

Atcoder Regular Contest 115

[比赛链接](#)

E - LEQ and NEQ

题意

给定序列 A 求所有满足下列条件的序列 X 数目：

- $X_i \leq A_i$
- $X_i \neq X_{i+1}$

题解

设 $\text{dp}(i,j)$ 表示满足限制条件且 $X_i=j$ 的序列 $X_{1 \sim i}$ 的数目
 $S_i = \sum_{j=1}^{A_i} \text{dp}(i,j)$

于是对于 $j \leq \min(A_i, A_{i+1})$ 有 $\text{dp}(i+1,j) = S_i - \text{dp}(i,j)$

如果 $A_i < A_{i+1}$ 对于 $A_i < j \leq A_{i+1}$ 有 $\text{dp}(i+1,j) = S_i$

于是维护一棵支持区间取负，区间加，区间赋值的线段树即可，时间复杂度 $O(n \log n)$

```
const int MAXN=5e5+5,Mod=998244353;
int
a[MAXN],b[MAXN],lef[MAXN<<2],rig[MAXN<<2],s[MAXN<<2],addTag[MAXN<<2],signTag
g[MAXN<<2],setTag[MAXN<<2];
bool setFlag[MAXN<<2];
void build(int k,int L,int R){
    lef[k]=L,rig[k]=R,signTag[k]=1;
    if(L==R)
        return;
    int M=L+R>>1;
    build(k<<1,L,M);
    build(k<<1|1,M+1,R);
}
int getLen(int k){
    return b[rig[k]]-b[lef[k]-1];
}
void toSet(int k,int v){
    s[k]=1LL*getLen(k)*v%Mod;
    addTag[k]=0;
    signTag[k]=1;
    setTag[k]=v;
    setFlag[k]=true;
}
```

```
}
void toNeg(int k){
    s[k]=(Mod-s[k]);
    if(setFlag[k])
        setTag[k]=(Mod-setTag[k])%Mod;
    else{
        signTag[k]*=-1;
        addTag[k]=(Mod-addTag[k])%Mod;
    }
}
void toAdd(int k,int v){
    s[k]=(s[k]+1LL*getLen(k)*v)%Mod;
    if(setFlag[k])
        setTag[k]=(setTag[k]+v)%Mod;
    else
        addTag[k]=(addTag[k]+v)%Mod;
}
void push_down(int k){
    if(setFlag[k]){
        toSet(k<<1,setTag[k]);
        toSet(k<<1|1,setTag[k]);
        setFlag[k]=false;
    }
    else{
        if(signTag[k]==-1){
            toNeg(k<<1);
            toNeg(k<<1|1);
            signTag[k]=1;
        }
        if(addTag[k]){
            toAdd(k<<1,addTag[k]);
            toAdd(k<<1|1,addTag[k]);
            addTag[k]=0;
        }
    }
}
void push_up(int k){
    s[k]=(s[k<<1]+s[k<<1|1])%Mod;
}
void update1(int k,int L,int R,int v){
    if(L<=lef[k]&&rig[k]<=R){
        toSet(k,v);
        return;
    }
    push_down(k);
    int mid=lef[k]+rig[k]>>1;
    if(mid>=L)
        update1(k<<1,L,R,v);
    if(mid<R)
        update1(k<<1|1,L,R,v);
}
```

```

    push_up(k);
}
void update2(int k,int L,int R,int v){
    if(L<=lef[k]&&rig[k]<=R){
        toNeg(k);
        toAdd(k,v);
        return;
    }
    push_down(k);
    int mid=lef[k]+rig[k]>>1;
    if(mid>=L)
        update2(k<<1,L,R,v);
    if(mid<R)
        update2(k<<1|1,L,R,v);
    push_up(k);
}
int query(int k,int L,int R){
    if(L<=lef[k]&&rig[k]<=R)
        return s[k];
    push_down(k);
    int mid=lef[k]+rig[k]>>1;
    if(mid>=R)
        return query(k<<1,L,R);
    else if(mid<L)
        return query(k<<1|1,L,R);
    else
        return (query(k<<1,L,R)+query(k<<1|1,L,R))%Mod;
}
int main()
{
    int n=read_int();
    _for(i,0,n)a[i]=b[i+1]=read_int();
    sort(b,b+n+1);
    int m=unique(b,b+n+1)-b;
    _for(i,0,n)a[i]=lower_bound(b,b+m,a[i])-b;
    build(1,1,m-1);
    update1(1,1,a[0],1);
    _for(i,1,n){
        int v=query(1,1,a[i-1]);
        if(a[i]<=a[i-1])
            update2(1,1,a[i],v);
        else{
            update1(1,a[i-1]+1,a[i],v);
            update2(1,1,a[i-1],v);
        }
    }
    enter(query(1,1,a[n-1]));
    return 0;
}

```

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

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:contest:arc_115&rev=1619837352 

Last update: **2021/05/01 10:49**