2-3 半数集问题

一、核心算法：

递归思想

二、代码实现：

```
 //半数集数量计算
int set(int n)
{
    int sum;
    if (n == 1) return 1;
    else
        for (int i = 1; i <= n / 2; i++) //不超过一半
            sum += set(i);
    return sum;
}
```

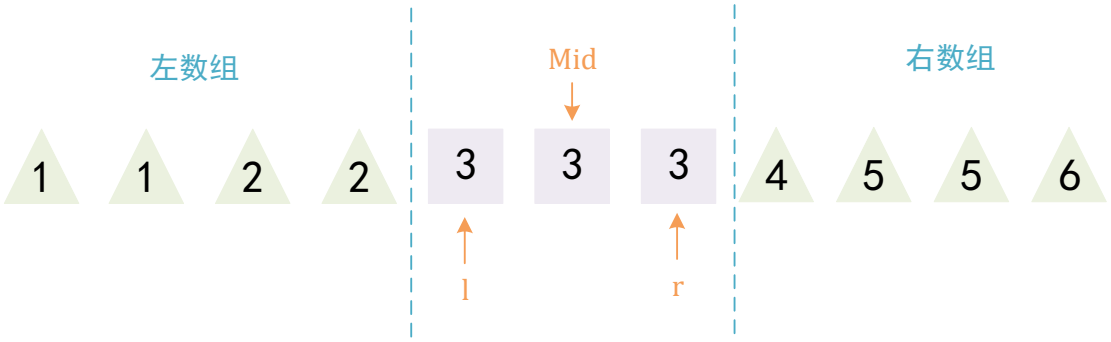
三、结果演示

2-1分治法求众数

一、核心算法：

（1）、在数组求出与中位数相同的元素的左右边界(l & r)，计算相同的元素数量Cnt

（2）、用相同的方法处理左右数组，找到最大Cnt.



二、代码实现：

```
//求出与中间数字相等的左右界限
void split(int a[], int n, int &l, int &r)
{
    int mid = n / 2;
    for (l = 0; l < n; l++){
        //求出a[mid]左边第一个与a[mid]相等的元素下标 l
        if (a[l] == a[mid]) break;
    }
    for (r = n; r > 0; r--){
        //求出a[mid]右边第一个与a[mid]相等的元素下标 r
        if (a[r] == a[mid]) break;
    }
}
//分治法找众数
void GetMaxCnt(int &mid, int &maxCnt, int a[], int n)
{
    int l, r; //左右边界
    split(a,n,l,r); //第一次分割
    int num = n / 2;
    int cnt = r - l; //中间元素的数量
    //update
    if (cnt > maxCnt){
        maxCnt = cnt;   mid = a[num];
    }
    //左边元素个数大于Cnt  搜寻左边
    if (l + 1 > maxCnt)   GetMaxCnt(mid,maxCnt,a,l+1);
    //右边元素个数大于Cnt  搜寻右边
    if (n - r > maxCnt)   GetMaxCnt(mid, maxCnt, &a[r], n - r);
}
```

三、结果演示