**1.Easy\03.Check_if_array_is_sorted_and_rotated.cpp**

```cpp
/*
QUESTION:-
Given an array nums, return true if the array was originally sorted in non-decreasing order, then rotated some number of positions (including zero). Otherwise, return false.
There may be duplicates in the original array.
Example 1:

Input: nums = [3,4,5,1,2]
Output: true
Explanation: [1,2,3,4,5] is the original sorted array.
You can rotate the array by x = 3 positions to begin on the the element of value 3: [3,4,5,1,2].
Example 2:

Input: nums = [2,1,3,4]
Output: false
Explanation: There is no sorted array once rotated that can make nums.

*/

/*
APPROACH:-
Compare all neignbour elements (a,b) in A,
the case of a > b can happen at most once.

Note that the first element and the last element are also connected.

If all a <= b, A is already sorted so answer is true.
If all a <= b but only one a > b, and the first element is greater than equal to last element we can rotate and make b the first element so answer is true.
Other case, return false.
*/

// CODE:-
bool check(vector<int> &nums)
{
    int cnt = 0;
    int n = nums.size();
    for (int i = 0; i < n - 1; i++)
    {
        if (nums[i] > nums[i + 1])
            cnt++;
    }
    if (cnt == 0)
        return true;
    else if (cnt == 1 && nums[0] >= nums[n - 1])
        return true;
    return false;
}

// TIME COMPLEXITY = O(N)
// SPACE COMPLEXITY = O(0)
```