

# Java方向编程题答案

## 第八周

### day46

题目ID: 23271

链接: <https://www.nowcoder.com/questionTerminal/e8a1b01a2df14cb2b228b30ee6a92163>

【题目解析】:

题目描述很清楚, 注意审题即可.

【解题思路】:

方法一:

直接对数组进行排序, 中间元素即为要求元素, 笔试的时候推荐这样实现, 代码简单不容易出错.

方法二:

采用阵地攻守的思想: 第一个数字作为第一个士兵, 守阵地; count = 1; 遇到相同元素, count++; 遇到不相同元素, 即为敌人, 同归于尽, count--; 当遇到count为0的情况, 又以新的i值作为守阵地的士兵, 继续下去, 到最后还留在阵地上的士兵, 有可能是主元素. 因为主元素数目超过了整个数组的一半, 因此其他的数字是不能把主数字给减为0的. 再加一次循环, 记录这个士兵的个数看是否大于数组一般即可.

面试的时候推荐使用这种方法. 比方法一更优化.

```
// 方法一代码
import java.util.Arrays;

public class Solution {
    public int MoreThanHalfNum_Solution(int [] array) {
        Arrays.sort(array);
        int count=0;

        for(int i=0;i<array.length;i++){
            if(array[i]==array[array.length/2]){
                count++;
            }
        }
        if(count>array.length/2){
            return array[array.length/2];
        }else{
            return 0;
        }
    }
}
```

```
// 方法二代码
```

```

public class Solution {
    public int MoreThanHalfNum_Solution(int [] numbers) {
        int n = numbers.length;
        if (n == 0) return 0;

        int num = numbers[0], count = 1;
        for (int i = 1; i < n; i++) {
            if (numbers[i] == num) {
                count++;
            } else {
                count--;
            }
            if (count == 0) {
                num = numbers[i];
                count = 1;
            }
        }
        // 经过上面的操作, 已经找到该数 num 了.
        // 下面的操作是为了确认 num 确实是出现次数超过一半.
        count = 0;
        for (int i = 0; i < n; i++) {
            if (numbers[i] == num) count++;
        }
        if (count * 2 > n) return num;
        return 0;
    }
};

```

### 36843 简单错误记录

链接: <https://www.nowcoder.com/questionTerminal/2baa6aba39214d6ea91a2e03dff3fbeb>

#### 【题目解析】：

题目描述很清楚, 注意审题即可.

注意! 题目中的净文件名指: 如果一个文件路径为 c:\test\test.c 那么净文件名就是 test.c

#### 【解题思路】：

实现过程比较简单, 按照题目需求操作即可.

记录个数, 故使用Map, 不需要排序故使用HashMap, 根据题意是要循环输出, 而且提交一次会有测试用例提示, 需要按照输入顺序输出, 故使用LinkedHashMap //在输出的时候, 题目的意思是循环八个, 但问题是必须记录全部错误个数, 否则刚记录完后弹出了, 错误数就对不上了, 跟操作系统里的缺页有那么点相似。

```

import java.util.LinkedHashMap;
import java.util.Map;
import java.util.Scanner;

public class Main{
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        Map<String, Integer> map=new LinkedHashMap<String, Integer>();
    }
}

```

```
while(sc.hasNext()){
    String str=sc.next();
    int linenum=sc.nextInt();
    String[] arr=str.split("\\\\"); //根据\切割
    String s=arr[arr.length-1];
    if(s.length()>16) //截取
        s=s.substring(s.length()-16);
    String key=s+" "+linenum;
    int value=1;
    if(map.containsKey(key))
        map.put(key, map.get(key)+1);
    else {
        map.put(key, value);
    }
}
int count=0;
for(String string:map.keySet()){
    count++;
    if(count>(map.keySet().size()-8)) //输出最后八个记录
        System.out.println(string+" "+map.get(string));
}
}
```