给目标序列强制排序，也就是说，不管目标序列里的数字有多乱，我反正按数组下标，强制把目标序列定义为有序数列，比如说1 5 2 3 4，我就反正按数组下标，1就对应第一，5就对应第二。。。然后回到原序列，1 2 3 4 5，根据我们自己定义的排序规则，即按照目标序列排序法，它的排序后的数列应该是 1 3 4 5 2.。。而且这个题目故意说每次只能挑选最后一位数字对序列任意一位进行插入，其实是降低了难度，因为我规定了目标序列是有序数列，其他数列就都得按照目标数列为标准看齐，所以，题目的要求这时候就变成 只要原序列转化出来的新数列是按升序排列，即可，也就是说，求出新数列的最长上升子序列，。。剩下的数字肯定需要通过插入到前方有序数列。。。因此，只要求出最长上升子序列的长度，剩下了多少数字，那就需要插几次。。。原题立解！

简单的总结这种思想，就是自定义规则，我不管目标序列有多乱，我就以他为标准，我就认可他是上升的序列，然后其他序列都得按照他的标准来排序。。。这样就行了

给两个不同的1-n的排列a和b，你每次可以把a数组最后一位的数字插入到任何位置，求从a转换到b的最小操作次数

首先将a相对于b排序，然后找第一个 i 满足a[i] > a[i + 1]，然后答案就是n - i

#include <bits/stdc++.h>

#include<cstring>

#define ll long long

using namespace std;

int a[200010],b[200010],f[200010];

int main()

{

int n;

scanf("%d",&n);

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

scanf("%d",&a[i]);

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

{

scanf("%d",&b[i]);

f[b[i]]=i;

}

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

a[i]=f[a[i]];

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

if(a[i]>a[i+1])

return cout<<n-i<<endl,0;

return 0;

}