<https://leetcode.com/problems/valid-palindrome/>

**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 \* 105

s consists only of printable ASCII characters.

**Method 1: (Brute Force)**

For

Time Complexity: O()

Space Complexity: O()

**Method 2: ()**

Use two pointers i and j to point to start and end index.

For non alphanumeric characters increment i and decrement j

For alphanumeric characters check if chars at index i and j are same.

Time Complexity: O(n) *[]*

Space Complexity: O(1) *[]*

bool isAlphanumeric(char ch){

        if(ch<'0'||(ch>'9' && ch<'A') || (ch>'Z' && ch<'a') || ch>'z')

            return false;

        else return true;

    }

    bool isPalindrome(string s) {

        int i=0, j=s.size()-1;

        while(i<j){

            while(i<j && !isAlphanumeric(s[i]))

                i++;

            while(j>i && !isAlphanumeric(s[j]))

                j--;

            if(isAlphanumeric(s[i]) && isAlphanumeric(s[j]) && tolower(s[i])!=tolower(s[j]))

                return false;

            else {

                i++;

                j--;

            }

        }

        return true;

    }