画解剑指 Offer - LeetBook - 力扣（LeetCode）全球极客挚爱的技术成长平台

剑指 Offer 39. 数组中出现次数超过一半的数字 - 解决方案

题目描述

数组中有一个数字出现的次数超过数组长度的一半，请找出这个数字。

你可以假设数组是非空的，并且给定的数组总是存在多数元素。

示例 1:

```
输入: [1, 2, 3, 2, 2, 2, 5, 4, 2]
输出: 2
```

限制：

`1 <= 数组长度 <= 50000`

解题方案

思路

- 标签：摩尔投票
- 本题常见解法共有 3 种
  - 数组排序：首先将 `nums` 排序，由于该数字超过数组长度的一半，所以数组的中间元素就是答案，时间复杂度为 O(nlogn)O(nlogn)
  - 哈希计数：遍历 `nums` 数组，将数字存在 `HashMap` 中，统计数字出现次数，统计完成后再遍历一次 `HashMap` ，找到超过一半计数的数字，时间复杂度为 O(n)O(n)
  - 摩尔投票：遍历 `nums` 数组，使用 `count` 进行计数，记录当前出现的数字为 `cur` ，如果遍历到的 `num` 与 `cur` 相等，则 `count` 自增，否则自减，当其减为 0 时则将 `cur` 修改为当前遍历的 `num` ，通过增减抵消的方式，最终达到剩下的数字是结果的效果，时间复杂度为 O(n)O(n)
- 摩尔投票是最优解法，时间复杂度：O(n)O(n)，空间复杂度：O(1)O(1)

算法流程

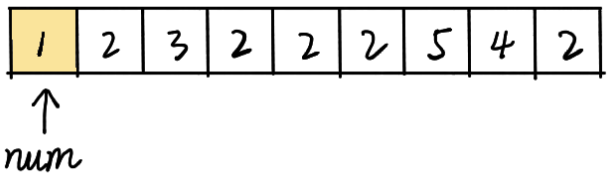
1. 初始化：预期结果 `cur = 0` 和计数器 `count = 0`
2. 遍历数组 `nums` ，遍历过程中取到的数字为 `num`
3. 当 `count` 为 0 时，表示不同的数字已经将当前的结果抵消掉了，可以换新的数字进行尝试，则 `cur = num`
4. 当 `num == cur` 时，表示遍历数字和预期结果相同，则计数器 `count++`

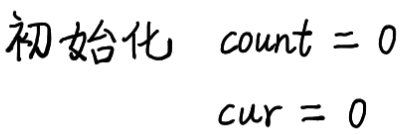
5. 当 `num != cur` 时，表示遍历数字和预期结果不同，则计数器 `count--`
6. 最终留下的数字 `cur` 就是最终的结果，出现次数超过一半的数字一定不会被抵消掉，最终得到了留存

代码

```
class Solution {
    public int majorityElement(int[] nums) {
        int cur = 0;
        int count = 0;
        for(int num : nums){
            if(count == 0) {
                cur = num;
            }
            if(num == cur) {
                count++;
            } else {
                count--;
            }
        }
        return cur;
    }
}
```

画解





1 / 11

最热

梦想家haima L3
来自北京2021-06-23

相互抵消，最终肯定会剩下超过半数的那个元素。如果这道题改成，求元素出现次数最多的 元素，那就不得行。

比如 [1,1,2,3,4,5,6]

5

木子亦大 L1
来自湖北2021-03-08

```
/*class Solution {
    public int majorityElement(int[] nums) {
        int count=0;
        int current=0;

        for(int i:nums){
            if(count==0){
                current=i;
            }
            if(i==current){
                count++;
            }
            else{
                count--;
            }
        }
        return current;
    }
}*/

/*class Solution {
    public int majorityElement(int[] nums) {
        Map<Integer,Integer> M=new HashMap<>();

        for(int i:nums){
            if(!M.containsKey(i)){
                M.put(i,1);

            }
            else{
                M.put(i, M.get(i)+1);
            }



        }
        Set<Integer> N=M.keySet();
        for(int i:N){
            if(M.get(i)>nums.length/2){
                return i;
            }
        }
        return 0;

    }
}*/
/*class Solution {
    public int majorityElement(int[] nums) {
        Arrays.sort(nums);
        return nums[nums.length/2];

    }
}*/
```

三种都实现了一下

4

Inspiring CannonctT

来自广东2022-02-27

既然出现最多的数超过一半，那么位于数组中间的数一定是那个数。
C++：

```
class Solution {
public:
    int majorityElement(vector<int>& nums) {
        sort(nums.begin(), nums.end());
        return nums[nums.size()/2];
    }
};
```

1

仙级圣纹师L1
来自北京2021-05-26

```
func majorityElement(nums []int) int {
    num,count:=0,0
    for _,n :=range nums{
        if count==0{
            num = n
        }
         if n==num{
            count++
        }else {
            count--
        }
    }
    return num
}
```

蔡徐坤L1

来自黑龙江2021-03-06

`

class Solution {
public int majorityElement(int[] nums) {
int count = 0;
Integer candidate = null;
for (int num:nums){
if (count==0){
candidate = num;
count++;
}else {
if (num==candidate){
count++;
}else {
count--;
}
}

```
    }

    return candidate;
}

}
```

`

FuyunWang L1
来自中国香港2021-02-27

`

```
    public int majorityElement(int[] nums) {
            Arrays.sort(nums);
            return nums[nums.length/2];
    }
```

`