

## 奇数下标都是奇数或者偶数下标都是偶数

### 【题目】

给定一个长度不小于 2 的数组 `arr`，实现一个函数调整 `arr`，要么让所有的偶数下标都是偶数，要么让所有的奇数下标都是奇数。

### 【要求】

如果 `arr` 的长度为  $N$ ，函数要求时间复杂度为  $O(N)$ ，额外空间复杂度为  $O(1)$ 。

### 【难度】

士 ★☆☆☆

### 【解答】

实现方法有很多，本书介绍一种易于实现的方法，步骤如下：

1. 设置变量 `even`，表示目前 `arr` 最左边的偶数下标，初始时 `even=0`。
2. 设置变量 `odd`，表示目前 `arr` 最左边的奇数下标，初始时 `odd=1`。
3. 不断检查 `arr` 的最后一个数，即 `arr[N-1]`。如果 `arr[N-1]` 是偶数，交换 `arr[N-1]` 和 `arr[even]`，然后令 `even=even+2`。如果 `arr[N-1]` 是奇数，交换 `arr[N-1]` 和 `arr[odd]`，然后令 `odd=odd+2`。继续重复步骤 3。
4. 如果 `even` 或者 `odd` 大于或等于  $N$ ，过程停止。

举例说明整个过程。比如 `[1,8,3,2,4,6]`，当前最后一个数记为 `end=6`，`even=0`，`odd=1`。此时 `end=6` 为偶数，所以 6 和 `arr[even=0]` 交换，数组变成 `[6,8,3,2,4,1]`，`even=even+2=2`。此时 `end=1` 为奇数，所以 1 和 `arr[odd=1]` 交换，数组变成 `[6,1,3,2,4,8]`，`odd=odd+2=3`。此时 `end=8` 为偶数，所以 8 和 `arr[even=2]` 交换，数组变成 `[6,1,8,2,4,3]`，`even=even+2=4`。此时 `end=3` 为奇数，所以 3 和 `arr[odd=3]` 交换，数组变成 `[6,1,8,3,4,2]`，`odd=odd+2=5`。此时 `end=2` 为偶数，所以 2 和 `arr[odd=4]` 交换，数组变成 `[6,1,8,3,2,4]`，`even=even+2=6`。此时 `even` 大于或等于长度 6，说明偶数下标已经都是偶数，过程停止。

再解释得直白一点，最后位置的数是偶数，就向偶数下标发送，最后位置的数是奇数，就向奇数下标发送，如果偶数下标或者奇数下标已经无法再向右移动，说明调整结束。调

整的全部过程请参看如下代码中的 `modify` 方法。

```
public void modify(int[] arr) {
    if (arr == null || arr.length < 2) {
        return;
    }
    int even = 0;
    int odd = 1;
    int end = arr.length - 1;
    while (even <= end && odd <= end) {
        if ((arr[end] & 1) == 0) {
            swap(arr, end, even);
            even += 2;
        } else {
            swap(arr, end, odd);
            odd += 2;
        }
    }
}

public void swap(int[] arr, int index1, int index2) {
    int tmp = arr[index1];
    arr[index1] = arr[index2];
    arr[index2] = tmp;
}
```

## 子数组的最大累加和问题

### 【题目】

给定一个数组 `arr`，返回子数组的最大累加和。

例如，`arr=[1,-2,3,5,-2,6,-1]`，所有的子数组中，`[3,5,-2,6]`可以累加出最大的和 12，所以返回 12。

### 【要求】

如果 `arr` 长度为  $N$ ，要求时间复杂度为  $O(N)$ ，额外空间复杂度为  $O(1)$ 。

### 【难度】

士 ★☆☆☆