```
class Solution {
    public int findUnsortedSubarray(int[] nums) {
        int min = Integer.MAX_VALUE, max = Integer.MIN_VALUE;
        boolean flag = false;
        for (int i = 1; i < nums.length; i++) {
            if (nums[i] < nums[i-1]) {
                flag = true;
            }
            if (flag) {
                min = Math.min(min, nums[i]);
            }
        }
        flag = false;
        for (int i = nums.length-2; i >= 0; i--) {
            if (nums[i] > nums[i+1]) {
                flag = true;
            }
            if (flag) {
                max = Math.max(max, nums[i]);
            }
        }

        int right, left;
        for (left = 0; left < nums.length; left++) {
            if (nums[left] > min) {
                break;
            }
        }

        for (right = nums.length-1; right >= 0; right--) {
            if (nums[right] < max) {
                break;
            }
        }
        return right - left < 0 ? 0 : right - left + 1;
        /*
        int[] numsCopy = nums.clone();
        Arrays.sort(numsCopy);
        int max = 0, min = nums.length;
        for (int i = 0; i < nums.length; i++) {
            if (nums[i] != numsCopy[i]) {
                max = Math.max(max, i);
                min = Math.min(min, i);
            }
        }
        return max - min >= 0 ? max - min + 1 : 0;
        */
        /*
        int len = nums.length, left = nums[0], right = nums[len-1], begin = -1, end = -2;

        for (int i = 1; i < len; i++) {
            left = Math.max(left, nums[i]);
            right = Math.min(right, nums[len - i - 1]);
            if (nums[i] < left)
```

```
                end = i;
            if (nums[len-i-1] > right)
                begin = len-i-1;

        }
        return end - begin + 1;
        */
    }
}
```