a string s, return true *if it is a* ***palindrome****, or* false *otherwise*.

Example 1:

Input: s = "A man, a plan, a canal: Panama"
Output: true
Explanation: "amanaplanacanalpanama" is a palindrome.

Example 2:

Input: s = "race a car"
Output: false
Explanation: "raceacar" is not a palindrome.

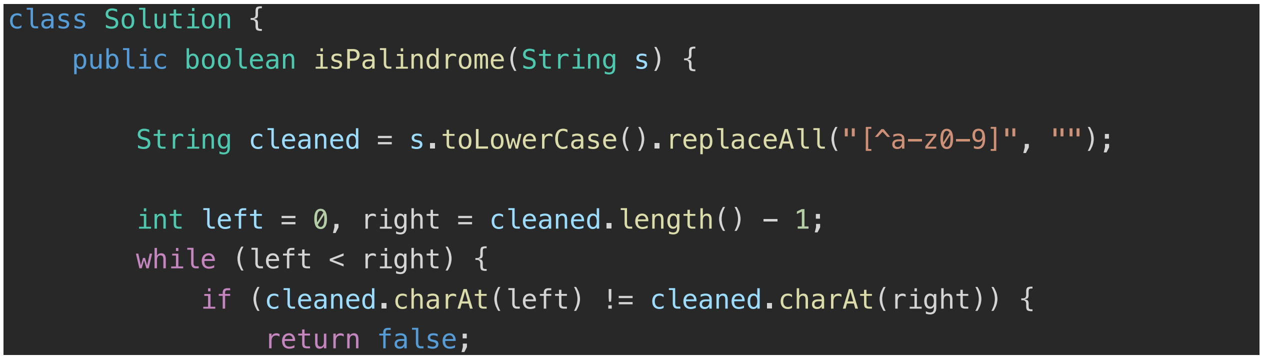
Example 3:

Input: s = " "
Output: true
Explanation: s is an empty string "" after removing non-alphanumeric characters.
Since an empty string reads the same forward and backward, it is a palindrome.

Constraints:

- 1 <= s.length <= 2 * 10^5
- s consists only of printable ASCII characters.

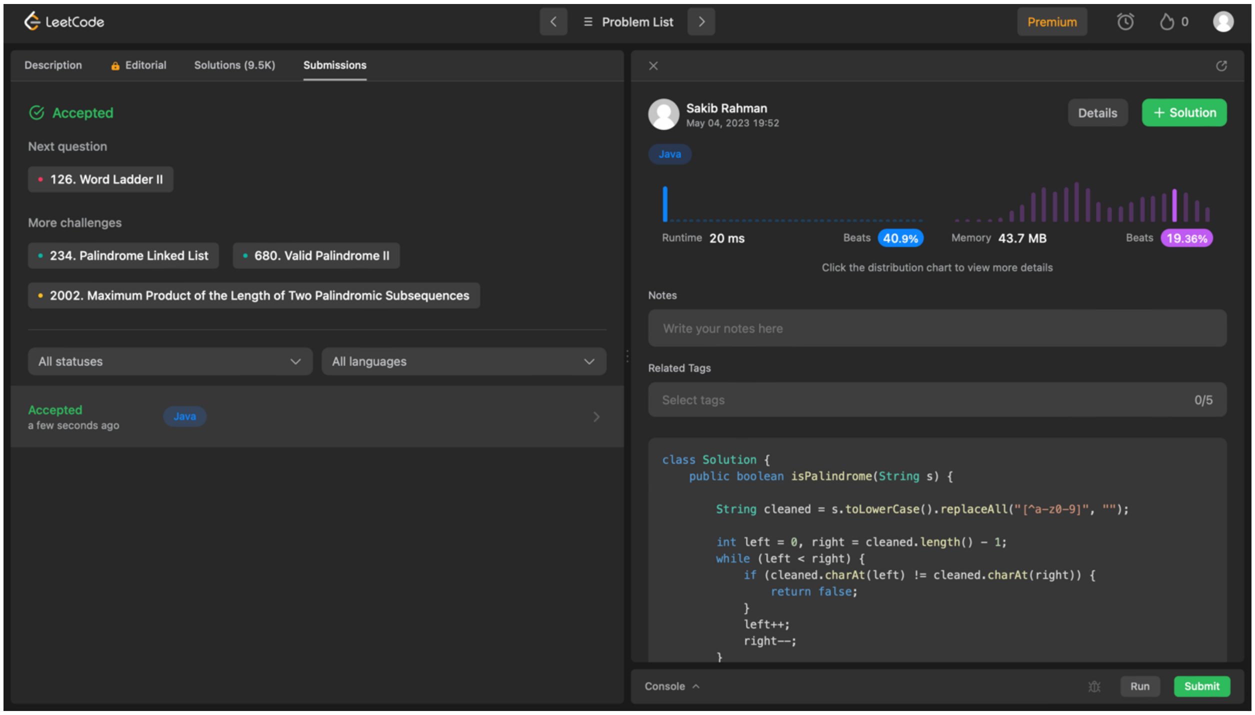
Link: https://leetcode.com/problems/valid-palindrome/



```
class Solution {
    public boolean isPalindrome(String s) {

        String cleaned = s.toLowerCase().replaceAll("[^a-z0-9]", "");

        int left = 0, right = cleaned.length() - 1;
        while (left < right) {
            if (cleaned.charAt(left) != cleaned.charAt(right)) {
                return false;
            }
            left++;
            right--;
        }
        return true;
    }
}
```



Accepted

Next question

- 126. Word Ladder II

More challenges

- 234. Palindrome Linked List
- 680. Valid Palindrome II
- 2002. Maximum Product of the Length of Two Palindromic Subsequences

Accepted
a few seconds ago
Java

Sakib Rahman
May 04, 2023 19:52

Java

Runtime 20 ms Beats 40.9%
Memory 43.7 MB Beats 19.36%

Click the distribution chart to view more details

Notes

Write your notes here

Related Tags

Select tags 0/5

```
class Solution {
    public boolean isPalindrome(String s) {

        String cleaned = s.toLowerCase().replaceAll("[^a-z0-9]", "");

        int left = 0, right = cleaned.length() - 1;
        while (left < right) {
            if (cleaned.charAt(left) != cleaned.charAt(right)) {
                return false;
            }
            left++;
            right--;
        }
```