Rabbit_hair

If you want something you have never had,you must be willing to do something you have never done.


```
/*zoj2112 http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemCode=2112
动态 kth
每一棵线段树是维护每一个序列前缀的值在任意区间的个数，
如果还是按照静态的来做的话，那么每一次修改都要遍历O(n)棵树，
时间就是O(2*M*nlogn)->TLE
考虑到前缀和，我们通过树状数组来优化，即树状数组套主席树，
每个节点都对应一棵主席树，那么修改操作就只要修改logn棵树，
o(nlognlogn+Mlognlogn)时间是可以的，
但是直接建树要nlogn*logn（10^7）会MLE
我们发现对于静态的建树我们只要nlogn个节点就可以了，
而且对于修改操作，只是修改M次，每次改变俩个值（减去原先的，加上现在的）
也就是说如果把所有初值都插入到树状数组里是不值得的，
所以我们分两部分来做，所有初值按照静态来建，内存O(nlogn)，
而修改部分保存在树状数组中，每次修改logn棵树，每次插入增加logn个节点
O(M*logn*logn+nlogn)


*/
#include<cstdio>
#include<cstring>
#include<cstdlib>
#include<iostream>
#include<algorithm>
#include<vector>
#include<cmath>
#define ls(i) T[i].ls
#define rs(i) T[i].rs
#define w(i) T[i].w
#define Find(i) (lower_bound(LX.begin(),LX.begin()+n1,i)-LX.begin())+1

using namespace std;
const int N=60000+10;
struct node{
    int ls,rs,w;
    node(){ls=rs=w=0;}
}T[2000000];
struct ope{
    int i,l,r,k;
}op[11000];
vector<int> LX,Q1,Q2;
int n,n1,m,cnt;
int a[61000],root[61000*2];
inline int lowbit(int x){
    return x&-x;
}
void build(int &i,int l,int r,int x){
    T[++cnt]=T[i]; i=cnt;
    w(i)++;
    if (l==r) return;
    int m=(l+r)>>1;
    if (x<=m) build(ls(i),l,m,x);
    else build(rs(i),m+1,r,x);
}
void ins(int &i,int l,int r,int x,int v){
    if (i==0){ T[++cnt]=T[i]; i=cnt; }
    w(i)+=v;
    if (l==r) return;
    int m=(l+r)>>1;
    if (x<=m) ins(ls(i),l,m,x,v);
    else ins(rs(i),m+1,r,x,v);
}
void my_ins(int pos,int x,int v){
    int t=Find(x);
    for (int i=pos;i<=n;i+=lowbit(i)){
        ins(root[i],1,n1,t,v);
    }
}
int Qy(vector<int> Q1,vector<int> Q2,int l,int r,int k){
    if (l==r) return l;
    int c=0;
    int m=(l+r)>>1;
```

```
72      for (int i=0;i<Q1.size();i++) c-=w(ls(Q1[i]));
73      for (int i=0;i<Q2.size();i++) c+=w(ls(Q2[i]));
74      for (int i=0;i<Q1.size();i++) Q1[i]=(c>=k?ls(Q1[i]):rs(Q1[i]));
75      for (int i=0;i<Q2.size();i++) Q2[i]=(c>=k?ls(Q2[i]):rs(Q2[i]));
76
77      if (c>=k) return Qy(Q1,Q2,l,m,k);
78      else return Qy(Q1,Q2,m+1,r,k-c);
79  }
80  void query(int l,int r,int k){
81      Q1.clear();Q2.clear();
82      Q1.push_back(root[l!=1?l-1+n:0]);
83      Q2.push_back(root[r+n]);
84      for (int i=l-1;i>0;i-=lowbit(i)) Q1.push_back(root[i]);
85      for (int i=r;i>0;i-=lowbit(i)) Q2.push_back(root[i]);
86
87      int t=Qy(Q1,Q2,1,n1,k);
88      printf("%d\n",LX[t-1]);
89  }
90  void work(){
91      cnt=0;
92      //for (int i=0;i<n1;i++) cout<<list[i]<<" ";cout<<endl;
93      memset(root,0,sizeof(root));
94      for (int i=1;i<=n;i++){
95          root[i+n]=root[i-1+n];
96          int t=Find(a[i]);
97          build(root[i+n],1,n1,t);
98      }
99      for (int i=0;i<m;i++){
100         if (op[i].i==0){
101             query(op[i].l,op[i].r,op[i].k);
102         //    cout<<"*** "<<i<<endl;
103         }else{
104             my_ins(op[i].l,a[op[i].l],-1);
105             my_ins(op[i].l,op[i].r,1);
106             a[op[i].l]=op[i].r;
107         }
108     }
109
110 }
111 int main(){
112     int Cas;scanf("%d",&Cas);
113     while (Cas--){
114         scanf("%d%d",&n,&m);
115         LX.clear();
116         for (int i=1;i<=n;i++){
117             scanf("%d",&a[i]);LX.push_back(a[i]);
118         }
119         char s[10];
120         for (int i=0;i<m;i++){
121             scanf("%s",s);
122             if (s[0]=='Q'){
123                 op[i].i=0;
124                 scanf("%d%d%d",&op[i].l,&op[i].r,&op[i].k);
125             }else{
126                 op[i].i=1;
127                 scanf("%d%d",&op[i].l,&op[i].r);
128                 LX.push_back(op[i].r);
129             }
130         }
131         sort(LX.begin(),LX.end());
132         n1=unique(LX.begin(),LX.end())-LX.begin();
133         work();
134     }
135
136
137     return 0;
138 }
```


Rabbit_hair
关注 - 3
粉丝 - 11
+加关注

« 上一篇：主席树 静态区间第k大
» 下一篇：zoj 3540

posted @ 2013-05-24 10:00 Rabbit_hair 阅读(1903) 评论(0) 编辑 收藏