<https://leetcode.com/problems/trapping-rain-water/>

**Trapping Rain Water**

**Given n non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it can trap after raining.**

Example 1:

Input: height = [0,1,0,2,1,0,1,3,2,1,2,1]

Output: 6

Explanation: The above elevation map (black section) is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped.

Example 2:

Input: height = [4,2,0,3,2,5]

Output: 9

Constraints:

n == height.length

1 <= n <= 2 \* 104

0 <= height[i] <= 105

**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;

    }