PULS CONTONT

汉诺塔

```
    public static  void hanoi(int n,char A,char B,char C){
if(n==1){System.out.println(A+"->"+C);

}
else {
    hanoi(n-1,A,C,B);//将A的n-1盘子由C中转到B
    System.out.println(A+"->"+C);//A最后一个盘子移至C
    hanoi(n-1,B,A,C);//将B的n-1盘子由A中转到C
}
    }
```

归并排序

时间复杂度为O(nlogn)，也是通过递归完成

```
private static void mergeSort(int[] nums, int left, int right) {
    if (left < right) {
        // 找出中间部分
        int mid = left + (right - left) / 2;
        // 对左半部分递归
        mergeSort(nums, left, mid);
        // 对右半部分递归
        mergeSort(nums, mid + 1, right);
        // 合并两个有序数组
        merge(nums, left, mid, right);
    }
}
// 合并两个数组
private static void merge(int[] nums, int left, int mid, int right) {
    int[] temp = new int[right - left + 1]; // 申请一个临时数组
    int i = left; // 左指针
    int j = mid + 1; // 右指针
    int k = 0; // 临时数组的指针
    // 作比较转移数组
    while (i <= mid && j <= right) {
        if (nums[i] <= nums[j]) {
            temp[k++] = nums[i++];
        } else {
            temp[k++] = nums[j++];
        }
    }
    // 把左边剩余的元素转移
    while (i <= mid) {
        temp[k++] = nums[i++];
```

```
    }
    // 把右边剩余的元素转移
    while (j <= right) {
        temp[k++] = nums[j++];
    }// 把新数组中的元素复制回原数组
    for (int p = 0; p < temp.length; p++) {
        nums[left + p] = temp[p];
    }
}
```

瑞士轮

写一个比赛函数，每一轮结束后调用排序后再进行比赛，最后输出名次 问题：我用的是冒泡排序，好写就是时间复杂度为n^2所以如果数据大时，容易超时，采用归并排序是一个比较好的办法，能够降低时间复杂度。

```
import java.util.Scanner;

public class Ruishilun {
    public static void main(String [] args)
        {Ruishilun ruishilun=new Ruishilun();
            int no[];/*定义数组，no为编号，s为得分，w为实力*/
            int s[];
            int w[];
            Scanner scanner=new Scanner(System.in); //创建输入对象
            String string=scanner.nextLine();//读输入的字符串
            String [] temp=string.split(" ");//删去空格并把剩下的当作临时数组存起来
            int n=Integer.parseInt(temp[0]);
            int r=Integer.parseInt(temp[1]);
            int q=Integer.parseInt(temp[2]);//存值
            no=new  int[2*n];
            s=new  int[2*n];
            w=new int[2*n];//根据给的n开一个数组
            string=scanner.nextLine();
            temp=string.split(" ");
            for(int i=0;i<temp.length;i++){
                s[i]=Integer.parseInt(temp[i]);
                no[i]=i+1;
            }//存入选手初始分数，并编号
            string=scanner.nextLine();
            temp=string.split(" ");
            for(int i=0;i<temp.length;i++){
                w[i]=Integer.parseInt(temp[i]);
            }//存入选手实力
            int ans=0;
            maopao(s,no,w);//最开始时就需要冒泡初始化数据
            while (ans<r)
            {
                ans++;
                ruishilun.sigleAfter(no,s,w);
            }//进行r轮的比赛
            System.out.println(no[q-1]);//输出第q名的编号
```

```
        }
        public void sigleAfter(int no[],int s[],int w[]){
        for (int i=0;i<s.length-1;i+=2){
            if(w[i]>w[i+1]){
                s[i]++;
            }
            else {
                s[i+1]++;
            }
        }//比赛函数，赢了加1分，输了不加分
        maopao(s,no,w);//每轮结束都进行冒泡排序
        }
        public  static void maopao(int s[],int no[],int w[])
        {
            for (int i=0;i<s.length-1;i++){
                for (int j=i+1;j<s.length-1;j++){
                    if(s[i]<s[j]){
                        int temp=s[i];//创建一个临时变量来中转
                        s[i]=s[j];
                        s[j]=temp;

                        temp=no[i];
                        no[i]=no[j];
                        no[j]=temp;

                        temp=w[i];
                        w[i]=w[j];
                        w[j]=temp;
                    } else if (s[i]==s[j]) {
                        if(no[i]>no[j]){
                            int temp=no[i];//让编号小的考前
                            no[i]=no[j];
                            no[j]=temp;

                            temp=w[i];
                            w[i]=w[j];
                            w[j]=temp;
                        }
                    }
                }
            }
        }
}
```