Java方向编程题答案

第八周

day48

> **25268 左右最值最大差**
>
> 链接：https://www.nowcoder.com/questionTerminal/f5805cc389394cf69d89b29c0430ff27
>
> **【题目解析】：**
>
> 基于贪心算法的思想 这两个数中有一个肯定是数组的最大值。要使得差值最大，那么另一边的最大值应尽可能的小。 假设最大值在左边，那么对于最大值右边的数组有很多种分法，每一种分法肯定都包含数组最后一个数字即A[n-1]。 如果不取A[n-1]，取最后一个数字和最大值中间的任一数字A[i]。 **若A[i]大于A[n-1]，那还不如取最后一个数字；若最A[i] 小于A[n-1]， 那右半边的最大值肯定不是A[i]，所以无论如何右半边取最右端数字。** 假设最大值在右边，同理左半边取最左端数字。 只需用数组最大值减去数组两端较小的那个值即可。
>
> 参考 https://blog.csdn.net/zd454909951/article/details/79058397
>
> 如果把思路理解清楚, 题目的代码其实很好实现.
>
> **【解题思路】：**
>
> 1. 先找到最大值
> 2. 再拿最大值和两端的值进行相减就行了.

```
import java.util.*;

public class MaxGap {
     public int findMaxGap(int[] A, int n) {
         int max=0;
         for(int i=0;i<A.length;i++) {          //找出数组中的最大值
             if(A[i]>max)
                 max=A[i];
         }
         int ans1=max-A[0];
         int ans2=max-A[n-1];
         if(ans1>ans2)
             return ans1;
         else
             return ans2;
     }
}
```

> 25282 顺时针打印矩阵
>
> 链接: https://www.nowcoder.com/questionTerminal/97e7a475d2a84eacb60ee545597a8407

【题目解析】：

问题本身不复杂, 主要是考虑周全各种边界条件.

【解题思路】：

1. 先记录左上角和右下角坐标(这两个坐标就描述了一个矩形)
2. 然后先按照顺时针打印这个矩形边上的元素
3. 缩小矩形(也就是调整左上和右下坐标位置)
4. 再次顺时针打印. 一直缩小到这个矩形为空即可.

```
// 问题本身不复杂，将思路考虑周全即可.
import java.util.*;

public class Printer {
    public int[] clockwisePrint(int[][] mat,int n,int m) {
       int[] a =new int[m*n];
         if( mat ==null)
             return a;
         int i =0;
         int j =0;
         int k =0;
         int startX =0;
         int startY =0;
         int endX = n -1;
         int endY = m -1;
         while(startX <= endX && startY <= endY) {
             //如果只剩下一行
             if(startX == endX){
                 for(; j <= endY; j++,k++){
                     a[k] = mat[startX][j];
                 }
                 return a;
             }
             //如果只剩下一列
             if(startY == endY){
                 for(; i <= endX; i++,k++){
                     a[k] = mat[i][startY];
                 }
                 return a;
             }
             //将矩阵上边除右顶点添加到返回的数组中
             for(; j < endY; j++,k++){
                 a[k] = mat[i][j];
             }
             //将矩阵右边除边下顶点添加到返回的数组中
             for(; i < endX; i++,k++){
                 a[k] = mat[i][j];
             }
             //将矩阵下边除边左顶点添加到返回的数组中
             for(; j > startX; j--,k++){
                 a[k] = mat[i][j];
             }
```

```
            //将矩阵左边除边上顶点添加到返回的数组中
            for(; i > startY; i--,k++){
                a[k] = mat[i][j];
            }
            i++;
            j++;
            startX++;
            startY++;
            endX--;
            endY--;
        }
        return a;
    }
}
```