

## 2-10:

若在习题2-9中，数组  $T$  中元素不存在序关系，只能测试任意两个元素是否相等，试设计一个有效算法确定  $T$  是否有一主元素。算法的计算复杂性应为  $O(n \log n)$ 。更进一步，能找到一个线性时间算法吗？

### 题目分析：

本题与2-9的区别在于，数组  $T$  中不存在序关系。因此，没办法利用2-9的思路，因为不再能线性时间选出中位数。因此，本题需要用别的方法求解。

对于本题，必有如下结论：

- 如果  $x$  为  $T[0 : n - 1]$  中的主元素，则  $x$  必为  $T_1[0 : \frac{n-1}{2}]$  或  $T_2[\frac{n-1}{2} : n - 1]$  中的主元素，如不然，使用反证法：若  $x$  不是  $T_1$  也不是  $T_2$  的主元素，即  $|S_1(x)| \leq \frac{n}{4}$  且  $|S_2(x)| \leq \frac{n}{4}$ ，则对于整个数组  $T$ ，有  $|S(x)| \leq \frac{n}{2}$ ，即  $x$  不是  $T$  的主元素。

既然只能比较任意两个元素是否相等，采用分治法，对原数组  $T$  进行左右对半划分，依次循环，直到数组大小为1。对于该“子数组”而言，显然剩下的那个元素为其主元素。之后再两两合并数组，对两个数组中的“主元素”都进行一次遍历，判断是否为合并后数组的主元素。如果是，则将该子元素继续向上传递；否则，返回一个无穷大表示该数组不存在主元素。

### 代码实现：

在本代码中，仍然使用 `int` 数组来模拟数组  $T$ ，但只比较大小，而不利用线性数组的序关系。可将  $T$  更换为其他类型，譬如结构、类等。

代码具体实现见2-10 Code

### 输出示例：

```
1  输入样例1:
2  16
3  3 9 8 6 11 21 49 38 5 4 7 1 2 15 23 8
4  输出1: 不存在
5  输入样例2:
6  16
7  2 3 5 2 1 8 2 3 2 2 2 2 5 2 3 2
8  输出2: 存在，且主元素为2（#因为有9个2）
9  输入样例3:
10 7
11 3 1 2 3 3 7 3
12 输出3: 存在，且主元素为3（#因为有4个3）
13 输入样例4:
14 7
15 1 7 3 2 3 3 1
```

### 算法分析:

该程序显然是一个递归式, 其复杂度为:

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n)$$

由主定理,  $f(n) = O(n) = \Theta(n^{\log_2 2})$  可得:  $T(n) = O(n \log n)$

### 更进一步:

能否找到线性时间的算法呢?

用递归的方法, 算法下界应为  $T(n) = O(n \log n)$  (不会证明, 但较为显然)。

思考一下, 2-9为什么能在  $O(n)$  时间内完成? 因为元素有序, 也就是说, 可以在  $O(n)$  时间内将数组的中位数选择出来。而本题由于元素不存在序关系, 既没有办法排序, 也没有办法选择。我推断, 本题无法在线性时间内求解。