Pontificia Universidad Católica del Perú - FCI

XieXieLucas Notebook - Froz/Phibrain/Ands

November 9, 2017

Contents

1 MinXOR

```
/*
       Mininum XOR-Pair on an array in O(n)
       Trie-based Implementation
*/


#define INT_SIZE 32

struct TrieNode{
    int value;
    TrieNode * Child[2];
};

TrieNode * getNode(){
    TrieNode * newNode = new TrieNode;
    newNode->value = 0;
    newNode->Child[0] = newNode->Child[1] = NULL;
    return newNode;
}

void insert(TrieNode *root, int key){
    TrieNode *temp = root;

    for (int i = INT_SIZE-1; i >= 0; i--){
        bool current_bit = (key & (1<<i));

        if (temp->Child[current_bit] == NULL)
            temp->Child[current_bit] = getNode();

        temp = temp->Child[current_bit];
    }

    temp->value = key ;
}

int  minXORUtil(TrieNode * root, int key){
    TrieNode * temp = root;

    for (int i=INT_SIZE-1; i >= 0; i--){

        bool current_bit = ( key & ( 1<<i) );

        if (temp->Child[current_bit] != NULL)
            temp = temp->Child[current_bit];

        else if(temp->Child[1-current_bit] !=NULL)
            temp = temp->Child[1-current_bit];
    }
    return key ^ temp->value;
}

int minXOR(int arr[], int n){
    int min_xor = INT_MAX;

    TrieNode *root = getNode();
    insert(root, arr[0]);
```

```
    for (int i = 1 ; i < n; i++){
        min_xor = min(min_xor, minXORUtil(root, arr[i]));
        insert(root, arr[i]);
    }
    return min_xor;
}

int main(){
    int arr[] = {9, 5, 3};
    int n = sizeof(arr)/sizeof(arr[0]);
    cout << minXOR(arr, n) << endl;
    return 0;
}
```

2 Offline Less K-Counting

```
//------------------inversiones en un rango (offline)---------------
// ar[]: arreglo, queries=queri.pb(l,r,valor)
//assignar n,q; ez[i] respuesta para la querie i
//hacer read y make;

struct ST{
  ll n,q;
  vector<tri> querie;
  ll t[2*N],ar[N];
  ll poar[N],pok[N],ark[N],ez[N];
  vii v,v1;
  inline ll Op(ll &a,ll &b){ return a+b;}
  inline void build (){
    RREP(i,n-1,1) t[i]=Op(t[i<<1],t[i<<1|1]);
  }
  inline void modify (ll p, ll val){
    for(t[p+=n]=val;p>1;p>>=1) t[p>>1]=Op(t[p],t[p^1]);
  }
  inline ll que(ll l, ll r){
    ll res=0;
    for(l+=n,r+=n;l<r;l>>=1,r>>=1){
      if(l&1) res+=t[l++];
      if(r&1) res+=t[--r];
    }
    return res;
  }
  ll p1=0, p2=0,po=0;
  inline void read(){
    REP(i,0,n) v.push_back({ar[i],i});
    sort(all(v));
    REP(i,0,n) poar[p1++]=v[i].snd;
    REP(u,0,q){
      ll k=querie[u].itm3;
      ark[u]=k;
      v1.push_back({k,u});
    }
    sort(all(v1));
    REP(i,0,q) pok[p2++]=v1[i].snd;
  }
  inline void make(){
    REP(i,0,n) t[i+n]=0; build();
    REP(i,0,q){
      ll x=pok[i];
      // <k, <= k en l,r(despues del &&)
      //inversa , hacer t[i+n]=1;
      while(po<n && ar[poar[po]]<=ark[x]) modify(poar[po++],1);
      ez[x]=que(querie[x].itm1-1,querie[x].itm2);
    }
  }
}st;

int main(){fastio;
  ll n; cin>>n;
  st.n=n;
  REP(i,0,n) cin>>st.ar[i];
  ll q; cin>>q;
  st.q=q;
  REP(i,0,q){
    ll l,r,k; cin>>l>>r>>k;
    st.querie.push_back({l,{r,k}});
  }
  st.read(); st.make();
  REP(i,0,q) cout<<st.ez[i]<<endl;
  return 0;
}
```

3 Online Less K-Counting

```
/*------------------inversiones en un rango (online)---------------
 construccion amortizada a nlog(n);
 cada querie en log^2(n);*/

struct T{
  vi v;
  T () {}
  T (vi v): v(v){}
};
struct ST{
  ll n,ans;
  T t[2*N];
  inline T Op(T &val1, T &val2 ){
    vi v;
    REP(i,0,val1.v.size()) v.pb(val1.v[i]);
    REP(i,0,val2.v.size()) v.pb(val2.v[i]);
    sort(all(v));
    T ty;
    ty.v=v;
    return ty;
  }
  inline ll Op1( T &val1,ll &k){
    ans=0;
    //usar upper_bound para valores mayores a k
    //usar quitar el val1.v.size() para valores menores o iguales a k
    // usar lower_bound para valores estrictamente menoes a k(sin el val1.
        v.size())
    ans+=val1.v.size()-(upper_bound(all (val1.v),k)-val1.v.begin());
    return ans;
  }
  inline void build(){
    RREP(i,n-1,1) t[i]=Op(t[i<<1],t[i<<1|1]);
  }
  inline ll que(ll l, ll r, ll k){
    ll ans=0;
    for(l+=n,r+=n;l<r;l>>=1,r>>=1){
      if(l&1) ans+=Op1(t[l++],k);
      if(r&1) ans+=Op1(t[--r],k);
    }
    return ans;
  }
}st;

int main(){fastio;
  ll n; cin>>n;
  st.n=n;
  REP(i,0,n) {
    ll x; cin>>x;
    st.t[i+n].v.push_back(x);
  }
  st.build();
  ll q,ans=0,l,r,k; cin>>q;
  REP(i,0,q){
    cin>>l>>r>>k;// queries 1 base
    ans=st.que(l-1,r,k);
    cout<<ans<<endl;
  }
  return 0;
}
```