A. 雷神之路

题意

其实就是爬楼梯，每一次可以上$1,2,3$个楼梯，问你爬到哪一个特定阶梯有多少种方法。其中有m个楼梯是不能走的。$n \le 10^5$
$m \le 500$。

分析

转移方程其实是很好想的，就是$dp[0]=0, dp[1]=1, dp[2]=2, dp[3]=4$，当$i>3$时，
$dp[i]=dp[i-1]+dp[i-2]+dp[i-3]$，这是一个非常好想的转移方程。这题目有两个限制，一是不能走的楼梯，当我们转移的时候，如果有的位置是不能走的，那么就是不要转移，把他当做0就可以了。二是n很大，$O(n)$的方法不能过，所以要采用矩阵快速幂来加速。由转移公式可以很快的得到转移矩阵：

$$\begin{bmatrix} dp_{n-2} \\ dp_{n-1} \\ dp_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}^{n-1} \begin{bmatrix} dp_1 \\ dp_2 \\ dp_3 \end{bmatrix} (n>3)$$

这样就可以转移了。

思考

不过就算知道了上面，还有一个问题就是因为矩阵快速幂的特点并不是顺序递推过去，如果直接进行计算的话，是没有办法规避那些不能走的楼梯的，所以这个时候我们就可以用每一个断点，把整个区间编程一个一个小的区间，对每个区间分别进行转移，我们只需要知道每一个区间的前三个，就可以算出后面的所有，而且都是log级别的，很快。

B. Snowdrop修长廊

题意

将序列分成几个子段，每个子段都有一个花费$cost(i,j)=W+(x[j]-x[i])^2$，问你如何划分，是的所有的花费总和最小。
$n \le 2*10^5$

分析

这个题目的转移方程就是$dp[i]=min(dp[j]+W+(x[i]-x[j+1])^2)\ \ j<i$,当然，这个x都是有序的。
那么这个状态转移的时候是n^2的，对于这样大的一个数据范围是不能接受的，于是乎就变成了斜率dp的模板题$\rightarrow_\rightarrow$。
我们来考虑当$k<j<i$时，假设j的决策要比k要好，那么就意味着
$dp[j]+W+(x[i]-x[j+1])^2<dp[k]+W+(x[i]-x[k+1])^2$展开
$dp[j]-2*x[i]*x[j+1]+x[j+1]^2<dp[k]-2*x[i]*x[k+1]+x[k+1]^2$
我们把上面的移项，会发现
$\frac{(dp[j]+x[j+1]^2)-(dp[k]-x[k+1]^2)}{2*(x[j+1]-x[k+1])}<x[i]$
我们然后令$y_j=dp[j]-x[j+1]^2, x_j=2*x[j+1]$,那么就变成了斜率dp的形式，我们再使用一个单调队列维护一个优先性就可以了。

思考

对于$dp[i]=dp[i]*dp[j]+****$，其中i和j无法分开时，就可以考虑斜率dp来优化，但是首先要能够把斜率的式子推出来。

C. TaoSama与煎饼

题意

一个煎饼开始在1号机器，然后有一些道具，每个道具可以让煎饼往前走几步，到达其他的机器，没到达一个机器，美味度就会增加$a[i]$，问你如何使用道具，可以得到最大的美味度。一个有四种m个道具，n个机器。
$n \le 350\quad m \le 120$

分析

最开始的时候，可以一下就得到一个状态$dp[pos][a][b][c][d]$表示在pos位置并且使用了四种道具各a,b,c,d个。转移的话就是

$$dp[pos][a][b][c][d]=min(dp[pos-1][a-1][b][c][d], dp[pos-2][a][b-1][c][d], dp[pos-3][a][b][c-1][d], dp[pos-4][a][b][c][d-1])$$

但是这样的无论是时间复杂度还是空间复杂度都是不允许的，所以我们要简化状态。因为是固定从1点开始出发，所以我们只需要知道了每种道具用了多少个，就可以知道当前位置，这样就可以吧第一维去掉，然后就可以很方便的转移。

思考

经常会碰到一些比较状态很复杂，包含了很多因素的状态，如果不化简的话，转移会很困难，所以我们就要对其中某些因素所表示的含义进行思考，尽量把其中重复的，可以互相得到的去掉，这样就可以化简的很简单。

D. 任务

题意

把n个任务分别放在两个机器上完成，不同的任务在不同的机器上需要不同的时间，问你如何分配可以使所有任务完成的时间最短。

分析

这个题目有一个很重要的条件，就是第i个任务可以被处理当且仅当前$i-1$个任务已经被处理或者正在被处理，所以说，在我们转移的时候，一般的转移方程是不对的。其实这个题和搭积木很像，都是往两个地方添加东西，只不过求的是$min(max(h_1, h_2))$，对于这个问题，我们可以很快想到一个状态$dp[i][j]$表示第i个任务放在机器1上时，机器1和机器2的完成时间的差值。如果$j < 0$，表示机器1的时间比机器2的时间短，反之则长。在专一的时候考虑上面所说的那个条件，转移一共分四种：

```
if(j >= 0) {    //1比2高
    dp[i][a[i] + off] = min(dp[i][a[i] + off], dp[i - 1][j + off] + a[i]); // 把任务继续放在1上
    dp[i][j - b[i] + off] = min(dp[i][j - b[i] + off], dp[i - 1][j + off] + max(0, b[i] - j)); // 放在2上
}
else {  // 2比1高
    dp[i][-b[i] + off] = min(dp[i][-b[i] + off], dp[i - 1][j + off] + b[i]);//  放在2上
    dp[i][j + a[i] + off] = min(dp[i][j + a[i] + off], dp[i - 1][j + off] + max(0, a[i] + j)); //  放在1上
}
```

因为二者的差值最多就是一个任务的全部时间，所以复杂度是可以接受的。

思考

这道题关键就在于对于那个条件的理解，画图的时候就可以发现，如果认为这个条件没用的话，就没有办法解决问题。

E. Goozy的积木

题意

有n个木块，用着两个木块搭两个塔，如果能把两个塔搭的一样高，问最大高度。

分析

以差值来表示状态，$dp[i][j]$表示把放第i个积木后，两个塔的高度差为j的最大高度。转移就比较简单了，放高塔，放低塔或者都不放，最后看$dp[n][0]$是否有答案。

思考

经典差值dp

G. Simple dp

题意

一棵树n个节点，给定每个节点的子树的节点个数，每个节点如果有子节点，子节点数必然大于等于2，求这样的树是否存在。

分析

因为数据量不大，所以一开始想到的是直接搜索，由树的特点可以发现，父亲的大小一定比子节点大，所以说，大小最大的那个节点一定是根节点。我们把所有的结点按照大小排序。搜索的时候每次假设一个点为一个根，然后在大小小于他的节点中枚举他的儿子，一但发现符合要求，就继续搜后面的，知道搜完为止。

思考

这个数据范围应该状压DP可以做，不过还没想到。。。

H. 又见背包

题意

有n种大小不同的数字a_i,每种m_i个，判断是否可以从这些数字中选出若干使它们的和恰好为k。

$1 \le n \le 100, \quad 1 \le k \le 100000$
$1 \le a_i \le 100000$
$1 \le m_i \le 10^9$

分析

经典01背包$+$二进制优化。。。听说梅小姐想卡掉这个，用单调队列结果失败233333。

思考

单调队列优化背包可以，但是需要数据量在一定的程度上，否则二者并不能看出差别。

L. 来签个到吧

题意

盒子里有n个球，每个球上面有数字，每次选两个球，如果$|x-y|$不存在，就放一个标有这个数字的球进去，知道不能放为止。然后开始摸球，求把所有的球都摸到一遍的期望操作次数。

分析

首先，对于第一个要求，是一个神奇的性质，最后这个集合里面的球的个数是max/gcd，这样第一个问题就解决了。对于期望次数，首先我们可以想到每次摸球的情况，假设要把i个球摸完，那么接下来摸球就有两种情况，摸到一个没出现过的球或者已经出现过的球，这两个的概率分别是$\frac{i-1}{n}$，　$\frac{n-i+1}{n}$，那么这个情况下的期望就是$dp[i] = \frac{i-1}{n}dp[i-1] + \frac{n-i+1}{n}dp[i] + 1$。

思考

这题题意很坑啊，次数居然还要加上添加球的次数。其次就是输入是有0的，在求gcd的时候，需要把0排除在外。