# **可并堆(**左偏堆**)：**

顾名思义就是可以合并的堆，堆的实现本身不难，而且有stl可用，但可并堆必须手写。可并堆是并查集和堆的合体，但是它只有2个叉，它和堆的不同是，可并堆不是一个堆，而是一群堆，初始是n个离散节点，各自是一个堆，操作如下：

合并操作：合并两个堆，核心，下面说

插入操作：插入操作也只是把某个单独节点与要插入的堆合并，本质还是合并，删除：只能删除某个堆顶端，要调整堆，不是真的删除节点，而是标记成被删除，为了节省内存，小顶堆可把val赋值成负无穷。

维护属性 ：左右和父亲地址，权值val，距离dis，它表示当前节点到最近叶子结点的距离，本身是叶子dis=0，注意到普通堆里没有dis这个属性。

以下用小跟堆来说。可并堆有如下性质：

1. 节点的权值小于等于它左右儿子的权值。(堆的基本性质)
2. 节点的左儿子的距离不小于右儿子的距离。(所以有些地方叫左偏堆)
3. 节点的距离等于右儿子的距离+1。(基于性质2的推论)
4. n个节点的可并堆距离最大为log(n+1)-1 （思考并计算得到）

## 合并：

对于两个堆A B，设A是根节点val较小的树，把A的根节点作为新树的根节点。

之后顺着A的右子节点，走到最右叶子，把新来的插在这个叶子上。

此时，会打破性质2，也就是距离会出现左小于右，就要递归的调整，遍历最右叶子到根节点的链每个节点，发现某个不满足性质2，就交换它的左右子树即可。

由于是左偏堆，这么做，会保证堆的尽可能平衡。

此外，对于给定2个点x和y,合并x和y所在树，应该先找到他们根节点。判断是不是本来就是一棵树，类似并查集那样。

## 删除：

删除一个堆顶后，调用合并操作，合并它的儿子即可。

删除一个堆顶肯定不是真的删除，而是标记成某个数。