```
/**
 * 这里只会描述一些注意点，如果想对原地快排有一个详细的了解，请参阅网络资源
 * 非原地快排过于简单，这里不予描述
 * 随机找出一个元素作为基准（pivot）
 * 然后遍历所有元素，按照升序规则（亦可按降序），将所有比基准小的值放在左边，将所有比基准大的值放在右边，与基准相同的值可以不予理睬
 * 最后递归处理两边的序列
 *
 * 举个简单的例子：
 * 比如原序列为：[5, 3, 9, 8, 4, (6), 44, 12, 3, 6]
 * 随机一个基准值，比如说：6，index 为 5
 * 根据规则做分配：[5, 3, 4, 3, (6), 8, 9, 44, 12, 6]
 * 快速排序最复杂的并不是二分，也不是递归，而是在[原地快速排序]时的元素的重排
 * 上面的例子中原本 pivotIndex 为 5，最终 pivotIndex 变为了 4
 * 同时也可以发现确实按照大小分列 pivot 两旁了，而且相等元素不进行处理
 *
 * 谨记：重拍算法多种多样，我们最终的目的只是需要把小于 pivot 的元素丢在左边，大的丢在右边，至于左边和右边序列的顺序不需要考虑
 * 有人说为什么 pivot 位置怎么变了，它变就变了，有什么问题吗
 * 还有人说左边不应该是 [3, 3, 4, 5, (6)]，这么理解就错了，我们才经历了一次二分，只能保证小在左，大在右，至于顺序，怎么样都无所谓
 * 有些人会在这里钻牛角尖，所以在这里提一下
 *
 * 上面已经做了二分了，下面就要递归了，通过下面的转化，你可以大致的明白这个算法：
 * 1: [5, 3, 9, 8, 4,       (6)      , 44, 12, 3, 6] 随机基准值
 * 2: [5, 3, 4, 3]          (6)      [8, 9, 44, 12, 6] 重排二分
 * 3: [5, (3), 4, 3]        (6)      [8, 9, (44), 12, 6] 对左右两边重复上边的操作，随机基准值
 * 4: (3)  [5, 4, 3]        (6)      [8, 9, 12, 6]  (44) 对左右两边重复上边的操作，重排二分
 * 5: (3)  [5, (4), 3]      (6)      [(8), 9, 12, 6]  (44)
 * 6: (3)  [3] (4) [5]      (6)      [6] (8) [9, 12]  (44)
 * 二分至不需要分的时候，合并所有片段 [3, 3, 4, 5, (6), 6, 8, 9, 12, 44]
 */

function quickSort (data) {
    let start = 0;
    let end = data.length;

    let core = function (data, start, end) {
        if (start < end) {
            let pivot = partition(data, start, end);
            core(data, start, pivot);
            core(data, pivot + 1, end);
        }
        return data;
    }

    return core(data, start, end);
}

function partition (data, start, end) {
    let pivot = Math.floor(start + Math.random() * (end - start));
    console.log(pivot);

    for (let i = 0; i < data.length; i++) {
        if (i < pivot && data[i] > data[pivot]) {
            swap(data, i, pivot);
            pivot = i;
```

```
        }

        if (i > pivot && data[i] < data[pivot]) {
            swap(data, i, pivot);
            swap(data, i, pivot + 1);
            pivot = pivot + 1;
        }
    }

    return pivot;
}

function swap (arr, l, r) {
    let temp = arr[l];
    arr[l] = arr[r];
    arr[r] = temp;
}
```