***Check if an Array is Sorted***

Given an array of size **n**, write a program to check if it is sorted in ascending order or not. Equal values are allowed in an array and two consecutive equal values are considered sorted.

**Examples:**

Input : 20 21 45 89 89 90

Output : Yes

Input : 20 20 45 89 89 90

Output : Yes

Input : 20 20 78 98 99 97

Output : No

**Naive Approach:**

C++Java

import java.util.\*;

import java.io.\*;

import java.lang.\*;

class GFG

{

static boolean isSorted(int arr[], int n)

{

for(int i = 0; i < n; i++)

{

for(int j = i + 1; j < n; j++)

{

if(arr[j] < arr[i])

return false;

}

}

return true;

}

public static void main(String args[])

{

int arr[] = {7, 2, 30, 10}, n = 4;

System.out.println(isSorted(arr, n));

}

}

**Output:**

false

**Iterative approach:**

C++Java

// Recursive approach to check if an

// Array is sorted or not

class GFG {

// Function that returns true if array is

// sorted in non-decreasing order.

static boolean arraySortedOrNot(int arr[], int n)

{

// Array has one or no element

if (n == 0 || n == 1)

return true;

for (int i = 1; i < n; i++)

// Unsorted pair found

if (arr[i - 1] > arr[i])

return false;

// No unsorted pair found

return true;

}

// driver code

public static void main(String[] args)

{

int arr[] = { 20, 23, 23, 45, 78, 88 };

int n = arr.length;

if (arraySortedOrNot(arr, n))

System.out.print("Yes\n");

else

System.out.print("No\n");

}

}

**Output**

Yes

**Time Complexity: O(n)**  
**Auxiliary Space: O(1)**