A. 雷神之路

题意

一个一维序列，长度为n，有m个地雷，每个地雷坐标已知，从0出发，每次只能走1,2,或者3步。问有多少种方法走到n点。$1 <= n <= 1e18, m <= 500$

分析

on算法很显然，但是也会超时，就得优化，得出的式子是$a_n = a_{n-1} + a_{n-2} + a_{n-3}$;显然是一个矩阵快速幂的形式，

$$\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{matrix}$$

称为A矩阵；
这个是从$a_{n-1}, a_{n-2}, a_{n-3}$ 向 a_n, a_{n-1}, a_{n-2}的正常转移，但是还有一些有地雷的怎么办，可以发现，有地雷的情况下，a_n是为0的，因此把上述矩阵第一行全部改为0即可，把这种矩阵称为B矩阵。接着我们发现，这种矩阵只会出现500次，也就是说，本是1e18个矩阵中只有500个b类矩阵，其他全部都是A矩阵，我们只需要算出间隔中的A矩阵快速幂，就可以了，复杂度完全可以承受。

思考

注意int，ll的转换，对于这种题目，有些地方换为ll时间会非常慢，但是有些地方又必须转为int，注意具体的数据类型。

B. Snowdrop修长廊

题意

我都不想说这个题目意思，太恶心了。~~制毒的舟老板，强行凑斜率DP~~
存在n个点，覆盖区间$[i, j]$的代价是$W + (x_j - x_i)^2, (i \le j)$，$W$是固有花费，问覆盖这些点的最小花费。
$1 \le n \le 2e5, 1 \le W \le 1e9, 1 \le x_i \le 1e9$

Attention：这些点不是连续覆盖的！！！自己和自己重叠着玩也算重叠了！！！修的不是长廊，是一堆线段！！！

分析

$dp_i = min(dp_j + W + (x_i - x_{j+1})^2), j < i.$
凡是形如$dp_i = min/max(dp_j + cost(i, j))$的递推式，一般都可以（我刷的也不多，只能说我见过的）用斜率dp优化，优化程度大概是降一维。
当然，并不会在这里讲斜率DP原理，因为要画图解释，lxc聚聚的博客有详细的解释，如果仍不理解的可以@我，我会解答。

然后我们通过得出的式子就可以在维护坐标为$(2x_{i+1}, dp_i + x_{i+1}^2)$的下凸壳，然后就可以愉快解题了。

思考

1.如果会了斜率DP就会发现（至少目前我这么认为），很多都是套路题，关键是推导式子以及建模。

C: TaoSama与煎饼

题意

有n个工作台，m个道具，道具可以前进1，2，3，4格，每个格子有对应的分数A_i，问怎么移动可以得到最高分。
$n \leq 350, m \leq 1000 \leq A_i \leq 100, 1 \leq B_i \leq 4;$
每种道具不超过40个，保证 $\sum_1^m B_i = n - 1.$

分析

会发现道具移动距离和为$n-1$,因此使用了多少个道具到达哪个位置都是确定的，因此想到记忆化，就是$xjbs$。先预处理一下$dp[i][j][k][l]$情况下到达的距离，就直接记忆化搜索即可。

思考

不知道为什么跑了2s……暴力跑的还比我快……不解……

D: 任务

题意

有两个机器处理任务，每个任务既可以在A机器，也可以在B机器运行，但是只可以在一个机器上运行，并且每个任务在每个机器运行的时间不同，任务i运行的时候，$j(j<i)$都必须运行完或正在运行，问完成所有任务需要的时间。
$1 \leq n \leq 2000, 1 \leq tA, tB \leq 3000$

分析

其实这道题和我出的一道题很相像的，就是维护两个机器处理时间的差值，但是这里有2000个任务，每个任务时间高达3000，如果单纯维护差值，最后复杂度是$O(2000*6000)$,爆炸。然后注意题目描述中加粗部分。当A机器处理时间大于B机器处理时间时，再在A机器上添加任务时，其实此时比B机器高的时间时A_i,并不是A_i+t,因为时间多出的部分时，$i-1$任务还没有运行，那i任务就不可以运行，因此A机器比B机器多出的时间是A_i,如果超过高度差超过3000,那么这种情况一定会有更优解覆盖它，具体要画图，要解释可以@我，当面画图解释。这样就证明了高度差一定在3000内，因此复杂度变为$O(2000*3000)$.

思考

注意题目给的条件，有时候读题不仔细会导致理解不仔细，最后导致对状态的建立有问题。

E: Goozy的积木

题意

用n个积木搭出双子塔，问能够得到的最高双子塔的高度，如果不能，输出-1。
$1 \leq n \leq 50, 1 \leq h_i \leq 500000,$
题目保证
$\sum_1^n h_i \leq 500000$

分析

和D题很类似，但是更好想，这个题目只要维护高度差即可。复杂度$O(50*500000)$,复杂度有点吓人，导致不敢写= =，其实只有一个大数据，而且跑的不算慢，可以卡内存，滚动数组才好一点。

思考

无

F: 先锋看烟花

题意

一个一维街道一共有n个房子，每个房子的距离为1,今晚会放m个烟花，每个烟花的地点a,时间t，观赏值b都已经给出，先锋观赏一个烟花的幸福度是$b_i - |a_i - cur|$,cur表示先锋当前所在位置，先锋每秒移动最远距离为d，问先锋今晚看烟花能获得的最大复杂度。

$1 \le n \le 150000, 1 \le m \le 300, 1 \le d \le n$
$1 \le a_i \le n, 1 \le b_i \le 1e9, 1 \le t_i \le 1e9$
输入保证$t_i \le t_{i+1}$

分析

可以推出式子
$dp[now][pos] = max(dp[pre][j] + b_i - |a_i - pos|),$
$|pos - j| \le d * (t_i - t_{i-1})$
这个式子强行写的话是$O(n^2m)$复杂度，爆炸。仔细观察这个式子，发现
$dp[now][pos] = max(dp[pre][j] + b_i - a_i + pos),$
$pos \le a_i;$
$dp[now][pos] = max(dp[pre][j] + b_i + a_i - pos),$
$pos \ge a_i;$
即是这两个等式中的最大值。
和斜率优化有所不同，形如：
$dp[i] = dp[j] + f[i], j \le i$的形式可以通过单调队列来优化，优化程度大约是一维，也就是说通过单调队列优化之后，复杂度变为了$O(nm)$,~~具体单调队列如何优化在这里我也不想讲，不会的@我~~。简单的讲一讲吧，队列中存放的元素有两个关键字，dp_j以及id，循环到i的时候，得到前面范围内的最大值，分两步：

- 把队首凡是id小于i要求的全都pop出来；
- 用队首来更新当前i之后，将dp_i更新到队列中，将队尾中凡是dp_j比dp_i要差的值pop出来，然后将dp_i填入队尾。

思考

都是套路。

G: Simple dp

题意

题意很迷，每个节点，如果有子树，那么它至少有2个子树，现在给出每个节点的子节点以及自己的个数，问这样的树是否存在。

$1 \le n \le 24, 1 \le x \le n$。

分析

这题我不会……我用了2个贪心搜索搜过去了……

1. 把点数多的优先放到子节点中，然后dfs下去；
2. 把点数多的优先放到当前节点，然后bfs，验证每个子树是否存在。

2种方式只要存在一个有解，就有解，否则无解。

思考

1.我也不知道怎么过的……水过去了……
2.想一想，状压弄一弄也是可以接受的。

H: 又见背包

题意

就是一个裸地多重背包，n个物品，每个物品价值a_i,有m_i个，背包容量为k。问是否能达到价值k
$1 \le n \le 100, 1 \le k \le 100000$
$1 \le q_i \le 100000$
$1 \le m_i \le 1e9$

分析

二进制优化被挂了，因为多了一个log级别。于是各种奇奇怪怪的优化都出来了。
因为这是一个$bool$类型的题目，如果使用int做法，是十分浪费的，于是$dp[i][j]$表示前i个物品表示价值j，物品i剩下的最多的数量。
于是更新就是：

- $dp[i][j]$存在，那么$dp[i+1][j] = m_{i+1}$;
- $j < a_i$ 或是 $dp[i+1][j-a_i] == -1$,那么$dp[i+1][j] = -1$;
- 其他情况便是$dp[i+1][j] = dp[i+1][j-a_i] - 1;$

最后检验$dp[n][k]$是否为正数即可。复杂度$O(nk)$

思考

1. 然而这道题让我见识了人民群众的力量，比如int浪费，先锋就开了64位的bool，把每一位当做一个$bool$，就是分块状压的意思，然后复杂度$O(nklogm/64)$。

I: Mingo's Game

题意

CF原题，一共有n个关卡，可以将关卡分为k个块，重复下列操作：

- 如果所有的关卡都被通关，那么这个游戏立即结束，否则系统会找到第一个包含还没有通过的关卡的那组，设X是这组的编号。
- 设在X组里，已经通关的关卡编号为i,i+1，i+2,...,j。那么j+1为最早的一个没有通关的关卡。每一个关卡都有一个权重ti。此时系统会随机以$P(k) = tk/\sum_i^{j+1} t_i, i \le k \le j+1$的概率进入关卡k。
- 每次通关时间都为一小时。
  求通关期望时间。
  $1 \le n \le 500000, 1 \le k \le 50$
  $1 \le t_i \le 100000$

分析

这题主要是解出公式……
首先，肯定要算出每个关卡消耗的期望时间。通过无穷级数可以证明出来是$1/p, p$是进入新关卡的概率。
然后就是推公式：

- sum[i]表示前i个关卡的t和;
- rev[i]表示前i个关卡的1/t和;
- con[i]表示前i个关卡的期望时间。

从i到j的一个分组消耗时间是$a_i/a_i + a_{i+1}/(a_i + a_{i+1}) + ... + a_j/(a_i + a_{i+1} + ... + a_j)$
就是$con[j] - con[i-1] - sum[i-1] * (rev[j] - rev[i-1])$;
然后这个满足之前提到的斜率优化式子。维护坐标$(sum_{i-1}, con_{i-1} + sum_{i-1} * rev_{i-1})$的一个下凸壳。复杂度$O(nk)$,要写滚动数组，很容易MLE。

思考

1. 推出了式子都成了模板题。

J: 奶牛还钱

题意

三个奶牛，互相欠钱，货币有1,5,10,20,50,100共六个种类，问最少交换几次就可以完成还债。

$-1000 \leq x_1, x_2, x_3 \leq 1000$
$k_{10} + k_5 + k_1 \leq 30$,
$b_{10} + b_5 + b_1 \leq 30$,
$d_{10} + d_5 + d_1 \leq 30$
并且保证3个牛共有的钱不超过1000

分析

一开始以为是每个牛有的钱不超过1000，然后怎么算复杂度都瞎了，后来发现是总和不超过，就直接瞎搞了。
首先预处理出每个钱种类的总和num，然后算出原来的钱sum,这样可以得到还债之后应有的钱now,下面开始xjb搞的辛酸历程。

首先要知道的是，计算交换次数的方法是：算出每个牛的钱币，和之间他有的各种钱币做差，计算出差的绝对值的和，然后除以2，就是了。

1. 首先用所有的钱币把第一个牛还债之后得到的钱当做背包塞满，然后一旦塞满，就去塞第二头牛，过程中只有第一头牛，或者第二头牛被塞满了之后，才可以更新答案，然后发现，这叫 ~~记忆化DP~~ 搜索；
2. 接着发现，凡是大于当前 牛原来拥有钱的数量，都是$+1$的，凡是小于的，都是-1的，写了一半发现这个更新只对大于的情况有用，GG；
3. 想了一下，大不了直接把塞两头牛的所有情况列举出来，然后暴力匹配(当然还是可以更优化一点的，但是我过了，我就没继续优化了，而且这个方法应该不是正解= =),开两个$vector$，然后逐个匹配，是否满足条件，因为知道了2头牛拥有的钱币种类，自然就知道了第三头牛的钱币情况，但是会超时……
4. 超时可以通过标记状态是否搜过来降低……但是开不下！！开不下！！他保证不超过30，但是对于我的做法，我是把3头牛的钱币加起来，那就是90，最后开出来的数组就是$vis[11][21][51][91][91][91]$，炸了，如果开小了，直接都不知道跑到哪里去了……就跪了……
5. 这时候我们会发现，交换之后的$now[0], now[1], now[2]$，最小的2个值一定是
$p[0] + p[1] + p[2] \leq 3 * p[2] \leq 1000$,也就是$p[0], p[1], p[2] \leq 333$,然后数组就可以开的小一些，
$leap[4][7][16][31][91][91]$完美水过……

思考

都是模拟，都是套路，只要胆子大，就要卡内存，反正不要钱。

K: 奶牛小镇

题意

我真的放弃了……

分析

思考

L: 来签个到吧

题意

有n个球，每个球上都有数字t,开始进行加球操作，取出两个球x,y,如果$|x-y|$不存在，那么就把$|x-y|$加入袋子中，一直到不能加为止。
接着摸球，直到把所有球都至少摸了一遍为止。
问两种操作和的期望向下取整是多少。
$2 \leq n \leq 60000, 0 \leq t < 100000$

分析

首先要算出有多少个球，然后惊奇的发现，这个就是辗转相除法，然后求出所有球的gcd，用最大值除以gcd，那么这些球便是最后得到的球的总数，至于摸球，就是一个无穷级数求概率而已，可以得到期望是$1/p$，p是摸到新球的概率。也就是第二种操作是$n/n+n/(n-1)+.....+n/1$。

~~trick就是0的情况，无论如何都会得到0这个球，但是如果原来已经出现了0，就要额外减1，我的做法鲁棒性不行，因此需要特判一下。~~我承认，是我傻逼了= =，直接减就行，没有trick……

思考

1. 强行变成了DP题，如果非要说推导公式是状态转移的话……