笛卡尔树和treap是一个东西，都是特殊的二叉搜索树,每个节点有key和val两个属性，key满足二叉搜索树性质，同时val满足堆性质。只不过在不同用法的情况下，二者叫法不同。在侧重于”平衡”的地方，比如求第k大,前驱等。堆的性质而有助于保持树的平衡，主要是用到key。而笛卡尔树，是侧重于val这个属性。Key只是为了查找。两个叫法实际是一个东西

## 数组构建笛卡尔树：

给定一个数组A,构建笛卡尔树,对于某个元素A[i],下标i作为key，A[i]作为val,构建完成后，一定要使得key满足二叉搜索树性质,且同时val满足堆性质。

可以证明(因为A数组排序后一定只要一种情况,再按照bst树插入构建)，对于给定一个数组A,和给定排列顺序(堆的比较规则),这样的树有且只要一种形态,

如何较快构建笛卡尔树呢？

以大顶堆为例子。

不知道栈先看单调栈使用。从根开始一直往右走的那一条链，用单调栈维护val值(单调递减),栈顶代表树最右边的元素，把数组按照下标的顺序一个个建树。

对于某个元素a,看最右链的叶子点网上找,找第一个比他大的元素b(如果都比它大,说明它就是最小的，直接插在最右就好了),把a元素插在b的右边。原先所有b的右树，都连到a的左边(实际就是二叉搜索树单旋转操作)。如果a比所有点都大，那么a成为新的根节点，把所有点放在a左边。

由于这个过程是按照A数组顺序构建的，所有key一定满足二叉搜索树性质。