```
/*
给定一个只包含整数的有序数组，每个元素都会出现两次，唯有一个数只会出现一次，找出这个数。

示例 1:

输入: [1,1,2,3,3,4,4,8,8]
输出: 2

示例 2:

输入: [3,3,7,7,10,11,11]
输出: 10

注意: 您的方案应该在 O(log n)时间复杂度和 O(1)空间复杂度中运行。

来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/single-element-in-a-sorted-array
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析:

- 方法一:位运算异或法,时间复杂度为$O(n)$;
- 方法二:逐二遍历查找法,时间复杂度为$O(n)$
- 方法三:二分查找法,时间复杂度为$O(\log n)$
  - 按照题意输入数组的长度一定是奇数,则一定有中间元素,另当前元素的索引为$mid, mid > 0$.
  - 若mid为数组头或数组尾巴,则直接返回$nums[mid]$;
  - 若mid是奇数
    - 若$nums[mid] == nums[mid-1]$,表明断点在右半部分,$left = mid + 1$;
    - 若$nums[mid] \neq nums[mid-1]$
      - 若$nums[mid] \neq num[mid+1]$,则已经找到了数量为1的元素;
      - 若$nums[mid] == num[mid+1]$,表明断点在左半部分,$right = mid - 1$;
  - 若k是偶数
    - 若$nums[mid] == nums[mid+1]$,则表明断点在右半部分,$left = mid + 1$
    - 若$nums[mid] \neq num[mid+1]$
      - 若$nums[mid] \neq nums[mid-1]$,则已经找到了数量为1的元素;
      - 若$nums[mid] == nums[mid-1]$,则表明果断点在左半部分,$right = mid - 1$.

方法一:C++_位运算异或法

```
class Solution
{
    public:
        int singleNonDuplicate(vector<int>& nums)
        {
            /*
                时间复杂度 O(n)
```

```
                空间复杂度 O(1)

            */
            int ret_val = 0x00;
            int i       = 0;
            for(i = 0 ; i < nums.size() ; i++)
            {
                ret_val ^= nums[i];
            }
            return ret_val;
        }
};

/*
执行结果:
通过
显示详情
执行用时 :12 ms, 在所有 cpp 提交中击败了76.04% 的用户
内存消耗 :9.5 MB, 在所有 cpp 提交中击败了9.73%的用户
*/
```

方法二:逐二遍历查找法_C++

```
class Solution
{
    public:
        int singleNonDuplicate(vector<int>& nums)
        {
            /*
                时间复杂度 O(n)
                空间复杂度 O(1)
            */
            int ret_val = 0x00              ;
            int i       = 0                 ;
            int left    = 0                 ;

            for(i=0;i<nums.size();i+=2)
            {
                if(i+1<nums.size())
                {
                    if(nums[i]!=nums[i+1])
                    {
                        ret_val = nums[i];
                        break;
                    }
                }
                else
                {
                    ret_val = nums[i];
                    break;
                }
            }
            return ret_val;
        }
};

```

```
/*
执行结果：
通过
显示详情
执行用时 :12 ms, 在所有 cpp 提交中击败了76.04% 的用户
内存消耗 :9.5 MB, 在所有 cpp 提交中击败了6.19%的用户
*/
```

方法三:二分查找法_C++

```
class Solution
{
    public:
        int singleNonDuplicate(vector<int>& nums)
        {
            int left    =   0               ;
            int right   =   nums.size()-1   ;
            int mid     =   0               ;


            while(left <= right)
            {
                mid = left + (right - left ) / 2;

                /*最右边*/
                if(   (mid == nums.size()-1)
                   || (mid == 0)
                )
                {
                    break;
                }
                else
                {
                    /*mid 是奇数*/
                    if(mid & 0x01)
                    {
                        if(nums[mid] == nums[mid-1])
                        {
                            left = mid + 1;
                        }
                        // nums[mid] != nums[mid-1]
                        else
                        {
                            if(nums[mid]!=nums[mid+1])
                            {
                                //找到了
                                break;
                            }
                            else
                            {
                                right = mid - 1;
                            }
                        }
                    }
                    /*mid是偶数*/
                    else
```

```
                    {
                        if(nums[mid] == nums[mid + 1])
                        {
                            left = mid + 1;
                        }
                        else
                        {
                            if(nums[mid] == nums[mid-1])
                            {
                                right = mid - 1;
                            }
                            else
                            {
                                // 找到了
                                break;
                            }
                        }

                    }
                }
            }

            return nums[mid];


        }
};

/*
执行结果：
通过
显示详情
执行用时 :8 ms, 在所有 cpp 提交中击败了97.08% 的用户
内存消耗 :9.6 MB, 在所有 cpp 提交中击败了5.31%的用户
*/
```

AlimyBreak
2019.10.27