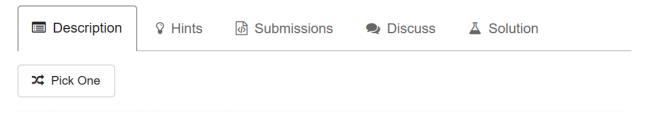
448. Find All Numbers Disappeared in an Array



Given an array of integers where $1 \le a[i] \le n$ (n = size of array), some elements appear twice and others appear once.

Find all the elements of [1, n] inclusive that do not appear in this array.

Could you do it without extra space and in O(n) runtime? You may assume the returned list does not count as extra space.

Example:

```
Input:
[4,3,2,7,8,2,3,1]

Output:
[5,6]
```

```
public class L448 {
     * 这道题目把nums(Math.abs(nums[i]) - 1)标负,第二遍iterate时若nums[i]非负
     * 就表明原array没有i+1,加到res中。
     public List<Integer> findDisappearedNumbers(int[] nums) {
         List<Integer> res = new ArrayList<Integer>();
         if(nums == null || nums.length == 0)
             return res;
         for(int i = 0; i < nums.length; i ++) {</pre>
             int index = nums[i] - 1;
             if(nums[index] > 0) {
                 nums[index] = -nums[index];
         }
         for(int i = 0; i < nums.length; i ++) {</pre>
             if(nums[i] > 0) {
                 res.add(i + 1);
         }
         return res;
     }
}
```