花火大会

我们注意到题目中说:"每个时刻可以向右(x轴正方向)移动到x轴任意的地方"。这代表我们不能向左往回走。

我们要计算的就是因为不能回头走而产生的不满值。

如果可以往左走,那解法就是对于每个时间发出的烟花,找到它们的中间点。

如何能求出中间点的位置呢?

可以将点根据p从小到大排序,对于每个点记一个q_i,初始设为-2。

例如:

$1, 3, 4, 6, 7$

i	**1**	**2**	**3**	**4**	**5**
q_i	-2	-2	-2	-2	-2
	-1	-2	-2	-2	-2
	0	-2	-2	-2	-2
	0	-1	-2	-2	-2
	0	0	-2	-2	-2
	0	0	-1	-2	-2

我们规定第一个出现-2的地方的i为pos,那么中间点的i就是$pos-1$。

回到这一题,我们把烟花按时间为第一关键字从大到小,按位置为第二关键字从小到大排序。

先处理时间最大(设为t时刻)的烟花,按照之前介绍的方法求出中间点后,对于$t-1$时刻的烟花,有两种情况:

1.$t-1$时刻的烟花完全在t时刻的烟花的左侧,即$\forall t_i=t-1$有$p_i<min\{p_j\}(t_j=t)$。

我们只需同样地求出$t-1$时刻的烟花的中间点即可,因为从上一个时间的中间点到这个时间的中间点一定是最优的。

2.$t-1$时刻的烟花不完全在t时刻的烟花的右侧。

这就相当于只有一次选择机会,将t和$t-1$时刻的烟花叠加一起考虑选取中间点。

如何实现呢?

```
for(int i=1; i<=n; i++){
    scanf("%lld%lld",&a[i].t,&a[i].p);
    ans+=a[i].p;
}
sort(a+1, a+1+n);
for(int i=1; i<=n; i++){
    q.push(-a[i].p);
    q.push(-a[i].p);
    ans+=q.top();
    q.pop();
}
```

每次将p_i压入一个小根堆里，将ans减去堆顶，就相当于加上了从p_i到$p_{q.top()}$的距离。

规划

新图E是单向边和双向边是相同的。因为你只需要从1开始走就可以了，遍历点的先后顺序无所谓。

不妨把它看做单向的。因为你如果从u->v,那么你肯定不会再去走v->u了。

首先可以发现在新图E中每个节点的入度一定为1.若为0则无法到达，若大于1则至少可以去掉一条边，在节点i既还能被到达又使得答案更优。

所以你的新图E一定是一棵树。

~~是不是树你也不一定要想清楚。~~

你只用在dij(spfa)松弛的时候，记录每一个点的前驱边的大小p[i].

如果dis[u]+w>dis[v]，则更新dis[v]。p[i]=w;

若dis[u]+w==dis[v]，则更新dis[v]。p[i]=min(p[i],w);

拼图

有两问，第一问就是有一个$2 \times n$的长方形，我们要在里面放上1×2的纸片来将它填满，问有多少种不同的方法。第二问还是有$2 \times n$的一个长方形，我们要用$1 \times k(k \in N_+)$的纸片来将它填满，问有多少种不同的方法。

第一问

对于第一问，我们考虑铺满$2 \times n$的长方形之前，时只有两种情况的。 我们注意到： 如果前面$n-1$列都已经铺满了，那么我们只用竖着填满就好了，如果是前面$n-2$列铺满了，我们就横着铺两个纸片就能填满。

令f_n表示填满填满n列的答案。 则有：

$$f_n = f_{n-1} + f_{n-2}$$

第二问

一个引理

我们首先考虑$1 \times n$的长方形，我们要将它用$1 \times k$的纸片来填满的方案数。

令g_n表示填满$1 \times n$的长方形的方案数。

注意到，对于格子，我们有两种决策方案：

1.放在前一个长条之中

2.新建一个长条，放在它里面 特别地，对于第一个格子，我们只有一种方案就是新建一个长条。

则有：

$$g_n = 2^{n-1}$$

特别地，我们定义$g_0 = 1$

具体推导

令有n列的方案数为t_n。

注意到，对于$2 \times n$的长方形，我们用的纸条都是$1 \times k$的，那么，我们将纸条竖着放的时候，它只可能是1×2的，我们称这样的竖纸条为S。

对于一个填满了纸条的方案来说，我们设从左开始，第一次出现S的列数为p，那么，在1到$p-1$这些列中，都没有S。

对于S之前的，由引理1可知，方案数为$g_{p-1} \times g_{p-1}$。对于S之后的，方案数就是t_{n-p}。

可知：

$$t_n = \sum_{p=1}^{n} (g_{p-1}^2 + t_{n-p}) + g_n^2$$

即：

$$t_n = g_0^2 \cdot t_{n-1} + g_1^2 \cdot t_{n-2} + \cdots + g_{n-2}^2 \cdot t_1 + g_{n-1}^2 \cdot t_0 + g_n^2$$

将g_n代入：

$$t_n = 4^0 \cdot t_{n-1} + 4^0 \cdot t_{n-2} + \cdots + 4^{n-3} \cdot t_1 + 4^{n-2} \cdot t_0 + 4^{n-1}$$

原式×4得:

$$4t_n = 4^1 \cdot t_{n-1} + 4^2 \cdot t_{n-2} + \cdots + 4^{n-2} \cdot t_1 + 4^{n-1} \cdot t_0 + 4^n \qquad ①$$

又推得:

$$t_{n+1} = \sum_{p=1}^{n+1} (g_{p-1}^2 + t_{n-p}) + g_{n+1}^2$$

即:

$$t_{n+1} = 4^0 \cdot t_n + 4^0 \cdot t_{n-1} + \cdots + 4^{n-2} \cdot t_1 + 4^{n-1} \cdot t_0 + 4^n \qquad ②$$

②-①得:

$$t_{n+1} - 4t_n = 4^0 \cdot t_n + 4^0 \cdot t_{n-1} - 4^1 \cdot t_{n-1}$$

整理，得:

$$t_{n+1} = 5t_n - 3t_{n-1}$$

变形，得:

$$t_n = 5t_{n-1} - 3t_{n-2}$$