```java
/*Author: Bochen (mddboc@foxmail.com)
Last Modified: Tue Apr 10 22:28:44 CST 2018*/

/*Given an array and a value, remove all instances of that value in-place and return the new length.

        Do not allocate extra space for another array, you must do this by modifying the input array in-place with O(1) extra memory.

        The order of elements can be changed. It doesn't matter what you leave beyond the new length.

        Example:

        Given nums = [3,2,2,3], val = 3,

        Your function should return length = 2, with the first two elements of nums being 2.*/


import java.lang.System;
import java.util.*;
import java.lang.Math;
import java.util.HashMap;


class ListNode
{
    int val;
    ListNode next;

    ListNode(int x)
    {
        val = x;
    }
}



public class Main
{
    public static void main(String[] args)
    {
        String s = "bbbbb";

        Solution solution = new Solution();

        int receive = solution.lengthOfLongestSubstring(s);

        System.out.println("haha");

    }


}


class Solution {
    public int removeElement(int[] nums, int val) {

        if ( nums == null || nums.length == 0)
        {
            return 0;
        }

        int numsLength = nums.length;
        int startIndex = 0, endIndex = numsLength - 1;


        while ( startIndex <= endIndex )
        {
            while (startIndex <= endIndex)
            {
```

```
                if (nums[startIndex] == val)
                {
                    break;
                }
                startIndex++;
            }

            while (startIndex <= endIndex)
            {
                if (nums[endIndex] != val)
                {
                    break;
                }
                endIndex--;
            }

            if (startIndex < endIndex)
            {
                int tempValue = nums[startIndex];
                nums[startIndex] = nums[endIndex];
                nums[endIndex] = tempValue;

                startIndex++;
                endIndex--;
            }
        }

        return startIndex;
    }
}
```