逆序对

逆序对的定义：在一个数组a中如果有两个元素满足i < j && a[i] > a[j]，那么(i, j)就称为数组中的一个逆序对

现在，我们要求的是，给定一个长度为N的数组a，求出该数组的逆序对的个数

比如 int a[4] = {7, 5, 6, 4} 逆序对为(7, 6), (7, 5), (7, 4), (6, 4), (5, 4)

我们知道,求逆序对最典型的方法就是树状数组，但是还有一种方法就是Merge\_sort(),即归并排序。

实际上归并排序的交换次数就是这个数组的逆序对个数，为什么呢？

我们可以这样考虑：

归并排序是将数列a[l,h]分成两半a[l,mid]和a[mid+1,h]分别进行归并排序，然后再将这两半合并起来。

在合并的过程中（设l<=i<=mid，mid+1<=j<=h），当a[i]<=a[j]时，并不产生逆序数；当a[i]>a[j]时，在

前半部分中比a[i]大的数都比a[j]大，将a[j]放在a[i]前面的话，逆序数要加上mid+1-i。因此，可以在归并

排序中的合并过程中计算逆序数.

4

7 5 6 4

#include<iostream>

using namespace std;

int n,a[10000],b[10000], ans;

void merge(int low,int mid,int high){

int h,i,j,k;

h=low; i=low; j=mid+1;

while(h<=mid && j<=high){

if(a[h]<=a[j])

b[i]=a[h++];

else{

ans+=mid-h+1;//统计逆序对

b[i]=a[j++];

}

i++;

}

if(h>mid)

for(k=j;k<=j;k++)

b[i++]=a[k];

else

for(k=h;k<=mid;k++)

b[i++]=a[k];

for(k=low;k<=high;k++)

a[k]=b[k];

}

void mergesort(int low, int high){

int mid;

if(low<high){

mid=(low+high)/2;

mergesort(low,mid);

mergesort(mid+1,high);

merge(low,mid,high);

}

}

void init(){

cin>>n;

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

cin>>a[i];

}

int main(){

init();

mergesort(1,n);

cout<<ans;

return 0;

}

# ****【洛谷P1908】逆序对****

## **题目描述**

猫猫TOM和小老鼠JERRY最近又较量上了，但是毕竟都是成年人，他们已经不喜欢再玩那种你追我赶的游戏，现在他们喜欢玩统计。最近，TOM老猫查阅到一个人类称之为“逆序对”的东西，这东西是这样定义的：对于给定的一段正整数序列，逆序对就是序列中ai>aj且i<j的有序对。知道这概念后，他们就比赛谁先算出给定的一段正整数序列中逆序对的数目。

## **输入输出格式**

输入格式：

第一行，一个数n，表示序列中有n个数。

第二行n个数，表示给定的序列。

输出格式：

给定序列中逆序对的数目。

## **输入输出样例**

输入样例#1：

6

5 4 2 6 3 1

输出样例#1：

11

## **说明**

对于50%的数据，n≤2500

对于100%的数据，n≤40000。

**题解：**

这应该算是树状数组入门的经典题了。

普及组的做法这就是归并的思想（就是在归并排序里加一句话，仅此而已）

那么今儿个我们来用用树状数组。

逆序对就是求一对（si,sj） 满足 i<j 且 si>sj 的对数。

先按大小从大到小sort一遍，

然后只要用树状数组求出第i个数前面位置比i小的个数就可以了，用的是前缀和思想。

注：c[i] 表示的是当前已经插入树状数组中的元素中比 i 小的元素的个数。（重要！！！）

入门题不赘述，代码硬上：

#include<bits/stdc++.h>

using namespace std;

int ans=0,n,c[40005];

struct node{

int val,pos;

}a[40005];

bool cmp(node a,node b)

{

return a.val>b.val;

}

void update(int x)

{

for (int i=x; i<=n; i+=i&(-i))

c[i]++;

}

int get(int x)

{

int cnt=0;

for (int i=x; i>0; i-=i&(-i))

cnt+=c[i];

return cnt;

}

int main()

{

scanf("%d",&n);

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

{

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

a[i].pos=i;

}

sort(a+1,a+1+n,cmp);

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

{

update(a[i].pos);

ans+=get(a[i].pos-1);

}

cout<<ans<<endl;

return 0;

}