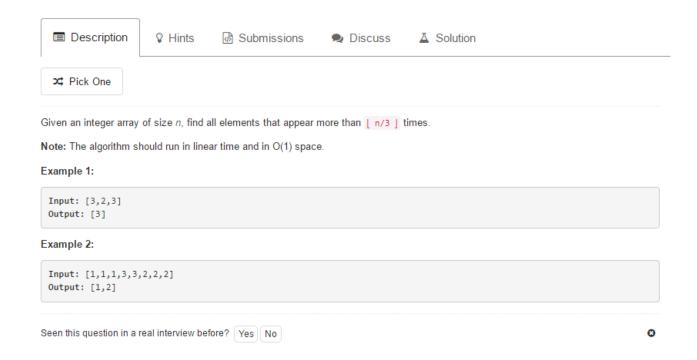
## 229. Majority Element II



## 题解:这个主要是通过摩尔投票算法

这个局部变量中定义的一个序列元素 (m) 和一个计数器i, 初始化的情况下计数器为0, 算法依次扫描序列中的元素, 当处理元素x的时候, 如果处理器为0, 那么将x赋值给m, 然后将计数器i设置为1, 如果计数器不为0, 那么将序列m和x 比较, 如果相等, 那么计数器加1, 如果不等, 那么计数器减1。处理之后, 最后存储的序列元素(m), 就是这个序列中最多的元素。

如果不确定是否存储的元素m是最多的元素,还需要进行第二遍扫描判断是否为最多的元素。

观察可知,数组中至多可能会有2个出现次数超过 | n/3 | 的众数

记变量n1, n2为候选众数; c1, c2为它们对应的出现次数

遍历数组,记当前数字为num

若num与n1或n2相同,则将其对应的出现次数加1

否则, 若c1或c2为0, 则将其置为1, 对应的候选众数置为num

否则,将c1与c2分别减1

最后,再统计一次候选众数在数组中出现的次数,若满足要求,则返回之。

```
public class L229 {
\Theta
        public List<Integer> majorityElement(int[] nums) {
)
            List<Integer> res = new ArrayList<Integer>();
)
            Integer n1 = null, n2 = null;
            int c1 = 0, c2 = 0;
            for (int i : nums) {
              if(n1 != null && i == n1.intValue()) {
              }else if (n2 != null && i == n2.intValue()) {
                  c2 ++;
              }else if (c1 == 0) {
                  c1 = 1;
                  n1 = i;
              }else if (c2 == 0) {
ţ
                  c2 = 1;
                  n2 = i;
              }else {
                  c1 --;
                  c2 --;
              }
          }
           c1 = c2 = 0;
           for(int i : nums) {
               if(i == n1.intValue()) {
                  c1 ++;
               }else if (i == n2.intValue()) {
                  c2 ++;
              }
           }
           if(c1 > nums.length / 3)
               res.add(n1);
           if(c2 > nums.length / 3)
               res.add(n2);
           return res;
        }
' }
```