思路：

最开始想的是以奶牛要求时间的结束点从小到大进行排序，但后来发现这样的想法是错误的。后来经过调整，应该先按奶牛要求的时间起始点进行从小到大排序，然后维护一个优先队列，里面以已经开始挤奶的奶牛的结束时间早为优先。然后每次只需要检查当前是否有奶牛的挤奶工作已经完成的机器即可，若有，则换那台机器进行工作。若没有，则加一台新的机器。

#include <iostream>

#include <string.h>

#include <algorithm>

#include <cmath>

#include<queue>

#include<cstdio>

#define ll long long

using namespace std;

struct node

{

int l,r,machine,num;

bool operator <(const node &a)const

{

if(r==a.r)

return l>a.l;

return r>a.r;

}

}a[50010];

bool cmp(node a,node b)

{

return a.l<b.l;

}

bool cmp1(node a,node b)

{

return a.num<b.num;

}

int main()

{

// freopen("input.txt","r",stdin);

ios::sync\_with\_stdio(false);

cin.tie(0);

int n;

cin>>n;

for(int i=0;i<n;i++)

{

cin>>a[i].l>>a[i].r;

a[i].num=i+1;

}

sort(a,a+n,cmp);

int cnt(1);

priority\_queue<node> q;

a[0].machine=1;

q.push(a[0]);

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

{

if(q.top().r>=a[i].l)

{

cnt++;

a[i].machine=cnt;

q.push(a[i]);

}

else

{

a[i].machine=q.top().machine;

q.pop();

q.push(a[i]);

}

}

cout<<cnt<<endl;

sort(a,a+n,cmp1);

for(int i=0;i<n;i++)

cout<<a[i].machine<<endl;

return 0;

}