**Maximum Length Bitonic Subarray Editorial**

## **Approach**

We create two arrays - 'inc' and 'dec'

1. inc[i] stores the length of increasing subarray till i.
2. dec[i] stores the length of decreasing subarray starting from index i.
3. Doing so gives us the length of increasing and decreasing subarray at each index in O(n) time.
4. We calculate the length of the longest bitonic subarray by finding the maximum inc[i] + dec[i] - 1
5. We subtract one since the current element at ith index is included in both the increasing and decreasing subarray lengths.

## **Algorithm**

1. Initialize inc[0] to 1 and dec[n-1] to 1
2. Creating inc[] array  
   a. Till end of the array ie, i=1 to n, if arr[i] > arr[i-1] then inc[i] = inc[i-1] + 1. else, inc[i] = 1
3. Creating dec[] array  
   a. From the end of the array ie, i = n-2 till i =0, if arr[i] > arr[i+1] then dec[i] = dec[i+1] +1 else, dec[i] = 1