Find the Minimum length Unsorted Subarray, sorting which makes the complete array sorted

Difficulty Level : Medium • Last Updated : 16 Jun, 2022

Given an unsorted array arr[0..n-1] of size n, find the minimum length subarray arr[s..e] such that sorting this subarray makes the whole array sorted.

Examples:

1. If the input array is [10, 12, 20, 30, 25, 40, 32, 31, 35, 50, 60], your program should be able to find that the subarray lies between the indexes 3 and 8.
2. If the input array is [0, 1, 15, 25, 6, 7, 30, 40, 50], your program should be able to find that the subarray lies between the indexes 2 and 5.

Solution:

No video with supported format and MIME type found.

1. **Find the candidate unsorted subarray**
    1. Scan from left to right and find the first element which is greater than the next element. Let *s* be the index of such an element. In the above example 1, *s* is 3 (index of 30).
    2. Scan from right to left and find the first element (first in right to left order) which is smaller than the next element (next in right to left order). Let *e* be the index of such an element. In the above example 1, e is 7 (index of 31).

2. **Check whether sorting the candidate unsorted subarray makes the complete array sorted or not. If not, then include more elements in the subarray.**
    1. Find the minimum and maximum values in *arr[s..e]*. Let minimum and maximum values be *min* and *max*. *min* and *max* for [30, 25, 40, 32, 31] are 25 and 40 respectively.
    2. Find the first element (if there is any) in *arr[0..s-1]* which is greater than *min*, change *s* to index of this element. There is no such element in above example 1.
    3. Find the last element (if there is any) in *arr[e+1..n-1]* which is

smaller than max, change *e* to index of this element. In the above example 1, e is changed to 8 (index of 35)

3. **Print *s* and *e*.**

Implementation:

C++

```
// C++ program to find the Minimum length Unsorted Subarray,
// sorting which makes the complete array sorted
#include<bits/stdc++.h>
using namespace std;

void printUnsorted(int arr[], int n)
{
int s = 0, e = n-1, i, max, min;

// step 1(a) of above algo
for (s = 0; s < n-1; s++)
{
    if (arr[s] > arr[s+1])
    break;
}
if (s == n-1)
{
    cout << "The complete array is sorted";
    return;
}

// step 1(b) of above algo
for(e = n - 1; e > 0; e--)
{
    if(arr[e] < arr[e-1])
    break;
}

// step 2(a) of above algo
max = arr[s]; min = arr[s];
for(i = s + 1; i <= e; i++)
{
    if(arr[i] > max)
```

```
        max = arr[i];
        if(arr[i] < min)
        min = arr[i];
    }

    // step 2(b) of above algo
    for( i = 0; i < s; i++)
    {
        if(arr[i] > min)
        {
        s = i;
        break;
        }
    }

    // step 2(c) of above algo
    for( i = n -1; i >= e+1; i--)
    {
        if(arr[i] < max)
        {
        e = i;
        break;
        }
    }

    // step 3 of above algo
    cout << "The unsorted subarray which"
         << " makes the given array" << endl
         << "sorted lies between the indices "
         << s << " and " << e;
    return;
    }

    int main()
    {
        int arr[] = {10, 12, 20, 30, 25,
                     40, 32, 31, 35, 50, 60};
        int arr_size = sizeof(arr)/sizeof(arr[0]);
        printUnsorted(arr, arr_size);
        getchar();
        return 0;
    }

    // This code is contributed
    // by Akanksha Rai
```

C

```c
// C program to find the Minimum length Unsorted Subarray,
// sorting which makes the complete array sorted
#include<stdio.h>

void printUnsorted(int arr[], int n)
{
  int s = 0, e = n-1, i, max, min;

  // step 1(a) of above algo
  for (s = 0; s < n-1; s++)
  {
    if (arr[s] > arr[s+1])
      break;
  }
  if (s == n-1)
  {
    printf("The complete array is sorted");
    return;
  }

  // step 1(b) of above algo
  for(e = n - 1; e > 0; e--)
  {
    if(arr[e] < arr[e-1])
      break;
  }

  // step 2(a) of above algo
  max = arr[s]; min = arr[s];
  for(i = s + 1; i <= e; i++)
  {
    if(arr[i] > max)
      max = arr[i];
    if(arr[i] < min)
      min = arr[i];
  }

  // step 2(b) of above algo
  for( i = 0; i < s; i++)
  {
    if(arr[i] > min)
    {
```

```
      s = i;
      break;
    }
  }

  // step 2(c) of above algo
  for( i = n -1; i >= e+1; i--)
  {
    if(arr[i] < max)
    {
      e = i;
      break;
    }
  }

  // step 3 of above algo
  printf(" The unsorted subarray which makes the given array "
         " sorted lies between the indees %d and %d", s, e);
  return;
}

int main()
{
  int arr[] = {10, 12, 20, 30, 25, 40, 32, 31, 35, 50, 60};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  printUnsorted(arr, arr_size);
  getchar();
  return 0;
}
```

Java

```
// Java program to find the Minimum length Unsorted Subarray,
// sorting which makes the complete array sorted
class Main
{
    static void printUnsorted(int arr[], int n)
    {
      int s = 0, e = n-1, i, max, min;

      // step 1(a) of above algo
      for (s = 0; s < n-1; s++)
      {
```

```
    if (arr[s] > arr[s+1])
      break;
  }
  if (s == n-1)
  {
    System.out.println("The complete array is sorted");
    return;
  }

  // step 1(b) of above algo
  for(e = n - 1; e > 0; e--)
  {
    if(arr[e] < arr[e-1])
      break;
  }

  // step 2(a) of above algo
  max = arr[s]; min = arr[s];
  for(i = s + 1; i <= e; i++)
  {
    if(arr[i] > max)
      max = arr[i];
    if(arr[i] < min)
      min = arr[i];
  }

  // step 2(b) of above algo
  for( i = 0; i < s; i++)
  {
    if(arr[i] > min)
    {
      s = i;
      break;
    }
  }

  // step 2(c) of above algo
  for( i = n -1; i >= e+1; i--)
  {
    if(arr[i] < max)
    {
      e = i;
      break;
    }
  }
```

```
        // step 3 of above algo
        System.out.println(" The unsorted subarray which"+
                           " makes the given array sorted lies"+
                          "  between the indices "+s+" and "+e);
        return;
    }

    public static void main(String args[])
    {
        int arr[] = {10, 12, 20, 30, 25, 40, 32, 31, 35, 50, 60};
        int arr_size = arr.length;
        printUnsorted(arr, arr_size);
    }
}
```

Python3

```
# Python3 program to find the Minimum length Unsorted Subarray,
# sorting which makes the complete array sorted
def printUnsorted(arr, n):
    e = n-1
    # step 1(a) of above algo
    for s in range(0,n-1):
        if arr[s] > arr[s+1]:
            break

    if s == n-1:
        print ("The complete array is sorted")
        exit()

    # step 1(b) of above algo
    e= n-1
    while e > 0:
        if arr[e] < arr[e-1]:
            break
        e -= 1

    # step 2(a) of above algo
    max = arr[s]
    min = arr[s]
    for i in range(s+1,e+1):
        if arr[i] > max:
            max = arr[i]
```

```
            if arr[i] < min:
                min = arr[i]

        # step 2(b) of above algo
        for i in range(s):
            if arr[i] > min:
                s = i
                break

        # step 2(c) of above algo
        i = n-1
        while i >= e+1:
            if arr[i] < max:
                e = i
                break
            i -= 1

        # step 3 of above algo
        print ("The unsorted subarray which makes the given array")
        print ("sorted lies between the indexes %d and %d"%( s, e))

arr = [10, 12, 20, 30, 25, 40, 32, 31, 35, 50, 60]
arr_size = len(arr)
printUnsorted(arr, arr_size)

# This code is contributed by Shreyanshi Arun
```

C#

```
// C# program to find the Minimum length Unsorted Subarray,
// sorting which makes the complete array sorted

using System;

class GFG
{
    static void printUnsorted(int []arr, int n)
    {
    int s = 0, e = n-1, i, max, min;

    // step 1(a) of above algo
    for (s = 0; s < n-1; s++)
    {
        if (arr[s] > arr[s+1])
```

```
        break;
    }
    if (s == n-1)
    {
        Console.Write("The complete " +
                            "array is sorted");
        return;
    }

    // step 1(b) of above algo
    for(e = n - 1; e > 0; e--)
    {
        if(arr[e] < arr[e-1])
        break;
    }

    // step 2(a) of above algo
    max = arr[s]; min = arr[s];

    for(i = s + 1; i <= e; i++)
    {
        if(arr[i] > max)
            max = arr[i];

        if(arr[i] < min)
            min = arr[i];
    }

    // step 2(b) of above algo
    for( i = 0; i < s; i++)
    {
        if(arr[i] > min)
        {
            s = i;
            break;
        }
    }

    // step 2(c) of above algo
    for( i = n -1; i >= e+1; i--)
    {
        if(arr[i] < max)
        {
            e = i;
            break;
```

```
        }
    }

    // step 3 of above algo
    Console.Write(" The unsorted subarray which"+
            " makes the given array sorted lies \n"+
              " between the indices "+s+" and "+e);
    return;
    }

    public static void Main()
    {
        int []arr = {10, 12, 20, 30, 25, 40,
                                32, 31, 35, 50, 60};
        int arr_size = arr.Length;

        printUnsorted(arr, arr_size);
    }
}

// This code contributed by Sam007
```

PHP

```
<?php
// PHP program to find the Minimum length Unsorted Subarray,
// sorting which makes the complete array sorted
function printUnsorted(&$arr, $n)
{
    $s = 0;
    $e = $n - 1;

    // step 1(a) of above algo
    for ($s = 0; $s < $n - 1; $s++)
    {
        if ($arr[$s] > $arr[$s + 1])
        break;
    }
    if ($s == $n - 1)
    {
        echo "The complete array is sorted";
        return;
    }
```

```
        // step 1(b) of above algo
        for($e = $n - 1; $e > 0; $e--)
        {
            if($arr[$e] < $arr[$e - 1])
            break;
        }

        // step 2(a) of above algo
        $max = $arr[$s];
        $min = $arr[$s];
        for($i = $s + 1; $i <= $e; $i++)
        {
            if($arr[$i] > $max)
                $max = $arr[$i];
            if($arr[$i] < $min)
                $min = $arr[$i];
        }

        // step 2(b) of above algo
        for( $i = 0; $i < $s; $i++)
        {
            if($arr[$i] > $min)
            {
                $s = $i;
                break;
            }
        }

        // step 2(c) of above algo
        for( $i = $n - 1; $i >= $e + 1; $i--)
        {
            if($arr[$i] < $max)
            {
                $e = $i;
                break;
            }
        }

        // step 3 of above algo
        echo " The unsorted subarray which makes " .
                        "the given array " . "\n" .
                " sorted lies between the indees " .
                                    $s . " and " . $e;
        return;
    }
```

```
// Driver code
$arr = array(10, 12, 20, 30, 25, 40,
             32, 31, 35, 50, 60);
$arr_size = sizeof($arr);
printUnsorted($arr, $arr_size);

// This code is contributed
// by ChitraNayal
?>
```

Javascript

```
<script>

// Javascript program to find the Minimum length Unsorted Subarray,
// sorting which makes the complete array sorted

    function printUnsorted(arr,n)
    {
        let s = 0, e = n-1, i, max, min;
        // step 1(a) of above algo
        for (s = 0; s < n-1; s++)
        {
            if (arr[s] > arr[s+1])
                break;
        }
        if (s == n-1)
        {
            document.write("The complete array is sorted");
            return;
        }
        // step 1(b) of above algo
        for(e = n - 1; e > 0; e--)
        {
            if(arr[e] < arr[e-1])
                break;
        }
        // step 2(a) of above algo
        max = arr[s]; min = arr[s];
        for(i = s + 1; i <= e; i++)
        {
            if(arr[i] > max)
                max = arr[i];
```

```
            if(arr[i] < min)
                min = arr[i];
        }
        // step 2(b) of above algo
        for( i = 0; i < s; i++)
        {
            if(arr[i] > min)
            {
                s = i;
                break;
            }
        }
        // step 2(c) of above algo
        for( i = n -1; i >= e+1; i--)
        {
            if(arr[i] < max)
            {
                e = i;
                break;
            }
        }
        // step 3 of above algo
        document.write(" The unsorted subarray which"+
                        " makes the given array sorted lies"+
                      "  between the indees "+s+" and "+e);
        return;
    }
    let arr=[10, 12, 20, 30, 25, 40, 32, 31, 35, 50, 60];
    let arr_size = arr.length;
    printUnsorted(arr, arr_size);

    // This code is contributed by avanitrachhadiya2155

</script>
```

Output :

```
 The unsorted subarray which makes the given array
sorted lies between the indexes 3 and 8
```

Time Complexity : O(n)

Auxiliary Space : O(1)

Please write comments if you find the above code/algorithm incorrect, or find better ways to solve the same problem.

Previous

Sort elements by frequency | Set 2

Next

Sort numbers stored on different machines

RECOMMENDED ARTICLES

Page : 1 2 3

01 Which sorting algorithm makes minimum number of memory writes?
01, Feb 11

02 Maximize partitions that if sorted individually makes the whole Array sorted
13, Jan 22

05 Find a pair of elements swapping which makes sum of two arrays same
30, Jun 17

06 Minimize cost to sort given array by sorting unsorted subarrays
03, Jan 22

03 **Check if removal of a subsequence of non-adjacent elements makes the array sorted**
23, May 21

04 **Maximize M such that swapping arr[i] with arr[i+M] makes Array sorted**
13, Jul 22

07 **Find index of first occurrence when an unsorted array is sorted**
05, Aug 18

08 **Why is it faster to process sorted array than an unsorted array ?**
11, Jul 17

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Medium

Easy | Normal | Medium | Hard | Expert

Improved By : ukasp, Akanksha_Rai, avanitrachhadiya2155, singhalrishabh0904, codewithshinchan, kothavvsaakash, hardikkoriintern

Article Tags : Arrays, Sorting

Practice Tags : Arrays, Sorting

Improve Article | Report Issue

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

Load Comments

GeeksforGeeks
A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

NodeJS