主席树 l-r l-r个数

#include <cstdio>

#include <iostream>

#include <algorithm>

#include <vector>

#include <set>

#include <map>

#include <string>

#include <stack>

using namespace std;

#define REP(I,N) for (I=0;I<N;I++)

#define rep(I,S,N) for (I=S;I<N;I++)

#define FOR(I,S,N) for (I=S;I<=N;I++)

#define INF 0x3f3f3f3f

#define INFF 0x3f3f3f3f3f3f3f3f

typedef unsigned long long ULL;

typedef long long LL;

const LL M=1e9+7;

const LL maxn=1e5+7;

const double eps=0.00000001;

LL gcd(LL a,LL b){return b?gcd(b,a%b):a;}

inline LL min(LL a,LL b) {return a>b?b:a;}

inline LL max(LL a,LL b) {return a<b?b:a;}

inline LL abs(LL a,LL b) {return a>0?a:-a;}

inline double abs(double a) {return a>0?a:-a;}

int cnt,root[maxn];

struct Tnode{

int l,r,sum;

}T[maxn\*40];

void update(int l,int r,int &x,int y,int pos){//主席树查找L~R数字个数

T[++cnt]=T[y];T[cnt].sum++;x=cnt;

if (l==r) return;

int mid=(l+r)/2;

if (mid>=pos) update(l,mid,T[x].l,T[y].l,pos);

else update(mid+1,r,T[x].r,T[y].r,pos);

}

int query(int l,int r,int x,int y,int L,int R){

if (L<=l&&r<=R) return T[y].sum-T[x].sum;

int mid=(l+r)/2;

int sum=0;

if (mid>=L) sum+=query(l,mid,T[x].l,T[y].l,L,R);

if (R>mid) sum+=query(mid+1,r,T[x].r,T[y].r,L,R);

return sum;

}

int n,m;

int i,j,k;

int l,r;

int main()

{

scanf("%d",&n);

FOR(i,1,n){

scanf("%d",&j);

update(1,n,root[i],root[i-1],j);

printf("%d ",root[i]);

}

FOR(i,1,1000){

scanf("%d%d%d%d",&j,&k,&l,&r);

printf("%d\n",query(1,n,root[j-1],root[k],l,r));

}

return 0;

}

/\*

\*/

Splay

int a[maxn],cnt;

struct splay\_tree{

struct node{

int val,min,add,size,son[2];//add=lazy

bool rev;

void init(int \_val){//开始时T[i].val==a[i-1](线性的);

val=min=max=\_val;size=1;

if (\_val==INF) max=-INF;

add=rev=son[0]=son[1]=0;

}

}T[maxn\*2];//内存池

int fa[maxn\*2],root,tot;

void pushup(int x){

T[x].min=T[x].max=T[x].val;T[x].size=1;

if (T[x].val==INF) T[x].max=-INF;

if (T[x].son[0]){

T[x].min=min(T[x].min,T[T[x].son[0]].min);

T[x].max=max(T[x].max,T[T[x].son[0]].max);

T[x].size+=T[T[x].son[0]].size;

}

if (T[x].son[1]){

T[x].min=min(T[x].min,T[T[x].son[1]].min);

T[x].max=max(T[x].max,T[T[x].son[1]].max);

T[x].size+=T[T[x].son[1]].size;

}

}

void pushdown(int x){

if (x==0) return;

if (T[x].add){

if (T[x].son[0]){

T[T[x].son[0]].val+=T[x].add;

T[T[x].son[0]].min+=T[x].add;

T[T[x].son[0]].max+=T[x].add;

T[T[x].son[0]].add+=T[x].add;

}

if (T[x].son[1]){

T[T[x].son[1]].val+=T[x].add;

T[T[x].son[1]].min+=T[x].add;

T[T[x].son[1]].max+=T[x].add;

T[T[x].son[1]].add+=T[x].add;

}

T[x].add=0;

}

if (T[x].rev){

if (T[x].son[0]) T[T[x].son[0]].rev^=1;

if (T[x].son[1]) T[T[x].son[1]].rev^=1;

swap(T[x].son[0],T[x].son[1]);

T[x].rev=0;

}

}

void rotate(int x,int kind){//zig(1->) zag(0<-)都行

int y=fa[x],z=fa[y];

T[y].son[!kind]=T[x].son[kind],fa[T[x].son[kind]]=y;

T[x].son[kind]=y,fa[y]=x;

T[z].son[T[z].son[1]==y]=x,fa[x]=z;

pushup(y);

}

void splay(int x,int goal){//node x->goal's son

if (x==goal) return;

while (fa[x]!=goal){

int y=fa[x],z=fa[y];

pushdown(z),pushdown(y),pushdown(x);

int rx=T[y].son[0]==x,ry=T[z].son[0]==y;

if (z==goal) rotate(x,rx);

else{

if (rx==ry) rotate(y,ry);

else rotate(x,rx);

rotate(x,ry);

}

}

pushup(x);

if (goal==0) root=x;

}

int select(int pos){//getnode

int u=root;

pushdown(u);

while (T[T[u].son[0]].size!=pos){//这里由于头节点有个-INF 所以不-1

if (pos<T[T[u].son[0]].size) u=T[u].son[0];

else{

pos-=T[T[u].son[0]].size+1;

u=T[u].son[1];

}

pushdown(u);

}

return u;

}

//下面是自己写的一点常用?函数

void update(int l,int r,int val){

int u=select(l-1),v=select(r+1);

splay(u,0);

splay(v,u);

T[T[v].son[0]].min+=val;

T[T[v].son[0]].max+=val;

T[T[v].son[0]].val+=val;

T[T[v].son[0]].add+=val;//lazy

}

void reverse(int l,int r){

int u=select(l-1),v=select(r+1);

splay(u,0);splay(v,u);

T[T[v].son[0]].rev^=1;

}

void revolve(int l,int r,int x){//l~r->循环往后x位

int u=select(r-x),v=select(r+1);

splay(u,0);splay(v,u);

int tmp=T[v].son[0];T[v].son[0]=0;

pushup(v);pushup(u);

u=select(l-1),v=select(l);

splay(u,0);splay(v,u);

fa[tmp]=v;

T[v].son[0]=tmp;

pushup(v);pushup(u);

}

void cut(int l,int r,int x){//l~r->去掉的x位置后 //HDU3487

int u=select(l-1),v=select(r+1);

splay(u,0);splay(v,u);

int tmp=T[v].son[0];

T[v].son[0]=0;

pushup(v);pushup(u);

u=select(x);v=select(x+1);

splay(u,0);splay(v,u);

fa[tmp]=v;

T[v].son[0]=tmp;

pushup(v);pushup(u);

}

int query\_min(int l,int r){

int u=select(l-1),v=select(r+1);

splay(u,0);

splay(v,u);

return T[T[v].son[0]].min;

}

void insert(int x,int val){

int u=select(x),v=select(x+1);

splay(u,0);

splay(v,u);

T[tot].init(val);

fa[tot]=v;

T[v].son[0]=tot++;

pushup(v);pushup(u);

}

void erase(int x){

int u=select(x-1),v=select(x+1);

splay(u,0);

splay(v,u);

T[v].son[0]=0;

pushup(v);pushup(u);

}

void exchange(int l1,int r1,int l2,int r2){//r1-l1+1?=r2-l2+1 OK

if (l1>l2){swap(l1,l2);swap(r1,r2);}

int u=select(l1-1),v=select(r1+1);

splay(u,0);splay(v,u);

int tmp=T[v].son[0];T[v].son[0]=0;

pushup(v);pushup(u);

l2-=T[tmp].size;r2-=T[tmp].size;

int \_u=select(l2-1),\_v=select(r2+1);

splay(\_u,0);splay(\_v,\_u);

fa[tmp]=\_v;

swap(T[\_v].son[0],tmp);

pushup(\_v);pushup(\_u);

u=select(l1-1),v=select(l1);

splay(u,0);splay(v,u);

fa[tmp]=v;

T[v].son[0]=tmp;

pushup(v);pushup(u);

}

int dfs(int x,int k){//小于k的值个数,会被卡

if (x==0) return 0;

if (T[x].min!=INF&&T[x].min>=k) return 0;

if (T[x].max!=-INF&&T[x].max<k) return T[x].size;

int ret=T[x].val<k;

if (T[x].son[0]) ret+=dfs(T[x].son[0],k);

if (T[x].son[1]) ret+=dfs(T[x].son[1],k);

return ret;

}

int query(int l,int r,int k){//小于k的值个数,会被卡 应该套主席树(但是太长两个log)

int u=select(l-1),v=select(r+1);

splay(u,0);splay(v,u);

return dfs(T[v].son[0],k);

}

int build(int l,int r){

if (l>r) return 0;

if (l==r) return l;

int mid=(l+r)/2;

T[mid].son[0]=build(l,mid-1);

T[mid].son[1]=build(mid+1,r);

fa[T[mid].son[0]]=fa[T[mid].son[1]]=mid;

pushup(mid);

return mid;

}

void init(int n){

tot=0;

int i;//0是虚的;

T[tot++].init(INF);//空的

T[tot++].init(INF);//前后两个-INF节点

FOR(i,1,n) T[tot++].init(a[i]);

T[tot++].init(INF);

root=build(1,tot-1);

fa[root]=0;

fa[0]=0;T[0].son[1]=root;T[0].size=0;

}

void print(int now=-1){

if (now==-1) now=root;

pushdown(now);

if (T[now].son[0]) print(T[now].son[0]);

if (T[now].val!=-INF){

if (cnt++) printf(" ");

printf("%d",T[now].val);

}

if (T[now].son[1]) print(T[now].son[1]);

pushup(now);

}

}T;

int n,m,i;

char s[50];

int main()

{

}