由于所选题目相对基础，在征求了几位教练的建议后，故将本次测试设置了 4 道题。

1. 精灵球 [ICPC2015 区域赛题目]

对于 n<=1000 的情况，可以使用 n^2 算法。依次枚举改变精灵球的颜色，然后计算魔法力，求最大值。

对于 n<=100000 的情况，对上述算法进行优化。每改变一个精灵球的颜色之后，只影响了相邻的精灵球序列，所以不需要全部重新计算，只计算被影响的这部分就行了。这样，没改变一个精灵球的颜色后，计算的时间差不多是 O(1)，总复杂度是 O(n)。

在进行处理过程中，可以计算每一个连续序列的长度，在处理时，分为序列长度为 1 和不为 1 的情况处理：如果序列长度为 1，则它与左边、右边的序列合成一个更长的序列；对于序列长度不为 1 的情况，只可能将该序列最左边的精灵球与左边的序列连接在一起或者该序列最右边的精灵球与右边的序列连接在一起，因为如果改变中间精灵球的颜色只能使魔法力更小。

最后注意 long long。

2. 球星卡 [某中学集训题目]

对所有的方案排序，然后每次询问进行二分查找，对大于张数的求最小值，直接查询即可。

3. 整除 [ZOJ 2836]

设 n, k 都是正整数，S={1,2,…,n}, 则 S 中能被 k 整除的正整数的个数为 [n/k].

设 Ai (i=1,2,…,n) 为有限集，则

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{1\le i_1\le n}\left|A_{i_1}\right| - \sum_{1\le i_1<i_2\le n}\left|A_{i_1}\cap A_{i_2}\right| + \cdots + (-1)^{k-1}\sum_{1\le i_1<i_2<\cdots<i_k\le n}\left|A_{i_1}\cap A_{i_2}\cap\cdots\cap A_{i_k}\right| + \cdots + (-1)^{n-1}\left|A_1\cap A_2\cap\cdots\cap A_n\right|$$

lcm(x,y)=xy/gcd(x,y)

lcm(x1,x2,⋯,xn)=lcm(lcm(x1,x2,⋯,xn-1),xn)

4. 树的问题 [HDOJ 2196]

对于一条链的情况，分别求每个结点到两个端点当中最长的距离即可。

对于另外 30%情况，分别进行 n 次 dfs 或者 bfs 即可，时间复杂度 O(n^2)。

对于 100%的数据：由于距离树中每个结点最远的点是树的直径的一个端点，所以可以先找直径，然后用直径两个端点分别 dfs 一遍求结点到直径端点的最大距离，即是所求答案，该中做法需要 3 遍 dfs。

对于 100%数据的另外一种解法：

思路：对于每个点来说它所能达到的最远距离，就是已自身为根结点向下 dfs 最大的距离，或者通过自身父节点，在加上父节点不经过自身所能达到的最远距离（这时又有两种情况，1.父节点最远距离是以它为根结点向下产生（如果最远距离经过这个子节点需要考虑父节点向下的第二远的距离），2.父节点也是通过它的父节点产生的最远距离）

比如下图，对于 2 点来说，它的最远距离可能是以它为根的子树（蓝色部分）产生的最大值，或者，通过它父节点不经过本身结点所能达到的最远点（也就是整棵树除蓝色部分，相当于红色部分）。

此时

设 f(x,0)表示以 x 为根形成的子树，x 所能达到的最远距离

f(x,1)表示，以 x 为根形成的子树，x 所能达到的第二远的距离

f(x,2)表示，x 通过父节点不经过自身结点，所能达到的最远距离

对于 f(x,0)和 f(x,1)可以通过一遍 dfs 一次得到。

对于 f(x,2)可以通过已知父节点的 f(x,0)f(x,1)f(x,2)的最优情况，来推出子节点的最优情况，还是从根向子节点遍历一遍求出答案。

每个点答案就是 Max(f(x,0),f(x,2))