Replace every element with the greatest element on right side

Given an array of integers, replace every element with the next greatest element (greatest element on the right side) in the array. Since there is no element next to the last element, replace it with -1. For example, if the array is {16, 17, 4, 3, 5, 2}, then it should be modified to {17, 5, 5, 5, 2, -1}.

The question is very similar to this post and solutions are also similar.

A **naive method** is to run two loops. The outer loop will one by one pick array elements from left to right. The inner loop will find the greatest element present after the picked element. Finally the outer loop will replace the picked element with the greatest element found by inner loop. The time complexity of this method will be O(n*n).

A **tricky method** is to replace all elements using one traversal of the array. The idea is to start from the rightmost element, move to the left side one by one, and keep track of the maximum element. Replace every element with the maximum element.

C/C++

```
// C Program to replace every element with the greatest
// element on right side
#include <stdio.h>

/* Function to replace every element with the
   next greatest element */
void nextGreatest(int arr[], int size)
{
  // Initialize the next greatest element
  int max_from_right =  arr[size-1];

  // The next greatest element for the rightmost element
  // is always -1
  arr[size-1] = -1;

  // Replace all other elements with the next greatest
  for(int i = size-2; i >= 0; i--)
  {
    // Store the current element (needed later for updating
    // the next greatest element)
    int temp = arr[i];
```

```
    // Replace current element with the next greatest
    arr[i] = max_from_right;

    // Update the greatest element, if needed
    if(max_from_right < temp)
       max_from_right = temp;
  }
}

/* A utility Function that prints an array */
void printArray(int arr[], int size)
{
  int i;
  for (i=0; i < size; i++)
    printf("%d ", arr[i]);
  printf("\n");
}

/* Driver program to test above function */
int main()
{
  int arr[] = {16, 17, 4, 3, 5, 2};
  int size = sizeof(arr)/sizeof(arr[0]);
  nextGreatest (arr, size);
  printf ("The modified array is: \n");
  printArray (arr, size);
  return (0);
}
```

Run on IDE

Java

```
// Java Program to replace every element with the
// greatest element on right side
import java.io.*;

class NextGreatest
{
    /* Function to replace every element with the
       next greatest element */
    static void nextGreatest(int arr[])
    {
        int size = arr.length;

        // Initialize the next greatest element
        int max_from_right =  arr[size-1];

        // The next greatest element for the rightmost
        // element is always -1
        arr[size-1] = -1;

        // Replace all other elements with the next greatest
        for (int i = size-2; i >= 0; i--)
        {
            // Store the current element (needed later for
            // updating the next greatest element)
            int temp = arr[i];

            // Replace current element with the next greatest
            arr[i] = max_from_right;
```

```
            // Update the greatest element, if needed
            if(max_from_right < temp)
            max_from_right = temp;
        }
    }

    /* A utility Function that prints an array */
    static void printArray(int arr[])
    {
        for (int i=0; i < arr.length; i++)
        System.out.print(arr[i]+" ");
    }

    public static void main (String[] args)
    {
        int arr[] = {16, 17, 4, 3, 5, 2};
        nextGreatest (arr);
        System.out.println("The modified array:");
        printArray (arr);
    }
}
/*This code is contributed by Devesh Agrawal*/
```

Run on IDE

Python

```
# Python Program to replace every element with the
# greatest element on right side

# Function to replace every element with the next greatest
# element
def nextGreatest(arr):

    size = len(arr)

    # Initialize the next greatest element
    max_from_right = arr[size-1]

    # The next greatest element for the rightmost element
    # is always -1
    arr[size-1] = -1

    # Replace all other elements with the next greatest
    for i in range(size-2,-1,-1):

        # Store the current element (needed later for updating
        # the next greatest element)
        temp = arr[i]

        # Replace current element with the next greatest
        arr[i]=max_from_right

        # Update the greatest element, if needed
        if max_from_right< temp:
            max_from_right=temp

# Utility function to print an array
def printArray(arr):
    for i in range(0,len(arr)):
        print arr[i],
```

```
# Driver function to test above function
arr = [16, 17, 4, 3, 5, 2]
nextGreatest(arr)
print "Modified array is"
printArray(arr)
# This code is contributed by Devesh Agrawal
```

Run on IDE

Output:

```
The modified array is:
17 5 5 5 2 -1
```

Time Complexity: O(n) where n is the number of elements in array.

Please write comments if you find any of the above codes/algorithms incorrect, or find other ways to solve the same problem.

44 Comments Category: Arrays

Related Posts:

- Longest Span with same Sum in two Binary arrays
- Count Inversions of size three in a give array
- Find the subarray with least average
- Count triplets with sum smaller than a given value
- Find zeroes to be flipped so that number of consecutive 1's is maximized
- Reorder an array according to given indexes
- Find maximum value of Sum(i*arr[i]) with only rotations on given array allowed

- Find maximum average subarray of k length

(Login to Rate and Mark)

1.7 Average Difficulty : **1.7/5.0**
Based on **4** vote(s)

Add to TODO List

Mark as DONE

Writing code in comment? Please use code.geeksforgeeks.org, generate link and share the link here.