格式

1. 英文/公式左右两边跟汉字需要空格
2. 结点 u, v 等也需要使用公式
3. log 使用 $\log$
4. 句子结束请使用句号
5. 建议写 merge，mergE 很怪

内容：个人感觉讲得不够清楚，需要反复看几遍，并结合代码才能大致看懂。建议用形式化的语言写明合并过程，在简介部分举例（有图最好）等方法改进。另外这个复杂度怎么是个玄学。。没有严谨一点的证明吗？建议再检索一下资料。

线段树合并

简介

~~线段树合并是怎么回事呢？线段树大家应该很熟悉，那么…~~

有时候会遇到树上每个点开一个桶或者一个线段树，然后在 dfs 过程中不断合并的情况，一般权值线段树可以替代桶，然后又因为线段树有较好的合并性质，所以就有了线段树合并的说法。

一般来说关键步骤就是**动态开点**和**线段树的合并操作**

因为不可能真的在每个点建一棵完整的线段树（而且一般情况下每个点所代表的权值都只是完整线段树的小部分），所以要动态开点

```
void update(int &id,int l,int r,int pos,int k)
{
    if(!id) id=++tot;  //动态开点
    if(l==r){
        //更新操作
        return ;
    }
    int mid = (l+r)>>1;
    if(pos<=mid) update(tr[id].lc,l,mid,pos,k);
    else update(tr[id].rc,mid+1,r,pos,k);
    push_up(id);
}
```

然后合并的操作主要基于两棵线段树具有相同的值域所以有相同的结构，那么递归的处理左右子树即可

```
int merge(int p,int q,int l,int r)
{
    if(!p)return q;if(!q)return p;
    if(l==r){
        // merge_leaf 操作
        return p;
    }
    int mid = (l+r)>>1;
    tr[p].lc = merge(tr[p].lc,tr[q].lc,l,mid);
    tr[p].rc = merge(tr[p].rc,tr[q].rc,mid+1,r);
```

```
    push_up(p);
    return p;
}
```

复杂度

空间复杂度比较要注意（因为涉及数组开多大QAQ□□每次 update 最多开 $logn$ 个点，所以一般开 $mlogn$ 的数组□m 表示 update 次数□n 表示权值线段树的值域上限

~~时间复杂度的话就不会搞子~~，主要开销在 merge 操作上，两个树的公共点越多开销越大（这好像没办法算啊），但注意一点是如果在一颗树上合并了 $n-1$ 次那么这个复杂度不会比直接建一棵完整的线段树来得大

例题

P4556雨天的尾巴

给出一棵树□m 次操作，每次操作让 path<u,v> 上的点都发放第 k 种粮食。所有 m 次操作完后，求每个点存放最多的粮食的种类是什么。

显然是树上点差分，但是值域很大，可以在每个点建一棵权值线段树，然后合并即可，注意差分在 update 上有常数 4，所以开数组要多乘以 4

P3605 [USACO17JAN]Promotion Counting P

一棵带点权的树，求每个点 u 的子树中有多少个点的点权比 u 自身大

也是每个点建一个权值线段树，权值线段树维护每个值出现的次数，然后dfs过程中合并完后 query 一下

P3224 HNOI2012]永无乡

n 个点，两个操作，一是连接两个点，二是求某个点和所有和他相连的点的第k小

第 k 小也是权值线段树干的事儿，连接的操作用并查集维护

CF600E. Lomsat gelral

树上每个点染色 c_i □好像树上染色可以用 dsu □原来我不会啊，那没事了），求每个点子树出现次数最多的颜色的和（可能多个颜色同时出险次数最多）

也是线段树瞎搞搞，最后 dfs 合并一下