给定一个无序的数组 `nums`，将它重新排列成 `nums[0] < nums[1] > nums[2] < nums[3]...` 的顺序。

示例 1:

输入：`nums = [1, 5, 1, 1, 6, 4]`
输出：一个可能的答案是 `[1, 4, 1, 5, 1, 6]`

示例 2:

输入：`nums = [1, 3, 2, 2, 3, 1]`
输出：一个可能的答案是 `[2, 3, 1, 3, 1, 2]`

说明:

你可以假设所有输入都会得到有效的结果。

进阶:

你能用 O(n) 时间复杂度和 / 或原地 O(1) 额外空间来实现吗?

方法一：对数组进行排序，然后将前半段倒序填入奇数下标，将后半段倒序填入偶数下标，其原理是如果题目有解，则对于排好序的数组，间隔超过n/2的两个元素必不相等。时间复杂度O(nlogn)，空间复杂度O(n)。

```
class Solution {
public:
    void wiggleSort(vector<int>& nums) {
        vector<int> a(nums);
        sort(a.begin(),a.end());
        int n=nums.size();
        int mid = (n-1) / 2;
        for(int i=0, j=mid, k=n-1;i<n;i+=2,j--,k--){
            if(i+1<n){
            nums[i+1]=a[k];
            }
            nums[i]=a[j];
        }
    }
};
```

一个数组大小减1除以2刚好是数组对称的位置。