要解决LIS问题，假设我们有一列a_n，那么可以设计f_i表示以i结尾的最长上升子序列长度。

容易设计转移方程：

$$f(i)=\max\{f(j)\,|\,a_j<a_i, j<i\}$$

通过二重循环做这个事。

例如对于序列$[1,3,2,4,5]$，容易找到f的值是$[1,2,1,3,4]$。

下面考虑如何优化复杂度。注意到每次转移的时候，以a_i为下标，选取了一个前缀最大值，这是可以用树状数组、线段树等数据结构来优化到$O(\log n)$插入和查询的。

下面简单介绍线段树的思路：一棵线段树的根节点维护范围$[1,n]$的信息，然后向左右方向递归，左儿子维护$[1,(1+n)/2]$的信息，右儿子维护$[(1+n)/2+1,n]$的信息，如此类推。

下面是修改时的代码：

```
void modify(int x,int l,int r,int pos,int val){
    if(l==r){
        c[pos]=val;
    }
    else{
        if(pos<=mid)modify(x*2,l,mid,pos,val);
        else modify(x*2+1,mid+1,r,pos,val);
        c[x]=max(c[x*2],c[x*2+1]);
    }
}
```

下面是查询时的代码：

```
void query(int x,int l,int r,int pos,int &val){
    if(pos>=r){
        val=max(val,c[pos]);
    }
    else{
        query(x*2,l,mid,pos,val);
        if(pos>mid)query(x*2+1,mid+1,r,pos,val);
    }
}
```

下面是转移部分：

```
for(int i=1;i<=n;i++){
    query(1,1,n,a[i]-1,f[i]);
    f[i]+=1;
    modify(1,1,n,a[i],f[i]);
}
```