比赛链接

补题情况

题目	蒋贤蒙	王赵安	王智彪
B	2	0	0
E	0	0	0
F	0	0	0
G	2	0	0
H	2	0	0
K	0	0	0

题解

B. discount

题意

给定 n 种商品，对于每个商品 i□有两种策略，一种是花费 a_i 购买同时赠送商品 j□一种是花费 b_i 购买□$(b_i\le a_i)$

问至少获得所有商品各一种的最小费用。

题解

假定花费 a_i 购买商品 i 可以赠送商品 j□则连边 $j\to i$□其中 i 是 j 的儿子结点。

于是可以得到基环树森林，定义每个结点的状态 $0/1/2$ 表示不购买该结点/以 b_i 购买该结点/以 a_i 购买该结点。

于是原问题等价于使得每个结点的状态如果为 0 则至少有一个儿子状态为 2 的最小费用。

先考虑树的解法，设 $\text{dp}(u,0/1/2)$ 为只考虑子树 u 的情况下的最小费用，不难得到状态转移方程。

接下来考虑每个基环树上的环，任取一个点，枚举该点在环上的子节点状态是否为 2，然后进行线性 dp□

```
const int MAXN=1e5+5;
const LL inf=1e18;
struct Edge{
    int to,next;
}edge[MAXN];
int head[MAXN],edge_cnt;
void Insert(int u,int v){
    edge[++edge_cnt]=Edge{v,head[u]};
```

```
    head[u]=edge_cnt;
}
int fa[MAXN],a[MAXN],b[MAXN];
bool inque[MAXN],node_vis[MAXN],node_cyc[MAXN];
LL dp[MAXN][3];
vector<int> cyc;
void dfs(int u){
    LL s1=0,s2=inf;
    node_vis[u]=true;
    for(int i=head[u];i;i=edge[i].next){
        int v=edge[i].to;
        if(node_cyc[v])continue;
        dfs(v);
        LL w=min({dp[v][0],dp[v][1],dp[v][2]});
        s1+=w;
        s2=min(s2,dp[v][2]-w);
    }
    dp[u][0]=s1+s2;
    dp[u][1]=b[u]+s1;
    dp[u][2]=a[u]+s1;
}
LL dp2[MAXN][3];
LL cal1(){
    int u=cyc[0];
    dp2[u][0]=min({dp[u][0],dp[u][1]-b[u],dp[u][2]-a[u]});
    dp2[u][1]=dp[u][1];
    dp2[u][2]=dp[u][2];
    _for(i,1,cyc.size()){
        int u=cyc[i],p=cyc[i-1];
        dp2[u][0]=dp2[p][2]+min({dp[u][0],dp[u][1]-b[u],dp[u][2]-a[u]});
        dp2[u][0]=min(dp2[u][0],dp[u][0]+min(dp2[p][0],dp2[p][1]));
        dp2[u][1]=dp[u][1]+min({dp2[p][0],dp2[p][1],dp2[p][2]});
        dp2[u][2]=dp[u][2]+min({dp2[p][0],dp2[p][1],dp2[p][2]});
    }
    return dp2[*cyc.rbegin()][2];
}
LL cal2(){
    int u=cyc[0];
    dp2[u][0]=dp[u][0];
    dp2[u][1]=dp[u][1];
    dp2[u][2]=dp[u][2];
    _for(i,1,cyc.size()){
        int u=cyc[i],p=cyc[i-1];
        dp2[u][0]=dp2[p][2]+min({dp[u][0],dp[u][1]-b[u],dp[u][2]-a[u]});
        dp2[u][0]=min(dp2[u][0],dp[u][0]+min(dp2[p][0],dp2[p][1]));
        dp2[u][1]=dp[u][1]+min({dp2[p][0],dp2[p][1],dp2[p][2]});
        dp2[u][2]=dp[u][2]+min({dp2[p][0],dp2[p][1],dp2[p][2]});
    }
    return
min({dp2[*cyc.rbegin()][0],dp2[*cyc.rbegin()][1],dp2[*cyc.rbegin()][2]});
```

```
}
LL solve(int u){
    cyc.clear();
    stack<int> st;
    int pos=u;
    while(!inque[pos]){
        inque[pos]=true;
        st.push(pos);
        pos=fa[pos];
    }
    node_cyc[pos]=true;
    cyc.push_back(pos);
    while(st.top()!=pos){
        int tmp=st.top();
        node_cyc[tmp]=true;
        cyc.push_back(tmp);
        st.pop();
    }
    reverse(cyc.begin(),cyc.end());
    for(int u:cyc)
    dfs(u);
    return min(cal1(),cal2());
}
int main()
{
    int n=read_int();
    _rep(i,1,n)
    a[i]=read_int();
    _rep(i,1,n)
    b[i]=a[i]-read_int();
    _rep(i,1,n){
        fa[i]=read_int();
        Insert(fa[i],i);
    }
    LL ans=0;
    _rep(u,1,n){
        if(!node_vis[u])
        ans+=solve(u);
    }
    enter(ans);
    return 0;
}
```

G. transform

二分答案 $+$ 滑动窗口，赛后一分钟过样例 $\to$ 过题，乐。

J. farm

题意

给定 $n\times m$ 的矩阵，每个位置一个植物，种类为 $a(i,j)$接下来 q 个操作，每次选定一个矩形区域施加种类为 k 的药水。

当植物的种类与被施加的药水种类不同时植物死亡。问最后死亡的植物数。

题解 1

二维线段树维护区间赋值，最后查询时将所有操作下放到子节点暴力修改，时间复杂度 $O(nm\log n\log m)$

题解 2

二维树状数组维护矩形区间加，先将所有操作加入矩阵，最后枚举种类，枚举种类 k 的植物时先消除 k 类药水的影响查询完成后再加回去。

时间复杂度同为 $O(nm\log n\log m)$但常数小。

题解 3

随机给每个种类 k 赋一个值 $f(k)$然后哈希处理矩阵加，当种类 k 的植物的所在位置的权值恰好为 $f(k)$ 的倍数时该植物存活。

如果不放心可以二重哈希，时间复杂度 $O(nm)$