125 Valid Palindrome

Company : AMazon apple bloomberg facebook microsoft oracle wayfair

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.

Approaches :

Brute force approach(Comparing with its reverse) :

First reverse the string and then compare with its original string if it matches then string is valid palindrome and if not then it is not valid palindrome.

For it time complexity is o(n)

Space complexity is o(n)

Approach i used Optimized approach(using two pointers) :

What we can do is take two pointer variables, *start* and *end* and point them with the two ends of the input string.

2. Now move the *start* pointer to right so it points to a alphanumeric character. Similarly move *end* pointer to left so it also points to a alphanumeric character.

3. Now check if both the characters are same or not (ignoring cases):

- If it is not equal then we know string is not a valid palindrome, hence return false.
- Else continue to next iteration and repeat the same process of moving both pointers to point to next alphanumeric character till *start<end*.

4. After loop finishes, the string is said to be palindrome, hence return true.

Time complexity : o(n)

Space complexity : o(1)

```
class Solution
{
```

```
public:
    bool isPalindrome(string s)
    {
        int start = 0,end = s,size()-1;

        //case 1 if string is empty then return true.
         if(s.size()==0)
         {
             return true;
         }
         while (start <= end) {
            while (!isalnum(s[start]) && start < end) {
                start++;
            }
            while (!isalnum(s[end]) && start < end) {
                end--;
            }
            if (lowercase(s[start]) != lowercase(s[end])) {
                return false;
            }
            start++, end--;
    }
            //check if character is alphabet and digits.
    bool isalnum(char c)
    {
        if(('A' <= c && c <= 'Z') || ('a' <= c && c <= 'z') || ('0' <= c && c <= '9'))
            return true;
        else
            return false;
    }
    //check if character is in uppercase then it converts to lowercase if it is not then remins same.
    bool lowercase(char c)
    {
        if('A' <= c && c <= 'Z')
            return c+32;
        else
            return c;
    }
};
```