Team notebook

Javalieron

July 21, 2018

Contents

1 DS

1.1 simpleSegtree

```
#include <bits/stdc++.h>
using namespace std;
const int MAXN = 1e5+5;
int a[MAXN],t[MAXN];
void build(int v, int tl, int tr){
        if(tl==tr)
                t[v]=a[tl];
        else{
                int tm = (tl+tr)/2;
                build(v*2,tl,tm);
                build(v*2+1,tm+1,tr);
                //depende la operacin a realizar.
                t[v]=t[v*2]+t[v*2+1];
        }
}
int get(int v, int tl, int tr, int l, int r){
        if(l>r)
                //retornar valor neutro de la opracin.
                return 0;
        if(tl==l&&tr==r)
                return t[v];
        int tm = (tl+tr)/2;
        return get(v*2,tl,tm,l,min(r,tm))+get(v*2,tm+1,tl,max(tm+1,l),r);

}
void update(int v,int tl, int tr, int pos, int new_val){
        if(tl==tr)
                t[v]=new_val;
        else{
                int tm = (tl+tr)/2;
                if(pos<=tm)
                        update(v*2,tl,tm,pos,new_val);
                else
                        update(v*2+1,tm+1,tr,pos,new_val);
                t[v]=t[v*2]+t[v*2+1];
        }
}
int main(){
        int n=100,val=0,pos=5,i=2,j=3;
        //leer arreglo
        //construir el segtree
        build(1,0,n-1);
        int res = get(1,0,n-1,i,j);
        update(1,0,n-1,pos,val);
        res = get(1,0,n-1,0,10);
        printf("%d\n",res);


}
```

2 Grafos

2.1 kosaraju

```
#include <bits/stdc++.h>
using namespace std;
//choose MAXN according to the problem.
const int MAXN = 100005;
vector<int> g[MAXN],gr[MAXN];
bool vis[MAXN];
stack<int> tp;
int n,m;
int scc = 0;
void dfs(int x){
        vis[x]=1;
        for(vector<int>::iterator it = g[x].begin(); it!=g[x].end(); ++it){
                int y = *it;
                if(!vis[y])
                        dfs(y);
        }
        tp.push(x);
}
void dfs2(int x){
        vis[x]=1;
        for(vector<int>::iterator it = gr[x].begin(); it!=gr[x].end();
            ++it){
                int y = *it;
                if(!vis[y])
                        dfs2(y);
        }
}
int main(){
        //read graph.
        //kosaraju
        memset(vis,0,sizeof(vis));
        for(int i = 0; i<n; i++)
                if(!vis[i])
                        dfs(i);
        memset(vis,0,sizeof(vis));
        while(!tp.empty()){
                int x = tp.top();
                tp.pop();
                if(!vis[x]){
                        scc++;
                        dfs2(x);
                        //do extra things like graph condensation.
                }
        }
        return 0;
}
```