

[比赛链接](#)

## 题解

### A. 序列

#### 题意

给定一个  $1 \sim n$  的排列  $a$ 。该排列的每个子序列  $s=(s_1, s_2 \dots s_p)$  对  $k$  的贡献为  $\sum_{i=1}^{p-1} [s_i \leq k < s_{i+1}] + [s_i > k > s_{i+1}]$

对  $k=1 \sim n$  问所有子序列对  $k$  的贡献总和。

#### 题解

每对  $(a_i, a_j) (i < j)$  对  $k \in (\min(a_i, a_j), \max(a_i, a_j))$  的贡献为  $2^{n-j+i+1}$ 。考虑枚举  $(i, j)$  然后差分计算贡献，于是得到  $O(n^2)$  暴力做法。

接下来考虑优化，不妨假设每对  $(a_i, a_j) (i < j)$  对所有  $k$  产生  $2^{n-j+i+1}$  的正贡献，然后对  $k \in [1, \min(a_i, a_j)] \cup [\max(a_i, a_j), n]$  产生负贡献。

正贡献总和不难发现为  $\sum_{i=1}^{n-1} (n-i) 2^{n-i-1}$ 。接下来考虑计算负贡献总和。

考虑从  $1 \sim n$  枚举  $j$ 。树状数组维护  $c_{a_i} = 2^{i-a_i} (i < j)$ 。于是  $j$  对  $[1, a_j]$  产生负贡献为  $2^{n-j} \sum_{i=1}^{a_j} c_i$ 。对  $[a_j, n]$  产生负贡献为  $2^{n-j} \sum_{i=a_j}^n c_i$ 。

同理，再从  $n \sim 1$  枚举  $i$  计算负贡献。最后用正贡献总和减去负贡献总和即可得到答案，时间复杂度  $O(n \log n)$ 。

```
const int MAXN=1e5+5,mod=1e9+7;
struct BIT{
    #define lowbit(x) ((x)&(-x))
    int c[MAXN];
    void add(int pos,int v){
        while(pos<MAXN){
            c[pos]=(c[pos]+v)%mod;
            pos+=lowbit(pos);
        }
    }
    int query(int pos){
        int ans=0;
        while(pos){
            ans=(ans+c[pos])%mod;
            pos-=lowbit(pos);
        }
        return ans;
    }
}
```

```
void clear(){mem(c,0);}
}tree1,tree2;
int query(int L,int R){
    int ans=tree1.query(R);
    if(L)ans=(ans-tree1.query(L-1)+mod)%mod;
    return ans;
}
void add(int L,int R,int v){
    tree2.add(L,v);
    tree2.add(R+1,mod-v);
}
int a[MAXN],pw[MAXN],s[MAXN];
int main()
{
    int n=read_int();
    _rep(i,1,n)
    a[i]=read_int();
    pw[0]=1;
    _for(i,1,MAXN)
    pw[i]=(pw[i-1]<<1)%mod;
    _rep(i,1,n){
        int v1=query(1,a[i]),v2=query(a[i],n);
        v1=1LL*v1*pw[n-i]%mod;
        v2=1LL*v2*pw[n-i]%mod;
        add(a[i],n,v1);
        add(1,a[i],v2);
        tree1.add(a[i],pw[i-1]);
    }
    _rep(i,1,n)
    s[i]=tree2.query(i);
    tree1.clear();
    tree2.clear();
    for(int i=n;i--){
        int v1=query(1,a[i]),v2=query(a[i],n);
        v1=1LL*v1*pw[i-1]%mod;
        v2=1LL*v2*pw[i-1]%mod;
        add(a[i],n,v1);
        add(1,a[i],v2);
        tree1.add(a[i],pw[n-i]);
    }
    _rep(i,1,n)
    s[i]=(s[i]+tree2.query(i))%mod;
    int ss=0;
    _rep(i,1,n-1)
    ss=(ss+1LL*(n-i)*pw[n-i-1])%mod;
    _rep(i,1,n)
    enter((ss-s[i]+mod)%mod);
    return 0;
}
```

# 总结

jxm有一半的时间在 debug

From: <https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link: [https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal\\_string:%E7%BB%84%E9%98%9F%E8%AE%AD%E7%BB%83%E6%AF%94%E8%B5%9B%E8%AE%B0%E5%BD%95:contest2&rev=1625968936](https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:%E7%BB%84%E9%98%9F%E8%AE%AD%E7%BB%83%E6%AF%94%E8%B5%9B%E8%AE%B0%E5%BD%95:contest2&rev=1625968936)

Last update: 2021/07/11 10:02