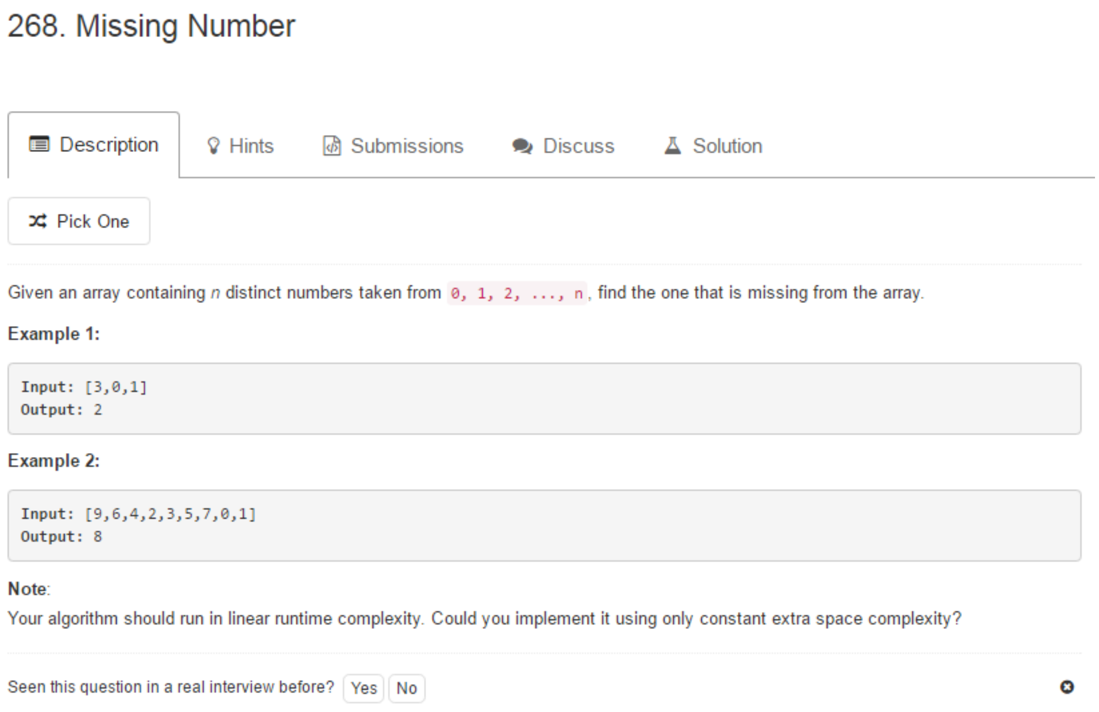


268. Missing Number

Given an array containing *n* distinct numbers taken from `0, 1, 2, ..., n`, find the one that is missing from the array.

Example 1:

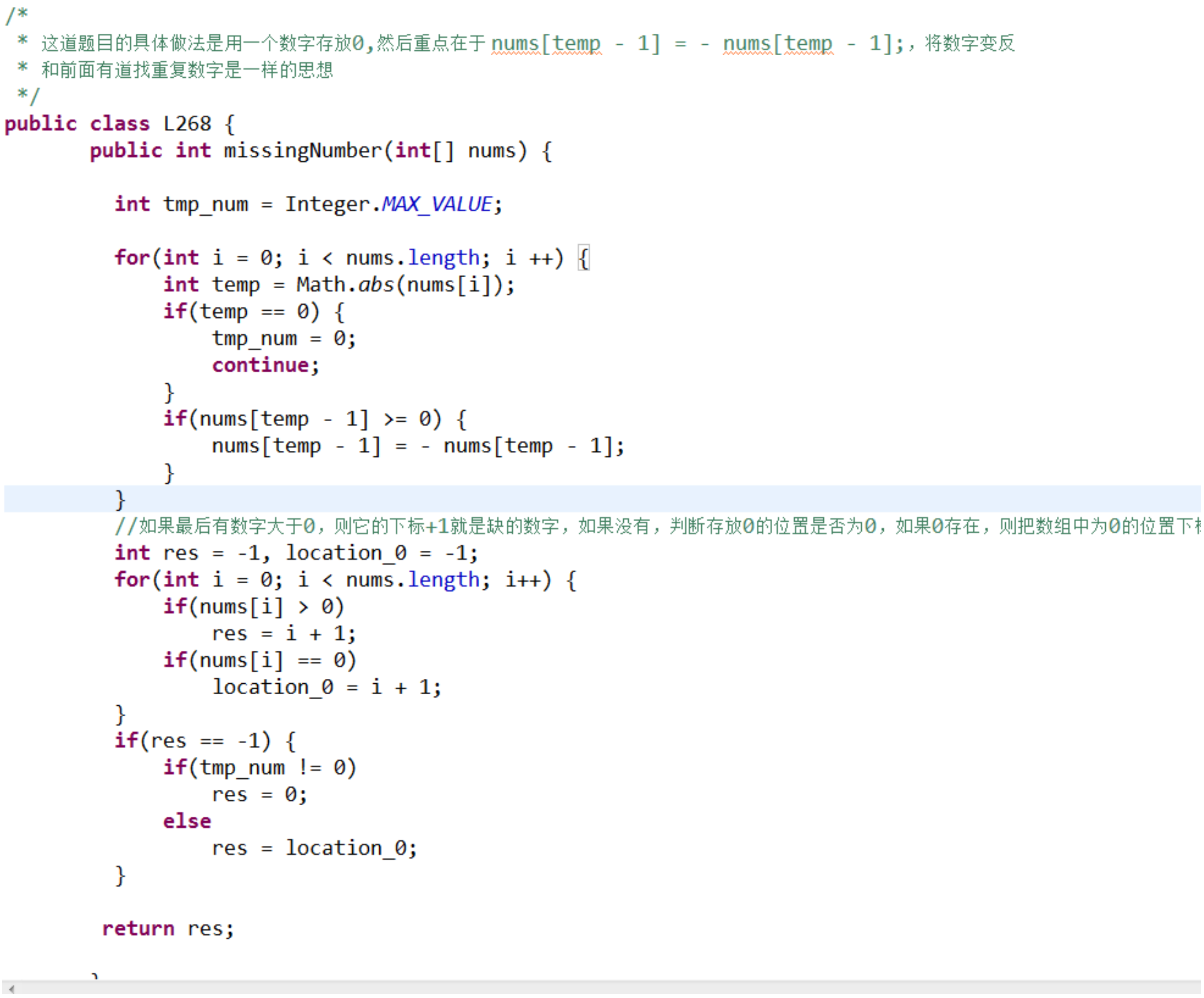
```
Input: [3,0,1]
Output: 2
```

Example 2:

```
Input: [9,6,4,2,3,5,7,0,1]
Output: 8
```

Note:
Your algorithm should run in linear runtime complexity. Could you implement it using only constant extra space complexity?

Seen this question in a real interview before? Yes No



```
/*
 * 这道题目的具体做法是用一个数字存放0,然后重点在于 nums[temp - 1] = - nums[temp - 1];，将数字变反
 * 和前面有道找重复数字是一样的思想
 */
public class L268 {
       public int missingNumber(int[] nums) {

         int tmp_num = Integer.MAX_VALUE;

         for(int i = 0; i < nums.length; i ++) {
             int temp = Math.abs(nums[i]);
             if(temp == 0) {
                 tmp_num = 0;
                 continue;
             }
             if(nums[temp - 1] >= 0) {
                 nums[temp - 1] = - nums[temp - 1];
             }
         }
         //如果最后有数字大于0，则它的下标+1就是缺的数字，如果没有，判断存放0的位置是否为0，如果0存在，则把数组中为0的位置下
         int res = -1, location_0 = -1;
         for(int i = 0; i < nums.length; i++) {
             if(nums[i] > 0)
                 res = i + 1;
             if(nums[i] == 0)
                 location_0 = i + 1;
         }
         if(res == -1) {
             if(tmp_num != 0)
                 res = 0;
             else
                 res = location_0;
         }

        return res;

       }
```