1001 Link with Bracket Sequence II

区间DP，设 $f_{i,j}$ 表示 $[i,j]$ 为合法括号序列且i,j上括号相互匹配的方案数， $g_{i,j}$ 表示 $[i,j]$ 区间形成一个合法括号序列的方案数，转移为：

- 枚举 i,j 位置上填写的内容，如果形成匹配的括号对，则：$f_{i,j}=k*g_{i+1,j-1}$ ，其中 k 为使得 i,j 上括号相匹配的方案数
- 一般地，$g_{i,j}=g_{i,k}+f_{k+1,j}$

答案取 $g_{1,n}$ 即可。

复杂度 $O(n^3)$。

1002 Link with Running

这是一道图论大杂烩题。

整体思路：在最短路图上跑最长路。

- 先用dijkstra在第一个权值上跑出最短路图来，并求出第一个答案（最短路图的边权为第二个权值）。
- 注意到第一个权上可能为0，因此它并不一定是个DAG，需要用trajan将强连通分量缩起来。
- 获得了DAG之后，只需要在DAG上求最长路即可得到第二个答案。

1003 Magic

差分约束

我们令$a[i]$为前i个魔法塔中加入的魔法原料之和。由于第i个魔法塔的魔法值只与有效半径k内的魔法原料数量有关，则魔法值需求p_i可以表示为
$a[min(n,i+k-1)]-a[max(0,i-k)]>=p_i$
对于魔法原料添加的限制L_j,R_j,B_j可以表示为
$a[R_j]-a[L_j-1]<=B_j$
由于每个魔法塔中的原料数量非负，需要满足
$a[i]-a[i-1]>=0$
将上述三组不等式使用差分约束求解即可。

1004 Link with Equilateral Triangle

输出一堆No即可。

证明：

对于一个合法的解，应当满足不存在同时包含0,1,2的三角形，下面我们证明这样的三角形一定存在。

左下角必然是1，右下角必然是0，底边不能含有2，则底边上必然有奇数条1-0的边，这些边都属于一个小三角形。考虑其他的0-1边，由于不在两个斜边上，其他的0-1边必然属于两个三角形。因此"每个三角形内0-1边的数量"的和必然为奇数。

但是，假设不存在0-1-2的三角形，则所有三角形都必然包含0条或2条的0-1边，产生了矛盾。

因此一定存在0-1-2的三角形。

1005 Link with Level Editor II

考虑用双指针求解，用矩阵的方式维护1~m的路径数量。

发现矩阵并非全部可逆，难以进行删除操作，使用对顶栈的技巧规避掉删除操作即可。

复杂度 $O(nm^3)$ 。

1006 BIT Subway

签到题，阅读理解题。

真实的情况很容易算，DLee的价格实际上是一个关于总原票价x的分段函数：

$$ans = \begin{cases} x & 0 \le x < 100 \\ (x-100)*0.8+100 & 100 \le x < 225 \\ (x-225)*0.5+200 & 225 \le x \end{cases}$$

1007 Climb Stairs

比较签的题。

由于题目要求必须把所有怪物打完，所以跳过的怪物还必须得走回去打败才行。我们要从当前位置x，找到一个右端点r，使得可以按照$r, r-1, r-2, \ldots, x+1$的顺序打败这一段的怪物。不难看出r更小的时候不会更劣。直接暴力维护当前想要先到达的右端点（满足$x+k \le r$），用线段树维护一下$[l,r]$的能力值一血量，一旦扩展到r位置，就将$[l,r]$区间加a_r。找到最小的满足要求的r就可以继续更新位置，更新位置后，相当于实际上完成了击杀$r, r-1, \ldots, l$的过程，所以将后面的区间加上相应的值。由于每个右端点只会被计算一次，所以时间复杂度是$O(n \log n)$的。

1008 Fight and upgrade

首先考虑只有一个士兵的情况：我们把士兵的每一种攻击方式映射在二维平面上(x,y)（物理血量作为x，魔法血量作为y），求一个凸包，则一个士兵通过攻击方式的'弱归一化线性组合'可以打出的所有伤害映射在二维平面的点都应该落在这个凸包内，且可以取遍凸包内的所有点。

接下来考虑多个士兵，假设第i个士兵打出的伤害为(x_i, y_i)，记作$\overrightarrow{a_i}$；则n个士兵打出的伤害为$(\sum_{i=1}^{n} x_i, \sum_{i=1}^{n} y_i) = \sum_{i=1}^{n} \overrightarrow{a_i}$。故问题转换为，是否能在$n$个士兵所对应的凸包中各取一个点，使得其与原点连线所成的向量的和等于怪兽血量。这等价于判断n个士兵凸包的闵可夫斯基和是否包含怪兽血量这个点。

考虑对某一个士兵添加一个攻击方式，用动态凸包维护即可，这里只需要维护添加点这个操作。维护出凸包上边的新增，消失情况，这个下面会用到。

考虑动态维护闵可夫斯基和，这里的凸包全都经过原点，如果只是求一次闵可夫斯基和，我们只要把所有凸包的边按照极角排序，从原点开始依次连接即可。考虑动态维护这个过程：我们把极角作为关键字，把这条边在x,y上延申距离作为信息插入平衡树。对于边的维护，通过动态凸包求出哪些边要删除/要新增哪些边，直接在平衡树上维护即可；对于查询一个点是否落在闵可夫斯基和中，我们可以在平衡树上通过维护前缀和二分查找到x所对应的那一条边，判断边与这个点的关系即可。

1009 Fall with Full Star

出题人的（想复杂了的）dp做法

首先注意到d为正数与负数的表现不同，容易发现先一直选正数，随后一直选负数的方案必定比混合选择更优，因此我们将正负数分开来讨论。

如果我们当前所有的游戏分数都是正数，那么就有一个简单结论，如果我们不打算满足一个游戏的需求，那么就可以将其直接放到整局游戏的最开头，一开始直接获取它增加的分数。而剩下的我们想要获取星星的游戏则必然是按照需求从小到大依次取得。如果我们把所有的游戏都按照需求排序，那么就相当于从整个游戏集合中选取出若干个吃分的游戏放到最开头，然后保证剩下的所有游戏能按顺序得分。

我们考虑一个朴素的dp：$f[i][j]$表示当前我们已经考虑了i之后的所有游戏，其中j局游戏被我们放在最前面能够额外吃到的分的最大值。转移就有两种，第一种是在当前额外吃到的分加上i前面能够吃到的总分超过了当前的需求，就能够再获得一颗星星，转移为$f[i][j]=\max(f[i][j],f[i+1][j])$；第二种是无论需求满不满足，我们都可以选择将当前这局游戏的分数放到最开始吃，转移为$f[i][j+1]=\max(f[i][j+1],f[i+1][j]+d_i)$。因此我们获得了一个$O(n^2)$的做法。

下面我们考虑如何对这个dp进行优化，我们先让所有的限制都减去它之前的分数和得到s'，这样我们只需要考虑$f[i][j]$与当前限制s'_i的大小关系来决定转移的状况。此时，一些写了n^2暴力的选手已经从dp值中发现了一个规律，如果将第二维下标看成x值，dp值视为y值，那么每一层的dp值在平面上总能构成一个单调递增的上凸壳。在以下证明中，我们将使用(x_j,y_j)来描述一个dp值(j,f_j)。

我们可以将构建这个凸壳的过程分为两个阶段，第一阶段中只有一个点存在合法的dp值，当$s'_i>y_j$时，仅有唯一的转移$(x'_j,y'_j)=(x_j+1,y_j+d_i)$，因此凸壳仍然保持在一个点的状态。另一种情况下，凸壳新增了一个点$(x'_{j+1},y'_{j+1})=(x_j+1,y_j+d_i)$，而$(x_j,y_j)$仍在凸壳中，此时我们注意到当前满足性质

1. $s'_i\le y_1$
2. $\forall y_{j+1}-y_j\in d$

下面我们将证明整个第二阶段中，以上两个性质都始终满足。

首先考虑性质2，如果不考虑s'_i的限制，全程使用两种转移，那么$y'_j=\max(y_{j-1}+d_i,y'_j)$，考虑到整个dp值当前形成了一个单调增的上凸壳，$y_j-y_{j-1}$必定为当前取得的某一个$d$值，且必定从大到小排列。若$d_i<y_j-y_{j-1}$，则$d_i+y_{j-1}<y_j$，则转移后相当于保持原状。在第一个满足$d_i>y_j-y_{j-1}$的点处，转移后的dp值变为了$(x_{j-1}+1,y_{j-1}+d_i)$，则新的$y'_j-y'_{j-1}=d_i$，而其后的所有dp值都遵循该转移，相邻的差值也变为了$(y'_{j-1}+d_i)-(y'_{j-2}+d_i)=y'_{j-1}-y'_{j-2}$，相当于我们在凸壳中合法的位置上插入了一段$(1,d_i)$的向量。因此性质2在经过不加限制的转移后仍然成立。

接下来我们考虑添加了限制的情况，考虑到我们本来所有的游戏是对s排过序的，因此我们有$s_i\le s_{i+1}$，而$s'_i=s_i-\sum_{j<i}d_j$，因此我们有：
$s'_i+\sum_{j<i}d_j\le s_{i+1}+\sum_{j\le i}d_j$
$s'_i-s'_{i+1}\le d_i$

随后我们考虑s'_i和y的大小关系。

若$s'_i<y_1$，则整个dp转移都没有限制，因此性质2仍然成立，而性质1也显然成立；

若$y_1\le s'_i<y_2$，则仅有1号点的转移受到限制，该点只能由前一个点的dp值转移过来，而不存在前一个dp值，因此(x_1,y_1)点直接由凸壳中被删除，而其后的所有点均遵循不加限制的转移，且dp值显然不会比原先更小，故$y'_1\ge y_2>s'_i$，两个性质仍然满足。

若$y_2\le s_i$，则我们考虑当前满足性质1，因此$s'_{i+1}\le y_1$，则有$s'_i-s'_{i+1}>y_2-y_1=d_{max}$，且我们已知$d_i\ge s'_i-s'_{i+1}$，则可得$d_j>d_{max}$，故除了1号点被删除外，其他所有转移均为$(x'_j,y'_j)=(x_j+1,y_j+d_i)$，同样满足性质2；又因为$s'_{i+1}+d_i\ge s_i$，而$y_1\ge s'_{i+1}$，则可得$y'_1=y_1+d_i\ge s_i$，满足性质2。

因此，在整个第二阶段转移中，无论限制如何，两个性质均被满足。容易发现，这两个性质限制下的转移让dp值的上凸壳性质在整个过程中也始终满足。这种优秀的性质让我们得以通过一些较为简单的数据结构来维护整个dp的转移。

事实上，考虑性质2，我们就可以简单地使用一个multiset维护dp值的差分，上述的所有转移均可以通过set的插入删除等操作实现，而最终的答案即为游戏总数减去被迫放弃的最小数量，即我们dp得到的合法凸壳的最小的x坐标。

最后我们发现，负数的情况实际上经过一些处理之后就是正数的情况倒过来做，将两部分答案合起来即可。

验题人sleep__的贪心做法

从后往前 如果当前已经提前的数和前缀和的和小于s，那就贪心地把后面最大的数提前。如果后面存在某个无法达到的数，那它必然会在算到它时就已经被提前了，所以现在要是还不够就说明至少要再提前一个。可以发现，提前的数的数量和总和一直是dp里合法的j的最小值对应的dp值。

其他

感谢验题人meyi和sleep__在验题过程中提供的诸多贪心思路，让数据强度得到了有效的提升，并最终得出了一个远比正解优秀的贪心做法。

1010 Fall with Intersection

题解

首先是最简单的情况，如果当前$n = 2m$，则游戏实际上已经结束，我们只需要数交点个数就能确定谁胜谁负。

对于双方能够操作的情况，我们要注意到，一个人总能够在“直线参数为整数”的限制条件下，画出过某个交点或者与某些直线相交或平行的直线，也就是实际上操作空间非常大，而且最后一个操作的人将占据很大的优势。

考虑本题的性质，由于Bob总是最后一个画线的人，因此如果他能够通过画线来控制交点的奇偶性，则他必胜。如果当前只剩下Bob还能再画一条线，那么Bob有两种情况能够控制奇偶性：

- 斜率为某个值的直线有奇数条
- 存在一个交点被奇数条直线经过

假如某个斜率只有奇数条直线，那么Bob可以选择与所有与直线相交或者与这组直线平行，这样就得到了奇偶性不同的两组交点。而若有一个交点被奇数条直线经过，那么Bob选择经过这个交点或者不经过就能改变有效的交点数奇偶性。而若这些都不满足，Bob无论如何都只能产生偶数个交点，我们只需要任意加一条线然后数交点即可，因此我们得到了$n = 2m - 1$时的答案情况。

接下来我们考虑$n = 2m - 2$的情况，此时Alice能够先操作一次，那么Alice想要获胜，就必须同时消除奇数条直线的斜率组与所有的奇数交点。当奇数组超过一个或者不存在时，Alice必然无法阻止Bob，当奇数组恰好为一个时，Alice所画的直线斜率必然为该组对应斜率，同时还需要经过所有的奇数点。我们实际上只需要先检查对应斜率过其中一个奇数点的直线，参数是否满足整数要求，是否出现重复直线，然后直接将其添加并调用$n = 2m - 1$时的判断即可。

随后我们考虑$n = 2m - 3$的情况，容易发现，假如在这种状态下存在一个偶数条直线经过的交点，Bob只需画一条斜率没出现过并且过该交点的直线，就为Alice同时创造出了一个奇数直线组和一个奇数交点，而Alice想要同时消去这两者，画出的唯一一条直线恰好与Bob刚画下的那一条重合，也就是说Alice没有任何操作空间，则Bob必胜。

而如果没有任何交点，也就是所有直线完全平行，那么考虑当前直线条数，偶数条的情况下，Alice接下来只需要画与Bob平行的直线，就能够封死Bob的所有操作，Alice胜利，否则显然Bob最后一次操作时直线有奇数条，Bob胜利。

当$n = 2m - 4$时，Alice的所有操作都为了导向唯一胜利的情况，即所有直线平行，简单判断即可。

当$n = 2m - 5$时，Bob可以在这一步任意发挥，创造交点，从而让自己倒数第二步必胜。

因此，当$n \leq 2m - 5$时，无论如何操作，Bob都已经抵达了胜利的真实。综合以上所有讨论，即可通过这题。

其他

- 这道题来源于一场其他比赛的签到题，但出题人自己脑补了一个“重复的交点不算”的条件，并且卡了很久。
- 题目中绝大多数数据都是出题人使用GeoGebra手造的。
- std为了避免被卡精度，使用了分数表示交点，并且全程没有出现过任何浮点数；而为了避免卡别人精度，所有的大数据都是在一定规则下随机得到的，数据强度主要来自于手造和hack。
- 感谢验题人meyi提供的对于$n = 2m - 3$和$n = 2m - 4$的hack数据。

1011 Link is as bear

是个思维+结论题，可以证明，从这n个数里任取一些数异或起来的方案，都是可以构造出对应的操作来做到的。
所以，**问题完全等价于**给n个数，从中选一些数，使得这些数的异或和最大。
这是**线性基**的板题，抄一个板子即可。

下面给出证明：

1、如果序列里有连续的两个0，那么一定都可以构造出操作方案。例如$0, 0, a_1, a_2$中，比如我们要保留a_1不保留a_2，就可以先两次操作变成$a_1, 0, 0, a_2$，再两次操作变成$a_1, 0, 0, 0$，因为上述操作总可以保证操作完后还有两个0，所以可以以此类推一直往下处理。如果两个0两边都有数字当然也没问题，我们先处理完一边儿的再回过头来处理另一边。

2、如果用1来表示一个需要保留的数字，0来表示一个想要删去的数字，则出现110或011的形状时，可以通过操作删掉想删的那个数并制造出两个0。以110为例：
$a1, a2, a3 \to a_1 \oplus a_2, a_1 \oplus a_2, a_3 \to a_1 \oplus a_2, 0, 0$
就可以做到只删掉a_3而产生两个0。

3、由上面2和1可以看出，只要出现110、011就可以完全解决问题，也就是出现连续两个1就完全可以构造出任意方案，而有连续的两个0也可以完全解决问题。综上，有连续两个0和连续两个1的序列，都可以造出来任意的异或方案。

4、所以现在唯一处理不了的就是既没有连续0也没有连续1的序列，也就是形如01010101或1010101这样的情况，这时就要用到两个数相同的条件，该条件实际上是保证了不会出现这种情况。假设相同的两个数都是x，如果最后方案里没有异或上X，则两个X的位置可以都当作0或都当作1；如果最后的方案里有异或上X，则两个X可以一个填0一个填1也可以一个填1一个填0。而上述两种情况下，X都有两种填法，且两种填法之一一定保证存在连续0或连续1，因此也可以构造出。

花絮：

标题重排之后，去掉一个多余的k，就是linear basis；
因为本场各种原因删掉了原有的两道题，本题是比赛前22小时才决定加到比赛里的（