What Goes Up Must Come Down (树状数组)

思路：树状数组

题意：求构造一个峰形数组的最小相邻交换操作次数。

一般有相邻交换操作的题目，可以考虑用树状数组做。

交换相邻操作的贡献，转化为每个数被其他数交换的次数。

一个数要么在峰左边，要么在峰右边。

在峰左边则左边的数都比它小，在右边则右边的数都比它小。

因此对于一个数我们需要考虑左边和右边比它大的数个数。

其个数就是该数被其他数交换的次数，哪边小就放在哪边。

因此树状数组左右扫一遍，每次取最小值即可。

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int N=1e5+5,M=2e4+5,inf=0x3f3f3f3f,mod=1e9+7;
#define mst(a,b) memset(a,b,sizeof a)
#define lx x<<1
#define rx x<<1|1
#define reg register
#define PII pair<int,int>
#define fi first
#define se second
#define pb push_back
#define il inline
#define lowbit(x) x&(-x)
int n,a[N],c[N],l[N],r[N];
ll ans;
void upd(int x,int v){
    while(x<N){
        c[x]+=v;x+=lowbit(x);
    }
}
int que(int x){
```

```
    int ans=0;
    while(x){
        ans+=c[x],x-=lowbit(x);
    }
    return ans;
}
int main(){
    scanf("%d",&n);
    for(int i=1;i<=n;i++) scanf("%d",&a[i]),l[i]=que(N-1)-que(a[i]),upd(a[i],1);
    mst(c,0);
    for(int i=n;i;i--)   r[i]=que(N-1)-que(a[i]),upd(a[i],1);
    for(int i=1;i<=n;i++) ans+=min(l[i],r[i]);
    printf("%lld\n",ans);
    return 0;
}
```

P1637 三元上升子序列(DP+离散化权值树状数组)

传送门

思路：dp+离散化转权值树状数组。

显然可以设$dp[i][j]$为长度为i以$a[j]$结尾的子序列的个数。

有转移方程：$dp[i][j]=\sum\limits_{k<j,a[k]<a[j]} dp[i-1][k]$

显然暴力时间复杂度：$O(n^2m)$

因为$a[i]\leq 2^{63}$,但$n\leq 3e4$考虑离散化$a[i]$，然后转权值线段树储存$dp[i-1][k]$。

先初始化一元上升子序列，然后从前往后遍历，

有转移方程：$dp[i][j]+=query(a[j]-1)$

再更新$update(a[j],dp[i-1][j])$。

时间复杂度：$O(nmlogn)$,m是几元上升子序列。

```
xxxxxxxxxx
#include<cstdio>
#include<iostream>
```

```
#include<algorithm>
#include<cstring>
using namespace std;
typedef long long ll;
const int N=3e4+5,inf=0x3f3f3f3f,mod=1e9+7;
#define mst(a) memset(a,0,sizeof a)
#define lx x<<1
#define rx x<<1|1
#define reg register
#define PII pair<int,int>
#define fi first
#define se second
int n,m;
ll a[N],b[N],tr[N];
ll dp[4][N];
#define lowbit(x) x&(-x)
void update(int x,int k){
    while(x<=m){
        tr[x]+=k;
        x+=lowbit(x);
    }
}
ll query(int x){
    ll ans=0;
    while(x){
        ans+=tr[x];
        x-=lowbit(x);
    }
    return ans;
}
int main(){
    scanf("%d",&n);
    for(int i=1;i<=n;i++){
        scanf("%lld",&a[i]),b[i]=a[i];
        dp[1][i]=1;
    }
    sort(b+1,b+n+1);
```

```
    m=unique(b+1,b+n+1)-b-1;
    for(int i=1;i<=n;i++) a[i]=lower_bound(b+1,b+m+1,a[i])-b;
    for(int i=2;i<=3;i++){
        mst(tr);
        for(int j=1;j<=n;j++)
        {
            dp[i][j]+=query(a[j]-1);
            update(a[j],dp[i-1][j]);
        }
    }
    ll ans=0;
    for(int i=1;i<=n;i++) ans+=dp[3][i];
    printf("%lld\n",ans);
    return 0;
}
```