树状数组与差分

NKOJ4263 1248 3886

helang

一. 复习：差分数组

差分数组

对于一个数组A[ ]，其差分数组D[i]=A[i]-A[i-1] (i>0)且D[0]=A[0]

令SumD[i]=D[0]+D[1]+D[2]+...+D[i]　（SumD[ ]是差分数组D[ ]的前缀和）

则SumD[i]=A[0]+A[1]-A[0]+A[2]-A[1]+A[3]-A[2]+...+A[i]-A[i-1]=A[i]

即A[i]的差分数组是D[i]，　而D[i]的前缀和是A[i]

对于将A数组的区间[L,R]整体加C，我们只需要将D[L]+C和D[R+1]-C
当修改完毕后，我们先求一遍差分前缀和就得到了修改后的数组A[ ]

普通差分数组往往用于离线操作！

二. 复习：树状数组

树状数组

树状数组的两个基本操作：

```
void modify(int x,int d)   //将所有包含A[x]的C[ ]增加d
{
        for( int i = x; i<=n; i+=lowbit(i) )C[i] += d;
}
```

```
int getSum(int x)       //求A[1]+A[2]+...+A[x]
{
      int Sum= 0;
      for ( int k = x; k > 0; k -= lowbit(k) )  Sum += C[k];
      return Sum;
}
```

树状数组一般用于单点修改，区间和查询。

三. 树状数组+差分

例1：区间修改，单点查询

有一长度为n的整数数列数A，有m次操作，操作分下列两种：
1. 查询A[x]的值。（x是输入的数，1<=x<=n）
2. 将A[L]，A[L+1]，…，A[R]都增加k（L，R，k都是输入的数，1<=L≤R<=n）

n，m<=100000

例1：区间修改，单点查询

设数组D[ ]，D[i] = A[i] - A[i-1]，那么A[i] = D[1] + D[2] + ... + D[i]
则D为A的差分数组。

那么求A[i]就可以用树状数组维护D[i]的前缀和。
根据D的定义，对[L,R]区间加上k，那么A[L]和A[L-1]的差增加了k，A[R+1]与A[R]的差减少了k，所以就对差分数组的前缀和进行修改。

于是我们用树状数组来维护差分数组D的前缀和。对于区间修改操作：

```
cin>>L>>R>>k;
Modify(L,k);
Modify(R+1,-k);
```

```
void Modify(int x,int k)
{
    for( int i = x; i<=n; i+=lowbit(i) )C[i] += k;
}
```

例1：区间修改，单点查询

查询操作：

```
cin>>x;
cout<<getSum(x);
```

```
int getSum(int x)
{
      int Sum= 0;
      for ( int i = x; i > 0; i -= lowbit(i) )   Sum += C[i];
      return Sum;
}
```

例2：区间修改，区间查询

有一长度为n的整数数列数A，有m次操作，操作分下列两种：

1. 查询A[L]+A[L+1]+…+A[R]的值。（L，R都是输入的数，1<=L≤R<=n）

2. 将A[L]，A[L+1]，…，A[R]都增加k（L，R，k都是输入的数，1<=L≤R<=n）

n，m<=100000

例2：区间修改，区间查询

设数组D[ ], D[i] = A[i] - A[i-1], 那么A[i] = D[1] + D[2] + ... + D[i]
则D为A的差分数组。

A[1] + A[2] + A[3] + ... + A[k] = D[1] + D[1] + D[2] + D[1] + D[2] + D[3] + ... + D[1] + D[2] + D[3] + ... + D[k]

$$= \Sigma\{(k - i + 1)*D[i]\} \qquad (1<=i<=k)$$

$$= \mathbf{(k+1)*\Sigma\{D[i]\} - \Sigma\{i*D[i]\}}$$

因为k+1是一个常数，所以我们需要维护D[i]的前缀和。也要维护i*D[i]的前缀和。
所以我们需要两个树状数组，一个维护D[i]的前缀和,一个维护i*D[i]的前缀和

```
cin>>L>>R>>k;
Modify(L,k);
Modify(R+1,-k);
```

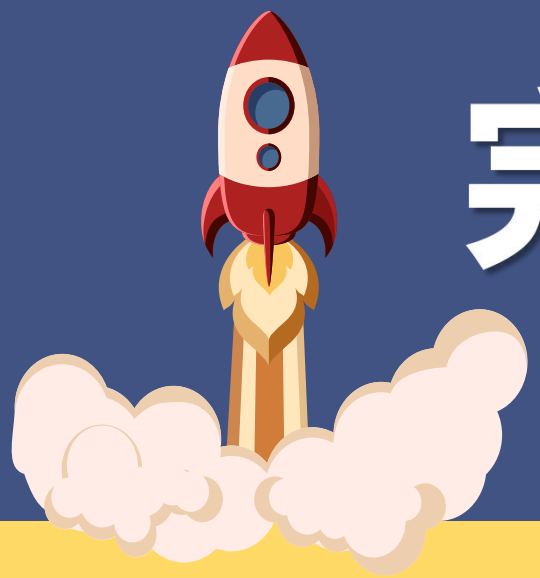
```
void Modify(int x,int k)
{
    for( int i = x; i<=n; i+=lowbit(i) )
    {  C1[i] += k;  C2[i] += x*k;  }
}
```

例2：区间修改，区间查询

查询操作：

```
cin>>L>>R;
cout<<getSum(R)-getSum(L-1);
```

```
/*
    计算A[1]+…+A[x]
       ==(x+1)*Σ{D[i]} - Σ{i*D[i]}
*/
int getSum(int x)
{
    int Sum1 = 0;
    int Sum2 = 0;
    for (int i = x;i > 0;i -= lowbit(i))
    {
        Sum1 += (x + 1) * C1[i];
        Sum2 += C2[i];
    }
    return Sum1 - Sum2;
}
```



完！

以梦为码 心之所往