# 线段树合并

## 算法简介

一种合并多个线段树(一般为权值线段树)的算法，主要用于解决染色问题，时空间复杂度 $O(n)$

## 算法思想

更新线段树时动态开点，合并时如果遇到叶子节点或空结点就直接 $\text{return}$否则跑子树。

## 代码模板

```
const int MAXS=MAXN*60;
int root[MAXN],tot;
struct Node{
    int max_cnt,ans;//自己需要维护的信息
    int ch[2];
}node[MAXS];
void push_up(int k){//自定义
    if(node[node[k].ch[0]].max_cnt>=node[node[k].ch[1]].max_cnt)
node[k].max_cnt=node[node[k].ch[0]].max_cnt,node[k].ans=node[node[k].ch[0]]
.ans;
    else
node[k].max_cnt=node[node[k].ch[1]].max_cnt,node[k].ans=node[node[k].ch[1]]
.ans;
}
void update(int &k,int lef,int rig,int pos,int v){
    if(!k) k=++tot;
    if(lef==rig) return node[k].max_cnt+=v,node[k].ans=pos,void();
    int mid=lef+rig>>1;
    if(pos>mid)
    update(node[k].ch[1],mid+1,rig,pos,v);
    else
    update(node[k].ch[0],lef,mid,pos,v);
    push_up(k);
}
void Merge(int &k1,int k2,int lef,int rig){
    if(!k1||!k2) return k1|=k2,void();
    if(lef==rig) return node[k1].max_cnt+=node[k2].max_cnt,void();
    int mid=lef+rig>>1;
    Merge(node[k1].ch[0],node[k2].ch[0],lef,mid);
    Merge(node[k1].ch[1],node[k2].ch[1],mid+1,rig);
    push_up(k1);
}
```

# 算法练习

## 习题一

洛谷p4556

### 题意

给定一棵 $n$ 个节点的数，$m$ 个操作。

每个操作三个参数 $x,y,z$，表示给结点 $x$ 到 $y$ 的树链上的每个点打上一个 $z$ 号标记。

经过所有操作后输出每个结点被打上的最多的标记的编号(满足条件的标记存在多个时输出编号最小的)，如果该结点未被标记过，输出 $0$。

### 题解 1

```

```

### 题解 2

```

```