这其实是一道“披着狼皮的背包题”

我们只需要对状态稍作调整就可以套背包啦~~~

我们先把骨牌翻转，调整至点数大的在上面

这样，我们就能保证上方的点数一定比下方大，并且保证每翻转一 次，都能使上下的点数之差变小，而变小的点数，就是上下点数之差乘以2。

把改变的点数看成物品的体积，初始上下方的点数之差看做背包体积，不难看出背包问题的模型。

那么物品的重量是什么呢？

因为我们一开始就把点数大的放在了上面，而每放一次，翻转次数就+1。考虑：要是我后来后悔了，我发现不翻这个骨牌更好怎么办？那我会把它翻回来，那么相当于没有翻这个骨牌。

因此，一开始翻过的骨牌重量就是-1，未翻过的骨牌重量就是1（重量等价于翻转次数）

当然，上下相同的骨牌就是体积为0，重量为0的物品，因为他们无论怎么翻，都不会对上下点数差造成影响。

至此，背包的模型就出来了。这个问题被简化成：有n个物品，给出每个物品的体积v[i],他们的重量是1或-1。背包的重量为base，体积为tot，现在请把这n个物品放到背包里去，总体积不能超过tot，体积最大的情况下使得物品重量之和最小。

其中，dp[i][j]表示前i件物品能装到体积为j的最小重量

vs[i][j]表示前i件物品能否装到j体积

上代码：```c

#include <cstdio>

int dp[1005][6005];

bool vs[1005][6005];

int w[1005];

int v[1005];

int min(int a,int b)

{

if(a<b) return a;

return b;

}

int main()

{

int n,i,j,x,y,base=0,tot=0;

//base表示背包重量，就是初始重量，初始翻转次数

scanf("%d",&n);

for(i=1;i<=n;i++) {

scanf("%d%d",&x,&y);

if(x>y){

v[i]=2\*(x-y);//点数变化量看做体积

w[i]=1;

tot+=x-y;

}

if(y>x) {

v[i]=2\*(y-x);

w[i]=-1;

tot+=y-x;

base++;//初始重量

}

}//用体积为v的物体装总体积为tot的背包,装的体积尽量多的情况下，总重量w最小 背包重量为base

for(i=1;i<=n;i++){

for(j=1;j<=tot;j++){

dp[i][j]=dp[i-1][j];

vs[i][j]=vs[i-1][j];

if(vs[i-1][j-v[i]]||j-v[i]==0){

if(!vs[i][j]){

dp[i][j]=dp[i-1][j-v[i]]+w[i];

vs[i][j]=1;

}

else dp[i][j]=min(dp[i][j],dp[i-1][j-v[i]]+w[i]);

}

}

}

//printf("%d",base+dp[n][tot]);

for(i=tot;i>=1;i--) if(vs[n][i]) break;

//找到第一个用所有物品可以装到的体积

printf("%d",base+dp[n][i]);

}