线段树

区间求和以及单点修改

void build(int i,int l,int r)

{

if(l==r)

{

cin>>sum[i];

return ;

}

int m=(l+r)/2;

build(i\*2,l,m);

build(i\*2+1,m+1,r);

sum[i]=sum[i\*2]+sum[i\*2+1];

}

int query(int ql,int qr,int i,int l,int r)

{

if(ql<=l&&r<=qr)

return sum[i];

int m=(l+r)/2;

int res=0;

if(ql<=m)

res+=query(ql,qr,i\*2,l,m);

if(m<qr)

res+=query(ql,qr,i\*2+1,m+1,r);

return res;

}

void update(int id,int val,int i,int l,int r)

{

if(l==r)

{

sum[i]+=val;

return ;

}

int m=(l+r)/2;

if(id<=m)

update(id,val,i\*2,l,m);

else

update(id,val,i\*2+1,m+1,r);

sum[i]=sum[i\*2]+sum[i\*2+1];

}

区间覆盖，区间求值

int query(int i,int l,int r)

{

if(sum[i]!=-1)

{

return sum[i]\*(r-l+1);

}

int mid=(l+r)/2;

return query(i\*2,l,mid)+query(i\*2+1,mid+1,r);

}

void update(int ll,int rr,int x,int i,int l,int r)

{

if(ll<=l&&rr>=r)

{

sum[i]=x;

return ;

}

if(sum[i]==x)

return ;

if(sum[i]!=-1)

{

sum[i\*2]=sum[i\*2+1]=sum[i];

sum[i]=-1;

}

int m=(l+r)/2;

if(ll>m)

update(ll,rr,x,i\*2+1,m+1,r);

else if(rr<=m)

update(ll,rr,x,i\*2,l,m);

else

{

update(ll,m,x,i\*2,l,m);

update(m+1,rr,x,i\*2+1,m+1,r);

}

}

区间修改，区间查询

每次操作把一个区间里的每个数都加上一个数，或者查询一个区间的和

query(int ql,int qr,int i,int l,int r)

{

if(ql<=l&&r<=qr)

return sum[i];

int m=(l+r)/2;

if(d[i])

{

d[i\*2]+=d[i];

d[i\*2+1]+=d[i];

sum[i\*2]+=(bl)d[i]\*(m-l+1);

sum[i\*2+1]+=(bl)d[i]\*(r-m);

d[i]=0;

}

bl res=0;

if(ql<=m)

res+=query(ql,qr,i\*2,l,m);

if(m<qr)

res+=query(ql,qr,i\*2+1,m+1,r);

sum[i]=sum[i\*2]+sum[i\*2+1];

return res;

}

void update(int ll,int rr,bl x,int i,int l,int r)

{

if(ll<=l&&rr>=r)

{

sum[i]+=x\*(r-l+1);

d[i]+=x;

return ;

}

int m=(l+r)/2;

if(d[i])

{

d[i\*2]+=d[i];

d[i\*2+1]+=d[i];

sum[i\*2]+=(bl)d[i]\*(m-l+1);

sum[i\*2+1]+=(bl)d[i]\*(r-m);

d[i]=0;

}

if(rr<=m)

update(ll,rr,x,i\*2,l,m);

else if(ll>m)

update(ll,rr,x,i\*2+1,m+1,r);

else

{

update(ll,m,x,i\*2,l,m);

update(m+1,rr,x,i\*2+1,m+1,r);

}

sum[i]=sum[i\*2]+sum[i\*2+1];

}