评分 (score)

直接对着题目模拟即可，将每个选手去掉最大最小得分之后算出平均分就行。由于评委个数一样，所以可以只算出分数和。

时间复杂度 $O(nm + n \log n)$。

数数 (count)

先将 a 排序，就有 $a_1 \le a_2 \le \cdots \le a_n$。

之后枚举较短的两根木棍 $i, j (i < j)$，合法的 k（最长的那根木棍）就要满足 $k > j, a_k < a_i + a_j$，这显然是一个区间，左端点是 $j + 1$。

当固定 i 的时候，随着 j 增大，k 能取的区间右端点也增大，于是双指针，在 j 增大时维护 k 能取的区间的右端点即可。

时间复杂度 $O(n^2)$。

行走 (walk)

题意即每次删一条边，问从 $(1,1)$ 每一步只向下或向右走一步，到 (n,n) 的最短路。

可以先求出来一条最短路，如果删除的边不在最短路上直接输出最短路即可。

现在问题变成了对某一条从 $(1,1)$ 走到 (n,n) 的边个数为 $2n-2$ 的路径，求出删除它每一条边之后的最短路。

注意到，如果将这个网格图分成若干条斜线，每条线上面的点 (x,y) 满足 $x+y$ 的值一样，那么每条边连接了相邻的两条斜线，一条路径一定是从 $(1,1)$ 开始，每次走到下一条斜线，一直走到 (n,n)。

那么对于一条被删除的边，直接枚举所有跟它在同一层之间的边来代替它即可。

时间复杂度 $O(n^2+q)$。

石子 (stone)

先考虑 $l = r$ 的情况，即给定一个点，每次向左向右合并的最小花费。

贪心地去考虑，如果在还没有合并的点中，最小值就在当前已经合并的区间的左边或者右边，那一定会先合并掉这个最小值。

但如果最小值不在它的左边或者右边呢?

比如最小值在它的左边，当最小值右边的点被合并之后，下一步一定是将这个最小值合并起来。

于是我们可以把最小值和它右边的合起来看成一个点，因为它们一定是紧接着一起合并的。

那么当目前合并的区间和合起来的若干个点合并起来时，设合起来的这若干个点有 x 个，那么耗费力气就是 $x\times$ 合并区间的 a_i 的和加上合起来的若干个点内部的贡献，即将它们一个一个合并起来，它们的 a_i 产生的贡献。

合起来的点之间也是可以类似 a_i 一样，比较贪心下我们按照之前想合并起来的先后顺序的。比较两个合起来的点 x, y 就是比较 x 先合并到最初点的花费和 y 先合并到最初点的花费。

之后能得到如果 x 内部点 a_i 的平均值较小，那么 x 先合并就更优。这时如果 y 左边是 x，初始点在 y 的右边，那么就可以将 x, y 合并起来，因为合并 y 之后一定立刻合并 x。

于是对于初始点左边的点，从左往右扫并且维护一个栈。每次加一个点到栈顶，如果栈顶的平均值大于栈顶下一个的平均值，就把栈顶和栈顶下一个合并起来。

对于初始点右边的点，从右往左这么做就行。

这样考虑初始点左边得到的栈和右边得到的栈，它们的平均值从栈顶到栈底递增。每次就选左边栈顶和右边栈顶平均值较小的那个合并起来就行。计算花费是容易的。

这样一个点就做到了 $O(n)$ 的复杂度。

由于数据随机，可以证明这个栈的大小是 $O(\log n)$ 的，所以对于一个点，它左边的栈和右边的栈都只有 $O(\log n)$ 个元素，直接对每个位置，将它左边/右边的栈记录下来，询问的时候像上面那样做就行。

时空复杂度均为 $O(n \log n)$。

其实数据不随机，即栈大小为 $O(n)$ 的情况多维护一点东西即可，但不是考察重点。