solution

——A Stupid Solution by a Stupid Oier l1ll5

[得分情况]

总：最高分：xxx 300 分 平均分：300 分

T1：最高分：xxx 100 分 平均分：100 分

T2：最高分：xxx 100 分 平均分：100 分

T3：最高分：xxx 100 分 平均分：100 分

由于不确定出题人是否可以去讲解这篇 sol，所以大家自发给每道题的最高分大爷鼓鼓掌吧……

如果你的算法对于任何数据都能干翻标程，欢迎指点一下大蒟蒻 l1ll5。若你的算法由于数据弱而获得了比预料中更好的分数，建议您在 D 翻 l1ll5 后还是看看这篇 solution 吧 qwq

注：期望得分均为仅使用该算法的期望得分，由于实现不同，期望得分不一定为您的真实得分。算法大概按照思维难度/.得分量排序，其中，算法零均为“不是算法的算法“，可以通过手动讨论得到，若您爆零了，真是不应该啊。

[分裂/split] 改编题/经典题

算法零：（期望得分 10 分）$n = 11$的话，手算也能得到答案是142吧。

算法一：（期望得分 40 分）对于这个数据范围，可以兹词一些线性时间复杂度的算法。看到兔子显然是斐波那契数列啦。如何处理死亡呢？std 的做法是递推的时候减去 f[i-11]，还有一种可行的做法是 f[i][j] 表示第 i 年 j 岁的兔子数量...无论怎么搞，递推就可以了。时间复杂度$O(n)$

算法二：（期望得分 10 分）给出了确定的n值，可以预处理出来。

算法三：（期望得分 70 分）观察到$n \leq 10^9$的数据范围是$O(n)$的递推无法接受的（当然您可以在 O2 下尝试暴力卡常，也许能过）。那怎么办呢？当然想到矩阵乘法啦。不会的话...emmmm，分块打表大法好！大概就是隔一段预处理出一个答案，计算的时候算一段即可。

算法四：（期望得分 10 分）给出了确定的n值，可以预处理出来。

算法五：（期望得分 100 分）$n \leq 10^{18}$好像不能打表了。所以老老实实写矩乘吧！构造矩阵的话，最好先写出朴素的递推式，不要强行构造。根据算法一的不同，矩阵也是不同的。在此不给出具体矩阵了，std 的实现可以参考标程，这样。

总结：随便考个矩乘好了。抱着这样的想法就出了这道题。暴力给到了良心的 80 分哦。另外，T1 换掉过一次，在这之前是一个毒瘤高精度...

[差题/bad] 原创题/经典题

算法零：（期望得分 10 分）只存在区间求和操作，直接前缀和维护即可。

算法一：（期望得分 20 分）对于$n \leq 5000$，可以直接按照题意模拟。时间复杂度$O(nm)$

算法二：（期望得分 30 分）不存在区间取反的情况，只需要支持单点修改，单调加，区间最值和区间和，线段树实现难度很低。

算法三：（期望得分 10 分）可以知道，一个点取反偶数次不变，奇数次等价于一次。所以可以先差分处理掉所有取反操作，O(n)扫一遍，维护前缀和即可。

算法四：（期望得分 100 分）考虑如何支持区间取反操作：很简单，max,min 取反后互换，sum 取反即可。区间修改需要上 lazy 标记。

总结：一道码农题，所以打上了差题的 tag，区间取反这个操作可能不是太常见（也可能是我见得少），但是思维难度不高。实现不太困难，码长不太短，就酱。

[读书/book] 改编题 （出处 world final 2017 C

算法零：（期望得分 10 分）哇我能看懂语文！答案显然就是0

算法一：（期望得分 20 分）哇我会搜索！直接暴力搜索每个金砖放在哪里。（真是耿直的搜索策略呢...）时间复杂度$O\left((rc)^{\sum h}\right)$

算法三：（期望得分 30 分）开始正经的考虑此问题。

如果不存在俯视图，这个问题将非常的简单。稍有常识的人都知道，解决简单的问题比解决难的问题要容易，我们就先从这个假设下手。

记一个$1*1*h$的长方体为一个高度为h的**金砖堆**，考虑输入中最高的**金砖堆**，设它的高度是h_1。那么它至少在正视图和侧视图中出现了一次，我们假设它在正视图出现了m_1次，在侧视图出现n_1次，

设$n_1 \le m_1$。在这种情况下，我们将会至少需要m_1堆高度为h_1的金砖。我们可以通过安排它们得到正确的侧视图和正视图，从而使这m_1列和n_1行中至少有一堆高度为h_1的金砖。

然后考虑下一个高度h_2，同样的，我们有m_2列和n_2行必须包含一堆高度为h_2的金砖堆的限制。但是这里有一点不同，m_2可能是0，在这种情况下我们将会把高度为h_2的金砖堆放到已经放了h_1堆的列中。

按照这种策略，我们将使用总计：$\sum h_i \cdot \max\left(m_i,n_i\right)$个金砖。这样将所有金砖摆放好后已经可以满足正视图和侧视图（如果忽略俯视图），多余的金砖就可以偷走了。

对于 4,5,6 测试点，满足俯视图一定是充满的，就可以套用上述做法了。

算法四：（期望得分 100 分）

现在我们考虑俯视图的存在，这改变了上述策略中的两点：

第一：在结束时，我们可能会不得不在其中一些剩余的空间里放一块金砖，以防止俯视图本来非空的格子变空。解决这一点是容易的：我们只需要记录我们在不考虑这一点时会使多少本来为非空的方块非空，最后减去需要填充的即可。

第二：由于某些空间在俯视图看来是空的，所以可能不再可能使得只用$\max\left(m_i,n_i\right)$,堆金砖以满足m_i列和n_i行的限制可行。但是我们依旧希望有尽可能多的堆可以同时满足行和列的限制。发现同时满足两个限制这是一个二分匹配模型：

当一个格子非空时，将这一行连到这一列上，以表示该格子可以同时满足行列限制。对于每个可能的高度，都进行这样的操作。然后跑二分图最大匹配就结束了。

总结：别看此题是个 WF 题，还是很水的，大概就是签到题的水平吧，不过放在这场考试里颇有些防 AK 的意味。二分图最大匹配是标准的 noip 知识点，对于这道题而言，如果意识到了二分图模型就很容易了，从有诱导性的部分分下手也不困难。行列连边是很常见的二分图模型，希望大家记住哦。

另外，数据没有针对性的去卡各种奇异的贪心做法，不知道有没有人艹掉 std

[总结]

本套题作为 STSO-Easy Round 系列的第二套题，应光老师要求，出了“NOIP”难度。

总体来说比上一套难一些，不过暴力分还是很多，为了避免 T3 太过不可做，给出了一点点思路上的提示。

沿袭上一套的传统，暴力分基本都对正解有启示性的作用，同时依旧给出了大样例，时限基本都在标程的 2-5 倍，可以说没什么很坑的地方。

总之，这篇 sol 就结束了，谢谢大家。