581. Shortest Unsorted Continuous Subarray

03 May 2022 04:59 PM

Given an integer array `nums`, you need to find one **continuous subarray** that if you only sort this subarray in ascending order, then the whole array will be sorted in ascending order.

Return *the shortest such subarray and output its length.*

Example 1:

```
Input: nums = [2,6,4,8,10,9,15]
Output: 5
Explanation: You need to sort [6, 4, 8, 10, 9] in ascending order to make the whole array sorted in ascending order.
```

Example 2:

```
Input: nums = [1,2,3,4]
Output: 0
```

Example 3:

```
Input: nums = [1]
Output: 0
```

Naive Approach:

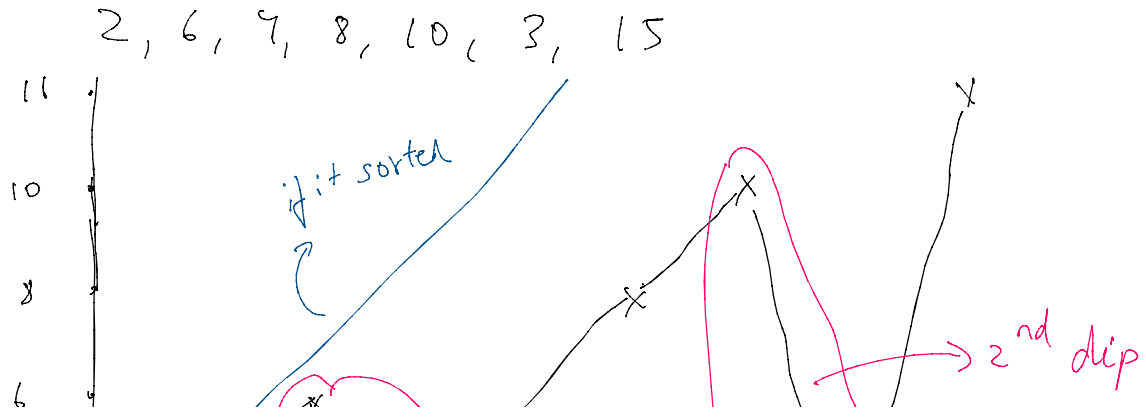
2 6 4 8 10 9 15

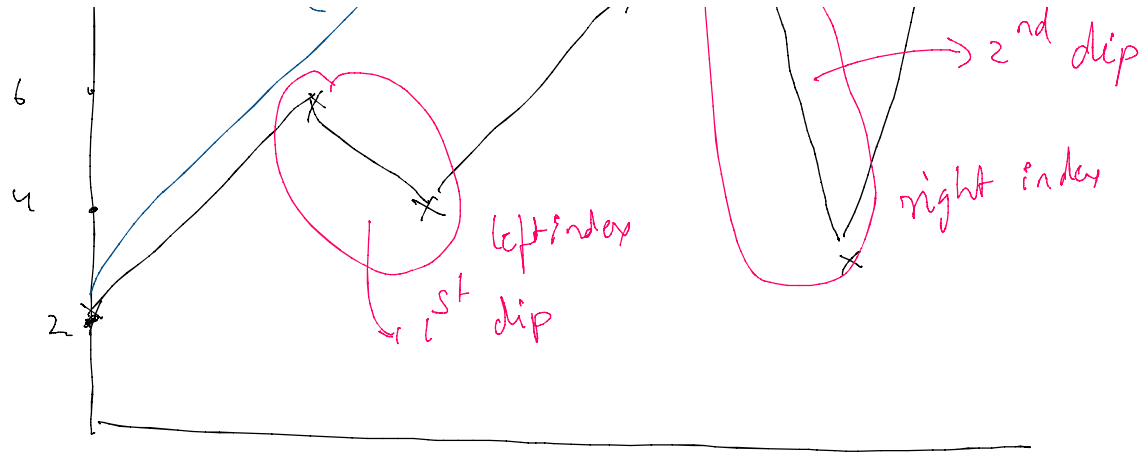
Sort ⇒ 2 4 6 8 9 10 15

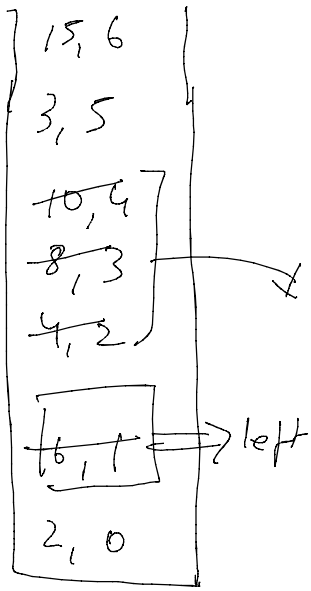
1st 5th

Compare and find the mismatch and find the difference

Output ⇒ 5



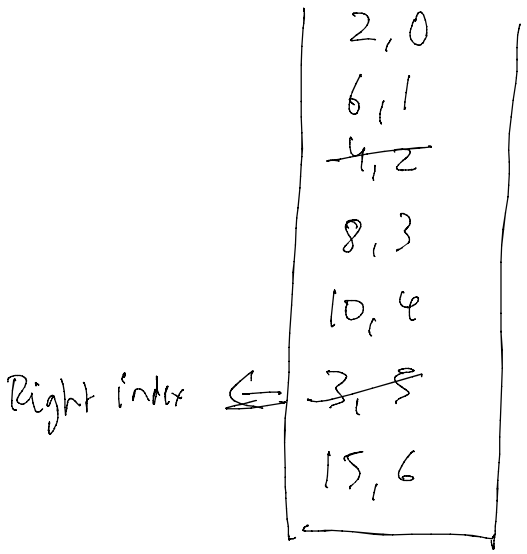




→ leftmost index where disturbance occurred

Left most index = 1

To find right most index we go Right → left :



R → 5

right - left + 1 = 5 - 1 + 1

= 5 output

```java
class Solution {
    public int findUnsortedSubarray(int[] nums) {
        Stack<Integer>st = new Stack<>();

        // find left most index
        int left = nums.length - 1;

        for(int i = 0; i < nums.length;){
            if(st.empty()){
                st.push(i);
                i++;
            } else {
                if(nums[st.peek()] > nums[i]){
                    left = Math.min(left, st.peek());
                    st.pop();
                } else {
                    st.push(i);
                    i++;
                }
            }
        }
        st.clear();

        // find right most index
        int right = 0;
        for(int i = nums.length-1; i>=0;){
            if(st.empty()){
                st.push(i);
                i--;
            } else {
                if(nums[st.peek()] < nums[i]){
                    right = Math.max(right, st.peek());
                    st.pop();
                } else {
                    st.push(i);
                    i--;
                }
            }
        }
        if(left >= right){
            return 0;
        } else {
            return right - left + 1;
        }
    }
}
```